A Symbolic Semantics for the π-calculus*

Michele Boreale and Rocco De Nicola
Dipartimento di Scienze dell'Informazione
Università di Roma "La Sapienza"
E-mail: michele@dsi.uniroma1.it, denicola@vm.cnuce.cnr.it

Abstract

We use symbolic transition systems as a basis for providing the π-calculus with an alternative semantics. The latter is more amenable to automatic manipulation and sheds light on the logical differences among different forms of bisimulation over algebras of name-passing processes. Symbolic transitions have the form $P \stackrel{\phi,\alpha}{\longmapsto} P'$, where ϕ is a boolean combination of equalities on names that has to hold for the transition to take place, and α is standard a π-calculus action. On top of the symbolic transition system, a symbolic bisimulation is defined that captures the standard ones. Finally, a sound and complete proof system is introduced for symbolic bisimulation.

1 Introduction

The π-calculus [MPW92] is a widely studied process description language with primitives for expressing the exchange of channel names (or simply *names*) among processes. The exchanged names can also be tested for identity. These features permit a natural description of systems with dynamic linking.

Like traditional process algebras, the π-calculus has undergone severe scrutiny and different semantics for it have been proposed, relying on the standard notions of bisimulation [MPW92, PS93] and testing [Hen91, BD92]. However, theoretical studies to support equivalence checking have just begun. In [San93], with efficiency motivations, a new form of π-calculus bisimulation equivalence, called *open*, is studied. In particular, a kind of *conditional* transition system is introduced to efficiently characterize the equivalence.

In this paper, we build on previous work by Hennessy and Lin on value-passing process algebras [HL92, HL93] and extend the work on open bisimulation of [San93] to capture the (*early* and *late*) bisimulations originally proposed in [MPW92]. For these equivalences, our framework yields alternative, more efficient characterizations and complete proof systems to reason about them. An additional advantage of the proposed framework is that it sheds light on the conceptual differences among the different forms of name-passing bisimulation. Our attention will be confined to strong bisimulations, but we do not see any serious obstacle in extending our results to the weak ones.

The basic theory of bisimulation for the π-calculus has been introduced in [MPW92]. The fundamental notion is that of *ground* bisimulation; it has the same conceptual simplicity as in CCS and suggests a natural strategy for equivalence checking. However, due to name passing, this definition-based verification technique runs into serious efficiency problems. On input actions, a case analysis on the received names is needed to check that receiving equal names leads to

*An extended abstract of this paper appears in: B. Jonsson and J. Parrow (eds.), *Proceedings of CONCUR '94*, Lectures Notes in Computer Science 836, Springer-Verlag, 1994. Work partially supported by Progetto Finalizzato Sistemi Informatici e Calcolo Parallelo Contract n. 93.01599.69, by Istituto di Elaborazione dell'Informazione of C.N.R. at Pisa and by the EXPRESS project within the HCM program of EEC.

"equivalent" states. To see this, consider the processes $P = a(y).P'$ and $Q = a(y).Q'$; here the input prefix operator $a(y).R$ is used to describe receipt of a name at channel a and its substitution for the formal parameter y within R. To check that P and Q are ground bisimilar, following the definition we would have to verify that P' and Q' are bisimilar for all possible instantiations of y with a name occurring free in P and Q or with a fresh name. Of course, performing multiple checks for each input action very often leads to a combinatorial explosion.

Input prefix also introduces another problem: it does not preserve the ground equivalence. This leads to considering the maximal *congruence* included in the ground equivalence, obtained by closing the latter under name substitutions [MPW92]. Checking for congruence of two terms by relying on the original definition would require performing several (one for each "relevant" substitution) ground equivalence checks.

Indeed, results obtained by Parrow and Jonsson [JP93], show that the problem of deciding bisimilarity for (even very simple) message-passing process algebras is NP-hard. Thus, combinatorial explosions are difficult to avoid in general. Nonetheless, a simple example is sufficient to appreciate that, when verifying π-calculus bisimulation with the above mentioned techniques, many performed checks are indeed useless.

Consider the two processes

$$P_1 = a(y).P \text{ and } P_2 = a(y).([y = z]P + [y \neq z]P)$$

where y and z are distinct and the operator $+$ stands for external choice. Both *match* $[z = y]$, and *mismatch* operators $[z \neq y]$, are used; $[z = y]P$ stands for a process that behaves like P if z and y are syntactically equal and is blocked otherwise, while $[z \neq y]P$ has exactly the opposite meaning.

It should be immediate to establish that P_1 and P_2 are equivalent. But, checking their equivalence by directly relying on its definition, requires checking that P and $[y = z]P + [y \neq z]P$ are bisimilar when y is replaced by z or by each of the free names of P or one fresh name. This could be very costly; it is however evident that, beside y, z is the only name that matters.

Therefore, a significant gain in efficiency could be obtained by finding a systematic way to prune non-essential cases when performing the case analysis. To this aim, the idea pursued in this paper, and originally proposed by Hennessy and Lin for value-passing processes, is that of setting up a framework where case analysis can be performed *incrementally*. First, a new symbolic transition system for the π-calculus is introduced, where logical conditions (equalities and inequalities among names) that make a transition possible are made explicit. Then, a new bisimulation, which we refer to as *symbolic*, is defined, that performs the case-analysis directly on these conditions. It will be proved that symbolic bisimulation can be used to establish the standard ones.

The symbolic transitions are of the form $P \stackrel{\phi, \alpha}{\longmapsto} P'$, where ϕ is a *boolean formula* over names that has to hold for the transition to take place, and α is a standard π-calculus action. A typical symbolic transition is $[x = y]\alpha.P \stackrel{[x=y], \alpha}{\longmapsto} P$, saying that action α can be performed under any interpretation of names satisfying $x = y$. A boolean formula is in general built from the basic formulae $[x = y]$ via the standard boolean connectives.

Symbolic bisimulation is defined on the top of the symbolic transition system. This leads to a family of distinct symbolic equivalences $\simeq^{\phi}$, depending on the boolean formula ϕ (equivalence under ϕ). Informally, to verify $P \simeq^{\phi} Q$, it is required to find, for each symbolic move $P \stackrel{\psi, \alpha}{\longmapsto} P'$ of P, a case-partition of the condition $\phi \wedge \psi$, such that each subcase entails a matching symbolic move for Q, and vice-versa for Q and P. As an example, the equivalence under *true* of the previously defined processes P_1 and P_2, $P_1 \simeq^{true} P_2$, is readily verified by partitioning condition *true* as $\{[z = y], [z \neq y]\}$; this is sufficient because each of these two conditions entails that $[y = z]P + [y \neq z]P$ is equivalent to P.

Symbolic equivalences are related to the standard ones by the following 'consistency and ade-

quacy' theorem:

$P \simeq^{\phi} Q$ if and only if for each name substitution σ satisfying ϕ, $P\sigma$ is ground equivalent to $Q\sigma$

where a name substitution σ *satisfies* ϕ if the result of applying σ to ϕ is a tautology; here, $P\sigma$ denotes the result of applying σ to P .

The above statement tells us that the symbolic equivalence $\simeq^{\phi}$ is the closure of the ground equivalence under all name substitutions satisfying ϕ. Thus, for example, the congruence (i.e. the closure w.r.t. *all* substitutions) will be recovered as the symbolic bisimulation $\simeq^{true}$. Ground equivalence of two specific processes P and Q will be instead recovered as a symbolic equivalence $P \simeq^{\phi(P,Q)} Q$, where $\phi(P, Q)$ is a condition imposing that all free names in P and Q be distinct.

In the paper, we also present a proof system to reason about symbolic bisimulation. The statements derivable within the system are of the form $\phi \rhd P = Q$ and, for finite processes, the system is sound and complete in the sense that $\phi \rhd P = Q$ is derivable if and only if $P \simeq^{\phi} Q$. By taking advantage of the symbolic transitional semantics, the proof of completeness is, by and large, a symbolic version of the classical proof for strong bisimulation over CCS [Mil89]. Additional complications are however introduced by the fact that the boolean condition ϕ may also constrain the communication capabilities of processes.

The symbolic characterization of the standard equivalences has an additional advantage; it sheds new light on the conceptual difference between different forms of bisimulations for the π-calculus. In [MPW92], two forms of ground bisimulation (each inducing a different congruence) were introduced, the *early* form and the *late* one. Intuitively, they correspond to two different instantiation strategies for the formal parameter of input actions: in the first strategy (early), the instantiation is performed at the moment of inferring the input action, while in the other (late) it is performed later, when a communication is actually inferred. In *open* bisimulation [San93], the instantiation may be delayed arbitrarily long; this yields an equivalence stronger than the early and late ones. Our symbolic formulation indicates that each of the mentioned strategies corresponds to a different degree of generality in performing case-analysis. This will be discussed in the concluding section.

Our work has strong connections with the papers [San93], [HL92], [HL93] and [PS93]. In [San93], conditional transition system was first used for the π-calculus. In [HL92], the notion of symbolic bisimulation was introduced within a syntax-free framework, where *symbolic transition graphs* are considered; a verification algorithm is also proposed. In [HL93], for a version of CCS with value-passing, an adequacy result and a sound and complete proof system similar to ours are presented. In [PS93], for the same name-passing language considered here, late and early ground equivalences and the induced congruences are equipped with four distinct algebraic proof systems; efficiency considerations are absent.

The present paper may be viewed as the extension of [HL92] and [HL93] to a name-passing calculus, for which more efficient characterizations of different bisimulation-based equivalences are obtained. A more detailed comparison with these and other mentioned papers is deferred to Section 7. Here we want only to point to what makes this extension non-trivial: the blurring of values and channel names. This is a distinctive feature of the π-calculus, that allows names to appear both in the actions, in the processes and in the boolean formulae; it gives rise to a subtle interplay between name-scoping and boolean formulae. This interplay is best revealed in the symbolic structural operational rules for one of the name-binders of the π-calculus, the *restriction* operator $(\nu\, y)$. In $(\nu\, y)P$, the name y is declared to be *new*, i.e. different from any other name. Therefore, when we have $P \stackrel{\phi,\alpha}{\longmapsto} P'$ as a premise of an inference rule for $(\nu\, y)P$, we have to discard from ϕ in the conclusion every assumption requiring y to be equal to other names, thus obtaining a new formula, $R_y(\phi)$, not containing y.

The rest of the paper is organized as follows. In Section 2, after introducing the π-calculus and the standard notions of bisimulation equivalences, the symbolic transitional semantics and

symbolic bisimulation are presented. In Section 3, some basic properties relating substitutions to boolean formulae, transitional semantics and bisimulation are established; these properties will be used in later sections. In Section 4, the main theorems, i.e. consistency and adequacy of symbolic equivalence w.r.t. standard equivalences, are proven. Section 5 presents the proof system and the corresponding theorems of soundness and completeness. In Sections 2 - 5 only early bisimulation is considered, while late bisimulation is treated in full detail in Section 6. The final section contains conclusions, comparisons with related work and suggestions for future research.

2 Symbolic Semantics

In this section the π-calculus [MPW92] and the standard bisimulation equivalences will be briefly reviewed; then the new symbolic semantics will be introduced.

2.1 The π-calculus and its standard bisimulation semantics

Definition 2.1 (Syntax) *Let $\mathcal{N}$ be a countable set and x, y range over it, let ϕ range over the language BF of* Boolean Formulae*:*

$$\phi ::= true \;\mid\; [x = y] \;\mid\; \neg\phi \;\mid\; \phi \wedge \phi$$

and let α range over actions silent move, input *and* free output*:*

$$\alpha ::= \tau \;\mid\; x(y) \;\mid\; \overline{x}y.$$

Let X range over a countable set of agent variables*. The language of* agent terms *is built by means of* agent variables, inaction, action prefix, summation, boolean guard, restriction, parallel composition *and* recursion *in the following way:*

$$P ::= X \;\mid\; \mathbf{0} \;\mid\; \alpha.P \;\mid\; P + P \;\mid\; \phi P \;\mid\; (\nu\, y)P \;\mid\; P|P \;\mid\; recX.P.$$

Note that, contrary to the original definition of the π-calculus [MPW92], our grammar does not allow name parameters in recursive definitions ($recX.P$). This is not due to particular difficulties in dealing with parameters. We keep the language as simple as possible, in order to focus on the relevant new concepts.

We fix now some basic notations. We shall use $false$ for $\neg true$, $[x \neq y]$ for $\neg[x = y]$ and $\phi_1 \vee \phi_2$ for $\neg(\neg\phi_1 \wedge \neg\phi_2)$. The *evaluation* of a boolean formula ϕ, $Ev(\phi)$, into the set $\{true, false\}$ is defined inductively as expected, once we set that $Ev(true) = true$, $Ev([x = x]) = true$ and $Ev([x = y]) = false$, for distinct names x and y. $n(\phi)$ will denote the set of names occurring in ϕ.

We use the *bound output prefix* $\overline{x}(y).P$, $x \neq y$, as a shorthand for $(\nu\, y)(\overline{x}y.P)$. Bound output *actions* are of the form $\overline{x}(y)$ with $x \neq y$. If $\alpha = x(y)$ or $\alpha = \overline{x}y$ or $\alpha = \overline{x}(y)$, we let $subj(\alpha) = x$ and $obj(\alpha) = y$. The π-calculus has two kinds of name *binders*: input prefix $x(y).P$ and restriction $(\nu\, y)P$ bind the name y in P; consequently, the notions of *free names, fn(.), bound names, bn(.)* and *α-equality*, over agent terms, formulae and actions are the expected ones (we define $fn(\phi) = n(\phi)$ for a boolean formula ϕ). We let $n(.) = fn(.) \cup bn(.)$.

A *process* is an agent term such that: (i) each occurrence of any agent variable X lies within the scope of a $recX.$ operator, and (ii) for each subterm of the form $recX.P$ and for each $y \in fn(P)$, no occurrence of X inside P lies within the scope of a binder for y. The requirement (ii) ensures that no occurrence of a free name becomes bound when "unfolding" $recX.P$ into $P[recX.P/X]$. We let $\mathcal{P}$ denote the set of processes. In the rest of the paper we confine ourselves to agent terms denoting processes.

Substitutions, ranged over by σ, ρ, are functions from $\mathcal{N}$ to $\mathcal{N}$; for any $x \in \mathcal{N}$, $\sigma(x)$ will be written as $x\sigma$. Given a substitution σ and $V \subseteq \mathcal{N}$, we define:

- $V\sigma = \{x\sigma |\, x \in V\}$
- $dom(\sigma) = \{x |\, x\sigma \neq x\}$

- $range(\sigma) = dom(\sigma)\sigma$
- $n(\sigma) = dom(\sigma) \cup range(\sigma)$

In the rest of the paper we confine ourselves to *finite* substitutions, i.e. those σ s.t. $n(\sigma)$ is finite. If t is either an action, a formula or a process, $t\sigma$ denotes the expression obtained by simultaneously replacing in t each $x \in fn(t)$ with $x\sigma$, with renaming of bound names possibly involved. Composition of substitutions is denoted by juxtaposition: given two substitutions σ_1 and σ_2, $\sigma_1\sigma_2$ denotes the substitution such that $t(\sigma_1\sigma_2) = (t\sigma_1)\sigma_2$. The set $\{y_1/x_1, \ldots, y_n/x_n\} = \{\tilde{y}/\tilde{x}\}$, with the x_i's pairwise distinct, will denote the following substitution σ: $x\sigma = y_i$ if $x = x_i$ for some $i \in \{1, \ldots, n\}$, $x\sigma = x$ otherwise. We also extend fn to substitutions and let $fn(\sigma)$ to be $n(\sigma)$; in this way, the function $fn(\,.\,)$ is defined over names, actions, processes, formulae and substitutions. Notations such as $fn(P, \alpha, \sigma)$ will be used to indicate $fn(P) \cup fn(\alpha) \cup fn(\sigma)$.

Unless otherwise stated, we will let $x, y, \ldots$ range over $\mathcal{N}$, $\alpha, \beta, \ldots$ over the set of actions (including bound output), $\phi, \psi, \ldots$ over BF, $P, Q, \ldots$ over $\mathcal{P}$ and $\rho, \sigma, \ldots$ over substitutions.

The standard "concrete" transitional semantics of $\mathcal{P}$ is given in Table 1(a). Following [PS93] *we shall work up to α-equivalence*; indeed *α-equivalent agents are deemed to have the same transitions.*

The definition of (standard) early ground bisimulation equivalence $\dot{\sim}$ and early bisimulation congruence $\sim$ can be given as follows:

Definition 2.2 (Early bisimulation)

- *A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a ground early bisimulation iff $(P, Q) \in \mathcal{R}$ and $P \xrightarrow{\alpha} P'$ with $bn(\alpha) \cap fn(P, Q) = \emptyset$, imply*
  - *if α is not an input action, then $\exists\ Q'$: $Q \xrightarrow{\alpha} Q'$ and $(P', Q') \in \mathcal{R}$.*
  - *if $\alpha = x(y)$, then $\forall z \in fn(P, Q, y)\ \exists\ Q' : Q \xrightarrow{\alpha} Q'$ and $(P'\{z/y\}, Q'\{z/y\}) \in \mathcal{R}$.*
- *$\dot{\sim} = \cup\{\mathcal{R} \,|\, \mathcal{R}$ is a ground early bisimulation $\}$.*
- *$P \sim Q$ iff $\forall\, \sigma :\ P\sigma \dot{\sim} Q\sigma$.*

The *late* version of the above definition is obtained by replacing the input-clause (the second clause of the first item) by the stronger:

- *if $\alpha = x(y)$, then $\exists Q' : Q \xrightarrow{\alpha} Q'$ and $\forall\ z \in fn(P, Q, y) : (P'\{z/y\}, Q'\{z/y\}) \in \mathcal{R}$.*

It has been shown [MPW92] that late bisimulation is strictly finer than early bisimulation.

2.2 Symbolic semantics

Before introducing the symbolic semantics, we need to fix some additional notation for boolean formulae and substitutions.

Definition 2.3 (Basic definitions)

- *$\sigma \models \phi$ stands for $Ev(\phi\sigma) = true$;*
- *$\phi \models \psi$ stands for $\forall\ \sigma$: $\sigma \models \phi$ implies $\sigma \models \psi$;*
- *$[\alpha = \beta]$ stands for*

$[x = z]$	*if for some y: $(\alpha = x(y)$ and $\beta = z(y))$ or $(\alpha = \overline{x}(y)$ and $\beta = \overline{z}(y))$;*
$[x = z] \wedge [y = w]$	*if $\alpha = \overline{x}y$ and $\beta = \overline{z}w$;*
$true$	*if $\alpha = \beta = \tau$;*
$false$	*otherwise.*

$$\texttt{Act}\ \frac{-}{\alpha.P \xrightarrow{\alpha} P}$$

$$\texttt{Sum}\ \frac{P_1 \xrightarrow{\alpha} P_1'}{P_1 + P_2 \xrightarrow{\alpha} P_1'} \qquad \texttt{Par}\ \frac{P_1 \xrightarrow{\alpha} P_1'}{P_1 \mid P_2 \xrightarrow{\alpha} P_1' \mid P_2}\ bn(\alpha) \cap fn(P_2) = \emptyset$$

$$\texttt{Com}\ \frac{P_1 \xrightarrow{\overline{x}z} P_1',\ P_2 \xrightarrow{x(y)} P_2'}{P_1 \mid P_2 \xrightarrow{\tau} P_1' \mid P_2'\{z/y\}} \qquad \texttt{Close}\ \frac{P_1 \xrightarrow{\overline{x}(y)} P_1',\ P_2 \xrightarrow{x(y)} P_2'}{P_1 \mid P_2 \xrightarrow{\tau} (\nu\, y)(P_1' \mid P_2')}$$

$$\texttt{Res}\ \frac{P \xrightarrow{\alpha} P'}{(\nu\, y)P \xrightarrow{\alpha} (\nu\, y)P'}\ y \notin n(\alpha) \qquad \texttt{Open}\ \frac{P \xrightarrow{\overline{x}y} P'}{(\nu\, y)P \xrightarrow{\overline{x}(y)} P'}\ x \neq y$$

$$\texttt{Guard}\ \frac{P \xrightarrow{\alpha} P'}{\phi P \xrightarrow{\alpha} P'}\ Ev(\phi) = true \qquad \texttt{Rec}\ \frac{P[recX.P/X] \xrightarrow{\alpha} P'}{recX.P \xrightarrow{\alpha} P'}$$

a) Standard SOS for $\mathcal{P}$; the symmetric rules for `Sum`, `Par`, `Com` and `Close` are omitted.

$$\texttt{S}-\texttt{Act}\ \frac{-}{\alpha.P \overset{true,\alpha}{\longmapsto} P}$$

$$\texttt{S}-\texttt{Sum}\ \frac{P_1 \overset{\phi,\alpha}{\longmapsto} P_1'}{P_1 + P_2 \overset{\phi,\alpha}{\longmapsto} P_1'} \qquad \texttt{S}-\texttt{Par}\ \frac{P_1 \overset{\phi,\alpha}{\longmapsto} P_1'}{P_1 \mid P_2 \overset{\phi,\alpha}{\longmapsto} P_1' \mid P_2}\ bn(\alpha) \cap fn(P_2) = \emptyset$$

$$\texttt{S}-\texttt{Com}\ \frac{P_1 \overset{\phi_1,\overline{x}z}{\longmapsto} P_1',\ P_2 \overset{\phi_2,w(y)}{\longmapsto} P_2'}{P_1 \mid P_2 \overset{\phi_1\wedge\phi_2\wedge[x=w],\tau}{\longmapsto} P_1' \mid P_2'\{z/y\}} \qquad \texttt{S}-\texttt{Close}\ \frac{P_1 \overset{\phi_1,\overline{x}(y)}{\longmapsto} P_1',\ P_2 \overset{\phi_2,w(y)}{\longmapsto} P_2'}{P_1 \mid P_2 \overset{\phi_1\wedge\phi_2\wedge[x=w],\tau}{\longmapsto} (\nu\, y)(P_1' \mid P_2')}$$

$$\texttt{S}-\texttt{Res}\ \frac{P \overset{\phi,\alpha}{\longmapsto} P'}{(\nu\, y)P \overset{R_y(\phi),\alpha}{\longmapsto} (\nu\, y)P'}\ y \notin n(\alpha) \qquad \texttt{S}-\texttt{Open}\ \frac{P \overset{\phi,\overline{x}y}{\longmapsto} P'}{(\nu\, y)P \overset{R_y(\phi),\overline{x}(y)}{\longmapsto} P'}\ x \neq y$$

$$\texttt{S}-\texttt{Guard}\ \frac{P \overset{\psi,\alpha}{\longmapsto} P'}{\phi P \overset{\phi\wedge\psi,\alpha}{\longmapsto} P'} \qquad \texttt{S}-\texttt{Rec}\ \frac{P[recX.P/X] \overset{\phi,\alpha}{\longmapsto} P'}{recX.P \overset{\phi,\alpha}{\longmapsto} P'}$$

b) Symbolic SOS for $\mathcal{P}$; the symmetric rules for $\texttt{S}-\texttt{Sum}$, $\texttt{S}-\texttt{Par}$, $\texttt{S}-\texttt{Com}$ and $\texttt{S}-\texttt{Close}$ are omitted.

Table 1: Standard and Symbolic SOS for $\mathcal{P}$

- $\bigvee D$, *where* $D = \{\phi_1, \ldots, \phi_n\} \subseteq_{fin} BF$, $n > 0$, *is the boolean formula* $\phi_1 \vee \ldots \vee \phi_n$. *A similar notation will be used for* $\bigwedge D$. *Furthermore, we let* $\bigvee \emptyset$ *denote* $false$ *and* $\bigwedge \emptyset$ *denote* $true$.

- *For each name* y, *the function* $R_y(\,) : BF \longrightarrow BF$ *is defined by structural induction on* ϕ *as follows:*

$$\begin{array}{rll} R_y(true) & = true & \\ R_y([w_1 = w_2]) & = [w_1 = w_2] & \text{if } y \notin \{w_1, w_2\} \\ R_y([y = y]) & = true & \\ R_y([y = w]) & = R_y([w = y]) = false & \text{if } y \neq w \\ R_y(\neg\phi) & = \neg R_y(\phi) & \\ R_y(\phi_1 \wedge \phi_2) & = R_y(\phi_1) \wedge R_y(\phi_2) & \end{array}$$

The symbolic transitional semantics of $\mathcal{P}$ is presented in Table 1(b). Each symbolic rule is the counterpart of a concrete one. Intuitively, the boolean formula ϕ in $P \stackrel{\phi,\alpha}{\longmapsto} P'$ collects the conditions on the free names of P necessary for α to take place. E.g., rule $\mathtt{S-Act}$ says that $\alpha.P$ can perform α unconditionally. In $\mathtt{S-Com}$ and $\mathtt{S-Close}$, the condition of matching channels ($[x = w]$) is moved into the boolean condition of the resulting symbolic transition; this is necessary to infer symbolic transitions such as:

$$(x(y).\mathbf{0}) \mid (\overline{w}y.\mathbf{0}) \stackrel{[x=w],\tau}{\longmapsto} \mathbf{0} \mid \mathbf{0}\,.$$

Rule $\mathtt{S-Res}$ reveals the interplay between name-scoping and boolean formulae: in $(\nu\, y)P$, the name y is declared to be *new*, i.e. different from any other name; thus the rule says that, given a symbolic transition $P \stackrel{\phi,\alpha}{\longmapsto} P'$ as a premise, under the assumption that y is new, every positive assumption about the identity of y has to be discarded from ϕ in the conclusion; as a result, the new formula $R_y(\phi)$ that does not contain y is obtained. An example of derivation using this rule is:

$$\mathtt{S-Res}\,\frac{P \stackrel{[y=z]\vee[z=w],\tau}{\longmapsto} P'}{(\nu\, y)P \stackrel{false\vee[z=w],\tau}{\longmapsto} (\nu\, y)P'}$$

where z and w are different from y (note that $false \vee [z = w]$ is equivalent to $[z = w]$). A similar comment holds for $\mathtt{S-Open}$. The other inference rules should be self-explanatory.

We are now set to introduce symbolic bisimulation for the π-calculus. For a variant of CCS with value passing, symbolic bisimulation has already been introduced in [HL92]. The underlying intuition is that of establishing equivalence of, say, P and Q under a condition ϕ, by matching symbolic transitions of P, $P \stackrel{\psi,\alpha}{\longmapsto} P'$, by *sets* of symbolic transitions of Q. More precisely, the condition $\phi \wedge \psi$, is partitioned into a set D of subcases, each of which entails a matching transition for Q. Due to the treatment of restricted names in the π-calculus, here we have to take into account the case when the performed action is a bound output $\overline{x}(y)$ which introduces a new name y. In that case, for subsequent reasoning, we also impose that y be logically different from any name known until that moment.

Before defining symbolic bisimulation, we introduce ϕ-*decompositions*.

Definition 2.4 (ϕ-decomposition) *Given* ϕ *and a finite set of formulae* $D = \{\phi_1, \ldots, \phi_n\}$, *we say that* D *is a* ϕ-decomposition *iff* $\phi \models \bigvee D$.

A similar concept (ϕ-*partitions*) was already present in [HL93]. The only difference is that the disjunction of a ϕ-partition is required to be *equivalent* to ϕ, whereas a ϕ-decomposition is just implied by ϕ.

Definition 2.5 (Symbolic early bisimulation)

- *A family* $\mathcal{F} = \{R_\phi \,|\, \phi \in BF\}$ *of symmetric binary relations over* $\mathcal{P}$*, indexed over the set* BF *of boolean formulae, is a* family of symbolic early bisimulations (FSEB) *iff* $\forall\ \phi \in BF$ *and* $(P,Q) \in R_\phi$*,* $P \stackrel{\psi,\alpha}{\longmapsto} P'$*, with* $bn(\alpha) \cap fn(P,Q,\phi) = \emptyset$ *, implies:*

  there exists a χ*-decomposition* D*, such that for all* $\phi' \in D$*, there is a transition* $Q \stackrel{\psi',\beta}{\longmapsto} Q'$ *with* $\phi' \models (\psi' \wedge [\alpha = \beta])$ *and* $(P',Q') \in R_{\phi'}$*, where:*

$$\chi = \begin{cases} \phi \wedge \psi \wedge \bigwedge_{z \in fn(P,Q,\phi)} [y \neq z] & \text{if } \alpha \text{ is a bound output action } \overline{x}(y) \\ \phi \wedge \psi & \text{otherwise.} \end{cases}$$

- $P \simeq^\phi Q$ *iff there exists a FSEB* $\{R_\psi |\, \psi \in BF\}$ *such that* $(P,Q) \in R_\phi$*.*

We now discuss informally the reason why symbolic bisimulation is more amenable to efficient automatic verification than the concrete one, leaving for future work a precise complexity analysis. Note that, on input actions, no multiple instantiation of the formal parameter is required; instead, a single instantiation with a *fresh* name suffices (the "freshness" condition is $bn(\alpha) \cap fn(P,Q,\phi) = \emptyset$[1]). The case analysis on the received value is now embodied in the decomposition D; by choosing this decomposition in an appropriate way, the number of cases to deal with can be significantly smaller than that arising with the original definition. This is shown in the following example.

Example 2.6 Consider the processes P_1 and P_2 given in the Introduction:

$$P_1 = x(y).P \text{ and } P_2 = x(y).([y = z]P + [y \neq z]P)$$

where y and z are different. $P_1 \sim P_2$ can be established by showing $P_1 \simeq^{true} P_2$, or by relying on the original definition. Below, we make use of both methods.

$P_1 \simeq^{true} P_2$ can be established as follows. Let $P' = [y = z]P + [y \neq z]P$ and Id denote the identity relation. It is easy to see that the family of relations below :

$$\begin{array}{rcl} \mathcal{R}_{true} & = & \{(P_1,P_2),(P_2,P_1)\} \\ \mathcal{R}_{[y=z]} & = & \{(P,P'),(P',P)\} \\ \mathcal{R}_{[y\neq z]} & = & \{(P,P'),(P',P)\} \\ \mathcal{R}_{\phi} & = & Id \ \text{ for } \phi \notin \{true,[y=z],[y\neq z]\} \end{array}$$

is an FSEB, corresponding to decomposing *true* into $\{[z = y],[z \neq y]\}$. Note that, formally, we have to exhibit infinitely many relations (one for each $\phi \in BF$); but it turns out that only finitely many are different from Id.

To establish $P_1 \sim P_2$ relying on the definition, we should prove that for each substitution σ, $P_1\sigma \dot{\sim} P_2\sigma$; the relation to exhibit for each σ is (here we assume for the sake of simplicity that $y \notin n(\sigma)$):

$$\mathcal{R} = \{(P_1\sigma, P_2\sigma)\} \cup \{(P\sigma\{w/y\}, P'\sigma\{w/y\}) |\, w \in fn(P'\sigma, P\sigma)\} \cup Id\,.$$

Actually, decompositions can be determined automatically for a large class of processes. In [HL92], an algorithm is presented to check symbolic bisimulation between two finite *standard* symbolic transition graphs. A symbolic graph is standard if the bound names of any transition does not occur free in its ancestor-nodes. Given two finite standard graphs G_1 and G_2, the algorithm calculates the *most general boolean expression* under which G_1 and G_2 are equivalent, i.e. it calculates a formula ϕ such that if $G_1 \simeq^\psi G_2$ then $\psi \models \phi$. Therefore, the equivalence problem for graphs is reduced to the implication problem for boolean formulae.

[1] $fn(\psi)$ need not to be considered because, as it will turn out (Lemma 3.6), $n(\psi) \subseteq fn(P)$.

By introducing minor modifications (that take into account the extra conditions due to bound output), Hennessy's and Lin's algorithm can be used to calculate the most general boolean expression of two π-calculus processes represented by finite standard symbolic transition graphs. If we consider, e.g., finite (i.e. *rec*-free) processes, it is sufficient to generate their symbolic graphs by considering one transition for each equivalence class of transitions and to use fresh names for input and bound output actions. We want also to point out that, as we shall see in Section 3, the implication problem between formulae is decidable for the considered language (BF).

3 Basic Properties of Substitutions

In this section, some basic facts about substitutions will be proved. The section is divided in two parts, where substitutions are related to properties of boolean formulae and of operational semantics, respectively. The most relevant results of this section are Lemma 3.2 and Lemma 3.10. The former enables us to verify the satisfaction relation $\models$ by only checking finitely many substitutions; an easy consequence of this (Corollary 3.3) is that $\models$ is decidable. The latter relates concrete and symbolic transitional semantics, and is crucial for proving both the consistency and the adequacy theorems.

3.1 Substitutions and boolean formulae

Lemma 3.1 states some elementary properties of substitutions and boolean formulae.

Lemma 3.1

1. *If t is a name, a boolean formula or a $\mathcal{P}$-term, $\tilde{w}$ is a vector of pairwise distinct names and $\tilde{w} \cap fn(t) = \emptyset$, then for any $\tilde{z}$: $t = t\{\tilde{w}/\tilde{z}\}\{\tilde{z}/\tilde{w}\}$.*
2. *If $Ev(\phi) = true$ and σ is injective over $n(\phi)$, then also $Ev(\phi\sigma) = true$.*
3. *Let σ be a substitution such that $y \notin n(\sigma)$. Then $\sigma \models \phi$ if and only if $\sigma \models R_y(\phi)$.*

Proof: By structural induction on t and ϕ. □

The next lemma is the core of the adequacy proof. Its essential meaning is that it enables us to verify the logical implication between any two formulae, $\phi \models \psi$, by testing a *finite* set S of substitutions. These are essentially the substitutions whose names range in $n(\phi, \psi)$ *plus* a reserve of fresh names F. The intuitive reason this works is that, given any substitution ρ over ϕ and ψ, the names of $range(\rho)$ can be injectively renamed to names in F, to obtain a new substitution in S, satisfying exactly the same formulae (with names in $n(\phi \wedge \psi)$) as ρ. The actual statement of the lemma is complicated by the presence of a distinct name y: in the completeness proof, it will represent the input or bound output formal parameter. In order to define injective renamings, set F has to be chosen large enough[2].

We adopt, in the sequel, the following notation: given a substitution σ, $\sigma_{|V}$ denotes the *restriction* of σ to V, i.e. the substitution defined as follows: for each $x \in \mathcal{N}$, if $x \in V$ then $x(\sigma_{|V}) = x\sigma$, $x(\sigma_{|V}) = x$ otherwise.

Lemma 3.2 *Let $y \in \mathcal{N}$, $V \subseteq_{fin} \mathcal{N} - \{y\}$ and $V_y = V \cup \{y\}$; moreover let $n(\phi, \psi) \subseteq V_y$, where $\phi, \psi \in BF$. Fix $F \subseteq_{fin} \mathcal{N} - V_y$ such that $|F| > 2 * |V_y|$ and consider the set of substitutions: $S = \{\sigma \,|\, n(\sigma) \subseteq V \cup F$ and $\sigma \models \phi\}$. We have $\phi \models \psi$ if:*

[2]We have used here a lower-bound of two times the number of all the other involved names; we could have considered a set F just as large as the involved names, at the cost of complicating the proof.

a. *$y \notin n(\phi)$ implies that $\sigma\{^z/_y\} \models \psi$, for each $\sigma \in S$ and for each $z \in V_y\sigma$, and*

b. *$y \in n(\phi)$ implies that $\phi \models \bigwedge_{x \in V}[x \neq y]$ and $\sigma \models \psi$, for each $\sigma \in S$.*

PROOF: Let ρ be any substitution such that $\rho \models \phi$. We have to show that $\rho \models \psi$. Let $V = \{x_1, \ldots, x_n\}$, with the x_i's pairwise distinct and $n \geq 0$. For each $i \in \{1, \ldots, n\}$, define $z_i = x_i\rho$. Thus

$$V\rho = \{z_1, \ldots, z_n\}$$

(where the z_i's, $1 \leq i \leq n$, are not necessarily pairwise distinct). Furthermore define $z_0 = y\rho$.

Fix now a set of names $\tilde{w} \subseteq F$, s.t. $|\tilde{w}| = |V\rho|$ and $\tilde{w} \cap V_y\rho = \emptyset$: such a $\tilde{w}$ does exist because F has been chosen big enough. Clearly it holds that $y \notin \tilde{w}$, since $y \notin F$. Let $\tilde{w} = \{w_1, \ldots, w_n\}$ (where the w_i's are not necessarily pairwise distinct). Define

$$\sigma_0 = \{w_1/z_1, \ldots, w_n/z_n\}$$

in such a way that $w_i = w_j$ iff $z_i = z_j$: this is possible because $|\tilde{w}| = |V\rho|$. Thus σ_0 is an injective substitution over $V_y\rho$. Therefore, since $Ev(\phi\rho) = true$, it also holds that (Lemma 3.1(2)):

$$Ev(\phi\rho\sigma_0) = true. \tag{1}$$

Now, it is easily checked that:

$$\forall x \in V_y : \ x\rho\sigma_0 = x(\rho\sigma_0)_{|V}\ \{^{z_0\sigma_0}/_y\}. \tag{2}$$

Letting $w_0 = z_0\sigma_0$ and $\sigma = (\rho\sigma_0)_{|V}$, we have now two cases:

- $z_0 \in V\rho$. Thus $\rho \not\models \bigwedge_{x \in V}[x \neq y]$. Recalling that $\rho \models \phi$, we have then that $\phi \not\models \bigwedge_{x \in V}[x \neq y]$, and this in turn implies $y \notin n(\phi)$ (condition a. of the hypotheses). Let us now check that $\sigma \in S$. Indeed, from $y \notin n(\phi)$, it follows $\phi\sigma = \phi\rho\sigma_0$, hence (from (1)): $Ev(\phi\sigma) = true$; furthermore we have $n(\sigma) = V \cup \tilde{w} \subseteq V \cup F$. Thus we are in case a. of the statement of the lemma. From $z_0 \in V\rho$, it follows $w_0 \in V\sigma$. Therefore, by hypothesis, it holds that $\sigma\{^{w_0}/_y\} \models \psi$. But, since $n(\psi) \subseteq V_y$, from (2) it follows that:

  $$Ev(\psi\rho\sigma_0) = Ev(\psi\sigma\{^{w_0}/_y\}) = true.$$

  Therefore, since $\sigma_0^{-1} = \{z_1/w_1, \ldots, z_n/w_n\}$ is injective over $V_y\rho\sigma_0 = V\rho\sigma_0 = \tilde{w}$, also it holds that (Lemma 3.1 (1) and (2)):

  $$Ev(\psi\rho) = Ev(\psi\rho\sigma_0\sigma_0^{-1}) = true.$$

  which is the thesis for this case.

- $z_0 \notin V\rho$. Hence $w_0 = z_0$ and, from (1) and (2), we have

  $$Ev(\phi\sigma\{^{z_0}/_y\}) = Ev(\phi\rho\sigma_0) = true.$$

  Recall now that $z_0 \notin \tilde{w}$, thus either $z_0 = y$ or $z_0 \notin V_y\sigma = \tilde{w} \cup \{y\}$. From these facts, relation above and Lemma 3.1 (1) and (2), it therefore follows:

  $$Ev(\phi\sigma) = Ev(\phi\sigma\{^{z_0}/_y\}\{^y/_{z_0}\}) = true.$$

  Therefore, since $n(\sigma) \subseteq V \cup F$, it holds that $\sigma \in S$. From the hypotheses it follows that $\sigma \models \psi$ and hence (Lemma 3.1(2)) also $\sigma\{^{z_0}/_y\} \models \psi$. But from (2) we get that $\psi\sigma\{^{z_0}/_y\} = \psi\rho\sigma_0$. Let σ_0^{-1} be defined as in the previous case and note that σ_0^{-1} is injective over $V_y\rho\sigma_0 = \tilde{w} \cup \{z_0\}$; therefore it also holds that (Lemma 3.1 (1) and (2)):

  $$Ev(\psi\rho) = Ev(\psi\rho\sigma_0\sigma_0^{-1}) = true$$

  which is the thesis for this case. □

As a corollary of the above lemma, we obtain the decidability of the relation $\models$.

Corollary 3.3 *The relation $\models \subseteq BF \times BF$ is decidable.*

PROOF: An easy application of the previous lemma. □

Next definition is useful to express the fact that a substitution is a *variant* of another one, i.e. that they are the same up to some injective renaming of names; $\rho \models \chi(\sigma, V)$ formalizes the fact that ρ is a variant of σ over the set V.

Definition 3.4 (Characteristic boolean formula) *Given $V \subseteq_{fin} \mathcal{N}$ and a substitution σ, we define the* characteristic boolean formula of σ over V *as:*

$$\chi(\sigma, V) = \bigwedge_{x,y \in V} m_{xy}$$

where $m_{xy} = [x = y]$ if $x\sigma = y\sigma$, $m_{xy} = [x \neq y]$ otherwise.

Lemma 3.5 below asserts that variant substitutions "behave the same" w.r.t. $\models$.

Lemma 3.5 *Let σ, ρ be substitutions, $V \subseteq_{fin} \mathcal{N}$ and suppose $\rho \models \chi(\sigma, V)$. If $\sigma \models \phi$ and $n(\phi) \subseteq V$ then $\rho \models \phi$.*

PROOF: Let $V = \{x_1, \ldots, x_n\}$, with the x_i's pairwise distinct, and $n \geq 0$ and let $\chi(\sigma, V) = \bigwedge_{x_i, x_j \in V} m_{x_i x_j}$. For each $i \in \{1, \ldots, n\}$ define $y_i = x_i\sigma$ and $z_i = x_i\rho$. We will show that, given any $i, j \in \{1, \ldots, n\}$, it holds that $z_i = z_j$ iff $y_i = y_j$. Indeed: $z_i = z_j$ iff $x_i\rho = x_j\rho$ iff (recalling that $\rho \models \chi(\sigma, V)$) $m_{x_i x_j} = [x_i = x_j]$ iff $x_i\sigma = x_j\sigma$ iff $y_i = y_j$.

Therefore the set $\sigma_0 = \{z_1/y_1, \ldots, z_n/y_n\}$ is a substitution, and furthermore:

1. for each $x \in V$, $x\rho = x\sigma\sigma_0$
2. σ_0 is injective over $V\sigma$.

From 1 it follows that $\phi\rho = \phi\sigma\sigma_0$, and since from 2 σ_0 is injective and $Ev(\phi\sigma) = true$, it follows from Lemma 3.1 that also $Ev(\phi\rho) = true$. □

3.2 Substitutions and operational semantics

The following elementary property of the two transitional semantics will prove useful in the sequel.

Lemma 3.6

1. *If $P \xrightarrow{\alpha} P'$ then: $fn(\alpha) \subseteq fn(P)$ and $fn(P') \subseteq fn(P) \cup bn(\alpha)$.*
2. *If $P \xmapsto{\phi, \alpha} P'$ then: $n(\phi) \subseteq fn(P)$, $fn(\alpha) \subseteq fn(P)$ and $fn(P') \subseteq fn(P) \cup bn(\alpha)$.*

PROOF: An easy transition induction on $P \xrightarrow{\alpha} P'$ and $P \xmapsto{\phi, \alpha} P'$. □

Next two lemmata will give us some freedom in renaming names in transitions, both standard and symbolic. Lemma 3.7 asserts that transitions are preserved by injective substitutions. Lemma 3.8 allows us to α-rename bound names with fresh names in transitions. Furthermore, in both cases, the renamed transitions have derivations just as complex as the original ones.

In what follows, we will use the following terminology (borrowed from [MPW92], part II). Sentences such as "if $P \xrightarrow{\alpha} P'$ then $Q \xrightarrow{\beta} Q'$ is derivable with the same depth as $P \xrightarrow{\alpha} P'$" will be abbreviated as "if $P \xrightarrow{\alpha} P'$ then *equally* $Q \xrightarrow{\beta} Q'$". A similar terminology will hold for symbolic transitions as well.

Lemma 3.7 *Let σ be an injective substitution over $fn(P)$ and suppose $bn(\alpha) \cap n(\sigma) = \emptyset$.*

1. *If $P \xrightarrow{\alpha} P'$ then equally $P\sigma \xrightarrow{\alpha\sigma} P'\sigma$.*
2. *If $P \overset{\phi,\alpha}{\longmapsto} P'$ then equally $P\sigma \overset{\phi\sigma,\alpha\sigma}{\longmapsto} P'\sigma$.*

PROOF: By transition induction. □

Lemma 3.8

1. *If $P \xrightarrow{x(y)} P'$ (resp. $P \xrightarrow{\overline{x}(y)} P'$) and $y_0 \notin fn(P)$, then equally $P \xrightarrow{x(y_0)} P'\{y_0/y\}$ (resp. $P \xrightarrow{\overline{x}(y_0)} P'\{y_0/y\}$) .*
2. *If $P \overset{\phi,x(y)}{\longmapsto} P'$ (resp. $P \overset{\phi,\overline{x}(y)}{\longmapsto} P'$) and $y_0 \notin fn(P)$, then equally $P \overset{\phi,x(y_0)}{\longmapsto} P'\{y_0/y\}$ (resp. $P \overset{\phi,\overline{x}(y_0)}{\longmapsto} P'\{y_0/y\}$) .*

PROOF: By transition induction. □

Remark 3.9 The assumption of identifying α-equivalent processes and the above two lemmata allow us to assume w.l.o.g. that *bound* names, both in processes and in transitions, are always fresh. In particular, we shall assume that given a bound name y and a substitution σ, y is fresh w.r.t. σ, i.e. $y \notin n(\sigma)$. A completely formalized argument would require, in each case, to rename y to a fresh y_0, by α-converting transitions (using Lemma 3.8) or processes. The latter could in turn imply using Lemma 3.7 to rename y throughout the part of a derivation in which y occurs free (e.g. when the last rule applied is a `Res` on y).

The main result of this part is Lemma 3.10, which relates symbolic to standard transitional semantics via substitutions. It will be crucial for the proof of both the consistency and the adequacy theorems of symbolic bisimulation.

Lemma 3.10 (Correspondence between symbolic and concrete SOS)

1. *If $P \overset{\phi,\alpha}{\longmapsto} P'$, with $bn(\alpha) \cap fn(P,\sigma) = \emptyset$ and $\sigma \models \phi$, then $P\sigma \xrightarrow{\alpha\sigma} P'\sigma$.*
2. *If $P\sigma \xrightarrow{\alpha} P'$, with $bn(\alpha) \cap fn(P,\sigma) = \emptyset$, then there exists a symbolic transition $P \overset{\phi,\beta}{\longmapsto} P''$, with $\sigma \models \phi$, $\beta\sigma = \alpha$ and $P''\sigma = P'$.*

PROOF: The proof of each of the two statements goes by transition induction.

1. We only deal with the cases when the last rule applied is $\mathtt{S-Res}$ or $\mathtt{S-Open}$. The other cases are easier or can be handled similarly.

- ($\mathtt{S-Res}$)

$$\mathtt{S-Res}\frac{P \overset{\phi,\alpha}{\longmapsto} P'}{(\nu\, y)P \overset{R_y(\phi),\alpha}{\longmapsto} (\nu\, y)P'}\ y \notin n(\alpha)\,.$$

By hypothesis we have $\sigma \models R_y(\phi)$ and $bn(\alpha) \cap fn((\nu\, y)P,\sigma) = \emptyset$. By Remark 3.9, we assume $y \notin n(\sigma)$. To apply the inductive hypothesis to the transition $P \overset{\phi,\alpha}{\longmapsto} P'$, we have to check that $bn(\alpha) \cap fn(P,\sigma) = \emptyset$ and that $\sigma \models \phi$. By hypothesis, $bn(\alpha) \cap fn((\nu\, y)P,\sigma) = \emptyset$; since $fn(P,\sigma) \subseteq fn((\nu\, y)P,\sigma,y)$ and $y \notin n(\alpha)$, it also holds that $bn(\alpha) \cap fn(P,\sigma) = \emptyset$. We now

check that $\sigma \models \phi$. Indeed, from $\sigma \models R_y(\phi)$, $y \notin n(\sigma)$ and Lemma 3.1 (3), it follows that $\sigma \models \phi$. Thus, the inductive hypothesis gives us:

$$P\sigma \xrightarrow{\alpha\sigma} P'\sigma\,.$$

Since $y \notin n(\alpha, \sigma)$, it also holds that $y \notin n(\alpha\sigma)$. We can therefore apply `Res` with y to the above transition:

$$(\nu\, y)P\sigma \xrightarrow{\alpha\sigma} (\nu\, y)P'\sigma.$$

Since $((\nu\, y)P)\sigma = (\nu\, y)P\sigma$ and $((\nu\, y)P')\sigma = (\nu\, y)P'\sigma$, we have the thesis.

- $(\mathtt{S-Open})$

$$\mathtt{S-Open}\frac{P \overset{\phi, \overline{x}y}{\longmapsto} P'}{(\nu\, y)P \overset{R_y(\phi), \overline{x}(y)}{\longmapsto} P'}\; y \neq x\,.$$

By hypothesis $\sigma \models R_y(\phi)$ and $y \notin n(\sigma)$. Thus, it holds that $((\nu\, y)P)\sigma = (\nu\, y)P\sigma$; furthermore, from Lemma 3.1 (3), we have that $\sigma \models \phi$. Since $bn(\overline{x}y) = \emptyset$, we can apply the inductive hypothesis to get:

$$P\sigma \xrightarrow{\overline{x\sigma}y} P'\sigma.$$

Since $x \neq y$ and $y \notin n(\sigma)$, it holds that $x\sigma \neq y$. We can therefore apply the `Open` rule with y to obtain:

$$(\nu\, y)P\sigma \xrightarrow{\overline{x\sigma}(y)} P'\sigma.$$

Since $((\nu\, y)P)\sigma = (\nu\, y)P\sigma$, we have the thesis for this case.

2. Again, the only non-trivial cases are those concerning the restriction operator. We show only the `Res` case, since the `Open` can be handled in a similar way.

- (`Res`) Consider the term $((\nu\, y)P)\sigma$; we assume by Remark 3.9 that $y \notin n(\sigma)$, so that $((\nu\, y)P)\sigma = (\nu\, y)P\sigma$. Therefore we have:

$$\mathtt{Res}\frac{P\sigma \xrightarrow{\alpha} P'}{(\nu\, y)P\sigma \xrightarrow{\alpha} (\nu\, y)P'}\; y \notin n(\alpha)$$

where, by hypothesis, $bn(\alpha) \cap fn((\nu\, y)P, \sigma) = \emptyset$. Since $fn(P, \sigma) \subseteq fn((\nu\, y)P, \sigma, y)$ and $y \notin n(\alpha)$, it holds that $bn(\alpha) \cap fn(P, \sigma) = \emptyset$. We can therefore apply the inductive hypothesis, obtaining that for some ϕ:

$$P \overset{\phi, \beta}{\longmapsto} P''$$

with $\sigma \models \phi$, $\beta\sigma = \alpha$ and $P''\sigma = P'$. Now, $y \notin n(\alpha, \sigma)$ implies $y \notin n(\beta)$. Therefore we can apply the rule $\mathtt{S-Res}$ to the above transition obtaining:

$$(\nu\, y)P \overset{R_y(\phi), \beta}{\longmapsto} (\nu\, y)P''\,.$$

Since $y \notin n(\sigma)$ and $\sigma \models \phi$, from Lemma 3.1 (3) we get $\sigma \models R_y(\phi)$; furthermore $((\nu\, y)P'')\sigma = (\nu\, y)P''\sigma = (\nu\, y)P'$; this is the thesis for present case.

□

We come now to two lemmata about bisimulation. The following one says that ground bisimulation $\dot{\sim}$ is closed under injective substitutions.

Lemma 3.11 *If $P \dot{\sim} Q$ and σ is injective over $fn(P,Q)$, then also $P\sigma \dot{\sim} Q\sigma$.*

PROOF: Routine, exploiting Lemma 3.7 (See also [MPW92]). □

Next lemma asserts that variant substitutions "behave the same" w.r.t. $\dot{\sim}$.

Lemma 3.12 *Let σ, ρ be substitutions, $V \subseteq_{fin} \mathcal{N}$ and suppose $\rho \models \chi(\sigma, V)$. If $P\sigma \dot{\sim} Q\sigma$ and $fn(P,Q) \subseteq V$ then $P\rho \dot{\sim} Q\rho$.*

PROOF: Let $V = \{x_1, \ldots, x_n\}$, with the x_i's pairwise distinct, and $n \geq 0$ and let $\chi(\sigma, V) = \bigwedge_{x_i, x_j \in V} m_{x_i x_j}$. For each $i \in \{1, \ldots, n\}$ define $y_i = x_i\sigma$ and $z_i = x_i\rho$. Proceeding exactly like in the proof of Lemma 3.5, we show that the set $\sigma_0 = \{z_1/y_1, \ldots, z_n/y_n\}$ is a substitution, and furthermore:

1. for each $x \in V$, $x\rho = x\sigma\sigma_0$,
2. σ_0 is injective over $V\sigma$.

From 1 it follows that $P\rho = P\sigma\sigma_0$ and $Q\rho = Q\sigma\sigma_0$. Furthermore, from $P\sigma \dot{\sim} Q\sigma$, from 2 above and from Lemma 3.11, we get that $P\sigma\sigma_0 \dot{\sim} Q\sigma\sigma_0$, from which our claim follows. □

4 Consistency and Adequacy of Symbolic Bisimulation

For stating both the consistency and the adequacy theorems, it is useful to fix the following definition:

Definition 4.1 (Closing $\dot{\sim}$ under ϕ) *For each $\phi \in BF$, let the relation $\sim^\phi$ be defined as follows:*

$$P \sim^\phi Q \text{ iff } \forall \sigma\colon \sigma \models \phi,\ P\sigma \dot{\sim} Q\sigma.$$

Theorem 4.2 (Consistency of symbolic bisimulation) *$P \simeq^\phi Q$ implies $P \sim^\phi Q$.*

PROOF: We will show that the relation:

$$\mathcal{R} = \{(P\sigma, Q\sigma) |\, \exists\, \phi.\, \sigma \models \phi \text{ and } P \simeq^\phi Q\}$$

is a ground bisimulation. Suppose $P \simeq^\phi Q$ and $\sigma \models \phi$. Let us see how the moves of $P\sigma$ are matched by $Q\sigma$. Suppose that $P\sigma \xrightarrow{\alpha} P'$, with $bn(\alpha) \cap fn(P\sigma, Q\sigma) = \emptyset$; by Remark 3.9, we assume that $bn(\alpha) \cap fn(P, Q, \phi, \sigma) = \emptyset$, as well. We distinguish the possible cases for α (α silent, input, free output or bound output action) and confine ourselves to the input and the bound output cases, since the others are easier.

- $\alpha = x(y)$. It suffices to show that for each $z \in fn(P\sigma, Q\sigma, y)$ there exists Q' such that $Q\sigma \xrightarrow{\alpha} Q'$ and $(P'\{z/y\}, Q'\{z/y\}) \in \mathcal{R}$. Thus fix $z \in fn(P\sigma, Q\sigma, y)$. From $P\sigma \xrightarrow{\alpha} P'$ and Lemma 3.10 we deduce that there is a symbolic transition:

  $$P \overset{\psi, w(y)}{\longmapsto} P''$$

  where $\sigma \models \psi$, $w\sigma = x$ and $P''\sigma = P'$. Since $P \simeq^\phi Q$, there exists a $\phi \wedge \psi$-decomposition $D = \{\phi_1, \ldots, \phi_n\}$ s.t. for each $i \in \{1, \ldots, n\}$ there exists a symbolic transition:

  $$Q \overset{\psi_i, w_i(y)}{\longmapsto} Q_i$$

where $\phi_i \models \psi_i \wedge [w = w_i]$ and $P'' \simeq^{\phi_i} Q_i$. Now, $\sigma \models \phi \wedge \psi$. But, since $n(\phi \wedge \psi) \subseteq fn(P, Q, \phi)$ (Lemma 3.6) and $y \notin fn(P, Q, \phi)$, we have that $(\phi \wedge \psi)\sigma\{z/y\} = (\phi \wedge \psi)\sigma$, hence $\sigma\{z/y\} \models (\phi \wedge \psi)$. Therefore, it holds that $\sigma\{z/y\} \models \phi_j$, for some $j \in \{1, \ldots, n\}$. Applying again Lemma 3.10 to transition $Q \overset{\psi_j, w_j(y)}{\longmapsto} Q_j$, we get the matching transition for $Q\sigma$:

$$Q\sigma \xrightarrow{x(y)} Q_j\sigma.$$

Now, define $Q' = Q_j\sigma$. We have $P'\{z/y\} = P''\sigma\{z/y\}$ and $Q'\{z/y\} = Q_j\sigma\{z/y\}$. Recall that $\sigma\{z/y\} \models \phi_j$ and $P'' \simeq^{\phi_j} Q_j$; thus, by definition, $(P'\{z/y\}, Q'\sigma\{z/y\}) \in \mathcal{R}$.

- $\alpha = \overline{x}(y)$. We will show that there exists Q' s.t. $P \xrightarrow{\alpha} Q'$ and $(P', Q') \in \mathcal{R}$. From $P\sigma \xrightarrow{\alpha} P'$ and Lemma 3.10, we deduce that there exists a symbolic transition:

$$P \overset{\psi, \overline{w}(y)}{\longmapsto} P_1$$

where $\sigma \models \psi$, $w\sigma = x$ and $P_1\sigma = P'$. Since $P \simeq^{\phi} Q$, there exists a $\phi \wedge \psi \wedge \bigwedge_{z \in fn(P,Q,\phi)} [y \neq z]$ - decomposition $D = \{\phi_1, \ldots, \phi_n\}$ s.t. for each $i \in \{1, \ldots, n\}$ there exists a symbolic transition:

$$Q \overset{\psi_i, \overline{w_i}(y)}{\longmapsto} Q_i$$

with $\phi_i \models \psi_i \wedge [w = w_i]$ and $P_1 \simeq^{\phi_i} Q_i$. Now, since $y \notin fn(P, Q, \phi, \sigma)$, it holds that $\sigma \models \bigwedge_{z \in fn(P,Q,\phi)} [y \neq z]$; hence, since $\sigma \models \phi \wedge \psi$, also $\sigma \models \phi \wedge \psi \wedge \bigwedge_{z \in fn(P,Q,\phi)} [y \neq z]$. Therefore there exists $\phi_j \in D$ s.t. $\sigma \models \phi_j$. Applying Lemma 3.10 to the transition $Q \overset{\psi_j, \overline{w_j}(y)}{\longmapsto} Q_j$, we get the matching transition for $Q\sigma$:

$$Q\sigma \xrightarrow{\overline{x}(y)} Q_j\sigma.$$

Now, define $Q' = Q_j\sigma$; recall that $P' = P_1\sigma$, that $P_1 \simeq^{\psi_j} Q_j$ and that $\sigma \models \psi_j$; thus by definition $(P', Q') \in \mathcal{R}$ and we are done.

□

We now come to prove that symbolic bisimulation is adequate for expressing concrete bisimulations, i.e. to prove that whenever $P \sim^{\phi} Q$ then $P \simeq^{\phi} Q$. To prove this adequacy theorem, we shall rely on the fact that only a suitable finite set of name-substitutions is "relevant" when working with a fixed collection of processes and formulae. All other substitutions are, in fact, variants of the considered ones, i.e. they can be obtained by injective renaming. This is a distinctive property of the π-calculus, since it relies on the absence of functions and predicates on names, apart from boolean combination of equalities.

An informal account of the proof might be the following. We show that $\{\sim^{\phi} \mid \phi \in BF\}$ is a family of symbolic bisimulations. Thus, given P, Q with $P \sim^{\phi} Q$ and $P \overset{\psi, \alpha}{\longmapsto} P'$, we have to find a decomposition of $\phi \wedge \psi$ (here for the sake of simplicity we assume that α is not a bound output), such that each subcase entails a matching symbolic transition for Q. Letting $V = fn(P, Q, \phi)$, the idea is to determine a *finite* set of substitutions, $S = \{\sigma_1, \ldots, \sigma_k\}$, such that each substitution satisfying $\phi \wedge \psi$ is a variant over $V \cup bn(\alpha)$ of some σ_i. By relying on Lemma 3.2, $\phi \wedge \psi$ is decomposed into a set $D = \{\phi_1, \ldots, \phi_k\}$, where we have one subcase for each $\sigma_i \in S$. Each $\phi' \in D$ is chosen to entail a symbolic matching transition for Q.

Theorem 4.3 (Adequacy of symbolic bisimulation) *$P \sim^{\phi} Q$ implies $P \simeq^{\phi} Q$.*

PROOF: We will show that the the family of relations

$$\mathcal{F} = \{\sim^{\phi} \mid \phi \in BF\}$$

is a family of symbolic (early) bisimulations. Thus let $\phi \in BF$ and $P \sim^{\phi} Q$ and suppose $P \stackrel{\psi,\alpha}{\longmapsto} P'$ with $bn(\alpha) \cap fn(P,Q,\phi) = \emptyset$. If $\alpha = \overline{x}(y)$, define $\chi = \bigwedge_{z \in fn(P,Q,\phi)}[y \neq z]$, otherwise define $\chi = true$. We have to find a $\phi \wedge \psi \wedge \chi$-decomposition $D = \{\phi_1, \ldots, \phi_n\}$ s.t. for each $i \in \{1, \ldots, n\}$ there exists a symbolic transition $Q \stackrel{\psi_i,\beta_i}{\longmapsto} Q_i$ with $\phi_i \models \psi_i \wedge [\alpha = \beta_i]$ and furthermore $P' \sim^{\phi_i} Q_i$.

Let $V = fn(P,Q,\phi)$ and $V_y = V \cup bn(\alpha)$. Take $F \subseteq_{fin} \mathcal{N} - V_y$ such that $|F| > 2 * |V_y|$. Define the set of substitutions:

$$S = \{\sigma |\, n(\sigma) \subseteq F \cup V \text{ and } \sigma \models \phi \wedge \psi \wedge \chi\}\,.$$

S is clearly finite. We now distinguish the possible alternatives for α (α equal to τ, $x(y)$, $\overline{x}y$ or $\overline{x}(y)$) and stick ourselves to $\alpha = x(y)$ and $\alpha = \overline{x}(y)$, since the others are easier.

- $\alpha = x(y)$. We let $D = \{\chi(\sigma\{z/y\}, V_y) |\, \sigma \in S,\, z \in V_y\sigma\}$. The fact that this is a $\phi \wedge \psi \wedge \chi$-decomposition follows from Lemma 3.2, case a. (read the "ϕ" of the statement of the lemma as $\phi \wedge \psi \wedge \chi$ and the "ψ" as $\bigvee D$). We now come to show that each $\chi(\sigma\{z/y\}, V_y) \in D$ entails a matching symbolic transition for Q. Thus fix $\sigma \in S$ and $z \in V_y\sigma$ and consider the following chain of implications:

  $$\begin{array}{rl}
  P \stackrel{\psi,\alpha}{\longmapsto} P' & \text{implies, in virtue of Lemma 3.10 and } \sigma \models \psi \text{ and } y \notin n(\sigma), \text{ that} \\
  P\sigma \stackrel{\gamma}{\longrightarrow} P'\sigma & \text{with } \alpha\sigma = \gamma; \text{ since } P \sim^{\phi} Q \text{ and } \sigma \models \phi, \text{ for some } Q' \text{ it holds that} \\
  Q\sigma \stackrel{\gamma}{\longrightarrow} Q' & \text{with } P'\sigma\{z/y\} \dot{\sim} Q'\{z/y\}. \text{ By Lemma 3.10, there exists a symbolic transition} \\
  Q \stackrel{\psi',\beta}{\longmapsto} Q'' & \text{with } \sigma \models \psi' \wedge [\beta = \alpha] \text{ and } Q''\sigma = Q'\,.
  \end{array}$$

  We check now that that $\chi(\sigma\{z/y\}, V_y) \models \psi' \wedge [\beta = \alpha]$ and that $P' \sim^{\chi(\sigma\{z/y\}, V_y)} Q''$.

  We prove the former fact. It holds that $n(\psi' \wedge [\beta = \alpha]) \subseteq fn(P,Q) \subseteq V$ (Lemma 3.6) and $y \notin n(\sigma)$; furthermore $\sigma \models \psi' \wedge [\beta = \alpha]$; these facts imply $\sigma\{z/y\} \models \psi' \wedge [\beta = \alpha]$; hence, from Lemma 3.5, given any ρ with $\rho \models \chi(\sigma\{z/y\}, V_y)$, we have $\rho \models \psi' \wedge [\beta = \alpha]$.

  We show now that $P' \sim^{\chi(\sigma\{z/y\}, V_y)} Q''$. Consider any substitution ρ s.t. $\rho \models \chi(\sigma\{z/y\}, V_y)$; we have to show that $P'\rho \dot{\sim} Q''\rho$. From above, we know that $P'\sigma\{z/y\} \dot{\sim} Q''\sigma\{z/y\}$. But $fn(P', Q'') \subseteq V_y$ (Lemma 3.6), thus from Lemma 3.12, $P'\rho \dot{\sim} Q''\rho$.

- $\alpha = \overline{x}(y)$. Define $D = \{\chi(\sigma, V_y) |\, \sigma \in S\}$. The fact that this D is a $\phi \wedge \psi \wedge \chi$-decomposition follows from Lemma 3.2, case b. (read the "ϕ" of the statement of the lemma as $\phi \wedge \psi \wedge \chi$ and the "ψ" as $\bigvee D$). We now come to show that each $\chi(\sigma, V) \in D$ entails a matching symbolic transition for Q. Thus fix $\sigma \in S$ and consider the following chain of implications:

  $$\begin{array}{rl}
  P \stackrel{\psi,\alpha}{\longmapsto} P' & \text{implies, in virtue of Lemma 3.10 and } \sigma \models \psi, \text{ that} \\
  P\sigma \stackrel{\gamma}{\longrightarrow} P'\sigma & \text{with } \alpha\sigma = \gamma; \text{ from } P \sim^{\phi} Q \text{ and } \sigma \models \phi, \text{ for some } Q' \text{ it holds that} \\
  Q\sigma \stackrel{\gamma}{\longrightarrow} Q' & \text{with } P'\sigma \dot{\sim} Q'. \text{ From Lemma 3.10, there exists a symbolic transition} \\
  Q \stackrel{\psi',\beta}{\longmapsto} Q'' & \text{with } \sigma \models \psi' \wedge [\beta = \alpha] \text{ and } Q''\sigma = Q'.
  \end{array}$$

  We now show that $\chi(\sigma, V_y) \models \psi' \wedge [\beta = \alpha]$ and that $P' \sim^{\chi(\sigma, V_y)} Q''$. Consider any ρ s.t. $\rho \models \chi(\sigma, V_y)$; since $\sigma \models \psi' \wedge [\beta = \alpha]$ and $n(\psi' \wedge [\beta = \alpha]) \subseteq V$ (Lemma 3.6), from Lemma 3.5 we have $\rho \models \psi' \wedge [\beta = \alpha]$ as well.

  We show now that $P' \sim^{\chi(\sigma, V_y)} Q''$. Consider any substitution ρ s.t. $\rho \models \chi(\sigma, V_y)$; we have to show that $P'\rho \dot{\sim} Q''\rho$. From above, we know that $P'\sigma \dot{\sim} Q''\sigma$. But $fn(P', Q'') \subseteq V_y$ (Lemma 3.6) and by hypothesis $\rho \models \chi(\sigma, V_y)$, thus from Lemma 3.12, $P'\rho \dot{\sim} Q''\rho$.

□

We end the section by showing that also ground bisimulation $\dot{\sim}$ can be characterized in terms of the symbolic one.

Theorem 4.4 *$P \dot{\sim} Q$ iff $P \simeq^{\phi} Q$, where $\phi = \bigwedge_{x,y \in fn(P,Q),\, x,y \text{ distinct}} [x \neq y]$.*

PROOF: The theorem follows immediately from the consistency and adequacy theorems for symbolic bisimulation and from the fact that $\dot{\sim}$ is closed under injective substitutions (Lemma 3.11). □

5 The Proof System

Let us now consider the *finite* fragment of the calculus, i.e. the calculus without the $recX.$ operator and discuss an equational axiomatization of symbolic bisimulation over it. It is well known that decidable axiomatizations cannot exist for the full language.

The statements derivable within the proof system are guarded equations of the form $\phi \rhd P = Q$, to be read as "under ϕ, P equals Q". In the sequel, we will write simply $\phi \rhd P = Q$ to mean that the equation $\phi \rhd P = Q$ is derivable within the proof system. Furthermore, we will abbreviate $true \rhd P = Q$ simply as $P = Q$. In this section, symbol $\equiv$ will be used for identity (up to α-conversion), in order to distinguish it from the proof-theoretic equality ($=$).

The inference rules of the system are presented in Table 2 (the standard inference rules for reflexivity, symmetry and transitivity have been omitted), while the axioms are in Table 3. Note that the new relevant axioms are *Subst*, *Res*2 and *Exp*, while the laws for Summation and Restriction are the usual ones, that we have borrowed from [MPW92].

Our proof system can be viewed as the result of merging that of [HL93] and [PS93]. More precisely, all of the inference rules, but the *Res* rule, are taken from [HL93], while the axioms are taken from [PS93]. In particular, our *Res*2 rule corresponds to rule RC5 of [PS93], once we interpret our $R_y(\,.\,)$ as their $Remove_y(\,.\,)$.

The *Cut* rule permits case analysis on ϕ: it says that if ϕ can be split into two subcases ϕ_1 and ϕ_2, and for each subcase we can prove $P = Q$, then $P = Q$ is derivable under ϕ. The *Res* and *Res*2 rules exhibit the same kind of logical "hiding" of the bound name y as the rules $\mathtt{S-Res}$ and $\mathtt{S-Open}$ of symbolic transitional semantics. The other rules and axioms should be self-explanatory; anyway, we refer the reader to [PS93, HL93] for explanations on their intuitive meaning.

We give below some derived inference rules and axioms that will be useful to manipulate terms.

Lemma 5.1 (Derived laws)

1. *(Guard2) $\phi \models \psi$ implies $\phi \rhd \psi P = P$.*
2. *(Guard3) $\phi \models \neg\psi$ implies $\phi \rhd \psi P = \mathbf{0}$.*
3. *(Guard4) $\phi(\psi P) = (\phi \wedge \psi)P$.*
4. *(Guard5) $\phi \models \psi$ and $\psi \models \phi$ imply $\phi P = \psi P$.*
5. *(Guard6) $\phi \models false$ implies $\phi P = \mathbf{0}$.*
6. *(Subst2) $\phi \models [x = y]$ implies $\phi \rhd \alpha.P = \alpha\{y/x\}.P$.*
7. *(Cons) $\psi \models \phi$ and $\phi \rhd P = Q$ imply $\psi \rhd P = Q$.*
8. *(Sum1) $\phi(P + Q) = \phi P + \phi Q$.*

$(Congr)$	$\dfrac{\phi \rhd P = Q}{\phi \rhd P' = Q'}$	where $P' = Q'$ stands for either of: $\tau.P = \tau.Q$, $\overline{x}y.P = \overline{x}y.Q$, $\psi P = \psi Q$, $P + R = Q + R$, $P \mid R = Q \mid R$.
(Res)	$\dfrac{\phi \rhd P = Q}{R_y(\phi) \rhd (\nu\, y)P = (\nu\, y)Q}$	
(Inp)	$\dfrac{\phi \rhd \sum_{i\in I} \tau.P_i = \sum_{i\in I} \tau.Q_i}{\phi \rhd \sum_{i\in I} x(y).P_i = \sum_{i\in I} x(y).Q_i}$	$y \notin n(\phi)$
$(Guard)$	$\dfrac{\phi \wedge \psi \rhd P = Q,\ \phi \wedge \neg\psi \rhd Q = \mathbf{0}}{\phi \rhd \psi P = Q}$	
$(False)$	$\dfrac{-}{false \rhd P = Q}$	
(Cut)	$\dfrac{\phi_1 \rhd P = Q,\ \phi_2 \rhd P = Q}{\phi \rhd P = Q}$	$\phi \models \phi_1 \vee \phi_2$
$(Axiom)$	$\dfrac{-}{true \rhd P = Q}$	for each axiom $P = Q$

Table 2: Inference Rules of the Proof System

Summation Laws

$(S0)$ $P + \mathbf{0} = P$

$(S1)$ $P + P = P$

$(S2)$ $P + Q = Q + P$

$(S3)$ $P + (Q + R) = (P + Q) + R$

Restriction Laws

$(R0)$ $(\nu\, x)P = P$, if $x \notin fn(P)$

$(R1)$ $(\nu\, x)(\nu\, y)P = (\nu\, y)(\nu\, x)P$

$(R2)$ $(\nu\, x)(P + Q) = (\nu\, x)P + (\nu\, x)Q$

$(R3)$ $(\nu\, x)\alpha.P = \alpha.(\nu\, x)P$, if $x \notin n(\alpha)$

$(R4)$ $(\nu\, x)\alpha.P = \mathbf{0}$, if $x = subj(\alpha)$

Axioms from [MPW92]

$(Subst)$ $[x = y]\alpha.P = [x = y]\alpha\{^{x}/_{y}\}.P$

$(Res2)$ $(\nu\, y)(\phi P) = (R_y(\phi))(\nu\, y)P$

(Exp)
If $P \equiv \sum_{i\in I} \phi_i\alpha_i.P_i$ and $Q \equiv \sum_{j\in J} \psi_i\beta_i.Q_i$ and
no α_i (resp. β_j) binds a name free in Q (resp. P).

$$P \mid Q = \sum_{i\in I} \phi_i\alpha_i.(P_i \mid Q) + \sum_{j\in J} \psi_j\beta_j.(P \mid Q_j) + \sum_{\alpha_i\, opp\, \beta_j}(\phi_i \wedge \psi_j \wedge [x_i = y_j])\tau.R_{ij}$$

where $\alpha_i\, opp\, \beta_j$ and R_{ij} are defined by:
1. $\alpha_i = \overline{x_i}z$ and $\beta_j = y_j(y)$; then $R_{ij} = P_i \mid Q_j\{^{z}/_{y}\}$
2. $\alpha_i = \overline{x_i}(y)$ and $\beta_j = y_j(y)$; then $R_{ij} = (\nu\, y)(P_i \mid Q_j)$
3. The converse of 1.
4. The converse of 2.

New Axioms

Table 3: Axioms of the Proof System

9. (Sum2) $\phi P + \psi P = (\phi \vee \psi)P$.

PROOF: Routine. As an example, we prove $Guard2$. Since $\phi \wedge \neg\psi \models false \vee false$ and, by $False$, $false \rhd P = \mathbf{0}$, applying the Cut rule we obtain: (a) $\phi \wedge \neg\psi \rhd P = \mathbf{0}$. On the other hand, by $Axiom$, it holds that $true \rhd P = P$ and hence, by Cut: (b) $\phi \wedge \psi \rhd P = P$. By (a) and (b), applying $Guard$ we get $\phi \rhd \psi P = P$, which is the wanted statement. The other cases are proven by similar techniques (see also [HL93], Proposition 2.2). □

Before proving its soundness and completeness, we give an elementary application of the proof system.

Example 5.2 We show that $x(y).P = x(y).([y = z]P + [y \neq z]P)$:

$$\begin{array}{rcll} P & = & true\, P & Guard2 \\ & = & ([y = z] \vee [y \neq z])P & Guard5 \\ & = & [y = z]P + [y \neq z]P & Sum2. \end{array}$$

From the latter equation, applying $Congr - \tau$, we get $\tau.P = \tau.([y = z]P + [y \neq z]P)$; finally applying Inp the wanted result follows.

Soundness is straightforward to prove by exploiting consistency and adequacy of symbolic bisimulation.

Theorem 5.3 (Soundness of the proof system) $\phi \rhd P = Q$ *implies* $P \simeq^{\phi} Q$.

PROOF: Relying on the definition of $\sim^{\phi}$, check soundness of the inference rules. All the cases are straightforward. As an example we consider Res. Suppose $P \sim^{\phi} Q$; we have to show that $(\nu\, y)P \sim^{R_y(\phi)} (\nu\, y)Q$ as well. Let $\sigma \models R_y(\phi)$; we show that $((\nu\, y)P)\sigma \dot{\sim} ((\nu\, y)Q)\sigma$.

By Remark 3.9, we assume that $y \notin n(\sigma)$; from Lemma 3.6 we get that $\sigma \models \phi$ as well. From $P \sim^{\phi} Q$ it follows that $P\sigma \dot{\sim} Q\sigma$; hence, from the fact that restriction preserves $\dot{\sim}$, we have:

$$((\nu\, y)P)\sigma = (\nu\, y)(P\sigma) \dot{\sim} (\nu\, y)(Q\sigma) = ((\nu\, y)P)\sigma$$

that is the wanted claim. □

The actual proof of completeness relies on a "customized" notion of head normal form.

Definition 5.4 (Head normal forms) *A process P is in* head normal form (HNF) *if it is of the form $\sum_{i \in I} \phi_i S_i$, where:*

- $\{\phi_i | \, i \in I\}$ *is a true-decomposition such that* $\phi_i \wedge \phi_j \models false$ *for each* $i, j \in I$ *with* $i \neq j$;
- *each* S_i, $i \in I$, *is of the form* $\sum_{j \in J_i} \alpha_j.P_j$.

Lemma 5.5 *For each process P, there exists a HNF H s.t. $P = H$.*

PROOF: The proof is by structural induction on P. The most interesting case is when $P = P_1 + P_2$. We check this case only. By the inductive hypothesis, there exist HNF's $H_1 \equiv \sum_{i \in I} \phi_i S_i$ and $H_2 \equiv \sum_{j \in J} \psi_j T_i$ that are provably equivalent to, respectively, P_1 and P_2; hence $P = H_1 + H_2$. Now, for each $I' \subseteq I$ and $J' \subseteq J$, define the formula

$$\chi_{I'J'} = \bigwedge_{i \in I', j \in J'} (\phi_i \wedge \psi_j) \wedge \bigwedge_{i \in I - I', j \in J - J'} (\neg\phi_i \wedge \neg\psi_j)\,.$$

It is easy to see that the set

$$D = \{\chi_{I'J'} | \, I' \subseteq I,\, J' \subseteq J\}$$

is a *true*-decomposition and furthermore that for any two distinct $\chi_1, \chi_2 \in D$, $\chi_1 \wedge \chi_2 = false$. Now we have

$$\begin{array}{lll} H_1 + H_2 & = true\,(H_1 + H_2) & Guard2 \\ & = (\bigvee D)(H_1 + H_2) & Guard5 \\ & = \sum_{\chi_{I'J'} \in D} (\chi_{I'J'}(H_1 + H_2)) & Sum2\ . \end{array} \tag{3}$$

Now observe that, from *Guard*4, *Guard*5 and *Guard*6, it follows that $\chi_{I'J'}\phi_i S_i = \chi_{I'J'} S_i$ if $i \in I'$, $\chi_{I'J'}\phi_i S_i = \mathbf{0}$ otherwise. From this, it follows that

$$\begin{array}{lll} \chi_{I'J'} H_1 & = \sum_{i \in I} \chi_{I'J'}\phi_i S_i & Sum1 \\ & = \sum_{i \in I'} \chi_{I'J'} S_i & \text{from the above fact and Summation laws} \\ & = \chi_{I'J'}(\sum_{i \in I'} S_i) & Sum2\ . \end{array}$$

In a similar manner, one shows that $\chi_{I'J'} H_2 = \chi_{I'J'}(\sum_{j \in J'} T_j)$. Hence, applying $Sum1$, we have:

$$\begin{array}{ll} \chi_{I'J'}(H_1 + H_2) & = \chi_{I'J'}(\sum_{i \in I'} S_i) + \chi_{I'J'}(\sum_{j \in J'} T_j) \\ & = \chi_{I'J'} U_{I'J'} \end{array}$$

where $U_{I'J'}$ is defined as $\sum_{i \in I'} S_i + \sum_{j \in J'} T_j$. From (3) it follows $H_1 + H_2 = \sum_{\chi_{I'J'} \in D} \chi_{I'J'} U_{I'J'}$; the latter process is a HNF. □

We need some other notions about substitutions. The next definition introduces the concepts of completeness for formulae and of equivalence relation induced by a formula. The former can be explained by saying that if ϕ is complete over V then under ϕ names in V can, in a sense, be treated as *constants*. The definition of the latter is self-explanatory.

Definition 5.6 *Let $V \subseteq_{fin} \mathcal{N}$ and $\phi \in BF$.*

1. *We say that ϕ is* complete over V *if for each $x, y \in V$, either $\phi \models [x = y]$ or $\phi \models [x \neq y]$.*
2. *We define $\mathcal{R}(\phi, V)$, the* equivalence relation induced by ϕ over V, *as: for each $x, y \in V$ $x\mathcal{R}(\phi, V)y$ iff $\phi \models [x = y]$.*

In the sequel we adopt the following notation: given a term P of the form $\sum_{i \in I} \alpha_i.P_i$ and a set $V \subseteq_{fin} \mathcal{N}$, P_V is $\sum_{i | i \in I,\, subj(\alpha_i) \in V} \alpha_i.P_i$.

Lemma 5.7 *Suppose that $P \equiv \sum_{i \in I} \alpha_i.P_i \simeq^{\phi} \sum_{j \in J} \beta_j.Q_j \equiv Q$ and that ϕ is complete over $V = \{subj(\alpha_i) \,|\, i \in I\} \cup \{subj(\beta_j) \,|\, j \in J\}$. Then, for any equivalence class C of $\mathcal{R}(\phi, V)$, we have $P_C \simeq^{\phi} Q_C$.*

Proof: As a direct consequence of the fact that ϕ is complete over V, we have that for each $x, y \in V$:

$$\text{not } x\mathcal{R}(\phi, V)y \text{ implies } \phi \models [x \neq y]. \tag{4}$$

Fix an equivalence class C of $\mathcal{R}(\phi, V)$ and any σ s.t. $\sigma \models \phi$; it is sufficient to show that $P_C\sigma \dot{\sim} Q_C\sigma$. Indeed, since $P\sigma \dot{\sim} Q\sigma$, for any summand $(\alpha_i.P_i)\sigma$ of $P_C\sigma$, there must exist a "matching" summand $(\beta_j.Q_j)\sigma$ of $Q\sigma$; but it must be that $subj(\beta_j) \in C$, because of (4), i.e $(\beta_j.Q_j)$ is a summand of Q_C. By symmetry we conclude that $P_C\sigma \dot{\sim} Q_C\sigma$. □

We now define a measure that will be used as induction parameter in the proof of the theorem.

Definition 5.8 (ϕ-depth) *Given P and ϕ, we define the* ϕ-depth of P *as:*

$$\begin{array}{ll} depth(\phi, P) = max\{k | & \text{for some } \sigma \models \phi,\ n \geq 0 \text{ and } \alpha_1, \cdots, \alpha_n \text{ different from } \tau, \\ & (P | \alpha_1.\cdots.\alpha_n.\mathbf{0})\sigma = R_0 \xrightarrow{\tau} R_1 \xrightarrow{\tau} \cdots \xrightarrow{\tau} R_k\ \}. \end{array}$$

This parameter has been defined so as to satisfy the three properties stated in the following lemma:

Lemma 5.9

1. *$depth(\phi, P) < depth(\phi, \alpha.P)$, for any action prefix α;*
2. *$P \sim^{\phi} Q$ implies $depth(\phi, P) = depth(\phi, Q)$;*
3. *$\phi \models \psi$ implies $depth(\phi, P) \leq depth(\psi, P)$.*

PROOF: Straightforward from the definition. □

We are now set for proving the main theorem of the section, i.e. that $P \simeq^{\phi} Q$ implies $\phi \rhd P = Q$. Our proof combines two major technical ideas: splitting the global condition ϕ into a set of complete sub-conditions and applying a symbolic variant of the classical proof by Milner [Mil89]. The latter idea comes entirely from [HL93]. The first one is present in [PS93]; there, head normal forms similar to ours are introduced, but it is required that the outermost formulae of summands be complete over the set of free names of the given processes. Here, the *Cut* rule permits using this concept at the proof system level, making it more explicit.

We give now an informal account of our proof, which goes by induction on $depth(\phi, P)$. We can assume that both P and Q are in HNF, in virtue of Lemma 5.5. The condition ϕ can be split into a decomposition D such that for each subcase $\psi \in D$:

1. $\psi \rhd P = \sum_{i \in I} \alpha_i.P_i$ and $\psi \rhd Q = \sum_{j \in J} \beta_j.Q_j$, that is under ϕ, P and Q are equal to some head normal form in the sense of [Mil89];
2. ψ is complete over the set V of names occurring in subject position in the α_i's and β_j's.

By Lemma 5.7, we have that for each equivalence class C of $\mathcal{R}(\phi, V)$, $(\sum_{i \in I} \alpha_i.P_i)_C \simeq^{\psi} (\sum_{j \in J} \beta_j.Q_j)_C$. Exploiting the symbolic transitional semantics and the inductive hypothesis, one easily shows that the latter two terms are provably equivalent under ψ; the proof is a symbolic version of the classical one of [Mil89]. From this it easily follows that also P and Q are provably equivalent under ψ. Since the latter holds for each $\psi \in D$, we can conclude that $\phi \rhd P = Q$ by applying the *Cut* rule.

Theorem 5.10 (Completeness of the proof system) *$P \simeq^{\phi} Q$ implies $\phi \rhd P = Q$.*

PROOF: The proof goes by induction on $k = depth(\phi, P) = depth(\phi, Q)$. The base case ($k = 0$) is easy and thus we omit it. Suppose $k > 0$; thus we can assume that the thesis holds for each P', Q' and ϕ' s.t. $P' \simeq^{\phi'} Q'$ and $depth(\phi', P') < k$.

In virtue of Lemma 5.5, of the correctness of the proof system and of Lemma 5.9(2), we can suppose w.l.o.g. that both P and Q are in HNF.

Therefore it holds that:

$$P \equiv \sum_{i \in K} \phi_i R_i \quad \text{and} \quad Q \equiv \sum_{j \in H} \psi_j S_j.$$

Since $\phi \models \bigvee_{i,j} \phi \wedge \phi_i \wedge \psi_j$ (recall that by definition of HNF both $\{\phi_i | \, i \in K\}$ and $\{\psi_j | \, j \in H\}$ are *true*-decompositions), it will suffice to show that for each $i \in K$ and $j \in H$

$$\phi \wedge \phi_i \wedge \psi_j \rhd P = Q \tag{5}$$

and then to apply the *Cut* rule. Thus fix $i \in K, j \in H$ and let $\chi = \phi \wedge \phi_i \wedge \psi_j$. Recall that (by definition of HNF) for $i' \neq i$ it holds that $\phi_i \models \neg\phi_{i'}$, and, for $j' \neq j$, $\psi_j \models \neg\psi_{j'}$; thus applying repeatedly *Guard*2, *Guard*3 and the Summation Axioms, we can write:

$$\chi \rhd P = R_i \quad \text{and} \quad \chi \rhd Q = S_j. \tag{6}$$

Thus it will suffice to show that:

$$\chi \rhd R_i = S_j. \tag{7}$$

By definition of HNF, for suitable non-negative integers M, N it holds that:

$$R_i \equiv \sum_{l=1}^{M} \alpha_l.P_l \;\text{ and }\; S_j \equiv \sum_{l=1}^{N} \beta_l.Q_l.$$

Define $\Gamma = \{\ \tau$, input, free-output, bound-output $\}$ and for each $\gamma \in \Gamma$

$$R_\gamma = \sum\{\alpha_l.P_l |\, \alpha_l \text{ is of kind } \gamma\} \text{ and } S_\gamma = \sum\{\beta_l.Q_l |\, \beta_l \text{ is of kind } \gamma\}.$$

Thus

$$R_i = \sum_{\gamma\in\Gamma} R_\gamma \;\text{ and }\; S_j = \sum_{\gamma\in\Gamma} S_\gamma.$$

To show (7), it will suffice to show that, for each $\gamma \in \Gamma$

$$\chi \rhd R_\gamma = S_\gamma. \tag{8}$$

Now, we can write, for suitable I and J:

$$R_\gamma \equiv \sum_{i\in I} \alpha_i.P_i \;\text{ and } S_\gamma \equiv \sum_{j\in J} \beta_j.Q_j.$$

Define $V = \{subj(\alpha_i) \,|\, i \in I\} \cup \{subj(\beta_j) \,|\, j \in J\}$ and assume $V = \{x_1, \ldots, x_k\}$, with the x_i's distinct. For each $x \in V$ and $V' \subseteq V$, define the formula:

$$\eta(x, V') = \bigwedge_{y\in V'} [x = y] \wedge \bigwedge_{y\in V-V'} [x \neq y].$$

Now, given k sets $V_1, \ldots, V_k \subseteq V$, define

$$\xi(V_1, \ldots, V_k) = \bigwedge_{x_i\in V} \eta(x_i, V_i)\,.$$

and let

$$\Xi = \{\xi(V_1, \ldots, V_k) |\, \text{with } V_1, \ldots, V_k \subseteq V\ \}.$$

It is easy to see that each $\xi \in \Xi$ is complete over V and that Ξ is a *true*-decomposition; the latter implies

$$\chi \models \bigvee\{\xi \wedge \chi |\, \xi \in \Xi\}\,. \tag{9}$$

Thus, to show (8), it will suffice to show that for each $\xi \in \Xi$:

$$\xi \wedge \chi \rhd R_\gamma = S_\gamma \tag{10}$$

and then to apply the *Cut* rule.

We will now show (10). Consider the equivalence relation $\mathcal{R}(\xi \wedge \chi, V)$: it will consist of t non-empty equivalence classes $C_1, \ldots, C_t$, thus, abbreviating $R_{\gamma C}$ as R_C, we can write:

$$R_\gamma = R_{C_1} + \cdots + R_{C_t} \;\text{ and }\; S_\gamma = S_{C_1} + \cdots + S_{C_t}.$$

We will now prove that for each equivalence class C:

$$\xi \wedge \chi \rhd R_C = S_C \tag{11}$$

and this will establish (10) and hence the theorem. Fix C. Note that, since ξ is complete over V, so is $\xi \wedge \chi$. From $P \sim^{\phi} Q$, it follows $P \sim^{\chi} Q$; from this, (6) and from the correctness of the proof system we get $R_i \sim^{\chi} S_j$; this in turn implies $R_\gamma \sim^{\chi} S_\gamma$; hence $R_\gamma \sim^{\xi\wedge\chi} S_\gamma$. From the latter fact and Lemma 5.7, it follows:

$$R_C \sim^{\chi\wedge\xi} S_C\,. \tag{12}$$

At this stage, we have to distinguish the possible cases for γ; we only deal with the case $\gamma =$ bound-output, since it is the most interesting. For other cases, from now on the proof parallels that of [HL93]. By α-equivalence, we can assume w.l.o.g. that for each $i \in I$ and $j \in J$, $bn(\alpha_i) = bn(\beta_j) = y$, with y fresh. Choose now $w \in C$. Since $\chi \wedge \xi \models [x = w]$ for each $x \in C$, by repeatedly applying $Subst2$, we can write

$$\chi \wedge \xi \rhd R_C = \sum_{i\in I\,|\,subj(\alpha_i)\in C} \overline{w}(y).P_i \stackrel{def}{=} F \;\text{ and }\; \chi \wedge \xi \rhd S_C = \sum_{j\in J\,|\,subj(\beta_j)\in C} \overline{w}(y).Q_j \stackrel{def}{=} G. \tag{13}$$

Now it is easy to show (11). We will show now that for each summand $\overline{w}(y).Q_j$ of G, it holds that

$$\chi \wedge \xi \rhd F = F + \overline{w}(y).Q_j. \tag{14}$$

From this, applying the congruence laws, it will follow that $\chi \wedge \xi \rhd F = F + G$; symmetrically, it will also be the case that $\chi \wedge \xi \rhd G = F + G$, from which it will follow $\chi \wedge \xi \rhd F = G$, that, together with (13), establishes (11).

We will now show (14) for an arbitrary $\overline{w}(y).Q_j$. Observe that $G \stackrel{true,\overline{w}(y)}{\longrightarrow} Q_j$ and $F \simeq^{\chi\wedge\xi} G$ (the latter is implied by (13) and (12)) imply, by the definition of symbolic bisimulation, that there exists a $(\chi \wedge \xi \wedge \bigwedge_{z\in fn(F,G,\chi\wedge\xi)}[y \neq z])$-decomposition D s.t. for each $\zeta \in D$ there exists a transition $F \stackrel{true,\overline{w}(y)}{\longrightarrow} P_{i_j}$ with

$$P_{i_j} \simeq^{\zeta} Q_j$$

hence also

$$P_{i_j} \simeq^{\zeta\wedge\chi\wedge\xi} Q_j.$$

Observe that $\zeta \wedge \chi \wedge \xi \models \chi \wedge \xi$, thus exploiting Lemma 5.9:

$$depth(\zeta \wedge \chi \wedge \xi, P_{i_j}) \leq depth(\chi \wedge \xi, P_{i_j}) < depth(\chi \wedge \xi, \overline{w}(y).P_{i_j}) \leq k$$

thus by the induction hypothesis:

$$\zeta \wedge \chi \wedge \xi \rhd P_{i_j} = Q_j$$

and hence:

$$\zeta \wedge \chi \wedge \xi \rhd \overline{w}y.P_{i_j} = \overline{w}y.Q_j.$$

Let $F' \stackrel{def}{=} \sum_{i\in I\,|\,subj(\alpha_i)\in C} \overline{w}y.P_i$. Applying the congruence laws and the axioms for $+$, we obtain:

$$\zeta \wedge \chi \wedge \xi \rhd F' = F' + \overline{w}y.Q_j.$$

Since the above relation holds for each $\zeta \in D$ and $\{\zeta\wedge\chi\wedge\xi \,|\, \zeta \in D\}$ is a $(\chi\wedge\xi\wedge\bigwedge_{z\in fn(F,G,\chi\wedge\xi)}[y \neq z])$-decomposition, an application of the Cut rule yields:

$$\chi \wedge \xi \wedge \bigwedge_{z\in fn(F,G,\chi\wedge\xi)} [y \neq z] \rhd F' = F' + \overline{w}y.Q_j.$$

Applying the Res inference rule with y and the law $R3$, we obtain

$$R_y(\chi \wedge \xi) \wedge R_y(\bigwedge_{z\in fn(F,G,\chi\wedge\xi)} [y \neq z]) \rhd F = F + \overline{w}(y).Q_j. \tag{15}$$

Now, since $y \notin fn(\chi \wedge \xi)$, it holds that $R_y(\chi \wedge \xi) = \chi \wedge \xi$; furthermore $R_y(\bigwedge_{z\in fn(F,G,\chi\wedge\xi)}[y \neq z])$ is equivalent to $true$; from these facts, $Cons$ and (15) the desired (14) follows. □

6 Dealing with Late Bisimulation

We report in this section the definition of symbolic late bisimulation and the related theorems of consistency and adequacy. After that, we discuss a symbolic characterization of ground bisimulation and a proof system for symbolic late bisimulation.

In the sequel, $\dot{\sim}_l$ will denote the standard ground late bisimulation and $\sim_l^\phi$ the closure of $\dot{\sim}_l$ under ϕ. We will only deal with the input case of each theorem, since the other cases are formally the same as early bisimulation.

Symbolic late bisimulation is obtained by simply adding the condition $bn(\alpha) \cap n(\bigvee D) = \emptyset$ to the first item of the Definition 2.5: this amounts to imposing that no alternative of the decomposition depends on the "value" of the formal parameter $bn(\alpha)$, i.e. to forbidding case-analysis on the actual value of $bn(\alpha)$.

Definition 6.1 (Symbolic late bisimulation)

- *A family $\mathcal{F} = \{R_\phi \,|\, \phi \in BF\}$ of symmetric binary relations over $\mathcal{P}$, indexed over the set BF of boolean formulae, is a* family of symbolic late bisimulations (FSLB) *iff $\forall\ \phi$ and $(P,Q) \in R_\phi$, $P \stackrel{\psi,\alpha}{\longmapsto} P'$, with $bn(\alpha) \cap fn(P,Q,\phi) = \emptyset$, implies:*

  there exists a χ-decomposition D, with $bn(\alpha) \cap n(\bigvee D) = \emptyset$, such that for all $\phi' \in D$, there is a transition $Q \stackrel{\psi',\beta}{\longmapsto} Q'$ with $\phi' \models (\psi' \wedge [\alpha = \beta])$ and $(P',Q') \in R_{\phi'}$, where:

$$\chi = \begin{cases} \phi \wedge \psi \wedge \bigwedge_{z \in fn(P,Q,\phi)} [y \neq z] & \text{if } \alpha \text{ is a bound output action } \overline{x}(y) \\ \phi \wedge \psi & \text{otherwise.} \end{cases}$$

- *$P \simeq_l^\phi Q$ iff there exists a FSEB $\{R_\psi | \, \psi \in BF\}$ such that $(P,Q) \in R_\phi$.*

Theorem 6.2 (Consistency of symbolic late bisimulation) *$P \simeq_l^\phi Q$ implies $P \sim_l^\phi Q$.*

Proof: We will show that the relation:

$$\mathcal{R} = \{(P\sigma, Q\sigma) | \, \exists\, \phi.\, \sigma \models \phi \ \text{ and } \ P \simeq_l^\phi Q\}$$

is a ground late bisimulation. Suppose $P \simeq_l^\phi Q$ and $\sigma \models \phi$. Let us see how the moves of $P\sigma$ are matched by $Q\sigma$. Suppose that $P\sigma \stackrel{\alpha}{\longrightarrow} P'$, with $bn(\alpha) \cap fn(P\sigma, Q\sigma) = \emptyset$; by Remark 3.9, we also assume $y \notin n(\sigma)$. We analyze the case when α is an input, $\alpha = x(y)$. It suffices to show that there exists Q' such that $Q\sigma \stackrel{\alpha}{\longrightarrow} Q'$ and for each $z \in fn(P\sigma, Q\sigma, y)$ $(P'\{z/y\}, Q'\{z/y\}) \in \mathcal{R}$. From $P\sigma \stackrel{\alpha}{\longrightarrow} P'$ and Lemma 3.10, we deduce that for some ψ and some w, it holds that:

$$P \stackrel{\psi,\, w(y)}{\longmapsto} P''$$

where $\sigma \models \psi$, $w\sigma = x$ and $P''\sigma = P'$. Since $P \simeq_l^\phi Q$, there exists a $\phi \wedge \psi$-decomposition $D = \{\phi_1, \ldots, \phi_n\}$ s.t. $y \notin n(\bigvee D)$ and for each $i \in \{1, \ldots, n\}$ there exists a symbolic transition:

$$Q \stackrel{\psi_i,\, w_i(y)}{\longmapsto} Q_i$$

where $\phi_i \models \psi_i \wedge [w = w_i]$ and $P'' \simeq_l^{\phi_i} Q_i$. Now, $\sigma \models \phi \wedge \psi$ implies that for some $j \in \{1, \ldots, n\}$, $\sigma \models \phi_j$. Applying again Lemma 3.10 to the transition $Q \stackrel{\psi_j,\, w_j(y)}{\longmapsto} Q_j$, we get the matching transition for $Q\sigma$:

$$Q\sigma \stackrel{x(y)}{\longrightarrow} Q_j\sigma.$$

Fix any $z \in fn(P\sigma, Q\sigma, y)$ and let $Q' = Q_j\sigma'$. We have $P'\{z/y\} = P''\sigma\{z/y\}$ and $Q'\{z/y\} = Q_j\sigma\{z/y\}$. Recall that $\sigma \models \phi_j$; since $y \notin n(\phi_j, \sigma)$, it also holds that $\phi_j\sigma\{z/y\} = \phi_j\sigma$ hence $\sigma\{z/y\} \models \phi_j$ as well. Recall also that $P'' \simeq^{\phi_j} Q_j$. Thus, by definition, $(P'\{z/y\}, Q'\{z/y\}) \in \mathcal{R}$. □

We come now to the adequacy theorem. Lemma 3.11 and Lemma 3.12 extend of course to the late case (i.e., they still hold if replacing $\dot{\sim}$ with $\dot{\sim}_l$).

Theorem 6.3 (Adequacy of symbolic late bisimulation) *$P \sim_l^\phi Q$ implies $P \simeq_l^\phi Q$.*

Proof: We will show that the the family of relations

$$\mathcal{F} = \{\sim_l^\phi \mid \phi \in BF\}$$

is a family of symbolic late bisimulations. Thus let $\phi \in BF$ and $P \sim_l^\phi Q$ and suppose $P \stackrel{\psi,\alpha}{\longmapsto} P'$ with $bn(\alpha) \cap fn(P, Q, \phi) = \emptyset$. We deal only with the case when α is an input. Thus we have to find a $\phi \wedge \psi$-decomposition $D = \{\phi_1, \ldots, \phi_n\}$ s.t. for each $i \in \{1, \ldots, n\}$ $y \notin n(\phi_i)$ and there exists a symbolic transition $Q \stackrel{\psi_i,\beta_i}{\longmapsto} Q_i$ with $\phi_i \models \psi_i \wedge [\alpha = \beta_i]$ and furthermore $P' \sim_l^{\phi_i} Q_i$.

Let $V = fn(P, Q, \phi)$ and $V_y = V \cup bn(\alpha)$. Take $F \subseteq_{fin} \mathcal{N} - V_y$ s.t. $|F| > 2 * |V_y|$ and define the set of substitutions:

$$S = \{\sigma |\, n(\sigma) \subseteq F \cup V \text{ and } \sigma \models \phi \wedge \psi\}\,.$$

S is clearly finite. We define D as $\{\chi(\sigma, V) |\, \sigma \in S\}$. The fact that D is a $\phi \wedge \psi$-decomposition follows from Lemma 3.2, case a. (read the "ϕ" of the statement of the lemma as $\phi \wedge \psi$ and the "ψ" as $\bigvee D$)[3]. We now prove that each $\chi(\sigma, V) \in D$ entails a matching symbolic transition for Q. Thus fix $\sigma \in S$ and consider the following chain of implications:

$P \stackrel{\psi,\alpha}{\longmapsto} P'$	implies, in virtue of Lemma 3.10, since $\sigma \models \psi$ and $y \notin n(\sigma)$
$P\sigma \stackrel{\gamma}{\longrightarrow} P'\sigma$	with $\alpha\sigma = \gamma$; from $P \sim_l^\phi Q$ and $\sigma \models \phi$, for some Q'
$Q\sigma \stackrel{\gamma}{\longrightarrow} Q'$	where for each $z \in fn(P\sigma, Q\sigma, y) : P'\sigma\{z/y\} \dot{\sim}_l Q'\{z/y\}$.
	By Lemma 3.10, there exists a symbolic transition:
$Q \stackrel{\psi',\beta}{\longmapsto} Q''$	with $\sigma \models \psi' \wedge [\beta = \alpha]$ and $Q''\sigma = Q'$.

We now prove that $\chi(\sigma, V) \models \psi' \wedge [\beta = \alpha]$ and that $P' \sim_l^{\chi(\sigma,V)} Q''$.

Consider any ρ s.t. $\rho \models \chi(\sigma, V)$; we have to show that $\rho \models \psi' \wedge [\beta = \alpha]$; in virtue of Lemma 3.6, $n(\psi' \wedge [\beta = \alpha]) \subseteq V$; since $\sigma \models \psi' \wedge [\beta = \alpha]$, by Lemma 3.5 we get $\rho \models \psi' \wedge [\beta = \alpha]$ as well.

We show now that $P' \sim_l^{\chi(\sigma,V)} Q''$. Consider any substitution ρ s.t. $\rho \models \chi(\sigma, V)$; we have to show that $P'\rho \dot{\sim}_l Q''\rho$. From above, we know that for each $z \in fn(P'\sigma, Q''\sigma, y)$, it is $P'\sigma\{z/y\} \dot{\sim}_l Q''\sigma\{z/y\}$. Taken any $w \in V$ (note that $V \neq \emptyset$, in that $x \in V$), it follows that $(w(y).P')\sigma \dot{\sim}_l (w(y).Q'')\sigma$. Since $fn(w(y).P', w(y).Q'') \subseteq V$ and by hypothesis $\rho \models \chi(\sigma, V)$, it follows by Lemma 3.12 that $(w(y).P')\rho \dot{\sim}_l (w(y)Q'')\rho$. From the latter it follows that $P'\rho \dot{\sim} Q''\rho$. □

The discussion of Section 2 on automatic verification extends to the late case as well, by considering the late version of Hennessy and Lin's algorithm.

Also ground late bisimulation $\dot{\sim}_l$ can be characterized in terms of the symbolic late one.

Theorem 6.4 *$P \dot{\sim}_l Q$ iff $P \simeq_l^\phi Q$, where $\phi = \bigwedge_{x,y \in fn(P,Q),\, x,y \text{ distinct}} [x \neq y]$.*

A sound and complete proof system for late bisimulation can be obtained by replacing the Inp rule of the system considered in Section 5 with the simpler rule:

$$(Inp - L)\frac{\phi \rhd P = Q}{\phi \rhd x(y).P = x(y).Q}\ y \notin n(\phi).$$

[3]Observe that in this case y plays essentially no role.

The corresponding provability relation is denoted by $\rhd_l$. The proofs of soundness and completeness can be obtained by slightly modifying those for the early case. In particular, the notion of head normal form remains the same, and, in the completeness part, only the case of input prefixes needs to be changed; this can be done along the same lines of [HL93]. We omit the proof and state:

Theorem 6.5 *$\phi \rhd_l P = Q$ iff $P \simeq_l^\phi Q$.*

7 Conclusions and Related Work

A symbolic transitional semantics for the π-calculus has been introduced and, on top of it, a notion of symbolic bisimulation has been defined, amenable to efficient checking. A sound and complete proof system for symbolic bisimulation has also been provided. Symbolic bisimulation has been related to the standard bisimulations of the π-calculus; this lays the basis for designing more efficient strategies for checking early and late bisimulations.

The symbolic characterization of the bisimulations has another major benefit: it sheds new light on the logical difference between π-calculus bisimulations based on different instantiation strategies, such as early, late and open. It is not difficult to see that different instantiation strategies correspond to different degrees of generality in the case analysis. Indeed, early bisimulation is the most general equivalence, since no restriction is imposed on the case decomposition D. Late bisimulation is obtained by requiring that the formal parameter of the input action is not in D, i.e. by forbidding case analysis on the actual value of the formal parameter.

It is strongly conjectured that, for the sublanguage without negation and disjunction over which open bisimilarity is defined, we can recast open bisimulation equivalence in our framework by simply omitting case analysis. Referring to Definition 2.5, this amounts to requiring that χ itself entails a matching transition for Q. In this case our symbolic formulation reduces essentially to the symbolic characterization of open bisimulation obtained by Sangiorgi ([San93], Definition 5.1). The main difference is that, in [San93], only the equivalence corresponding to *true* is considered; the condition χ on the derivative processes resulting from matching the transitions is encoded as a particular substitution σ_χ, applied to these processes. This technique permits avoiding explicit parameterization of bisimulations over boolean formulae. A formal proof of equivalence between the definition we sketched and that by Sangiorgi should not be too difficult.

The original idea of symbolic bisimulation has been presented in [HL92]. There, a verification algorithm is proposed for a class of symbolic transition graphs and a theorem relating symbolic bisimulations to concrete bisimulations over a version of CCS with value-passing is presented. In [HL93], the same language has then been equipped with a sound and complete proof system. The results obtained by Hennessy and Lin are the direct inspiration of our work, but they cannot be directly extended to the π-calculus for two main reasons:

1. the lack of distinction between variables, values and channels proper of the π-calculus;
2. the absence of a specific language for boolean formulae in the work by Hennessy and Lin.

It is easier to deal with a static value-passing process algebra, because channel names are neatly separated from the exchanged values and thus channels do not appear in the boolean formulae. Of course, this is no longer true in a name-passing calculus, where a subtle interplay between name-scoping and boolean formulae is present. An example of such interplay is offered by the symbolic structural operational rules for the restriction operator:

$$\mathtt{S-Res}\frac{P \stackrel{\phi,\alpha}{\longmapsto} P'}{(\nu\, y)P \stackrel{R_y(\phi),\alpha}{\longmapsto} (\nu\, y)P'}\ y \notin n(\alpha).$$

The symbolic framework of [HL92] and [HL93] is parameterized on the language of boolean formulae, in other words they do not have a specific language for them. In order to establish the relationship between symbolic and concrete bisimulation, they just assume existence of an extremely expressive language, capable of describing any given collection of environments (associations of variables with values). This is admittedly [HL92] a very strong requirement. It is at least not obvious, in the presence of non-trivial value types, that such a language exists. Here, we had to consider a *specific* language (BF) and had to deal with name substitutions rather than with environments. Indeed, it must be said that our solution heavily depends on the specific features of the π-calculus: only finitely many substitutions are important when dealing with a fixed set of formulae and processes. This property does not hold for languages that, besides names, permit exchanging other kinds of values (e.g. integers) and make use of predicates (e.g. $\leq$) over them.

In [PS93], the ground equivalence and the corresponding congruence, for the early and late cases, are separately axiomatized, via four distinct algebraic proof systems. If we confine ourselves to one specific form of bisimulation (be it early or late), the main differences between our proof system and theirs can be summarized as follows. In [PS93], the ground equivalence and the congruence are considered separately; in our framework, all equivalences obtainable as substitution-closures of the ground one (including, as particular cases, the ground equivalence itself and the congruence) are considered at once. As a consequence, it is possible to reason about each such equivalence, just by selecting the appropriate ϕ (though proof systems for different ϕ's depend on each other). Furthermore, in many cases, the symbolic formulation makes it possible a gain in efficiency. As an example, if it has to be proven that $x(y).P$ is ground bisimilar to $x(y).Q$, within the symbolic framework it just suffices to derive $\phi \rhd P = Q$, for some ϕ not containing y and not stronger than the formula corresponding to our symbolic characterization of $\dot{\sim}$. Within the proof system of [PS93], it is necessary to apply the input-prefix rule:

$$IP: \frac{\forall z \in fn(P,Q,y).\ P\{z/y\} = Q\{z/y\}}{x(y).P = x(y).Q}$$

whose premise *always* requires as many sub-proofs as the cardinality of $fn(P,Q,y)$. This example indicates that making reasoning assumptions explicit can often avoid a number of useless checks. An accurate comparison between the two approaches w.r.t. efficient deduction strategies would be interesting.

Between the publication of the short version of this paper [BD94] and the final revision, we learnt of related work by Huimin Lin [Lin94] (successively extended to the weak case in [Lin95]). There, a symbolic semantics very similar to ours is proposed. The considered calculus does not contain mismatch and disjunction, which are nonetheless used in the meta-language for boolean conditions. The absence of negation permits a simpler treatment of restriction, whose symbolic operational rules just contain a side condition ensuring that the restricted name does not occur on the premise's transition. Beside that, the formulation of symbolic bisimulation is slightly simpler, in that a particular "canonical" decomposition is always forced. The latter is taken to be the set of all conditions complete over $fn(P,Q)$ (in the sense of our Definition 5.6) that consistently extend $\phi \wedge \psi$. However, it is not clear to us whether such a formulation can be exploited to gain efficiency when proving equivalence of processes. In this respect, we feel that the advantage of symbolic bisimulation as presented in [HL92] and in our work lies in the fact that, by *choosing* appropriate decompositions, the size and the number of the relations to exhibit can be possibly reduced.

Our work is somewhat related also to [Dam93] and to [FMQ94], where other symbolic transitional semantics for the π-calculus have been presented. In [Dam93], a symbolic semantics is used as a basis for developing a model checker; first-order (rather than boolean) formulae are used; in the operational rules for the restriction operator, the "hiding" of a name y in a formula is modeled using an existential quantifier $\exists y$. The aim of [FMQ94] is to define a uniform framework, within

which different kinds of strategies (such as early, late, open) can be described by just setting certain parameters. The problem of efficiently representing the considered equivalences is not tackled.

Acknowledgments

We thank David Walker for providing helpful suggestions and Rosario Pugliese for careful reading of an earlier draft. Two anonymous referees provided valuable suggestions for improvements.

References

[BD92] M. Boreale and R. De Nicola. Testing equivalence for mobile processes. *Information and Computation*, 120:279–303, 1995. Extended abstract in: R. Cleaveland (ed.), *Proceedings of CONCUR '92, LNCS 630*.

[BD94] M. Boreale and R. De Nicola. A symbolic semantics for the π-calculus - Extended abstract. In B. Jonsson, J. Parrow (eds.) *Proceedings of CONCUR '94, LNCS 836*, Springer-Verlag, Berlin, 1994.

[Dam93] M. Dam. Model checking mobile processes. In E. Best, editor, *Proceedings of CONCUR '93, LNCS 715*. Springer-Verlag, Berlin, 1993.

[FMQ94] G. Ferrari, U. Montanari, and P. Quaglia. π-calculus with explicit substitutions. Technical report, Università di Pisa, 1994. Short version in *Proceedings of MFCS'94*. To appear in *Theoretical Computer Science*.

[Hen91] M. Hennessy. A model for the π-calculus. Technical report 8/91, Computer Science, University of Sussex, 1991.

[HL92] M. Hennessy and H. Lin. Symbolic bisimulations. in *Theoretical Computer Science* 138:353-389.

[HL93] M. Hennessy and H. Lin. Proof systems for message-passing process algebras. In E. Best, editor, *Proceedings of CONCUR '93, LNCS 715*. Springer-Verlag, Berlin, 1993.

[JP93] B. Jonsson and J. Parrow. Deciding bisimulation equivalences for a class of non-finite state programs. *Information and Computation*, 107:272–302, 1993.

[Lin94] H. Lin. Symbolic bisimulations and proof systems for the π-calculus. Technical report 7/94, Computer Science, University of Sussex, 1994.

[Lin95] H. Lin. Complete inference systems for weak bisimulation equivalences in the π-calculus. In *Proceedings of TAPSOFT '95*, LNCS series, Springer-Verlag, Berlin, 1995.

[Mil89] R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.

[MPW92] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, part 1 and 2. *Information and Computation*, 100:1-77, 1992.

[PS93] J. Parrow and D. Sangiorgi. Algebraic theories for name-passing calculi. *Information and Computation*, 120:174–197, 1995.

[San93] D. Sangiorgi. A theory of bisimulation for the π-calculus. Technical report ECS-LFCS 93-270, University of Edinburgh, 1993. To apper in *Acta Informatica*. Short version in E. Best, editor, *Proceedings of CONCUR '93, LNCS 715*. Springer-Verlag, Berlin, 1993.