On Bisimulations for the Asynchronous pi-calculus

Roberto M. Amadio, Ilaria Castellani, Davide Sangiorgi

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

On Bisimulations for the Asynchronous π-calculus

Roberto M. Amadio, Ilaria Castellani, Davide Sangiorgi

N° 2913

June 1996

THÈME 1

ISSN 0249-6399

On Bisimulations for the Asynchronous π-calculus

Roberto M. Amadio, Ilaria Castellani, Davide Sangiorgi

Thème 1 — Réseaux et systèmes
Projet MEIJE

Rapport de recherche n°2913 — June 1996 — 39 pages

Abstract: The *asynchronous π-calculus* is a variant of the π-calculus where message emission is non-blocking. Honda and Tokoro have studied a semantics for this calculus based on bisimulation. Their bisimulation relies on a modified transition system where, at any moment, a process can perform any input action.

In this paper we propose a new notion of bisimulation for the asynchronous π-calculus, defined on top of the standard labelled transition system. We give several characterizations of this equivalence including one in terms of Honda and Tokoro's bisimulation, and one in terms of *barbed equivalence*. We show that this bisimulation is preserved by name substitutions, hence by input prefix. Finally, we give a complete axiomatization of the (strong) bisimulation for finite terms.

Key-words: process calculi, π-calculus, asynchronous communications, bisimulation

(Résumé : tsvp)

Roberto M. Amadio: CNRS, Sophia-Antipolis, e-mail: amadio@cma.cma.fr.
Ilaria Castellani: INRIA, Sophia-Antipolis, e-mail: ic@cma.cma.fr.
Davide Sangiorgi: INRIA, Sophia-Antipolis, e-mail: davide@cma.cma.fr.

Unité de recherche INRIA Sophia-Antipolis
2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex (France)
Téléphone : (33) 93 65 77 77 – Télécopie : (33) 93 65 77 65

Bisimulations pour le pi-calcul asynchrone

Résumé : Le π-calcul asynchrone est une variante du π-calcul de Milner, Parrow et Walker où l'émission de message n'est pas bloquante. Honda et Tokoro ont donné une sémantique fondée sur la notion de bisimulation pour ce calcul. Leur bisimulation est définie sur un système de transitions étiquetées non standard.

Nous proposons ici une nouvelle notion de bisimulation pour le π-calcul asynchrone, qui se base sur le système de transitions étiquetées habituel du π-calcul. Nous donnons plusieurs caractérisations de cette équivalence, en particulier une par équivalence à barbes. D'autre part nous montrons que notre bisimulation coïncide avec celle de Honda et Tokoro, et qu'elle est préservée par substitution et donc qu'il s'agit d'une congruence. Enfin, nous donnons une caractérisation axiomatique de la bisimulation (forte) pour les processus finis.

Mots-clé : calculs de processus, pi-calcul, communication asynchrone, bisimulation

1 Introduction

Process interaction in a distributed system without global clock is usually modelled by message passing. In this context, one often distinguishes between *synchronous* and *asynchronous* message passing. In the former, the send and receive events can be regarded as happening at the same time. In the latter, one can imagine that messages are sent and travel in the ether till they reach their destination, while the sending process accomplishes other tasks.

In the *distributed algorithms* community the distinction synchronous vs. asynchronous communication is not considered a very important issue. For instance [Tel95], pp 44 says:

> Messages in distributed systems can be passed either synchronously or asynchronously. (...) For many purposes synchronous message passing can be regarded as a special case of asynchronous message passing (...)

Indeed one can simulate a synchronous communication with two asynchronous ones. On the other hand in the *language design* community the distinction seems to be quite relevant. Basically, asynchronous communication is easier to implement than the synchronous one as it is closer to the communication primitives offered by available distributed systems. In particular, asynchronous communication has become a popular choice in the design of languages for the programming of distributed applications. An early proposal is Agha's actors model [Agh86], while more recent contributions based on the theory of the π-calculus include Pict [PT96] and the join calculus [FG96].

A second community where the distinction synchronous vs. asynchronous is gaining momentum is that concerned with the *semantics of programs*. In this community one is often interested in comparing calculi. Certain translations turn out to be fully abstract in an asynchronous setting, where the observer has less power. Examples include the encoding of input-guarded choice [NP96] into the asynchronous π-calculus and the encoding of the asynchronous π-calculus into the join calculus [FG96].

A way to restrict a process calculus to asynchronous communications is to remove output prefixing. In other terms, an asynchronous output $\overline{a}$ followed by a process P is the same as the parallel composition $\overline{a} \mid P$. If the calculus has a non-deterministic sum, then we also disallow output guards. We can justify this decision as follows: (i) An output on a choice point forces synchronizations at the implementation level, this seems to contradict the very essence of asynchronous communication (we are not aware of any programming language which allows this). (ii) At the semantic level a calculus with output guards is more discriminating, in particular certain desirable equations such as (2) in section 4 fail to hold.

The resulting calculus is still quite expressive when working in a framework where channel names are transmissible values, e.g. the π-calculus [MPW92]. Indeed it is quite easy to simulate the synchronous π-calculus in the asynchronous one: the sending process waits for an acknowledgment from the receiving process on a private channel. Basic results on the expressiveness of the asynchronous π-calculus can be found in the works by Honda and Tokoro, and Boudol [HT91, Bou92], where the asynchronous π-calculus was first proposed.

When communications are asynchronous, the sender of an output message does not know when the message is actually consumed. In other words, an asynchronous observer, as opposed to a synchronous one, cannot directly detect the input actions of the observed process. Consequently, the asynchronous calculus requires the development of an appropriate semantic framework.

In this paper we develop a theory of bisimulation for the asynchronous π-calculus both in the strong and in the weak case. Our starting point is an original notion of asynchronous bisimulation over the standard labelled transition system. As a first contribution, we provide several characterizations of this bisimulation, and in particular we study under which conditions it coincides with *barbed equivalence*. We also show that our asynchronous bisimulation coincides with that proposed by Honda and Tokoro, which is based on a modified transition system for the π-calculus, on the sublanguage that they consider. As a second result, we observe that asynchronous bisimulation is preserved by the input prefix of the π-calculus (a similar property is proved in [HT92]) and coincides with *ground* bisimulation (a bisimulation where only *one* fresh name is considered in the input clause). Finally, we give a complete axiomatization of asynchronous bisimulation in the strong case for finite terms.

Insensitivity to name instantiation (and hence the possibility of using ground forms of bisimulation) appears to depend on having no output prefixing. It does not depend on having asynchronous, rather than synchronous, bisimulation (see [BS96] for a study of insensitivity to name instantiation for various forms of synchronous bisimulations).

Forms of asynchronous π-calculus have also been studied in [HKH95], but the bisimilarity used is the standard (synchronous) one. Part of our theory, in particular axioms and normal forms, is related to that in [HKH95]. Our formulation of asynchronous bisimulation has been recently used by Nestmann and Pierce [NP96] to prove the full abstraction of the above-mentioned encoding of input-guarded choice.

The paper is organized as follows. In section 2 we provide the basic definitions. In section 3 we present various characterizations and properties of *strong* asynchronous bisimulation. In section 4 we study an equational theory which characterizes strong asynchronous bisimulation for finite terms. In section 5 we adapt some of the results in section 3 to the *weak* case. Appendix A provides a detailed comparison of our work with that of Honda and Tokoro and appendix B contains longer proofs for sections 3-5.

2 Asynchronous π-calculus

The *asynchronous* π-calculus is defined as a subset of the π-calculus where: (i) There is no output prefixing, and (ii) outputs cannot be on a choice point (formally sums are allowed

$$(cong)\ \frac{P \equiv P' \quad P' \xrightarrow{\alpha} Q' \quad Q' \equiv Q}{P \xrightarrow{\alpha} Q} \qquad (\tau)\ \frac{\cdot}{\tau.P \xrightarrow{\tau} P}$$

$$(in)\ \frac{\cdot}{a(b).P \xrightarrow{a\,c} [c/b]P} \qquad (out)\ \frac{\cdot}{\overline{a}b \xrightarrow{\overline{a}b} \mathbf{0}}$$

$$(out_{ex})\ \frac{P \xrightarrow{\overline{a}b} P' \quad a \neq b}{\nu b\, P \xrightarrow{\overline{a}(b)} P'} \qquad (\nu)\ \frac{P \xrightarrow{\alpha} P' \quad a \notin n(\alpha)}{\nu a\, P \xrightarrow{\alpha} \nu a\, P'}$$

$$(sync)\ \frac{P \xrightarrow{\overline{a}b} P' \quad Q \xrightarrow{ab} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'} \qquad (sync_{ex})\ \frac{P \xrightarrow{\overline{a}(b)} P' \quad Q \xrightarrow{ab} Q' \quad b \notin fn(Q)}{P \mid Q \xrightarrow{\tau} \nu b\,(P' \mid Q')}$$

$$(comp)\ \frac{P \xrightarrow{\alpha} P' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P \mid Q \xrightarrow{\alpha} P' \mid Q} \qquad (sum)\ \frac{G \xrightarrow{\alpha} P}{G + G' \xrightarrow{\alpha} P}$$

$$(rep)\ \frac{G \xrightarrow{\alpha} P}{!G \xrightarrow{\alpha} P \mid !G}$$

Figure 1: Labelled transition system with early instantiation

only on input prefixes and τ's). Our language differs from the one proposed in [HT91, Bou92] for the presence of a form of choice. This will be important in the axiomatisation (section 4).

We assume a countable collection Ch of channel names, say $a, b, \ldots$ We distinguish between general processes $P, Q, \ldots$ and guards $G, H, \ldots$ as specified in the following grammars:

$$P ::= \overline{a}b \;\big|\; P \mid P \;\big|\; \nu a\, P \;\big|\; !G \;\big|\; G \qquad G ::= \mathbf{0} \;\big|\; a(b).P \;\big|\; \tau.P \;\big|\; G + G$$

In figure 1 we define a labelled transition system with early instantiation (rule (in)). The actions α are specified as follows: $\alpha ::= \tau \;\big|\; \overline{a}b \;\big|\; \overline{a}(b) \;\big|\; ab$. Conventionally we set $n(\alpha) = fn(\alpha) \cup bn(\alpha)$ where:

$$\begin{array}{lll} fn(\tau) = \emptyset & fn(\overline{a}(b)) = \{a\} & fn(\overline{a}b) = fn(ab) = \{a, b\} \\ bn(\tau) = \emptyset & bn(\overline{a}(b)) = \{b\} & bn(\overline{a}b) = bn(ab) = \emptyset \end{array}$$

The rules $(sync)$, $(sync_{ex})$, $(comp)$, and (sum) have a symmetric version which is omitted. Indeed, parallel composition and sum should be understood as commutative operators. We denote with $\equiv$ syntactic identity modulo α-renaming and with $fn(P)$ the names free in P.

The notion of *weak* transition is defined as usual:

$$\begin{array}{lll} P \stackrel{\tau}{\Rightarrow} P' & \text{iff} & P(\xrightarrow{\tau})^* P' \\ P \stackrel{\alpha}{\Rightarrow} P' & \text{iff} & P \stackrel{\tau}{\Rightarrow} \cdot \xrightarrow{\alpha} \cdot \stackrel{\tau}{\Rightarrow} P' \quad (\text{for } \alpha \neq \tau) \end{array}$$

We write $\rightarrow$ and $\Rightarrow$ as abbreviations for $\xrightarrow{\tau}$ and $\stackrel{\tau}{\Rightarrow}$, respectively. The relations $\rightarrow$ and $\Rightarrow$ are often called *reduction* relations.

The first important technical point arises in the definition of *commitment*. In the asynchronous case it seems natural to restrict the observation to the *output* commitments. The intuition is that an observer has no direct way of knowing if the message he has sent has been received. All the sender can do is to introduce an output particle in the system, unless there is an explicitly programmed acknowledgment mechanism there is no way for him to know when the particle is actually consumed.

Definition 2.1 (commitment) *The strong commitment of a process on a channel expresses the fact that the process is ready to send a message on that channel. Formally, $P \downarrow \overline{a}$ if P can make an output action whose subject is a, that is if there exist P', b such that $P \stackrel{\overline{a}b}{\rightarrow} P'$ or $P \stackrel{\overline{a}(b)}{\rightarrow} P'$. The weak commitment is then defined as:*

$$P \Downarrow \overline{a} \text{ if } P \Rightarrow P' \text{ and } P' \downarrow \overline{a}$$

From the definition of reduction and commitment the notion of barbed bisimulation is derived in a canonical way.

Definition 2.2 (barbed bisimulation) *A symmetric relation S on π-terms is a (strong) barbed bisimulation if whenever PSQ the following holds:*

1. *If $P \downarrow \overline{a}$ then $Q \downarrow \overline{a}$.*
2. *If $P \rightarrow P'$ then $Q \rightarrow Q'$ and $P'SQ'$.*

Let $\stackrel{\bullet}{\sim}$ be the largest barbed bisimulation. The notion of weak barbed simulation is obtained by replacing everywhere the commitment $\downarrow$ with $\Downarrow$ and the transition $\rightarrow$ with $\Rightarrow$. We denote with $\stackrel{\bullet}{\approx}$ the largest weak barbed bisimulation.

A more refined notion of bisimulation can be obtained if we also allow observation of output transitions.

Definition 2.3 ($o\tau$-bisimulation) *A symmetric relation S on π-terms is a (strong) $o\tau$-bisimulation if PSQ, $P \stackrel{\alpha}{\rightarrow} P'$, α is not an input action, and $bn(\alpha) \cap fn(Q) = \emptyset$ implies $Q \stackrel{\alpha}{\rightarrow} Q'$ and $P'SQ'$. Let $\sim_{o\tau}$ be the largest $o\tau$-bisimulation. Again, the notion of weak $o\tau$-bisimulation is obtained by replacing strong transitions with weak transitions. We denote with $\approx_{o\tau}$ the largest weak $o\tau$-bisimulation.*

Both barbed bisimulation and $o\tau$-bisimulation are too rough to distinguish processes such as $a(b).\overline{c}b$ and $a(b).\overline{d}b$. Clearly these processes exhibit different behaviours when they are put in parallel with a process $\overline{a}b$. It is then natural to refine barbed bisimulation to an equivalence which is preserved by parallel composition. Following [MS92], we call it barbed equivalence.

Definition 2.4 (barbed equivalence) *The relations of strong and weak barbed equivalence are defined as follows:*

$$P \sim_b Q \quad \text{if} \quad \forall R (P \mid R \stackrel{\bullet}{\sim} Q \mid R)$$
$$P \approx_b Q \quad \text{if} \quad \forall R (P \mid R \stackrel{\bullet}{\approx} Q \mid R)$$

Another approach consists in looking for a variant of the input clause. This leads to the following notion of asynchronous bisimulation. We will see later (definition 3.6) that several other equivalent definitions are possible.

Definition 2.5 (asynchronous bisimulation) *A relation S is an asynchronous bisimulation if it is an $o\tau$-bisimulation and whenever PSQ and $P \stackrel{ab}{\rightarrow} P'$ the following holds:*

- *either* $Q \stackrel{ab}{\rightarrow} Q'$ *and* $P' S Q'$
- *or* $Q \stackrel{\tau}{\rightarrow} Q'$ *and* $P' S (Q' \mid \overline{a}b)$.

Let $\sim_a$ *be the largest asynchronous bisimulation. The definition of weak asynchronous bisimulation is obtained by replacing the strong labelled transitions with the weak labelled transitions everywhere. We denote with* $\approx_a$ *the largest weak asynchronous bisimulation.*

Since asynchronous bisimulation is the basic bisimulation considered in this paper, we will call it simply bisimulation in what follows.

Remark 2.6 (comparison with [HT91]) *Definition 2.5 relies on a standard labelled transition system. Honda and Tokoro [HT91] take a different approach. They modify the labelled transition system by replacing the input rule with the following rule for the* **0** *process (which to some extent allows one to observe the behaviour of a process after an input):*

$$\frac{\cdot}{\mathbf{0} \stackrel{ab}{\rightarrow} \overline{a}b} \tag{1}$$

Since rules in [HT91] are applied modulo a structural equivalence $\equiv_{HT}$*, and* $P \equiv_{HT} P \mid \mathbf{0}$*, this implies that any process* P *can perform any input* ab.

We think that rule 1 is not so appealing because: (i) it introduces an infinite branching, (ii) it is not obviously compatible with a calculus including choice or other dynamic operators (in particular **0** *fails to be a unit for the choice operator, at least with the usual rule for choice), and (iii) it does not reflect the computational content of processes.*

Honda and Tokoro's bisimulation coincides with ours; the proof is easy, using the characterisation of our asynchronous bisimulation as 1-bisimulation (definition 3.6). A detailed analysis is given in appendix A.

The following properties are specific to the asynchronous π-calculus (properties 1 and 2 also depend on the absence of outputs on choice points):

Lemma 2.7 *1. If* $P \stackrel{\overline{a}b}{\rightarrow} P'$ *then* $P \sim_a P' \mid \overline{a}b$.

2. If $P \stackrel{\overline{a}(b)}{\rightarrow} P'$ *then* $P \sim_a \nu b\,(P' \mid \overline{a}b)$.

3. If $P \stackrel{\overline{a}b}{\rightarrow} \cdot \stackrel{\alpha}{\rightarrow} P'$ *then* $P \stackrel{\alpha}{\rightarrow} \cdot \stackrel{\overline{a}b}{\rightarrow} P'$.

4. If $P \stackrel{\overline{a}(b)}{\rightarrow} \cdot \stackrel{\alpha}{\rightarrow} P'$ *and* $b \notin n(\alpha)$ *then* $P \stackrel{\alpha}{\rightarrow} \cdot \stackrel{\overline{a}(b)}{\rightarrow} P'$.

5. If $P \stackrel{\overline{a}b}{\rightarrow} \cdot \stackrel{ac}{\rightarrow} P'$ *and* c *is fresh, then* $P \stackrel{\tau}{\rightarrow} [b/c]\,P'$.

6. If $P \stackrel{\overline{a}(b)}{\rightarrow} \cdot \stackrel{ac}{\rightarrow} P'$ *and* c *is fresh, then* $P \stackrel{\tau}{\rightarrow} \nu b\,([b/c]\,P')$.

3 Asynchronous bisimulation, strong case

In this section we study some properties of strong asynchronous bisimulation (definition 2.5). In section 5 we will discuss how these results can be lifted to the weak case. Since most proofs for the weak case can be trivially adapted to the strong case we delay all proofs to that section. The contributions of the present section can be summarized as follows:

1. We show that bisimulation is preserved by name substitution.
2. We provide several equivalent definitions of bisimulation.
3. We prove that bisimulation and barbed equivalence coincide.

The definition of bisimulation has been given in an *early* style, and thus contemplates the substitution of the bound name of an input with all possible names. In the *ground* [1] style [San95], on the other hand, *no* name instantiation is needed in the input clause.

Definition 3.1 (ground bisimulation) *A relation S is a ground bisimulation if it is an $o\tau$-bisimulation and whenever PSQ, $P \stackrel{ab}{\rightarrow} P'$ and $b \notin fn(P \mid Q)$ the following holds:*

- *either $Q \stackrel{ab}{\rightarrow} Q'$ and $P'SQ'$*
- *or $Q \stackrel{\tau}{\rightarrow} Q'$ and $P'S(Q' \mid \overline{a}b)$.*

We denote with $\sim_g$ the largest ground bisimulation. Weak ground bisimulation is obtained by replacing transitions with weak transitions. We denote with $\approx_g$ the largest weak ground bisimulation.

Theorem 3.2 *Strong ground bisimulation is preserved by name substitutions.*

An important corollary of theorem 3.2 is that bisimulation and ground bisimulation coincide.

Corollary 3.3 *Strong bisimulation and strong ground bisimulation coincide: $\sim_a = \sim_g$.*

A second corollary is that bisimulation is preserved by input prefix (a property which fails in the synchronous calculus). We can then easily conclude as follows.

Corollary 3.4 *Strong bisimulation is a congruence.*

Besides early and ground, other variants of bisimulation which have been studied in the literature are *late* and *open*. The difference among all these variants is in the requirements on closure under name instantiations. Late bisimulation requires that matching input transitions should be adequate for all instantiations of the bound name. In open [San93] bisimulation the only constraints on equalities among names are those imposed by name extrusion and are recorded as a distinction in the bisimulation clauses. Moreover, in the synchronous π-calculus strong late and early bisimulations are not congruences because they are not preserved by input prefixes, hence the induced congruences, called late and early congruences, have been introduced. In the asynchronous π-calculus, bisimulation is preserved by name instantiations, and therefore all the above forms of bisimulation coincide. We omit the definitions of late and open (which are best defined on a late transition system) and we simply state the result.

[1] We use the adjective ground to emphasize the fact that in this bisimulation the formal parameter of an input prefix is treated as a fresh constant. Note that the terminology ground equivalence was used in [MPW92], pp 28, with quite a different meaning.

Corollary 3.5 *Late and open variants of strong (asynchronous) bisimulation coincide with the early strong (asynchronous) bisimulation.*

We have thus demonstrated some interesting mathematical properties of our notion of bisimulation. Our next task will be to give an intuitive justification of this notion. First, we introduce three further definitions of bisimulation, which differ in the formulation of the input clause, and we show them all equivalent to definition 2.5. Roughly, 1-bisimulation requires preservation under parallel composition with an output, while 2,3-bisimulations propose variants of the diagram chasing in the input clause (cf. definition 2.5).

Definition 3.6 (variants of bisimulation) *An i-bisimulation ($i = 1, 2, 3$) is an $o\tau$-bisimulation S such that:*

- (1-bisimulation) PSQ *implies* $(P \mid \overline{a}b)\, S\, (Q \mid \overline{a}b)$, *for all* $\overline{a}b$.
- (2-bisimulation) PSQ *and* $P \stackrel{ab}{\rightarrow} P'$ *implies*
  - *either* $Q \stackrel{ab}{\rightarrow} Q'$ *and* $P'SQ'$
  - *or* $Q \stackrel{\tau}{\rightarrow} Q'$ *and there is* P'' *s.t.* $P' \stackrel{\overline{a}b}{\rightarrow} P''$ *and* $P''SQ'$.
- (3-bisimulation) PSQ *and* $P \stackrel{ab}{\rightarrow} P'$ *implies*
  - *either* $Q \stackrel{ab}{\rightarrow} Q'$ *and* $P'SQ'$
  - *or there are* P'', P''' *s.t.* $P' \stackrel{\overline{a}b}{\rightarrow} P''$, $P \stackrel{\tau}{\rightarrow} P'''$ *and* $P''SP'''$.

We denote with $\sim_i$ the largest i-bisimulation, for $i = 1, 2, 3$.

Theorem 3.7 (characterization) *All definitions of bisimulation are equivalent. That is:* $\sim_a = \sim_1 = \sim_2 = \sim_3$.

Our last result connects bisimulation with barbed equivalence. It should be noted that our definition of barbed equivalence follows [MS92]. Honda and Yoshida [HY95] rely on a stronger notion of barbed equivalence, where the preservation under parallel composition with outputs is required at each step.

Theorem 3.8 *Let P, Q be processes. Then $P \sim_b Q$ iff $P \sim_a Q$.*

4 Equational theory, strong case

We present now an equational theory which characterizes strong asynchronous bisimulation on finite terms. In the rest of this section we shall concentrate on the restricted language without replication. In this case the following equation summarizes the differences between the synchronous and the asynchronous bisimulations:

$$a(b).(\overline{a}b \mid P) + \tau.P = \tau.P \qquad b \notin fn(P) \tag{2}$$

The reader should pause to formally verify this equation according to definition 2.5. A particular instance of equation 2 is $a(b).\overline{a}b + \tau = \tau$ which intuitively says that the process that emits what it has just received can be "absorbed" in an internal action.

Our axiom system is reported in figure 2. The proof of completeness relies on a non-standard notion of normal form. Let us first observe that, due to the absence of output prefix in the syntax, the parallel operator cannot be completely eliminated via an expansion theorem. Unrestricted outputs will continue to be present as parallel components in normal forms, and their possible communications with the rest of the process will remain potential (that is, they will not give rise to an explicit τ-action in the normal form). A related notion of normal form is introduced in [HKH95]. In this work the equational theory captures strong synchronous, rather than *a*synchronous, bisimulation; the axiom system is essentially the same as that in figure 2 but without equation 2.

We introduce some notation. Let $\prod_{i\in I}\overline{a_i}b_i$ denote a product of outputs, defined up to the laws (P1)–(P3) in figure 2 (monoid laws for $|$). We shall use $\vec{c}$ to denote a sequence of names $c_1,\ldots,c_m$. If $\vec{c} = c_1,\ldots,c_m$, we let $\nu\vec{c}\,P$ stand for $\nu c_1 \ldots \nu c_m\, P$. If $\vec{c} = \varepsilon$ (the empty sequence), we let by convention $\nu\varepsilon\, P \equiv P$. With a slight abuse of notation, we will sometimes use $\vec{c}$ also to represent the set $\{c_1,\ldots,c_m\}$ (this is justified by axiom (R3)). We define now the set $Fire(\nu\vec{c}\prod_{i\in I}\overline{a_i}b_i)$ of indices of firable outputs of $\prod_{i\in I}\overline{a_i}b_i$ when all names in $\vec{c}$ are restricted.

Definition 4.1 *Let $P \equiv \nu\vec{c}\prod_{i\in I}\overline{a_i}b_i$. Then $Fire(P) = \bigcup_n Fire_{n\in\omega}(P)$, where $Fire_n(P)$ is the set of indices of outputs that can be fired after exactly n steps, given by:*

$$\begin{aligned} Fire_0(P) &= \{\, i \mid a_i \notin \vec{c}\,\} \\ Fire_{n+1}(P) &= \{\, i \mid \exists\, k \in Fire_n(P)\ \ b_k = a_i\,\} \setminus \bigcup_{m\le n} Fire_m(P) \end{aligned}$$

Example 4.2 *Let $P = \nu b\,\nu c\,\Pi_{i\in I}\overline{a_i}b_i$ with $I = \{1,2,3,4\}$ and $\overline{a_1}b_1 = \overline{a}b$, $\overline{a_2}b_2 = \overline{a}c$, $\overline{a_3}b_3 = \overline{b}c$, and $\overline{a_4}b_4 = \overline{c}b$. Then $Fire_0(P) = \{1,2\}$, $Fire_1(P) = \{3,4\}$, and $Fire_n(P) = \emptyset$ for $n \ge 2$. Hence $Fire(P) = I$. Note that by construction $Fire_n(P) \cap Fire_m(P) = \emptyset$ if $n \ne m$.*

Let $=_{SP}$ be the congruence induced by the laws (S1)–(S4), (P1)–(P3) in figure 2 (commutative monoid laws and idempotence for $+$, and commutative monoid laws for $|$).

Definition 4.3 *A* normal form *is a term defined up to (S1)–(S3) and (P1)–(P3) of the form:*

$$\nu\vec{c}\left(\prod_{i\in I}\overline{a_i}b_i \;\mid\; \Big(\sum_{j\in J}\tau.P_j \;+\; \sum_{k\in K}a_k(b).P_k\Big)\right)$$

where the sets I, J, K are pairwise disjoint, each P_j, P_k is a normal form, and supposing $\vec{c} = c_1,\ldots,c_m$, the following conditions are satisfied:

1. *(All restricted names are emitted)* $\forall \ell \in \{1,\ldots,m\}\ \exists\, i \in I\ \ b_i = c_\ell$
2. *(All outputs are firable)* $Fire(\nu\vec{c}\prod_{i\in I}\overline{a_i}b_i) = I$

3. (Non-redundancy) $\forall k \forall j \;\; P_k \neq_{SP} (\overline{a_k} b \mid P_j)$.

By convention $\prod_{i\in I} \overline{a_i} b_i \equiv \mathbf{0}$ *if* $I = \emptyset$ *(and similarly for the sums* $\sum_{j\in J} \tau.P_j$ *and* $\sum_{k\in K} a_k(b).P_k$*). Thus* $\mathbf{0}$ *is a normal form, when* $\vec{c} = \varepsilon$ *and* $I = J = K = \emptyset$*. A* guarded normal form *is a normal form such that* $\vec{c} = \varepsilon$ *and* $I = \emptyset$*.*

We will show that each term P can be reduced to a normal form using axioms $\mathcal{A}$ in figure 2. Most axioms are standard: (EXP) is an instance of the expansion theorem applied to guards, (OABS) is a form of expansion in which the output particles which are not firable are forced to synchronize or to be postponed. Let $=_{\mathcal{A}}$ denote the congruence induced by these axioms. The proof of normalisation uses nested induction on the depth and on the structure of P.

Definition 4.4 *The* depth *of a process* P*,* $d(P)$*, is defined inductively by:*

$$
\begin{aligned}
&d(\mathbf{0}) \;=\; 0; \quad d(\overline{a}b) \;=\; 1;\\
&d(a(b).P) \;=\; d(\tau.P) \;=\; 1 + d(P);\\
&d(P \mid Q) \;= d(P) \;+\; d(Q);\\
&d(\nu a\, P) \;= d(P);\\
&d(G \;+\; F) \;=\; max\,\{\, d(G),\, d(F)\,\}.
\end{aligned}
$$

Remark 4.5 *$d(P)$ is an upper bound on the length of the transition sequences of P. It is easy to see that if P' is a subterm of P then $d(P') \leq d(P)$.*

Lemma 4.6 (normalisation lemma) *For any process P there exists a normal form:*

$$\lceil P \rceil \;\equiv\; \nu\vec{c}\left(\prod_{i\in I} \overline{a_i} b_i \;\mid\; (\sum_{j\in J} \tau.P_j \;+\; \sum_{k\in K} a_k(b).P_k\,)\right)$$

such that $P =_{\mathcal{A}} \lceil P \rceil$ and $d(\lceil P \rceil) \leq d(P)$. In particular, every guarded sum G can be reduced to a guarded normal form $\lceil G \rceil \equiv \sum_{j\in J} \tau.P_j \;+\; \sum_{k\in K} a_k(b).P_k$.

In the proof of our completeness result, we shall use also the following:

Lemma 4.7 (separation) *Let P and Q be two normal forms:*

$$P \;\equiv\; \nu\vec{u}\,(\prod_{i\in I} \overline{a_i} b_i \;\mid\; P_\Sigma\,) \qquad \text{and} \qquad Q \;\equiv\; \nu\vec{v}\,(\prod_{h\in H} \overline{c_h} d_h \;\mid\; Q_\Sigma\,)$$

where $P_\Sigma \;\equiv\; (\sum_{j\in J} \tau.P_j \;+\; \sum_{k\in K} a_k(b).P_k\,)$ *and* $Q_\Sigma \;\equiv\; (\sum_{\ell\in L} \tau.Q_\ell \;+\; \sum_{m\in M} c_m(d).Q_m\,)$.

If $P \sim_a Q$ then there exists an injective substitution σ that renames the set $\vec{v}$ into $\vec{u}$ and acts as the identity otherwise, such that:

$$\prod_{i\in I} \overline{a_i} b_i \;\equiv\; \sigma \prod_{h\in H} \overline{c_h} d_h \qquad \text{and} \qquad P_\Sigma \;\sim_a\; \sigma\, Q_\Sigma$$

(S1) $G + \mathbf{0} = G$
(S2) $G + G' = G' + G$
(S3) $G + (G' + G'') = (G + G') + G''$
(S4) $G + G = G$

(P1) $P \mid \mathbf{0} = P$
(P2) $P \mid Q = Q \mid P$
(P3) $P \mid (Q \mid R) = (P \mid Q) \mid R$

(R1) $\nu a\,(\sum_{i\in I} \alpha_i.\, P_i) = \sum\{\alpha_i.\, \nu a\, P_i \mid i \in I,\ a \notin fn(\alpha_i)\} \qquad \forall i\ \ a \notin bn(\alpha_i)$

(R2) $\nu a\,(P \mid Q) = P \mid \nu a\, Q \qquad$ if $a \notin fn(P)$

(R3) $\nu a\, \nu b\ P = \nu b\, \nu a\ P$

(EXP) (*Expansion Theorem*) Let $J \cap K = \emptyset = L \cap M$, $b \notin fn(Q)$, $d \notin fn(P)$.

$$P = (\sum_{j\in J} \tau.\, P_j + \sum_{k\in K} a_k(b).\, P_k) \quad \text{and} \quad Q = (\sum_{\ell\in L} \tau.\, Q_\ell + \sum_{m\in M} c_m(d).\, Q_m). \quad \text{Then :}$$

$$P \mid Q = \sum_{j\in J} \tau.\, (P_j \mid Q) + \sum_{k\in K} a_k(b).\, (P_k \mid Q) + \sum_{\ell\in L} \tau.\, (P \mid Q_\ell) + \sum_{m\in M} c_m(d).\, (P \mid Q_m)$$

(OABS) (*Output Absorption*) Let I, J, K be disjoint, $h \in I \setminus Fire\,(\nu\vec{u} \prod_{i\in I} \overline{a_i} b_i\,)$ and $b \notin \vec{u}$. Then:

$$\nu\vec{u}\left(\prod_{i\in I} \overline{a_i} b_i \ \mid\ (\sum_{j\in J} \tau.\, P_j + \sum_{k\in K} a_k(b).\, P_k\,)\right) =$$

$$\nu\vec{u}\left(\prod_{i\in I\setminus\{h\}} \overline{a_i} b_i \ \mid\ (\sum_{j\in J} \tau.\, (\overline{a_h} b_h \mid P_j) + \sum_{k\in K} a_k(b).\, (\overline{a_h} b_h \mid P_k) + \sum_{\substack{k\in K \\ a_k = a_h}} \tau.\, [b_h/b]\, P_k\,)\right)$$

(IABS) (*Input Absorption*) $a(b).\, (\overline{a} b \mid P) + \tau.\, P = \tau.\, P \qquad b \notin fn(P)$

Figure 2: Axioms $\mathcal{A}$

Theorem 4.8 *On finite terms, the equivalence $\sim_a$ is the congruence generated by the axioms $\mathcal{A}$.*

PROOF. *Soundness*: $P =_{\mathcal{A}} Q \;\Rightarrow\; P \sim_a Q$. This is the easy part: it is proved by exhibiting appropriate bisimulations for each axiom.

Completeness: $P \sim_a Q \;\Rightarrow\; P =_{\mathcal{A}} Q$. Given the normalisation lemma and the soundness of the axioms, it is enough to prove the statement for normal forms. This point relies on lemma 4.7 and is developed in appendix B. □

5 Asynchronous bisimulation, weak case

In an asynchronous world a process can make an input and then emit it again on the same channel without changing the overall behaviour of the system. Some interesting equations that hold in the weak semantics and that further motivate its study are the following:

$$\begin{array}{ll} !(a(b).\overline{a}b) & = \mathbf{0} \\ a(b).(\overline{a}b \mid a(b).P) & = a(b).P \\ a(b).(\overline{a}b \mid G) + G & = a(b).G \end{array}$$

We present the weak versions of theorems 3.2 and 3.7. Our first task is to show that (weak) bisimulation is preserved by substitutions and coincides with ground bisimulation. To this end we first establish some elementary properties whose proof is not completely standard, in particular some work needs to be done to prove transitivity of $\approx_a$ (cf. section B). In the following $P, Q, R \ldots$ denote processes.

Lemma 5.1 *Bisimulation is preserved by parallel composition, restriction, replication and guarded sum, and it is included in ground bisimulation:*

1. *If $P \approx_a Q$ then $P \mid R \approx_a Q \mid R$, $\nu a\, P \approx_a \nu a\, Q$, $\alpha.P + R \approx_a \alpha.Q + R$, and $!P \approx_a !Q$.*
2. *If $P \approx_a Q$ then $P \approx_g Q$.*

Let σ denote a name substitution which is almost everywhere the identity. Whenever we apply a substitution to a process or an action we suppose that the bound names have been renamed so that no conflict can arise, in particular σ acts as an identity on bound names and if $\sigma(c) \neq c$ then $\sigma(c)$ is not a bound name either.

Lemma 5.2 *The transitions of P and σP can be related as follows:*

1. *If $P \xrightarrow{\alpha} P'$ then $\sigma P \xrightarrow{\sigma\alpha} \sigma P'$.*
2. *If $\sigma P \xrightarrow{\alpha'} P''$ and $\alpha' \neq \tau$ then for some P', $P \xrightarrow{\alpha} P'$, $\sigma P' \equiv P''$, and $\sigma\alpha = \alpha'$.*
3. *If $\sigma P \xrightarrow{\tau} P''$ then:*
   (a) *either $P \xrightarrow{\tau} P'$ and $\sigma P' \equiv P''$.*
   (b) *or $\sigma a = \sigma d$, $P \xrightarrow{\overline{a}b} \cdot \xrightarrow{dc} P'$ and $[b/c]\sigma P' \sim_a P''$ (c fresh).*

(c) or $\sigma a = \sigma d$, $P \stackrel{\overline{a}(b)}{\rightarrow} \cdot \stackrel{dc}{\rightarrow} P'$ *and* $\nu b\,([b/c]\sigma P') \sim_a P''$ *(*c *fresh).*

We are now ready to prove the crucial lemma.

Lemma 5.3 *If* $P \approx_g Q$ *then* $\sigma P \approx_a \sigma Q$.

Proof. We show that the following relation is a bisimulation *up to* $\sim_a$ *and restriction*:

$$S = \{(\sigma P, \sigma Q) \mid P \approx_g Q, \sigma \text{ substitution } \} \tag{3}$$

Suppose $\sigma P \stackrel{\alpha}{\rightarrow} P'$. If α is a τ or output action then the "up to" means that there are $\vec{d}$, P'', Q'', Q' such that $\sigma Q \stackrel{\alpha}{\Rightarrow} Q'$ and

$$P' \sim_a \nu\vec{d}\,P'' \quad P'' S Q'' \quad \nu\vec{d}\,Q'' \sim_a Q' \tag{4}$$

If $\alpha \equiv ab$ is an input action then the "up to" means that there are $\vec{d}$, P'', Q'', Q' such that:

- either $\sigma Q \stackrel{ab}{\Rightarrow} Q'$ and condition 4 holds.
- or $\sigma Q \stackrel{\tau}{\Rightarrow} Q'$ and
$$P' \sim_a \nu\vec{d}\,P'' \quad P'' S Q'' \quad \nu\vec{d}\,Q'' \sim_a (Q' \mid \overline{a}b) \tag{5}$$

The various cases are considered in appendix B. □

Theorem 5.4 *Weak ground bisimulation and weak bisimulation coincide and they are preserved by substitution.*

Proof. From lemma 5.1(2) and lemma 5.3 applied with the identity substitution we know that $P \approx_g Q$ iff $P \approx_a Q$. From lemma 5.3 we can conclude that both bisimulations are preserved by substitution. □

It follows that weak bisimulation is preserved by all operators but sum (as usual) and that late and open variants of the weak bisimulation coincide with the early bisimulation studied here.

Corollary 5.5 *If* $P \approx_a Q$ *then* $a(b).P \approx_a a(b).Q$.

We can generalise the characterization of asynchronous bisimulation in terms of 1-bisimulation to the weak case.

Definition 5.6 *Let* S *be a weak* $o\tau$*-bisimulation. We say that* S *is a* weak 1-bisimulation *if* PSQ *implies* $(P \mid \overline{a}b)\, S\, (Q \mid \overline{a}b)$. *We denote with* $\approx_1$ *the largest weak 1-bisimulation.*

Theorem 5.7 (characterization) *The 1-bisimulation coincides with (asynchronous) bisimulation. That is:* $\approx_a = \approx_1$.

We now relate barbed equivalence and bisimulation. In the weak case our results rely crucially on the matching operator which we introduce next (in the strong case matching is not needed). We suppose that the grammar of the calculus is extended by the clause: $P ::= \cdots \mid [a = b]P$. The rule associated to matching in the labelled transition system is:

$$(match) \quad \frac{P \stackrel{\alpha}{\rightarrow} P'}{[c = c]P \stackrel{\alpha}{\rightarrow} P'}$$

We will concentrate on the weak case first. In appendix B we indicate how to eliminate matching in the strong case (hence providing a proof for theorem 3.8).

Proposition 5.8 *Let P, Q, R be processes. Then:*

1. *If σ is an injective substitution on $fn(P \mid Q)$ then $P \approx_a Q$ iff $\sigma P \approx_a \sigma Q$.*
2. *If $P \approx_a Q$ then $P \mid R \approx_a Q \mid R$, for any process R.*
3. *If $P \approx_a Q$ then $P \approx_b Q$.*

Proof. The proof of (1) is standard. The proof of (2) is shaped upon the one for lemma 5.1 (we cannot use directly this lemma because we have extended the calculus with matching). The proof of (3) follows by:

$$\begin{array}{ll} P \approx_a Q & \Rightarrow \forall R(P \mid R \approx_a Q \mid R) \\ & \Rightarrow \forall R(P \mid R \approx_b Q \mid R) \\ & \Rightarrow P \stackrel{\bullet}{\approx} Q \end{array}$$

□

We recall that a lts $(Pr, Act, \mapsto)$ is *image finite* if for any process P and action α the set $\{P' \mid P \stackrel{\alpha}{\mapsto} P'\}$ is finite. We say that a process P is image finite if the lts generated by P is image finite. Image finite processes form an interesting class: w.r.t. strong reduction all processes are image finite (up to renaming of bound names), and w.r.t. weak reduction all finite control processes (cf. [Dam93]) are image finite modulo the equation $\nu a\, P = P$ for $a \notin fn(P)$.

Theorem 5.9 *If P and Q are image finite processes, and $P \approx_b Q$ then $P \approx_a Q$.*

Proof. Let $\mathcal{F}$ be the monotone operator over $\mathcal{P}(Pr \times Pr)$ associated with the definition of asynchronous bisimulation. Suppose $\approx_a^0 = Pr \times Pr$, $\approx_a^{k+1} = \mathcal{F}(\approx_a^k)$, and $\approx_a^\omega = \bigcap_{k<\omega} \approx_a^k$. It is well-known that on an image finite lts the operator $\mathcal{F}$ preserves co-directed sets. In particular, $\mathcal{F}(\approx_a^\omega) = \approx_a^\omega$. It follows that on image finite processes $\approx_a = \approx_a^\omega$. We show that $P \approx_b Q$ implies $P \approx_a^\omega Q$. From the previous remark the theorem follows.

More precisely, we define a collection of tests $R(n, L)$ depending on $n \in \omega$ and L finite set of channel names, and show by induction on n that:

$$\exists L, L' \, (L \supseteq fn(P \mid Q), L' \subseteq L \text{ and } \nu L' \, (P \mid R(n, L)) \stackrel{\bullet}{\approx} \nu L' \, (Q \mid R(n, L))))$$
implies $P \approx_a^n Q$.

Strong case (without matching):

- $\overset{\bullet}{\sim} \supset \sim_{o\tau} \supset \sim_a = \sim_g = \sim_1 = \sim_2 = \sim_3 = \sim_b$.
- $\sim_a$ is a congruence.
- Axiom which distinguishes asynchronous from synchronous bisimulation: $a(b).(\overline{a}b \mid P)+\tau.P = \tau.P, \quad \text{if } b \notin fn(P)$.

Weak case:

- $\overset{\bullet}{\approx} \supset \approx_{o\tau} \supset \approx_a = \approx_1$.
- Without matching: $\approx_g = \approx_a$ is a congruence and $\approx_a \subset \approx_b$.
- With matching on image finite processes: $\approx_a = \approx_b$.

Figure 3: Summary of results

If the property above holds then we can conclude the proof by observing:

$$\begin{aligned} P \approx_b Q \quad &\Rightarrow \forall R\,(P \mid R \overset{\bullet}{\approx} Q \mid R) \\ &\Rightarrow \forall n \in \omega\,(P \mid R(n,L) \overset{\bullet}{\approx} Q \mid R(n,L)) \quad \text{with } L = fn(P \mid Q),\ L' = \emptyset \\ &\Rightarrow \forall n \in \omega\,(P \approx_a^n Q) \\ &\Rightarrow P \approx_a^\omega Q \end{aligned}$$

Full definitions of the tests $R(n,L)$ are given in appendix B. □

Remark 5.10 *(1) In the proof for the strong case one can achieve the effect of matching with synchronization. Therefore theorem 5.9 holds also for a calculus without matching. In the weak case matching plays an essential role, for instance the terms $\overline{a}b$ and $\overline{a}c$ cannot be separated when put in parallel with the process $!(b(d).\overline{c}d) \mid !(c(d).\overline{b}d)$ (which is an equalizer in Honda-Tokoro terminology).*

(2) The definition of the tests $R(n,L)$ does not *involve the guarded sum. This implies that the characterization theorem still holds for an asynchronous calculus without guarded sum.*

(3) In the asynchronous calculus with matching *the various notions of bisimulation do not collapse. For instance consider $P \equiv a(c).\overline{b}e + a(c).\mathbf{0}$ and $Q \equiv P + a(c).[c = d]\overline{b}e$. The processes P and Q are early equivalent but late distinct. Moreover asynchronous bisimulation and barbed equivalence fail to be congruences. If we refine asynchronous bisimulation to an asynchronous congruence (by asking invariance under substitution) and if we refine barbed equivalence to barbed congruence (by considering contexts including the input prefix) then we can show that asynchronous congruence coincides with barbed congruence.*

6 Conclusion

Our contributions are summarized in figure 3. We leave open the problem of finding an axiomatization of weak asynchronous bisimulation (with or without matching), and the

problem of determining the counterpart in the weak case of the characterisations of strong asynchronous bisimulation in terms of $\sim_2$ and $\sim_3$. In another direction, it would be worth investigating the applications of theorem 5.4 (bisimulation equals ground bisimulation) to automatic verification. For instance, one may wonder if it is possible to speed up current verification techniques by compiling into the asynchronous π-calculus and applying ground bisimulation. For this, it would be useful to find syntactic conditions under which asynchronous and synchronous bisimulations coincide.

Acknowledgements

The authors were partially supported by France Télécom, CTI-CNET 95-1B-182 Modélisation de Systèmes Mobiles.

We would like to thank David N. Turner for interesting initial discussions and Marco Pistore and the anonymous referees for helpful comments.

References

[Agh86] G. Agha. *Actors: a model of concurrent computation in distributed systems.* MIT-Press, 1986.

[Bou92] G. Boudol. Asynchrony and the π-calculus. Research Report 1702, INRIA, Sophia-Antipolis, 1992.

[BS96] M. Boreale and D. Sangiorgi. Some congruence properties for π-calculus bisimilarities. Research Report 2870, INRIA, Sophia-Antipolis, 1996.

[Dam93] M. Dam. Model checking mobile processes. In *Proc. CONCUR'93,* Lecture Notes in Computer Science, 715:22–36, 1993. Full version in SICS report RR94:1, 1994.

[FG96] C. Fournet and G. Gonthier. The reflexive CHAM and the join-calculus. *Proc. ACM-POPL*, 1996.

[HKH95] M. Hansen, J. Kleist, and H. Hüttel. Bisimulations for asynchronous mobile processes. In *Proceedings of the Tbilisi Symposium on Language, Logic, and Computation*, 1995. Research paper HCRC/RP-72, Human Communication Research Centre, University of Edinburgh.

[HT91] K. Honda and M. Tokoro. An object calculus for asynchronous communication. *Proc. ECOOP 91, Geneve*, 1991.

[HT92] K. Honda and M. Tokoro. On asynchronous communication semantics. *Object-based concurrent computing, SLNCS 612*, 1992.

[HY95] K. Honda and N. Yoshida. On reduction based process semantics. *Theoretical Computer Science*, 151:437–486, 1995.

[MPW92] R. Milner, J. Parrow, and D. Walker. A Calculus of Mobile Process, Parts 1-2. *Information and Computation*, 100(1):1–77, 1992.

[MS92] R. Milner and D. Sangiorgi. Barbed bisimulation. In *Proc. ICALP 92, SLNCS 623*, 1992.

[NP96] U. Nestmann and B. Pierce. Decoding choice encodings. In *CONCUR 96, SLNCS to appear*, Pisa, 1996.

[PT96] B. Pierce and D. Turner. Pict: a programming language based on the π-calculus. U. Cambridge, 1996.

[San93] D. Sangiorgi. A theory of bisimulation for the π-calculus. in *Proc. CONCUR'93* Lecture Notes in Computer Science, 715:127–142, 1993.

[San95] D. Sangiorgi. Lazy functions and mobile processes. Research Report RR-2515, INRIA, Sophia-Antipolis, 1995. Available via anonymous ftp from `cma.cma.fr` as `pub/papers/davide/RR-2515.ps`.

[Tel95] G. Tel. *Introduction to distributed algorithms.* Cambridge University Press, 1995.

A Coincidence of $\sim_a$ with Honda and Tokoro's bisimulation

In [HT91] Honda and Tokoro define a bisimulation based on a modified transition system for the asynchronous π-calculus without sum. We will show that on this restricted language their bisimulation coincides with our asynchronous bisimulation. We first recall some facts about Honda and Tokoro's transition system. Note that since there is no sum in the language, guarded sums G are reduced to guarded processes of the form $\tau.P$ or $a(b).P$, and replication is limited to such processes (in practice this is no restriction, since replicated input guarded processes are sufficient to simulate general replication).

In Honda and Tokoro's transition system (HT-transition system, for short) the transition relations, which we denote by $\xrightarrow{\alpha}_{HT}$, are defined up to a *structural equivalence* $\equiv_{HT}$. This is the smallest equivalence such that:[2]

$P \equiv Q \;\Rightarrow\; P \equiv_{HT} Q$ ($\equiv$ is syntactic identity modulo α-conversion)
$P \mid \mathbf{0} \equiv_{HT} P$
$P \mid Q \equiv_{HT} Q \mid P$
$P \mid (Q \mid R) \equiv_{HT} (P \mid Q) \mid R$
$\nu a\,(P \mid Q) \equiv_{HT} P \mid \nu a\, Q$ if $a \notin fn(P)$
$\nu a\, \nu b\, P \equiv_{HT} \nu b\, \nu a\, P$
$!G \equiv_{HT} G \mid !G$
$P \equiv_{HT} Q \;\Rightarrow\; P \mid R \equiv_{HT} Q \mid R$ and $\nu a\, P \equiv_{HT} \nu a\, Q$

Then the transitions $\xrightarrow{\alpha}_{HT}$ are inferred using the system of rules in figure 1 (without the rule (sum)), with the following changes:

1. The congruence rule $(cong)$ is replaced by the rule:

$$(cong_{HT}) \qquad \frac{P \equiv_{HT} P' \quad P' \xrightarrow{\alpha}_{HT} Q' \quad Q' \equiv_{HT} Q}{P \xrightarrow{\alpha}_{HT} Q}$$

2. The input rule (in) is replaced by an input rule for the $\mathbf{0}$ process:

$$(in_{HT}) \qquad \frac{\cdot}{\mathbf{0} \xrightarrow{ab}_{HT} \overline{a}b}$$

3. The communication rule $(sync)$ is replaced by the rule:

$$(sync_{HT}) \qquad \frac{\cdot}{\overline{a}c \mid a(b).P \xrightarrow{\tau}_{HT} [c/b]P}$$

4. The rule $(sync_{ex})$ is not needed anymore, since all restrictions can be brought outside terms using $\equiv_{HT}$.

[2] We take here a slightly simpler equivalence than that used in [HT91], keeping only the clauses that are necessary to infer transitions.

The (strong) bisimulation equivalence[3] based on this transition system, noted $\asymp_{HT}$, is defined as the largest HT-bisimulation.

Definition A.1 (HT-bisimulation) *A relation S is a HT-bisimulation if it is an $o\tau$-bisimulation and whenever PSQ and $P \stackrel{ab}{\rightarrow}_{HT} P'$ then $Q \stackrel{ab}{\rightarrow}_{HT} Q'$ and $P'SQ'$, for any ab.*

Note the rather special role played by input transitions in the HT-transition system: the transitions $\stackrel{ab}{\rightarrow}_{HT}$ are never consumed in communications; they are only used in the bisimulation to create contexts $[\,] \mid \overline{a}b$ for testing processes. In fact, every process can perform any input and it is easy to show the following.

Lemma A.2 $\quad P \stackrel{ab}{\rightarrow}_{HT} P' \quad \Leftrightarrow \quad P' \equiv_{HT} P \mid \overline{a}b$

Proof. ($\Leftarrow$) Suppose $P' \equiv_{HT} P \mid \overline{a}b$. Then, using (in_{HT}), $(comp)$ and $(cong)$, we have $(\, P \equiv_{HT} P \mid \mathbf{0} \stackrel{ab}{\rightarrow}_{HT} P \mid \overline{a}b \equiv_{HT} P' \,) \;\Rightarrow\; P \stackrel{ab}{\rightarrow}_{HT} P'$.

($\Rightarrow$) By induction on the proof of the transition. If the only rule used for deducing $P \stackrel{ab}{\rightarrow}_{HT} P'$ is (in_{HT}) then $P \equiv \mathbf{0}$ and the result is immediate. If the last rule used is $(comp)$, the result is also immediate by induction. Suppose now the last rule is (ν), that is, $\nu c\, P \stackrel{ab}{\rightarrow}_{HT} \nu c\, P'$ is deduced from $P \stackrel{ab}{\rightarrow}_{HT} P'$, $a, b \neq c$. By induction $P' \equiv_{HT} P \mid \overline{a}b$. Then $\nu c\, P' \equiv_{HT} \nu c (P \mid \overline{a}b) \equiv_{HT} \nu c\, P \mid \overline{a}b$. Let now the last rule be (rep). This means that $!G \stackrel{\alpha}{\rightarrow}_{HT} Q \mid !G$ is deduced from $G \stackrel{\alpha}{\rightarrow}_{HT} Q$. By induction $Q \equiv_{HT} G \mid \overline{a}b$. Then $Q \mid !G \equiv_{HT} G \mid \overline{a}b \mid !G \equiv_{HT} !G \mid \overline{a}b$. Suppose finally that the last rule used is $(cong)$, that is, $P \stackrel{ab}{\rightarrow}_{HT} P'$ is deduced from $P \equiv_{HT} Q \stackrel{ab}{\rightarrow}_{HT} Q' \equiv_{HT} P'$. By induction $Q' \equiv_{HT} Q \mid \overline{a}b$. Then, since $\equiv_{HT}$ is preserved by parallel composition, also $P' \equiv_{HT} Q' \equiv_{HT} Q \mid \overline{a}b \equiv_{HT} P \mid \overline{a}b$. □

This property will be the basis for an alternative definition of the HT-transition system, where there is no recourse to a structural equivalence. This new transition system, which we call *direct* HT-transition system, will be easier to compare with ours. It includes two kinds of input transitions:

- Those generated by $\mathbf{0}$ processes, noted $\stackrel{ab}{\mapsto}_0$, which are only used in the bisimulation to create contexts $[\,] \mid \overline{a}b$.
- Those corresponding to input guards $a(b).P$, noted $\stackrel{ab}{\mapsto}_1$, which are only used in communications and never tested directly by the bisimulation.

We will use $\stackrel{\alpha}{\mapsto}$ to denote a generic transition in the direct HT-transition system. The transition relations $\stackrel{\alpha}{\mapsto}$ are defined by the system of rules in figure 4 (where we omit the symmetric rules for (in_0), $(sync')$, $(sync'_{ex})$ and $(comp)$) and in rules $(cong)$, (ν), $(comp)$ and (rep) we use $\stackrel{ab}{\mapsto}$ to denote either kind of input transition. Note that the communication rules $(sync')$ and $(sync'_{ex})$ are based uniquely on the input transitions $\stackrel{ab}{\mapsto}_1$ corresponding to input guards. The input transitions $\stackrel{ab}{\mapsto}_0$ satisfy a slightly weaker property than that expressed by lemma A.2.

[3] In fact Honda and Tokoro define directly the weak bisimulation.

The transition relations $\stackrel{\alpha}{\mapsto}$ are the smallest relations such that:

$$(cong)\quad \frac{P \equiv P' \quad P' \stackrel{\alpha}{\mapsto} Q' \quad Q' \equiv Q}{P \stackrel{\alpha}{\mapsto} Q}$$

$$(in_0)\quad \frac{\cdot}{P \stackrel{ab}{\mapsto}_0 P \mid \overline{a}b} \qquad (in_1)\quad \frac{\cdot}{a(b).P \stackrel{ac}{\mapsto}_1 [c/b]P}$$

$$(\tau)\quad \frac{\cdot}{\tau.P \stackrel{\tau}{\mapsto} P} \qquad (out)\quad \frac{\cdot}{\overline{a}b \stackrel{\overline{a}b}{\mapsto} \mathbf{0}}$$

$$(out_{ex})\quad \frac{P \stackrel{\overline{a}b}{\mapsto} P' \quad a \neq b}{\nu b\, P \stackrel{\overline{a}(b)}{\mapsto} P'} \qquad (\nu)\quad \frac{P \stackrel{\alpha}{\mapsto} P' \quad a \notin n(\alpha)}{\nu a\, P \stackrel{\alpha}{\mapsto} \nu a\, P'}$$

$$(sync')\quad \frac{P \stackrel{\overline{a}b}{\mapsto} P' \quad Q \stackrel{ab}{\mapsto}_1 Q'}{P \mid Q \stackrel{\tau}{\mapsto} P' \mid Q'} \qquad (sync'_{ex})\quad \frac{P \stackrel{\overline{a}(b)}{\mapsto} P' \quad Q \stackrel{ab}{\mapsto}_1 Q' \quad b \notin fn(Q)}{P \mid Q \stackrel{\tau}{\mapsto} \nu b\,(P' \mid Q')}$$

$$(comp)\quad \frac{P \stackrel{\alpha}{\mapsto} P' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P \mid Q \stackrel{\alpha}{\mapsto} P' \mid Q} \qquad (rep)\quad \frac{G \stackrel{\alpha}{\mapsto} P}{!G \stackrel{\alpha}{\mapsto} P \mid !G}$$

Figure 4: Direct HT labelled transition system

Lemma A.3 *The input transitions $\stackrel{ab}{\mapsto}_0$ satisfy the following:*

- $P \stackrel{ab}{\mapsto}_0 P' \;\Rightarrow\; P' \equiv_{HT} P \mid \overline{a}b$
- $P' \equiv_{HT} P \mid \overline{a}b \;\Rightarrow\; \exists P''\ (\, P'' \equiv_{HT} P' \text{ and } P \stackrel{ab}{\mapsto}_0 P''\,)$

We show next that the transitions $\stackrel{\alpha}{\mapsto}$ preserve the structural equivalence $\equiv_{HT}$.

Lemma A.4 *The transitions $\stackrel{\alpha}{\mapsto}$ satisfy the property:*

$$P \equiv_{HT} Q \stackrel{\alpha}{\mapsto} Q' \;\Rightarrow\; \exists\, P'\ (\, P \stackrel{\alpha}{\mapsto} P' \equiv_{HT} Q'\,)$$

We can now prove the correspondence between the two HT-transition systems.

Lemma A.5 *The two HT-transition systems are related as follows:*

1. *If α is an output or τ action, then:*

   (i) $P \stackrel{\alpha}{\rightarrow}_{HT} P' \;\Rightarrow\; \exists P''\ (\, P \stackrel{\alpha}{\mapsto} P'' \equiv_{HT} P'\,)$

   (ii) $P \stackrel{\alpha}{\mapsto} P' \;\Rightarrow\; P \stackrel{\alpha}{\rightarrow}_{HT} P'$

2. *Moreover, for output transitions* $P \overset{\overline{a}b}{\mapsto} P'$ *or* $P \overset{\overline{a}(b)}{\mapsto} P'$ *we have:*

   (i) $P \overset{\overline{a}b}{\mapsto} P' \;\Rightarrow\; P \equiv_{HT} \nu\vec{u}\,(\overline{a}b \mid R),\;\; a,b \notin \vec{u} \;\; and \;\; P' \equiv_{HT} \nu\vec{u}\,R$

   (ii) $P \overset{\overline{a}(b)}{\mapsto} P' \;\Rightarrow\; P \equiv_{HT} \nu\vec{u}\,(\overline{a}b \mid R),\;\; a \notin \vec{u},\; b \in \vec{u} \;\; and \;\; P' \equiv_{HT} \nu(\vec{u}\backslash b)\,R$

3. *Case of input transitions* $P \overset{ab}{\mapsto}_0 P'$*:*

   (i) $P \overset{ab}{\rightarrow}_{HT} P' \;\Rightarrow\; \exists P''\;\; (\, P \overset{ab}{\mapsto}_0 P'' \equiv_{HT} P' \,)$

   (ii) $P \overset{ab}{\mapsto}_0 P' \;\Rightarrow\; P \overset{ab}{\rightarrow}_{HT} P'$

4. *Case of input transitions* $P \overset{ab}{\mapsto}_1 P'$*:*

   (i) Let $a,b,c \notin \vec{u}$*. Then* $\nu\vec{u}\,(a(c).Q \mid R) \overset{ab}{\mapsto}_1 \nu\vec{u}\,([b/c]\,Q \mid R)$.

   (ii) $P \overset{ab}{\mapsto}_1 P' \;\Rightarrow\; P \equiv_{HT} \nu\vec{u}\,(a(c).Q \mid R),\;\; a,b,c \notin \vec{u},\;\; P' \equiv_{HT} \nu\vec{u}\,([b/c]\,Q \mid R)$

Proof. Lemma A.4 is used in all cases to care for the fact that the transitions $\overset{\alpha}{\rightarrow}_{HT}$ are defined up to the structural equivalence $\equiv_{HT}$. Then the proof for output transitions is straightforward, since apart from the congruence rule all their defining rules are the same in the two transition systems. Point 2 is an easy consequence of lemma A.3. The proof of 3.(i) is immediate. Point 3.(ii) is shown by induction on the proof of $P \overset{ab}{\mapsto}_1 P'$. We give here the proof for τ-transitions, which relies on 3.

- We show first that $P \overset{\tau}{\rightarrow}_{HT} P' \;\Rightarrow\; \exists P''\;\; (\, P \overset{\tau}{\mapsto} P'' \equiv_{HT} P' \,)$.

– *Basis*: there are two cases to consider, $\tau.P \overset{\tau}{\rightarrow}_{HT} P$ and $\overline{a}c \mid a(b).P \overset{\tau}{\rightarrow}_{HT} [c/b]\,P$. The first case is immediate, since the defining rule is the same in the direct transition system. For the communication case, using rules (out), (in_1) and $(sync')$ we can deduce $\overline{a}c \mid a(b).P \overset{\tau}{\mapsto} \mathbf{0} \mid [c/b]\,P \equiv_{HT} [c/b]\,P$.

– *Inductive step*: the cases where the last rule used is one of $(comp)$, (ν), (rep) are straightforward, since the rules are the same in the two transition systems. Suppose now the last rule used is $(cong)_{HT}$. This means that $P \overset{\tau}{\rightarrow}_{HT} P'$ is inferred from $P \equiv_{HT} Q \overset{\tau}{\rightarrow}_{HT} Q' \equiv_{HT} P'$. By induction we have $Q \overset{\tau}{\mapsto} Q'$. Then by lemma A.4 there exists P'' such that $P \overset{\tau}{\mapsto} P'' \equiv_{HT} Q' \equiv_{HT} P'$.

- We show now that $P \overset{\tau}{\mapsto} P' \;\Rightarrow\; P \overset{\tau}{\rightarrow}_{HT} P'$.

– *Basis*: there is only one case to consider, $\tau.P \overset{\tau}{\mapsto} P$, which is immediate.

– *Inductive step*: cases where the last rule used is one of $(comp)$, (ν), (rep), $(sync')$, $(sync'_{ex})$. We examine the last two cases:

  $(sync')$: Suppose $P \mid Q \overset{\tau}{\mapsto} P' \mid Q'$ because $P \overset{\overline{a}b}{\mapsto} P'$ and $Q \overset{ab}{\mapsto}_1 Q'$. By point 2.$(i)$ we have $P \equiv_{HT} \nu\vec{u}\,(\overline{a}b \mid R)$, $a,b \notin \vec{u}$ and $P' \equiv_{HT} \nu\vec{u}\,R$. Similarly, by point 4.$(ii)$ $Q \equiv_{HT} \nu\vec{v}\,(a(c).S \mid S')$, $a,b,c \notin \vec{v}$, $Q' \equiv_{HT} \nu\vec{v}\,([b/c]\,S \mid S')$. Then, supposing $\vec{u} \cap \vec{v} = \emptyset$ and $\vec{u} \cap fn(Q) = \emptyset = \vec{v} \cap fn(P)$, we have, by rule $(sync_{HT})$:

$$\begin{aligned} P \mid Q \;&\equiv_{HT}\; \nu\vec{u}\vec{v}\;(\, R \mid \overline{a}b \mid a(c).S \mid S' \,) \;\overset{\tau}{\rightarrow}_{HT}\; \nu\vec{u}\vec{v}\;(\, R \mid [b/c]\,S \mid S' \,) \\ &\equiv_{HT}\; \nu\vec{u}\;R \mid \nu\vec{v}\,(\,[b/c]\,S \mid S'\,) \;\equiv_{HT}\; P' \mid Q' \end{aligned}$$

  whence, by rule $(cong_{HT})$, $P \mid Q \overset{\tau}{\rightarrow}_{HT} P' \mid Q'$.

($sync'_{ex}$) : Suppose $P \mid Q \stackrel{\tau}{\mapsto} \nu b\,(P' \mid Q')$ because $P \stackrel{\overline{a}(b)}{\mapsto} P'$ and $Q \stackrel{ab}{\mapsto}_1 Q'$, $b \notin fn(Q)$. By 2.(ii) $P \equiv_{HT} \nu\vec{u}(\overline{a}b \mid R)$, $a \notin \vec{u}$, $b \in \vec{u}$, $P' \equiv_{HT} \nu(\vec{u}\backslash b)\,R$, and by 4.($ii$) $Q \equiv_{HT} \nu\vec{v}(a(x).S \mid S')$, $a, b, c \notin \vec{v}$, $Q' \equiv_{HT} \nu\vec{v}([b/c]\,S \mid S')$. Then, supposing $\vec{u} \cap \vec{v} = \emptyset$ and $\vec{u} \cap fn(Q) = \emptyset = \vec{v} \cap fn(P)$, we have, using rule ($sync_{HT}$) again:

$$\begin{aligned} P \mid Q \quad &\equiv_{HT} \quad \nu\vec{u}\vec{v}\,(\,R \mid \overline{a}b \mid a(c).S \mid S'\,) \stackrel{\tau}{\rightarrow}_{HT} \nu\vec{u}\vec{v}\,(\,R \mid [b/c]\,S \mid S'\,) \\ &\equiv_{HT} \quad \nu b\,(\nu(\vec{u}\backslash b)\,R \mid \nu\vec{v}(\,[b/c]\,S \mid S'\,) \equiv_{HT} \nu b\,(P' \mid Q') \end{aligned}$$

□

The bisimulation equivalence based on the direct transitions $\stackrel{\alpha}{\mapsto}$, noted $\sim_{HT}$, is defined as may be expected.

Definition A.6 (direct HT-bisimulation) *A relation S is a direct HT-bisimulation if it is an $o\tau$-bisimulation and whenever PSQ and $P \stackrel{ab}{\mapsto}_0 P'$ then $Q \stackrel{ab}{\mapsto}_0 Q'$ and $P'SQ'$.*

Using lemma A.5 it is easy to show the following.

Proposition A.7 $\asymp_{HT} = \sim_{HT}$.

We shall now prove the coincidence of $\sim_{HT}$ with our asynchronous bisimulation $\sim_a$. The correspondence between the direct HT-transition system and ours is very easy to establish (note that there is no counterpart for the transitions $\stackrel{ab}{\mapsto}_0$ in our system):

Lemma A.8 *The lts in figure 1 and the direct HT-transition system are related as follows:*

1. $P \stackrel{\overline{a}b}{\mapsto} P' \quad \Leftrightarrow \quad P \stackrel{\overline{a}b}{\rightarrow} P'$
2. $P \stackrel{ab}{\mapsto}_1 P' \quad \Leftrightarrow \quad P \stackrel{ab}{\rightarrow} P'$
3. $P \stackrel{\tau}{\mapsto} P' \quad \Leftrightarrow \quad P \stackrel{\tau}{\rightarrow} P'$

Proof. Immediate, since the defining rules are the same in all cases. □

We are now ready to show that $\sim_{HT}$ coincides with $\sim_a$. The proof is straightforward if we take the characterization of $\sim_a$ as $\sim_1$. In fact the coincidence of $\sim_{HT}$ (in its original formulation $\asymp_{HT}$) with $\sim_1$ was already stated in [HT92] for the weak versions of the bisimulations. The proof is based on the following observation.

Remark A.9 *In the direct HT-transition system any two processes P and Q have the same inputs $\stackrel{ab}{\mapsto}_0$, so checking the correspondence of the transitions $\stackrel{ab}{\mapsto}_0$ reduces to checking the correspondence of the resulting processes; by lemma A.3 these are always of the form $P \mid \overline{a}b$ and $Q \mid \overline{a}b$ (modulo $\equiv_{HT}$, but $\equiv_{HT} \subseteq (\sim_{HT} \cap \sim_1)$).*

Proposition A.10 $\sim_{HT} = \sim_1$.

PROOF.

- $\sim_1 \subseteq \sim_{HT}$. We show that $\sim_1$ is a (direct) HT-bisimulation. Suppose $P \sim_1 Q$. We only have to check the input clause, so let $P \stackrel{ab}{\mapsto}_0 P'$. By lemma A.3 $P' \equiv_{HT} P \mid \overline{a}b$. By rule (in_0) we have $Q \stackrel{ab}{\mapsto}_0 Q \mid \overline{a}b$. Since $P \sim_1 Q$, by definition also $P \mid \overline{a}b \sim_1 Q \mid \overline{a}b$, and thus, since $\equiv_{HT} \subseteq \sim_1$, we conclude $P' \sim_1 P \mid \overline{a}b \sim_1 Q \mid \overline{a}b$.

- $\sim_{HT} \subseteq \sim_1$. Suppose $P \sim_{HT} Q$. We want to show that $P \mid \overline{a}b \sim_{HT} Q \mid \overline{a}b$. But this is immediate because $P \stackrel{ab}{\mapsto}_0 P \mid \overline{a}b$, and since $P \sim_{HT} Q$, there exists Q' such that $Q \stackrel{ab}{\mapsto}_0 Q'$ and $P \mid \overline{a}b \sim_{HT} Q'$. By lemma A.3 we have $Q' \equiv_{HT} Q \mid \overline{a}b$. Thus, since $\equiv_{HT} \subseteq \sim_{HT}$, $P \mid \overline{a}b \sim_{HT} Q' \sim_{HT} Q \mid \overline{a}b$. □

B Proofs

B.1 Proofs of section 3

Preliminaries to the proof of theorem 3.7.

Lemma B.1 *The relations $\sim_a, \sim_1, \sim_2, \sim_3$ are equivalence relations.*

PROOF. The only nontrivial property to show is transitivity. The transitivity of $\sim_1$ is immediate. That of $\sim_a$ is proved for the weak case, see proposition B.8. We prove here the transitivity of $\sim_2$. The transitivity of $\sim_3$ is shown in a similar way.

Transitivity of $\sim_2$. We show that the relation $(\sim_2 \circ \sim_2)$ is a 2-bisimulation. This will imply $(\sim_2 \circ \sim_2) \subseteq \sim_2$ and therefore the transitivity of $\sim_2$. Suppose that $P \sim_2 T \sim_2 Q$. The two interesting cases are:

- $P \stackrel{ab}{\rightarrow} P'$ and T answers by $T \stackrel{\tau}{\rightarrow} T'$ such that for some P'' we have $P' \stackrel{\overline{a}b}{\rightarrow} P''$ and $P'' \sim_2 T'$. Then Q must have a transition $Q \stackrel{\tau}{\rightarrow} Q'$ such that $T' \sim_2 Q'$. Therefore $P'' (\sim_2 \circ \sim_2) Q'$ as required.

- $P \stackrel{ab}{\rightarrow} P'$ and $T \stackrel{ab}{\rightarrow} T'$ with $P' \sim_2 T'$. If $T \stackrel{ab}{\rightarrow} T'$ is matched by $Q \stackrel{ab}{\rightarrow} Q'$ we have finished. So suppose we are in the case where $Q \stackrel{\tau}{\rightarrow} Q'$ and for some T'' we have $T' \stackrel{\overline{a}b}{\rightarrow} T''$ and $T'' \sim_2 Q'$. Then P' must have a transition $P' \stackrel{\overline{a}b}{\rightarrow} P''$ such that $P'' \sim_2 T''$. Therefore $P'' (\sim_2 \circ \sim_2) Q'$ and this concludes the proof. □

Let $\equiv_{HT}$ be the structural equivalence defined in page 19. Clearly $\equiv_{HT}$ is included in all the equivalences $\sim_a, \sim_1, \sim_2, \sim_3$. The following property holds (it should be noted that this property depends on not having outputs on choice points).

Lemma B.2 *If $P \stackrel{\overline{a}b}{\rightarrow} P'$ then $P \equiv_{HT} P' \mid \overline{a}b$.*

Lemma B.3 *The relations $\sim_a$ and $\sim_2$ are preserved by parallel composition with outputs.*

PROOF. The proof for $\sim_a$ is given in lemma B.7 for the weak case. We give here the proof for $\sim_2$. We show that the relation:

$$R = \{ (P \mid \overline{a}b,\ Q \mid \overline{a}b) \mid P \sim_2 Q \} \cup \sim_2$$

is a 2-bisimulation up to $\equiv_{HT}$. We check that the bisimulation condition is satisfied by the pairs $(P \mid \overline{a}b,\ Q \mid \overline{a}b)$.

Consider first the case of *output or τ actions*:

- Case where P moves alone: $P \mid \overline{a}b \xrightarrow{\alpha} P' \mid \overline{a}b$ is inferred from $P \xrightarrow{\alpha} P'$. Since $P \sim_2 Q$, this implies $Q \xrightarrow{\alpha} Q'$ with $P' \sim_2 Q'$. Then $Q \mid \overline{a}b \xrightarrow{\alpha} Q' \mid \overline{a}b$ is the required matching move, since $(P' \mid \overline{a}b,\ Q' \mid \overline{a}b) \in R$.
- Case where $\overline{a}b$ moves alone: $P \mid \overline{a}b \xrightarrow{\overline{a}b} P \mid \mathbf{0}$. Then $Q \mid \overline{a}b \xrightarrow{\overline{a}b} Q \mid \mathbf{0}$ is the matching move, since $(P \mid \mathbf{0},\ Q \mid \mathbf{0}) \in (\equiv_{HT} \circ R \circ \equiv_{HT})$.
- Communication case: $P \mid \overline{a}b \xrightarrow{\tau} P' \mid \mathbf{0}$ is inferred from $P \xrightarrow{ab} P'$ and $\overline{a}b \xrightarrow{\overline{a}b} \mathbf{0}$. There are two possibilities for Q to answer to $P \xrightarrow{ab} P'$:
  - $Q \xrightarrow{ab} Q'$, with $P' \sim_2 Q'$. Then $Q \mid \overline{a}b \xrightarrow{\tau} Q' \mid \mathbf{0}$ is the required move, since $(P' \mid \mathbf{0},\ Q' \mid \mathbf{0}) \in (\equiv_{HT} \circ R \circ \equiv_{HT})$.
  - $Q \xrightarrow{\tau} Q$ and there exists P'' such that $P' \xrightarrow{\overline{a}b} P''$ and $P'' \sim_2 Q'$. By lemma B.2 $P' \equiv_{HT} P'' \mid \overline{a}b$ and then also $P' \mid \mathbf{0} \equiv_{HT} P'' \mid \overline{a}b$. Hence $Q \mid \overline{a}b \xrightarrow{\tau} Q' \mid \overline{a}b$ is the matching move, since $P' \mid \mathbf{0}\ (R \circ \equiv_{HT})\ Q' \mid \overline{a}b$.

Case of *input actions*: here $P \mid \overline{a}b \xrightarrow{cd} P' \mid \overline{a}b$ is inferred from $P \xrightarrow{cd} P'$. Then Q can answer in two ways:

- $Q \xrightarrow{cd} Q'$ and $P' \sim_2 Q'$. In this case we have $Q \mid \overline{a}b \xrightarrow{cd} Q' \mid \overline{a}b$ and $(P' \mid \overline{a}b,\ Q' \mid \overline{a}b) \in R$.
- $Q \xrightarrow{\tau} Q'$ and there exists P'' such that $P' \xrightarrow{\overline{c}d} P''$ and $P'' \sim_2 Q'$. Then $Q \mid \overline{a}b \xrightarrow{\tau} Q' \mid \overline{a}b$ and $P' \mid \overline{a}b \xrightarrow{\overline{c}d} P'' \mid \overline{a}b$, where $(P'' \mid \overline{a}b,\ Q' \mid \overline{a}b) \in R$.

Proof of theorem 3.7: *All the equivalences $\sim_a, \sim_1, \sim_2, \sim_3$ coincide.*

PROOF. We show the three equalities: 1. $\sim_a = \sim_1$, 2. $\sim_a = \sim_2$, 3. $\sim_2 = \sim_3$.

The proof of 1. is given in appendix B.3 for the weak case: let us just mention that the direction $\sim_a \subseteq \sim_1$ uses the fact that $\sim_a$ is preserved by parallel composition with outputs, and the direction $\sim_1 \subseteq \sim_a$ uses transitivity of $\sim_1$. The proof of 3. is straightforward: the direction $\sim_2 \subseteq \sim_3$ uses the transitivity of $\sim_2$, and the direction $\sim_3 \subseteq \sim_2$ uses the transitivity of $\sim_3$. We give here the proof of 2, which relies on lemmas B.2 and B.3 and uses the transitivity of $\sim_2$.

Proof of 2. $\sim_a = \sim_2$. We first show that $\sim_a$ is a 2-bisimulation. Let $P \sim_a Q$. Suppose $P \xrightarrow{ab} P'$ and Q answers by $Q \xrightarrow{\tau} Q'$ such that $P' \sim_a Q' \mid \overline{a}b$. Then P' must be able to simulate the move $Q' \mid \overline{a}b \xrightarrow{\overline{a}b} Q' \mid \mathbf{0}$ by a move $P' \xrightarrow{\overline{a}b} P''$ such that $P'' \sim_a Q' \mid \mathbf{0} \sim_a Q'$.

We show now that $\sim_2$ is an a-bisimulation. Let $P \sim_2 Q$. Suppose $P \stackrel{ab}{\rightarrow} P'$ and Q answers by a transition $Q \stackrel{\tau}{\rightarrow} Q'$ such that for some P'' we have $P' \stackrel{ab}{\rightarrow} P''$ and $P'' \sim_2 Q'$. By lemma B.2 $P' \equiv_{HT} P'' \mid \overline{a}b$ and thus also $P' \sim_2 P'' \mid \overline{a}b$. By lemma B.3, $P'' \sim_2 Q'$ implies $P'' \mid \overline{a}b \sim_2 Q' \mid \overline{a}b$. Whence, by transitivity of $\sim_2$, also $P' \sim_2 Q' \mid \overline{a}b$. □

B.2 Proofs of section 4

Proof of lemma 4.6.

By lexicographic induction on the depth $d(P)$ and on the structure of P. For a given depth, we proceed by structural induction. Axioms S1, S2, S3 and P1, P2, P3 will be used implicitly in the proof, in particular the relation $\equiv$ should be intended as syntactic identity modulo α-renaming, and the axioms above.

- *Case* $n = 0$. If $d(P) = 0$, P is built with the operators $\mathbf{0}$, $\mid$ and νa. If we define $\lceil P \rceil = \mathbf{0}$, then we have $P =_{\mathcal{A}} \lceil P \rceil$ by axioms (P1) and (R1).
- *Case* $n \geq 1$. We proceed by induction on the structure of P.
  1. $P \equiv \overline{a}b$. This is already a normal form.
  2. $P \equiv \alpha.Q$, where $\alpha = a(b)$ or $\alpha = \tau$. By induction there exists a normal form $\lceil Q \rceil$ such that $Q =_{\mathcal{A}} \lceil Q \rceil$ and $d(\lceil Q \rceil) \leq d(Q)$. Then $\lceil P \rceil \equiv \alpha.\lceil Q \rceil$ is a guarded normal form, and $d(\lceil P \rceil) = d(\lceil Q \rceil) + 1 \leq d(Q) + 1 = d(P)$.
  3. $P \equiv G + F$. By induction there exist guarded normal forms $\lceil G \rceil, \lceil F \rceil$ such that $G =_{\mathcal{A}} \lceil G \rceil$, $F =_{\mathcal{A}} \lceil F \rceil$, $d(\lceil G \rceil) \leq d(G)$, $d(\lceil F \rceil) \leq d(F)$. Assume $\lceil G \rceil = \sum_{j \in J} \tau.R_j + \sum_{k \in K} a_k(b).R_k$ and $\lceil F \rceil = \sum_{\ell \in L} \tau.S_\ell + \sum_{m \in M} c_m(d).S_m$. Let now $K' = \{k \in K \mid \exists\, \ell \in L \;\; R_k =_{SP} (\overline{a_k}b \mid S_\ell)\}$, $M' = \{m \in M \mid \exists\, j \in J \;\; S_m =_{SP} (\overline{c_m}d \mid R_j)\}$. Define $\lceil P \rceil = \sum_{j \in J} \tau.R_j + \sum_{k \in K \setminus K'} a_k(b).R_k + \sum_{\ell \in L} \tau.S_\ell + \sum_{m \in M \setminus M'} c_m(d).S_m$. Then $\lceil P \rceil$ is a guarded normal form and we can deduce $P =_{\mathcal{A}} \lceil G \rceil + \lceil F \rceil = \lceil P \rceil$ by repeated use of the absorption law (IABS). Also, we have $d(\lceil P \rceil) \leq max\{\, d(\lceil G \rceil), d(\lceil F \rceil)\,\} \leq max\{\, d(G), d(F)\,\} = d(P)$.
  4. $P \equiv R \mid S$. By induction there exist normal forms $\lceil R \rceil, \lceil S \rceil$ such that $R =_{\mathcal{A}} \lceil R \rceil$, $S =_{\mathcal{A}} \lceil S \rceil$ and $d(\lceil R \rceil) \leq d(R)$, $d(\lceil S \rceil) \leq d(S)$. Suppose that:

$$\lceil R \rceil \;\equiv\; \nu \vec{u} \left(\prod_{i \in I} \overline{a_i} b_i \;\mid\; R_\Sigma \right) \qquad \lceil S \rceil \;\equiv\; \nu \vec{v} \left(\prod_{h \in H} \overline{c_h} d_h \;\mid\; S_\Sigma \right)$$

where

$$R_\Sigma \equiv (\sum_j \tau.R_j + \sum_k a_k(b).R_k) \qquad S_\Sigma \equiv (\sum_\ell \tau.S_\ell + \sum_m c_m(d).S_m)$$

are the guarded parts of $\lceil R \rceil$ and $\lceil S \rceil$. By induction on the depth, all the terms $(R_j \mid S_\Sigma)$, $(R_k \mid S_\Sigma)$, $(R_\Sigma \mid S_\ell)$ and $(R_\Sigma \mid S_m)$ have normal forms (induction can be applied because $d(\lceil R \rceil) \leq d(R)$, $d(\lceil S \rceil) \leq d(S)$). For instance $d(R_j \mid$

$S_\Sigma) < (d(R) + d(S)) = d(P)$ follows from $d(R_j) < d(\lceil R \rceil) \leq d(R)$ and $d(S_\Sigma) \leq d(\lceil S \rceil) \leq d(S)$. Let now $K' = \{k \in K \mid \exists\, \ell \in L \cup J \;\; \lceil R_k \mid S_\Sigma \rceil =_{SP} (\overline{a_k} b \mid \lceil R_\Sigma \mid S_\ell \rceil)\}$ and $M' = \{m \in M \mid \exists\, j \in L \cup J \;\; \lceil R_\Sigma \mid S_m \rceil =_{SP} (\overline{c_m} d \mid \lceil R_j \mid S_\Sigma \rceil)\}$. We can assume that $b \notin fn(\lceil S \rceil)$ and $d \notin fn(\lceil R \rceil)$, and also $\vec{u} \cap \vec{v} = \emptyset$ and $\vec{u} \cap fn(\lceil S \rceil) = \emptyset = \vec{v} \cap fn(\lceil R \rceil)$. We now define:

$$\lceil P \rceil \;\equiv\; \nu \vec{u}\vec{v} \left(\prod_{i \in I} \overline{a_i} b_i \;\mid\; \prod_{h \in H} \overline{c_h} d_h \;\mid \left(\sum_{j \in J} \tau . \lceil R_j \mid S_\Sigma \rceil \;+\; \sum_{\ell \in L} \tau . \lceil R_\Sigma \mid S_\ell \rceil \;+ \right.\right.$$
$$\left.\left. +\; \sum_{k \in K \setminus K'} a_k(b) . \lceil R_k \mid S_\Sigma \rceil \;+\; \sum_{m \in M \setminus M'} c_m(d) . \lceil R_\Sigma \mid S_m \rceil \right) \right)$$

This is indeed a normal form. In particular, since $\vec{v} \cap fn(\prod_i \overline{a_i} b_i) = \emptyset = \vec{u} \cap fn(\prod_h \overline{c_h} d_h)$, we have:

$$\begin{aligned} Fire\,(\nu \vec{u}\vec{v}\, (\prod_{i \in I} \overline{a_i} b_i \mid \prod_{h \in H} \overline{c_h} d_h)) \;&=\; Fire\,(\nu \vec{u} \prod_{i \in I} \overline{a_i} b_i) \cup Fire\,(\nu \vec{v} \prod_{h \in H} \overline{c_h} d_h) \\ &=\; I \;\cup\; H \end{aligned}$$

Using laws (R2), (EXP) and (IABS), we can easily deduce that:

$$\begin{aligned} P \;&=_{\mathcal{A}}\; \lceil R \rceil \mid \lceil S \rceil \;=_{R2}\; \nu \vec{u}\vec{v} \left(\prod_{i \in I} \overline{a_i} b_i \;\mid\; \prod_{h \in H} \overline{c_h} d_h \;\mid R_\Sigma \mid S_\Sigma \right) \\ &=_{EXP}\; \nu \vec{u}\vec{v} \left(\prod_{i \in I} \overline{a_i} b_i \;\mid\; \prod_{h \in H} \overline{c_h} d_h \;\mid \left(\sum_{j \in J} \tau . (R_j \mid S_\Sigma) \;+\; \sum_{\ell \in L} \tau . (R_\Sigma \mid S_\ell) \right.\right. \\ &\qquad \left.\left. +\; \sum_{k \in K} a_k(b) . (R_k \mid S_\Sigma) \;+\; \sum_{m \in M} c_m(d) . (R_\Sigma \mid S_m) \right) \right) \\ &=_{IABS}\; \lceil P \rceil . \end{aligned}$$

Moreover, we have (using max_j as an abbreviation for $max_{j \in J}$):

$$\begin{aligned} d(\lceil P \rceil) \;&\leq\; |I| + |H| + 1 + max_{j,k,\ell,m} \{ d(R_j) + d(S_\Sigma),\; d(R_k) + d(S_\Sigma),\; d(R_\Sigma) + d(S_\ell),\; d(R_\Sigma) + d(S_m) \} \\ &=\; |I| + |H| + 1 + max \{ max_{j,k} \{ d(R_j),\; d(R_k) \} + d(S_\Sigma), max_{\ell,m} \{ d(S_\ell),\; d(S_m) \} + d(R_\Sigma) \} \\ &=\; |I| + |H| + max \{ d(S_\Sigma) + d(R_\Sigma),\; d(S_\Sigma) + d(R_\Sigma) \} \\ &=\; |I| + |H| + d(S_\Sigma) + d(R_\Sigma) \;=\; d(\lceil R \rceil) + d(\lceil S \rceil) \;\leq\; d(P) \end{aligned}$$

5. $P \equiv \nu a\, Q$. By induction $Q =_{\mathcal{A}} \lceil Q \rceil$ and $d(\lceil Q \rceil) \leq d(Q)$. Assume that:

$$\lceil Q \rceil \;\equiv\; \nu \vec{u} \left(\prod_{i \in I} \overline{a_i} b_i \;\mid\; \left(\sum_{j \in J} \tau . Q_j \;+\; \sum_{k \in K} a_k(b) . Q_k \right) \right)$$

We consider separately the two cases where $a \notin fn(\nu \vec{u} \prod_{i \in I} \overline{a_i} b_i)$ and $a \in fn(\nu \vec{u} \prod_{i \in I} \overline{a_i} b_i)$. Note that we can assume $a \notin \vec{u}$, and in this case $a \in fn(\nu \vec{u} \prod_{i \in I} \overline{a_i} b_i) \Leftrightarrow a \in fn(\prod_{i \in I} \overline{a_i} b_i)$.

- If $a \notin fn(\prod_{i\in I} \overline{a_i}b_i)$, we set:

$$\lceil P \rceil \;\equiv\; \nu\vec{u}\left(\prod_{i\in I}\overline{a_i}b_i \;\mid\; (\sum_{j\in J}\tau.\lceil \nu a\, Q_j\rceil \;+\; \sum_{\substack{a_k\neq a\\ k\in K\setminus K'}} a_k(b).\lceil \nu a\, Q_k\rceil\,)\right)$$

where the normal forms $\lceil \nu a\, Q_j\rceil$, $\lceil \nu a\, Q_k\rceil$ exist by induction on the depth, and $K' = \{\, k \in K \mid \exists j \in J \;\; \lceil \nu a\, Q_k\rceil =_{SP} (\overline{a_k}b \mid \lceil \nu a\, Q_j\rceil)\,\}$. This is by definition a normal form. Suppose that both $J \neq \emptyset$ and $K \neq \emptyset$. Then we have:

$$\begin{aligned}
P \;&=_{\mathcal{A}}\; \nu a\vec{u}\left(\prod_{i\in I}\overline{a_i}b_i \;\mid\; (\sum_{j\in J}\tau.Q_j \;+\; \sum_{k\in K} a_k(b).Q_k\,)\right)\\
&=_{\substack{R3\\R2}}\; \nu\vec{u}\left(\prod_{i\in I}\overline{a_i}b_i \;\mid\; \nu a\,(\sum_{j\in J}\tau.Q_j \;+\; \sum_{k\in K} a_k(b).Q_k\,)\right)\\
&=_{\substack{R1\\IABS}}\; \nu\vec{u}\left(\prod_{i\in I}\overline{a_i}b_i \;\mid\; (\sum_{j\in J}\tau.(\nu a\, Q_j) \;+\; \sum_{\substack{a_k\neq a\\ k\in K\setminus K'}} a_k(b).(\nu a\, Q_k)\,)\right)\\
&=_{\mathcal{A}}\; \lceil P\rceil
\end{aligned}$$

The cases where one or both of J, K are empty are simpler, since we do not need to apply (IABS). We have thus shown that $P =_{\mathcal{A}} \lceil P\rceil$ using laws (R1)–(R3) and (IABS). Moreover it is easy to see that:

$$d(\lceil P\rceil) \;\leq\; |I| + 1 + max_{j,k}\{d(Q_j), d(Q_k)\} \;=\; d(\lceil Q\rceil) \;\leq\; d(Q) \;=\; d(P)$$

- If $a \in fn(\prod_{i\in I} \overline{a_i}b_i)$, define $F = Fire\,(\nu a\vec{u}\prod_{i\in I}\overline{a_i}b_i\,)$, $\overline{F} = I\setminus F$, and let $\vec{v}$, $\vec{w}$ be the projections of $a\vec{u}$ on the names that bind, respectively do not bind, some $\overline{a_i}b_i$ such that $i \in F$. Formally, if $\vec{u'} = \{\, u_\ell \;\big|\; \exists\, i \in F \;\; (a_i = u_\ell \;\vee\; b_i = u_\ell)\,\}$ and $\vec{u''} = \vec{u}\setminus\vec{u'}$, we define:

$$\vec{v} \;=\; \begin{cases} a\vec{u'} & \text{if } \exists\, i \in F \;\; (a_i = a \;\vee\; b_i = a)\\ \vec{u'} & \text{otherwise}\end{cases}$$

$$\vec{w} \;=\; \begin{cases} a\vec{u''} & \text{if } \nexists\, i \in F \;\; (a_i = a \;\vee\; b_i = a)\\ \vec{u''} & \text{otherwise}\end{cases}$$

Supposing $b \notin \vec{v}\vec{w}$, let now:

$$\begin{aligned}
\lceil P\rceil \;\equiv\; \nu\vec{v}\Bigg(\prod_{i\in F}\overline{a_i}b_i \;\mid\; (&\sum_{j\in J}\tau.\lceil \nu\vec{w}\,(\prod_{i\in\overline{F}}\overline{a_i}b_i \mid Q_j)\rceil \;+\; \sum_{\substack{k\in K\setminus K'\\ a_k\notin\vec{w}}} a_k(b).\lceil \nu\vec{w}\,(\prod_{i\in\overline{F}}\overline{a_i}b_i \mid Q_k)\rceil\\
&+\; \sum_{\substack{k\in K\\ a_k = a_h, h\in\overline{F}}} \tau.\lceil \nu\vec{w}\,(\prod_{\substack{i\in\overline{F}\\ i\neq h}}\overline{a_i}b_i \mid [b_h/b]Q_k)\rceil\,)\Bigg)
\end{aligned}$$

where all the required normal forms exist by induction on the depth, and $K' = \{ k \in K \mid \exists j \in J \ \lceil \nu\vec{w}\,(\prod_{i\in\overline{F}} \overline{a_i}b_i \mid Q_k)\rceil =_{SP} (\overline{a_k}b \mid \lceil \nu\vec{w}\,(\prod_{i\in\overline{F}} \overline{a_i}b_i \mid Q_j)\rceil \text{ or } \exists k \in K \ \lceil \nu\vec{w}\,(\prod_{i\in\overline{F}} \overline{a_i}b_i \mid Q_k)\rceil =_{SP} (\overline{a_k}b \mid \lceil \nu\vec{w}\,(\prod_{\substack{i\in\overline{F}\\ i\neq h}} \overline{a_h}b_h \mid [b_r/b]Q_k)\,)\,\rceil\}$. Then, if $\lceil Q\rceil \equiv \nu\vec{u}\,Q'$ and $\lceil P\rceil \equiv \nu\vec{v}\,P'$ we have, by applying (OABS) until all unfirable outputs have been pushed under the guards, and then using (R2), (R1) to push under the guards the restrictions in $\vec{w}$:

$$P =_{\mathcal{A}} \nu a\vec{u}\,Q' =_{R3} \nu\vec{v}\vec{w}\,Q' = \underset{\substack{OABS\\ R2,\,R1\\ IABS}}{} \nu\vec{v}\,P' \equiv \lceil P\rceil$$

Moreover:

$$\begin{aligned} d(\lceil P\rceil) &\leq |F| + 1 + max\{\, max_j\{\, |\overline{F}| + d(\lceil Q_j\rceil)\,\}, max_k\{\, |\overline{F}| + d(\lceil Q_k\rceil)\,\}\,\} \\ &= |I| + 1 + max_{j,k}\{\, d(\lceil Q_j\rceil), d(\lceil Q_k\rceil)\,\} \leq d(Q) = d(P) \end{aligned}$$

□

Preliminaries to the proof of lemma 4.7.

Let us look back at the definition of $Fire_n(P)$ for $P \equiv \nu\vec{c}\prod_{i\in I}\overline{a_i}b_i$. Note that the sets $Fire_n(P)$ partition $Fire(P)$. Note also that, since I is finite, there exists a minimal r such that $Fire_{r+1}(P) = \emptyset$. We then have $Fire(P) = \bigcup_{n=0}^{r} Fire_n(P)$. We shall use $\overline{a}\langle b\rangle$ to stand for either $\overline{a}b$ or $\overline{a}(b)$, and $P \xrightarrow{s} P'$ to denote a sequence of transitions $P \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_k} P'$ such that $\alpha_1, \ldots, \alpha_k = s$. The following fact can be easily proved by induction on n.

Remark B.4 *Let $P \equiv \nu\vec{u}\prod_{i\in I}\overline{a_i}b_i$ be a normal form such that $I \neq \emptyset$, and define $I_n = Fire_n(P)$ and $N_n = |I_n|$. If $r = min\{\, n \mid Fire_{n+1}(P) = \emptyset\,\}$, then P has a transition sequence*

$$P \equiv P_0 \xrightarrow{s_0} P_1 \cdots P_r \xrightarrow{s_r} P_{r+1} \equiv \mathbf{0}$$

such that for any $j = 1, \ldots, r+1$, $P_j \equiv \nu\vec{u}^j \prod_{i\in I\setminus I_0,\ldots I_{j-1}} \overline{a_i}b_i$ where $\vec{u}^{j+1} = \vec{u}^j \setminus bn(s_j)$, and, letting $\vec{u}^0 = \vec{u}$, for any $j = 1, \ldots, r$ the sequence $s_j = \overline{a}^j_1\langle b^j_1\rangle, \ldots, \overline{a}^j_{N_j}\langle b^j_{N_j}\rangle$ is a sequentialisation of the outputs in $Fire_j(P)$ such that for any $k = 1, \ldots, N_j$:

$$\overline{a}^j_k\langle b^j_k\rangle = \begin{cases} \overline{a}^j_k(b^j_k) & \text{if } (\, b^j_k \in \vec{u}^j \text{ and } \forall\, \ell \leq k \ \ b^j_\ell \neq b^j_k\,) \\ \overline{a}^j_k b^j_k & \text{otherwise} \end{cases}$$

Remark B.5 *If $|I| = N$, we can assume w.l.o.g. that $I = \{1, \ldots, N\}$. Then we can build a canonical transition sequence $P \equiv P_0 \xrightarrow{s_0} P_1 \cdots P_r \xrightarrow{s_r} P_{r+1} \equiv \mathbf{0}$ where outputs within the same sequence s_i are sequentialised according to the ordering of I.*

Proof of lemma 4.7.

We apply remark B.4. Let the canonical transition sequence associated with P be:

$$P \xrightarrow{\overline{a_1}\langle b_1\rangle} \nu(\vec{u}\backslash b_1)\,(\prod_{i\in I\setminus\{1\}} \overline{a_i}b_i \mid P_\Sigma)\cdots \xrightarrow{\overline{a_N}\langle b_N\rangle} \nu\varepsilon\,(\prod_{i\in\emptyset} \overline{a_i}b_i \mid P_\Sigma) \;\equiv\; P_\Sigma$$

Since $P \sim_a Q$, we can find a matching sequence for Q, possibly using α-conversion. Let σ be a renaming on the names of $\vec{v}$ such that $\sigma\vec{v} = \vec{u}$ and the process $Q' \equiv \nu\vec{u}\,(\,\sigma(\prod_{h\in H} \overline{c_h}d_h \mid Q_\Sigma)\,)$ has the following matching sequence, deduced without using α-conversion:

$$Q' \xrightarrow{\sigma\overline{c_{i_1}}\langle d_{i_1}\rangle} \nu\sigma(\vec{v}\backslash d_{i_1})\,(\,\sigma(\prod_{h\in H\setminus\{i_1\}} \overline{c_h}d_h \mid Q_\Sigma)\,)\cdots \xrightarrow{\sigma\overline{c_{i_N}}\langle d_{i_N}\rangle} \nu\varepsilon\,\sigma(\prod_{h\in\emptyset} \overline{c_h}d_h \mid Q_\Sigma) \equiv \sigma Q_\Sigma$$

where for any $k = 1,\ldots,i_N$, $\sigma\overline{c_{i_k}}\langle d_{i_k}\rangle = \overline{a_k}\langle b_k\rangle$. This shows that $P_\Sigma \sim_a \sigma\, Q_\Sigma$.

Let now $P_\Pi \equiv \prod_{i\in I} \overline{a_i}b_i$ and $Q_\Pi \equiv \prod_{h\in H} \overline{c_h}d_h$. To obtain $P_\Pi \equiv \sigma\, Q_\Pi$ it is enough to show that the two multisets of actions in P_Π and $\sigma\, Q_\Pi$ are the same. But this is an immediate consequence of the above and of the fact that $Fire(\nu\vec{u}\,P_\Pi) = I$ and $Fire(\nu\vec{u}\,\sigma Q_\Pi) = H$ (because P and Q are normal forms).

Complement to the proof of theorem 4.8.

So assume $P \equiv \nu\vec{u}\,(\prod_{i\in I} \overline{a_i}b_i \mid P_\Sigma)$ and $Q \equiv \nu\vec{v}\,(\prod_{h\in H} \overline{c_h}d_h \mid Q_\Sigma)$, where as usual P_Σ and Q_Σ are the guarded parts of P and Q:

$$P_\Sigma \equiv (\sum_{j\in J} \tau.\,P_j + \sum_{k\in K} a_k(b).\,P_k), \quad Q_\Sigma \equiv (\sum_{\ell\in L} \tau.\,Q_\ell + \sum_{m\in M} c_m(d).\,Q_m)$$

By the separation lemma we know that there exists a substitution σ such that $\sigma\vec{v} = \vec{u}$, $\sigma w = w$ if $w \notin \vec{v}$, and:

$$\prod_{i\in I} \overline{a_i}b_i \;\equiv\; \sigma \prod_{h\in H} \overline{c_h}d_h \qquad\qquad P_\Sigma \sim_a \sigma\, Q_\Sigma$$

We will show, by induction on the sum of depths of P and Q, that $P_\Sigma =_{S2} \sigma\, Q_\Sigma$. This will imply the required result, namely:

$$P \;=_{S2}\; \nu\vec{u}\,(\,\sigma \prod_{h\in H} \overline{c_h}d_h \mid \sigma\, Q_\Sigma) \;\equiv\; Q$$

Note that, if P is a normal form and $P \xrightarrow{\alpha} P'$ (where α is any action), then P' is a normal form such that $d(P') < d(P)$.[4] We will show that:

[4] On the other hand, P' is not in general a subterm of P, so we could not use structural induction on normal forms.

$$(*) \qquad P_\Sigma =_{S2} P_\Sigma + \sigma\, Q_\Sigma =_{S2} \sigma\, Q_\Sigma$$

To this end it is enough to prove:

$$(i) \quad P_\Sigma =_{S2} P_\Sigma + \tau.\sigma\, Q_\ell$$
$$(ii) \quad P_\Sigma =_{S2} P_\Sigma + \sigma\, c_m(d).\, Q_m$$

Then (*) will follow by iteration and by symmetry.

(i) Suppose $P_\Sigma \xrightarrow{\tau} P_j$. Since $P_\Sigma \sim_a \sigma\, Q_\Sigma$, there exists $\ell \in L$ such that $\sigma\, Q_\Sigma \xrightarrow{\tau} \sigma\, Q_\ell$ and $P_j \sim_a \sigma\, Q_\ell$. By induction $P_j =_{S2} \sigma\, Q_\ell$ and thus also $\tau.\, P_j =_{S2} \tau.\sigma\, Q_\ell$. Then $P_\Sigma =_{S2} P_\Sigma + \tau.\sigma\, Q_\ell$.

(ii) Let now $P_\Sigma \xrightarrow{a_k b_k} [b_k/b]P_k$. We show first that $\sigma\, Q_\Sigma$ is forced to match this move by a transition of the form $\sigma\, Q_\Sigma \xrightarrow{a_k b_k} [b_k/d]\,\sigma\, Q_m$ for some m such that $\sigma\, c_m d_m = a_k b_k$. For suppose $\sigma\, Q_\Sigma$ responds with a transition $\sigma\, Q_\Sigma \xrightarrow{\tau} \sigma\, Q_\ell$ for some Q_ℓ such that $[b_k/b]P_k \sim_a \overline{a_k} b_k \mid \sigma\, Q_\ell$. Since $d(P_k) < d(P)$ and $d(\overline{a_k} b_k \mid \sigma\, Q_\ell) \leq d(Q)$, we have by induction that $P_k =_{S2} \overline{a_k} b_k \mid \sigma\, Q_\ell$. But then, since $P_\Sigma \sim_a \sigma\, Q_\Sigma$, there must be $j \in J$ such that $P_\Sigma \xrightarrow{\tau} P_j$ and $P_j \sim_a \sigma\, Q_\ell$. By induction this implies $P_j =_{S2} \sigma\, Q_\ell$ and hence $P_k =_{S2} \overline{a_k} b_k \mid P_j$, contradicting the hypothesis that P_Σ is a normal form.

Thus a transition $P_\Sigma \xrightarrow{a_k b_k} [b_k/b]P_k$ is always matched by a transition $\sigma\, Q_\Sigma \xrightarrow{a_k b_k} [b_k/d]\sigma\, Q_m$ such that $\sigma\, c_m d_m = a_k b_k$ and $[b_k/b]P_k \sim_a [b_k/d]\sigma\, Q_m$. By induction $[b_k/b]P_k =_{S2} [b_k/d]\sigma\, Q_m$ and therefore also $a_k(b).\, P_k =_{S2} \sigma\, c_m(d).\, Q_m$. Then $P_\Sigma =_{S2} P_\Sigma + \sigma\, c_m(d).\, Q_m$. □

B.3 Proofs of section 5

Preliminaries to the proof of theorem 5.7.

Remark B.6 $P \approx_a Q \;\Leftrightarrow\; P \approx_a (Q \mid \mathbf{0})$.

Lemma B.7 *The relation $\approx_a$ is preserved by parallel composition with outputs:*

$$P \approx_a Q \;\Rightarrow\; P \mid \overline{a}b \;\approx_a\; Q \mid \overline{a}b$$

PROOF. Let $\equiv_P$ be the congruence induced by the commutativity and associativity laws for $\mid$ (laws (P2), (P3) of our axiom table). We show that the relation:

$$R \;=\; \{\, (P \mid \overline{a}b,\, Q \mid \overline{a}b) \;\mid\; P \approx_a Q \,\} \cup \;\approx_a$$

is an a-bisimulation up to $\equiv_P$. We check that the bisimulation condition is satisfied by the pairs $(P \mid \overline{a}b,\, Q \mid \overline{a}b)$.

Consider first the case of *output or τ actions*:

- Case where P moves alone: $P \mid \overline{a}b \stackrel{\alpha}{\Rightarrow} P' \mid \overline{a}b$ is inferred from $P \stackrel{\alpha}{\Rightarrow} P'$. Since $P \approx_a Q$, this implies $Q \stackrel{\alpha}{\Rightarrow} Q'$ with $P' \approx_a Q'$. Then $Q \mid \overline{a}b \stackrel{\alpha}{\Rightarrow} Q' \mid \overline{a}b$ is the required matching move, since $(P' \mid \overline{a}b,\ Q' \mid \overline{a}b) \in R$.

- Case where $\overline{a}b$ moves alone: $P \mid \overline{a}b \stackrel{\overline{a}b}{\Rightarrow} P \mid \mathbf{0}$. Then $Q \mid \overline{a}b \stackrel{\overline{a}b}{\Rightarrow} Q \mid \mathbf{0}$ is the matching move, since $P \mid \mathbf{0} \approx_a Q \mid \mathbf{0}$ by remark B.6, and $\approx_a \subseteq R$.

- Case where both P and $\overline{a}b$ move, independently. Similar to the previous case: $P \mid \overline{a}b \stackrel{\overline{a}b}{\Rightarrow} P' \mid \mathbf{0}$ is inferred from $P \stackrel{\tau}{\Rightarrow} P'$ and $\overline{a}b \stackrel{\overline{a}b}{\rightarrow} \mathbf{0}$. Then $Q \stackrel{\tau}{\Rightarrow} Q'$ with $P' \approx_a Q'$, and thus $Q \mid \overline{a}b \stackrel{\overline{a}b}{\Rightarrow} Q' \mid \mathbf{0}$ is the matching move.

- Communication case: $P \mid \overline{a}b \stackrel{\tau}{\Rightarrow} P' \mid \mathbf{0}$ is inferred from $P \stackrel{\tau}{\Rightarrow} P_1 \stackrel{ab}{\rightarrow} P_2 \stackrel{\tau}{\Rightarrow} P'$ and $\overline{a}b \stackrel{\overline{a}b}{\rightarrow} \mathbf{0}$. There are two possibilities for Q to answer:

  - $Q \stackrel{\tau}{\Rightarrow} Q_1 \stackrel{\tau}{\Rightarrow} Q_1' \stackrel{ab}{\rightarrow} Q_2' \stackrel{\tau}{\Rightarrow} Q_2 \stackrel{\tau}{\Rightarrow} Q'$, with $P_i \approx_a Q_i$ and $P' \approx_a Q'$. Hence $Q \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q_1' \mid \overline{a}b \stackrel{\tau}{\rightarrow} Q_2' \mid \mathbf{0} \stackrel{\tau}{\Rightarrow} Q' \mid \mathbf{0}$, which is the required move since $P' \mid \mathbf{0} \approx_a Q' \mid \mathbf{0}$.
  - $Q \stackrel{\tau}{\Rightarrow} Q'$ with $P' \approx_a Q' \mid \overline{a}b$. Hence $Q \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q' \mid \overline{a}b$ is the matching move, since $P' \mid \mathbf{0} \approx_a Q' \mid \overline{a}b$.

Consider now the case of *input actions*. There are two possibilities:

- Case where P moves alone: $P \mid \overline{a}b \stackrel{cd}{\Rightarrow} P' \mid \overline{a}b$ is inferred from $P \stackrel{cd}{\Rightarrow} P'$. Then Q can answer in two ways:

  - $Q \stackrel{cd}{\Rightarrow} Q'$ and $P' \approx_a Q'$. In this case we have $Q \mid \overline{a}b \stackrel{cd}{\Rightarrow} Q' \mid \overline{a}b$ and $(P' \mid \overline{a}b,\ Q' \mid \overline{a}b) \in R$.
  - $Q \stackrel{\tau}{\Rightarrow} Q'$ and $P' \approx_a Q' \mid \overline{c}d$. Then $Q \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q' \mid \overline{a}b$ is the required move since $(P' \mid \overline{a}b,\ (Q' \mid \overline{a}b) \mid \overline{c}d) \in (R \circ \equiv_P)$.

- Case where P communicates with $\overline{a}b$, before or after doing the input.

– Suppose the communication occurs earlier, that is $P \stackrel{ab}{\Rightarrow} P_1 \stackrel{cd}{\Rightarrow} P'$ and $P \mid \overline{a}b \stackrel{\tau}{\Rightarrow} P_1 \mid \mathbf{0} \stackrel{cd}{\Rightarrow} P' \mid \mathbf{0} = P''$. By the communication case above, we know that $P \mid \overline{a}b \stackrel{\tau}{\Rightarrow} P_1 \mid \mathbf{0}$ is matched either by $Q \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q_1 \mid \mathbf{0}$ such that $P_1 \mid \mathbf{0} \approx_a Q_1 \mid \mathbf{0}$ or by $Q \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q_1 \mid \overline{a}b$ such that $P_1 \mid \mathbf{0} \approx_a Q_1 \mid \overline{a}b$.

  - In the first case, $P_1 \mid \mathbf{0} \stackrel{cd}{\Rightarrow} P''$ can be matched by $Q_1 \mid \mathbf{0} \stackrel{cd}{\Rightarrow} Q''$ such that $P'' \approx_a Q''$, in which case $P \mid \overline{a}b \stackrel{cd}{\Rightarrow} P''$ is matched by $Q \mid \overline{a}b \stackrel{cd}{\Rightarrow} Q''$; or $P_1 \mid \mathbf{0} \stackrel{cd}{\Rightarrow} P''$ is matched by $Q_1 \mid \mathbf{0} \stackrel{\tau}{\Rightarrow} Q''$ such that $P'' \approx_a Q'' \mid \overline{c}d$, in which case $P \mid \overline{a}b \stackrel{cd}{\Rightarrow} P''$ is matched by $Q \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q''$.
  - The second case is similar. One can compose the move $Q \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q_1 \mid \overline{a}b$ with either $Q_1 \mid \overline{a}b \stackrel{cd}{\Rightarrow} Q''$ such that $P'' \approx_a Q''$ or with $Q_1 \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q''$ such that $P'' \approx_a Q'' \mid \overline{c}d$.

– The case where the communication occurs later is slightly more involved. Suppose $P \stackrel{cd}{\Rightarrow} P_1 \stackrel{ab}{\Rightarrow} P'$ and $P \mid \overline{a}b \stackrel{cd}{\Rightarrow} P_1 \mid \overline{a}b \stackrel{\tau}{\Rightarrow} P' \mid \mathbf{0}$. By the case where P moves alone (first

item of input case) we know that the input transition $P \mid \overline{a}b \stackrel{cd}{\Rightarrow} P_1 \mid \overline{a}b$ is matched either by $Q \mid \overline{a}b \stackrel{cd}{\Rightarrow} Q_1 \mid \overline{a}b$ for some Q_1 such that $P_1 \approx_a Q_1$ and $(P_1 \mid \overline{a}b, Q_1 \mid \overline{a}b) \in R$, or by $Q \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q_1 \mid \overline{a}b$ for some Q_1 such that $P_1 \approx_a Q_1 \mid \overline{c}d$.

- In the first case, by the communication case above (fourth item of output and τ case) we know that $P_1 \mid \overline{a}b \stackrel{\tau}{\Rightarrow} P' \mid \mathbf{0}$ can be matched either by $Q_1 \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q' \mid \mathbf{0}$ such that $P' \mid \mathbf{0} \approx_a Q' \mid \mathbf{0}$, in which case $P \mid \overline{a}b \stackrel{cd}{\Rightarrow} P' \mid \mathbf{0}$ is matched by $Q \mid \overline{a}b \stackrel{cd}{\Rightarrow} Q' \mid \mathbf{0}$; or by $Q_1 \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q' \mid \overline{a}b$ such that $P' \mid \mathbf{0} \approx_a Q' \mid \overline{a}b$, in which case $P \mid \overline{a}b \stackrel{cd}{\Rightarrow} P' \mid \mathbf{0}$ is matched by $Q \mid \overline{a}b \stackrel{cd}{\Rightarrow} Q' \mid \overline{a}b$.
- In the second case, we have $P_1 \approx_a Q_1 \mid \overline{c}d$. Then $Q_1 \mid \overline{c}d$ can simulate the move $P_1 \stackrel{ab}{\Rightarrow} P'$ in two ways:
  1. $Q_1 \mid \overline{c}d \stackrel{ab}{\Rightarrow} Q'$ for some Q' such that $P' \approx_a Q'$. Then there are again two possibilities:
     1a. $Q_1 \mid \overline{c}d \stackrel{ab}{\Rightarrow} Q'$ because $Q_1 \stackrel{ab}{\Rightarrow} Q''$ and $Q' = Q'' \mid \overline{c}d$. In this case $P_1 \mid \overline{a}b \stackrel{\tau}{\Rightarrow} P' \mid \mathbf{0}$ is matched by $Q_1 \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q'' \mid \mathbf{0}$, where $P' \approx_a (Q'' \mid \mathbf{0}) \mid \overline{c}d$ because $P' \mid \mathbf{0} \approx_a Q'$. Thus $P \mid \overline{a}b \stackrel{cd}{\Rightarrow} P' \mid \mathbf{0}$ is matched by $Q \mid \overline{a}b \stackrel{cd}{\Rightarrow} Q'' \mid \mathbf{0}$.
     1b. The transition $Q_1 \mid \overline{c}d \stackrel{ab}{\Rightarrow} Q'$ consumes the output $\overline{c}d$. This can be because $Q_1 \stackrel{ab}{\Rightarrow} Q_2 \stackrel{cd}{\Rightarrow} Q''$ or because $Q_1 \stackrel{cd}{\Rightarrow} Q_2 \stackrel{ab}{\Rightarrow} Q''$. In both cases we have $Q_1 \mid \overline{a}b \stackrel{cd}{\Rightarrow} Q'' \mid \mathbf{0} = Q'$ and thus $Q \mid \overline{a}b \stackrel{cd}{\Rightarrow} Q'$ is the matching move.
  2. $Q_1 \mid \overline{c}d \stackrel{\tau}{\Rightarrow} Q'$ for some Q' such that $P' \approx_a Q' \mid \overline{a}b$. Again, there are two subcases:
     2a. $Q_1 \mid \overline{c}d \stackrel{\tau}{\Rightarrow} Q'$ because $Q_1 \stackrel{\tau}{\Rightarrow} Q''$ and $Q' = Q'' \mid \overline{c}d$. Then $Q_1 \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q'' \mid \overline{a}b$ and thus $Q \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q'' \mid \overline{a}b$ is the matching move, since $P' \approx_a Q' \mid \overline{a}b \approx_a (Q'' \mid \overline{a}b) \mid \overline{c}d$.
     2b. $Q_1 \mid \overline{c}d \stackrel{\tau}{\Rightarrow} Q'$ because $Q_1 \stackrel{cd}{\Rightarrow} Q''$ and $Q' = Q'' \mid \mathbf{0}$. Then $Q_1 \mid \overline{a}b \stackrel{cd}{\Rightarrow} Q'' \mid \overline{a}b$ and thus $Q \mid \overline{a}b \stackrel{cd}{\Rightarrow} Q'' \mid \overline{a}b$ is the matching move, since $P' \approx_a Q' \mid \overline{a}b \approx_a Q'' \mid \overline{a}b$. □

Proposition B.8 *The relation $\approx_a$ is an equivalence relation.*

PROOF. The only nontrivial property is transitivity. We show that the relation $(\approx_a \circ \approx_a)$ is an a-bisimulation. This will imply $(\approx_a \circ \approx_a) \subseteq \approx_a$ and therefore the transitivity of $\approx_a$. Suppose that $P \approx_a T \approx_a Q$. The two interesting cases are:

- $P \stackrel{ab}{\Rightarrow} P'$ and T answers by $T \stackrel{\tau}{\Rightarrow} T'$ with $P' \approx_a T' \mid \overline{a}b$. Then Q must have a transition $Q \stackrel{\tau}{\Rightarrow} Q'$ such that $T' \approx_a Q'$. By lemma B.7 we have then $T' \mid \overline{a}b \approx_a Q' \mid \overline{a}b$ and thus $P' (\approx_a \circ \approx_a) Q' \mid \overline{a}b$ as required.
- $P \stackrel{ab}{\Rightarrow} P'$ and $T \stackrel{ab}{\Rightarrow} T'$ with $P' \approx_a T'$. Now if $T \stackrel{ab}{\Rightarrow} T'$ is matched by $Q \stackrel{ab}{\Rightarrow} Q'$ we are done. So suppose we are in the case where $Q \stackrel{\tau}{\Rightarrow} Q'$ and $T' \approx_a Q' \mid \overline{a}b$. Then we have $P' (\approx_a \circ \approx_a) Q' \mid \overline{a}b$ as required. □

Let $\approx_a^1$ be the variant of $\approx_a$ obtained by replacing $\stackrel{\alpha}{\Rightarrow}$ with $\stackrel{\alpha}{\rightarrow}$ in the hypothesis of the clauses. We show that it is an equivalent formulation for $\approx_a$. It will be used to show that $\approx_a$ coincides with $\approx_1$ and thus with Honda and Tokoro's bisimulation.

Lemma B.9 (simpler formulation of $\approx_a$) $\approx_a = \approx_a^1$.

PROOF.

- $\approx_a \subseteq \approx_a^1$. This is immediate, since $\stackrel{\alpha}{\rightarrow}$ is a particular case of $\stackrel{\alpha}{\Rightarrow}$.

- $\approx_a^1 \subseteq \approx_a$. Let $P \approx_a^1 Q$ and suppose $P \stackrel{\alpha}{\Rightarrow} P'$. We consider first the case where α is an output action or a τ-action:

  - The case $P \stackrel{\tau}{\Rightarrow} P$ is trivial (just take $Q \stackrel{\tau}{\Rightarrow} Q$ as the matching move). Suppose now $P = P_0 \stackrel{\tau}{\rightarrow} \cdots P_i \stackrel{\alpha}{\rightarrow} P_{i+1} \cdots \stackrel{\tau}{\rightarrow} P_n = P'$. Since $P \approx_a^1 Q$ we have then $Q = Q_0 \stackrel{\tau}{\Rightarrow} \cdots Q_i \stackrel{\alpha}{\Rightarrow} Q_{i+1} \cdots \stackrel{\tau}{\Rightarrow} Q_n$, where $P_k \approx_a^1 Q_k$ for each $k = 0, \ldots, n$. In particular $P_n \approx_a^1 Q_n$.

Consider now the case where α is an input action:

  - Let $P = P_0 \stackrel{\tau}{\rightarrow} \cdots P_i \stackrel{ab}{\rightarrow} P_{i+1} \cdots \stackrel{\tau}{\rightarrow} P_n = P'$. Then $Q = Q_0 \stackrel{\tau}{\Rightarrow} \cdots Q_i$ with $P_k \approx_a^1 Q_k$ for each $k = 0, \ldots, i$. Now if $P_i \stackrel{ab}{\rightarrow} P_{i+1}$ is matched by $Q_i \stackrel{ab}{\Rightarrow} Q_{i+1}$ we proceed as above. So suppose we are in the case where $Q_i \stackrel{\tau}{\Rightarrow} Q_{i+1}$ and $P_{i+1} \approx_a^1 Q_{i+1} \mid \overline{a}b$. Then there are two ways in which $Q_{i+1} \mid \overline{a}b$ can match the move $P_{i+1} \stackrel{\tau}{\Rightarrow} P'$:

    - Q_{i+1} moves alone: $Q_{i+1} \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q' \mid \overline{a}b$ because $Q_{i+1} \stackrel{\tau}{\Rightarrow} Q'$. In this case we have $Q \stackrel{\tau}{\Rightarrow} Q'$ and $P' \approx_a^1 Q' \mid \overline{a}b$ as required.
    - Q_{i+1} consumes the output $\overline{a}b$ in a communication step. In this case the sequence $P_{i+1} \stackrel{\tau}{\rightarrow} \cdots P_j \stackrel{\tau}{\rightarrow} P_{j+1} \cdots \stackrel{\tau}{\rightarrow} P_n = P'$ is matched by $Q_{i+1} \mid \overline{a}b \stackrel{\tau}{\Rightarrow} \cdots Q_j \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q_{j+1} \mid \mathbf{0} \cdots \stackrel{\tau}{\Rightarrow} Q' \mid \mathbf{0}$ where $Q_j \stackrel{ab}{\Rightarrow} Q_{j+1}$ and $Q_{j+1} \stackrel{\tau}{\Rightarrow} Q'$. Then we have $Q \stackrel{ab}{\Rightarrow} Q'$ and $P' \approx_a^1 Q' \mid \mathbf{0} \approx_a^1 Q'$, which is the required matching transition. □

Lemma B.10 (simpler formulation of $\approx_1$) $\approx_1 = \approx_1^1$.

PROOF. The only difference between the two definitions is in the output and τ clauses, and the proof for this case goes exactly as for $\approx_a$. □

Remark B.11 $P \approx_1 Q \;\Leftrightarrow\; P \approx_1 (Q \mid \mathbf{0})$.

Proof of theorem 5.7: $\approx_a = \approx_1$.

PROOF. We will use the characterisations of $\approx_a$ and $\approx_1$ as $\approx_a^1$ and $\approx_1^1$ respectively. For the sake of simplicity, we keep the notations $\approx_a$ and $\approx_1$.

- $\approx_a \subseteq \approx_1$. It is immediate to see that $\approx_a$ is a 1-bisimulation, since the output and τ clauses are the same and $P \approx_a Q \;\Rightarrow\; P \mid \overline{a}b \approx_a Q \mid \overline{a}b$ by lemma B.7.

• $\approx_1 \subseteq \approx_a$. We show that $\approx_1$ is an a-bisimulation. Again, there is nothing to prove for the output and τ-clauses. As for the input clause, suppose that $P \stackrel{ab}{\rightarrow} P'$. Then $P \mid \overline{a}b \stackrel{\tau}{\rightarrow} P' \mid \mathbf{0}$. Since $P \approx_1 Q$, by definition of $\approx_1$ also $P \mid \overline{a}b \approx_1 Q \mid \overline{a}b$. Therefore there exists Q' such that $Q \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q'$ and $P' \mid \mathbf{0} \approx_1 Q'$. By remark B.11 we have then $P' \approx_1 Q'$. Now there are three possibilities for the transition $Q \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q'$:

- $Q \mid \overline{a}b$ does not move: $Q' = Q \mid \overline{a}b$ and $P' \approx_1 Q \mid \overline{a}b$. In this case we just take $Q \stackrel{\tau}{\Rightarrow} Q$ and we are in the second case of the input clause of a-bisimulation.
- Q consumes the output $\overline{a}b$: $Q \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q'$ because $Q \stackrel{\tau}{\Rightarrow} Q_1 \stackrel{ab}{\rightarrow} Q_2 \stackrel{\tau}{\Rightarrow} Q''$ and $Q' = Q'' \mid \mathbf{0}$. Then by remark B.11 we have $P' \approx_1 Q''$ as required.
- Q moves alone: $Q \mid \overline{a}b \stackrel{\tau}{\Rightarrow} Q'$ is inferred from $Q \stackrel{\tau}{\Rightarrow} Q_1 \stackrel{\tau}{\rightarrow} Q_2 \stackrel{\tau}{\Rightarrow} Q''$ and $Q' = Q'' \mid \overline{a}b$. Then $P' \approx_1 Q'' \mid \overline{a}b$, and we are again in the second case of the input clause of a-bisimulation. □

Complement to the proof of lemma 5.3.

• Suppose α is a τ or output action, $\sigma P \stackrel{\alpha}{\rightarrow} P'$ and $P \stackrel{\alpha'}{\rightarrow} P_1$, where $\sigma\alpha' = \alpha$ and $\sigma P_1 \equiv P'$.

This is the simplest case (as given by lemma 5.2(2) or 5.2(3.a)). From the hypothesis $P \approx_g Q$ we derive that $Q \stackrel{\alpha'}{\Rightarrow} Q_1$ and $P_1 \approx_g Q_1$. From lemma 5.2(1) it follows that $\sigma Q \stackrel{\alpha}{\Rightarrow} \sigma Q_1$.

• Suppose $\sigma P \stackrel{\tau}{\rightarrow} P'$ and we are not in the previous case. According to lemma 5.2(3) we have to consider two cases:

output: Suppose $P \stackrel{\overline{a}b}{\rightarrow} \cdot \stackrel{dc}{\rightarrow} P_1$, where $P' \sim_a [b/c]\sigma P_1$, c is fresh and $\sigma a = \sigma d$. We have to consider two subcases:

input : Suppose $Q \stackrel{\overline{a}b}{\Rightarrow} \cdot \stackrel{dc}{\Rightarrow} Q_1$ and $P_1 \approx_g Q_1$. This means that $Q \stackrel{\tau}{\Rightarrow} \cdot \stackrel{\overline{a}b}{\rightarrow} \cdot \stackrel{\tau}{\Rightarrow} \cdot \stackrel{dc}{\rightarrow} \cdot \stackrel{\tau}{\Rightarrow} Q_1$. By lemma 2.7(3) we have then $Q \stackrel{\tau}{\Rightarrow} \cdot \stackrel{\overline{a}b}{\rightarrow} \cdot \stackrel{dc}{\rightarrow} \cdot \stackrel{\tau}{\Rightarrow} Q_1$, whence, by lemma 5.2(1), $\sigma Q \stackrel{\tau}{\Rightarrow} \cdot \stackrel{\sigma\overline{a}b}{\rightarrow} \cdot \stackrel{\sigma dc}{\rightarrow} \cdot \stackrel{\tau}{\Rightarrow} \sigma Q_1$. Then, by lemma 2.7(5) we conclude that $\sigma Q \stackrel{\tau}{\Rightarrow} \cdot \sim_a [b/c]\sigma Q_1$.

τ : Let $Q \stackrel{\overline{a}b}{\Rightarrow} \cdot \stackrel{\tau}{\Rightarrow} Q_1$ and $P_1 \approx_g (Q_1 \mid \overline{d}c)$. By lemma 2.7(3) we have $Q \stackrel{\tau}{\Rightarrow} \cdot \stackrel{\overline{a}b}{\rightarrow} Q_1$, and then by lemma 5.2(1) there exists S such that $\sigma Q \stackrel{\tau}{\Rightarrow} S \stackrel{\sigma\overline{a}b}{\rightarrow} \sigma Q_1$. By lemma 2.7(1) we know that $S \sim_a (\sigma Q_1 \mid \sigma\overline{a}b) \equiv [b/c]\sigma(Q_1 \mid \overline{d}c)$. Then $\sigma Q \stackrel{\tau}{\Rightarrow} \cdot \sim_a [b/c]\sigma(Q_1 \mid \overline{d}c)$ is the matching move.

bound output: Suppose $P \stackrel{\overline{a}(b)}{\rightarrow} \cdot \stackrel{dc}{\rightarrow} P_1$, where $P' \sim_a \nu b\,([b/c]\sigma P_1)$, c is fresh and $\sigma a = \sigma d$. As before we have to consider two subcases:

input : Suppose $Q \stackrel{\overline{a}(b)}{\Rightarrow} \cdot \stackrel{dc}{\Rightarrow} Q_1$ and $P_1 \approx_g Q_1$. By the same reasoning as above, using lemmas 2.7(4), 5.2(1) and 2.7(6) we deduce that $\sigma Q \stackrel{\tau}{\Rightarrow} \cdot \sim_a \nu b\,([b/c]\sigma Q_1)$.

τ : Suppose $Q \stackrel{\overline{a}(b)}{\Rightarrow} \cdot \stackrel{\tau}{\Rightarrow} Q_1$ and $P_1 \approx_g (Q_1 \mid \overline{d}c)$. Again, by the same reasoning as above, using lemmas 2.7(4), 5.2(1) and 2.7(2) we deduce that $\sigma Q \stackrel{\tau}{\Rightarrow} \cdot \sim_a \nu b\,([b/c]\sigma(Q_1 \mid \overline{d}c))$.

• The last case to consider is when $\sigma P \stackrel{ab}{\rightarrow} P'$. Then we have $P \stackrel{a'c}{\rightarrow} P_1$ where c is a fresh name, $\sigma a' = a$ and $[b/c]\sigma P_1 \equiv P'$. Again there are two cases:

input : If $Q \stackrel{a'c}{\Rightarrow} Q_1$ and $P_1 \approx_g Q_1$ then $\sigma Q \stackrel{ab}{\Rightarrow} [b/c]\sigma Q_1$.

τ : $Q \stackrel{\tau}{\Rightarrow} Q_1$ and $P_1 \approx_g (Q_1 \mid \overline{a'}c)$. Then the matching move is $\sigma Q \stackrel{\tau}{\Rightarrow} \sigma Q_1$, since $\sigma Q_1 \mid \overline{a}b \equiv [b/c]\sigma(Q_1 \mid \overline{a'}c)$. □

Complement to the proof of theorem 5.9.

We define the tests $R(n, L)$. To this end we introduce an internal choice operator $\oplus$. This is a derived operator defined as follows:

$$P_1 \oplus \cdots \oplus P_n \equiv \nu a\,(a.P_1 \mid \cdots \mid a.P_n \mid \overline{a}) \quad a \notin fn(P_1 \mid \cdots \mid P_n)$$

If $X = \{P_1, \ldots, P_n\}$ is a set of processes then $\oplus X$ is an abbreviation for $P_1 \oplus \cdots \oplus P_n$. We suppose that the collection of channel names Ch has been partitioned in two infinite well-ordered sets Ch' and Ch''. In the following we have $L' \subseteq L \subset_{finite} Ch''$. We also assume the following sequences of distinct names in Ch':

$$\begin{array}{l}\{b_n, b'_n \mid n \in \omega\}\\ \{c_n^\beta \mid n \in \omega \text{ and } \beta \in \{\tau, aa', a, \overline{a}a', \overline{a} \mid a, a' \in Ch''\}\}\\ \{c_n'^\beta \mid n \in \omega \text{ and } \beta \in \{aa', a \mid a, a' \in Ch''\}\}\\ \{d_n^\beta \mid n \in \omega \text{ and } \beta \in \{a \mid a \in Ch''\}\}\\ \{e_n \mid n \in \omega\}\end{array}$$

The test $R(n, L)$ is defined by induction on n as follows, where we pick a'' to be the first name in the well-ordered set $Ch''\backslash L$. When emitting or receiving a name which is not in L we work up to injective substitution to show that $P \approx_a^n Q$.

$$R(0, L) = \overline{b}_0 \oplus \overline{b'}_0$$

$$\begin{array}{l}R(n, L) = \overline{b}_n \oplus \overline{b'}_n \oplus \quad (\text{for } n > 0)\\ \quad (\overline{c}_n^\tau \oplus R(n-1, L)) \oplus\\ \quad \oplus \{\overline{c}_n^{\overline{a}a'} \oplus (\overline{a}a' \mid R(n-1, L)) \mid a, a' \in L\} \oplus\\ \quad \oplus \{\overline{c}_n^{\overline{a}} \oplus \nu a''\,(\overline{a}a'' \mid R(n-1, L \cup \{a''\})) \mid a \in L\} \oplus\\ \quad \oplus \{\overline{c}_n^{aa'} \oplus a(a'').(\overline{c'}_n^{aa'} \oplus ([a'' = a']\overline{d}_n^{a'} \oplus R(n-1, L))) \mid a, a' \in L\} \oplus\\ \quad \oplus \{\overline{c}_n^{a} \oplus a(a'').(\overline{c'}_n^{a} \oplus (\oplus\{[a'' = a']\overline{d}_n^{a'} \mid a' \in L\} \oplus \overline{e}_n \oplus R(n-1, L \cup \{a''\}))) \mid a \in L\}\end{array}$$

• We suppose $n > 0$, $\nu L'\,(P \mid R(n, L)) \stackrel{\bullet}{\approx} \nu L'\,(Q \mid R(n, L))$, and $P \stackrel{\alpha}{\Rightarrow} P'$. We proceed by case analysis on the action α to show that Q can match the action α (in the asynchronous sense).

$\alpha \equiv \tau$ Then:

$$\nu L'\,(P \mid R(n, L)) \stackrel{\tau}{\Rightarrow} \nu L'\,(P \mid (\overline{c}_n^\tau \oplus R(n-1, L)))$$

To match this reduction up to barbed bisimulation we have to have:

$$\nu L'\,(Q \mid R(n, L)) \stackrel{\tau}{\Rightarrow} \nu L'\,(Q_1 \mid (\overline{c}_n^\tau \oplus R(n-1, L)))$$

We make a further reduction on the lhs:

$$\nu L'\,(P \mid (\overline{c}_n^{\tau} \oplus R(n-1,L))) \stackrel{\tau}{\Rightarrow} \nu L'\,(P' \mid R(n-1,L))$$

Again this has to be matched by (note that we cannot run $R(n,L)$ without losing a commitment $\overline{b}_n$ or $\overline{b'}_n$):

$$\nu L'\,(Q_1 \mid (\overline{c}_n^{\tau} \oplus R(n-1,L))) \stackrel{\tau}{\Rightarrow} \nu L'\,(Q' \mid R(n-1,L))$$

We observe $Q \stackrel{\tau}{\Rightarrow} Q_1 \stackrel{\tau}{\Rightarrow} Q'$. We can conclude by applying the inductive hypothesis.

$\alpha \equiv aa'$ We suppose $a' \in L$. Then:

$$\nu L'\,(P \mid R(n,L)) \stackrel{\tau}{\Rightarrow} \nu L'\,(P \mid (\overline{c}_n^{\overline{a}a'} \oplus (\overline{a}a' \mid R(n-1,L))))$$

This has to be matched by:

$$\nu L'\,(Q \mid R(n,L)) \stackrel{\tau}{\Rightarrow} \nu L'\,(Q_1 \mid (\overline{c}_n^{\overline{a}a'} \oplus (\overline{a}a' \mid R(n-1,L))))$$

We make a further reduction on the lhs:

$$\nu L'\,(P \mid (\overline{c}_n^{\overline{a}a'} \oplus (\overline{a}a' \mid R(n-1,L)))) \stackrel{\tau}{\Rightarrow} \nu L'\,(P' \mid R(n-1,L))$$

This is matched by:

$$\nu L'\,(Q_1 \mid (\overline{c}_n^{\overline{a}a'} \oplus (\overline{a}a' \mid R(n-1,L)))) \stackrel{\tau}{\Rightarrow} Q''$$

Now we have two possibilities:

- $Q_1 \stackrel{\tau}{\Rightarrow} Q'$ and $Q'' \equiv \nu L'\,(Q' \mid \overline{a}a' \mid R(n-1,L))$. Then $Q \stackrel{\tau}{\Rightarrow} Q_1 \stackrel{\tau}{\Rightarrow} Q'$ and $P' \approx_a^{n-1} Q' \mid \overline{a}a'$ by inductive hypothesis.
- $Q_1 \stackrel{aa'}{\Rightarrow} Q'$ and $Q'' \equiv \nu L'\,(Q' \mid R(n-1,L))$. Then $Q \stackrel{\tau}{\Rightarrow} Q_1 \stackrel{aa'}{\Rightarrow} Q'$ and $P' \approx_a^{n-1} Q'$ by inductive hypothesis.

$\alpha \equiv aa''$ We suppose $a'' \notin L$. Up to an injective substitution we may suppose a'' is the first name in $Ch''\backslash L$. Then:

$$\nu L'\,(P \mid R(n,L)) \stackrel{\tau}{\Rightarrow} \nu L'\,(P \mid \overline{c}_n^{\overline{a}} \oplus \nu a''\,(\overline{a}a'' \mid R(n-1,L \cup \{a''\})))$$

This has to be matched by:

$$\nu L'\,(Q \mid R(n,L)) \stackrel{\tau}{\Rightarrow} \nu L'\,(Q_1 \mid \overline{c}_n^{\overline{a}} \oplus \nu a''\,(\overline{a}a'' \mid R(n-1,L \cup \{a''\})))$$

We make a further reduction on the lhs:

$$\nu L'\,(P \mid \overline{c}_n^{\overline{a}} \oplus \nu a''\,(\overline{a}a'' \mid R(n-1,L \cup \{a''\}))) \stackrel{\tau}{\Rightarrow} \nu L' \cup \{a''\}\,(P' \mid R(n-1,L \cup \{a''\}))$$

This is matched by:

$$\nu L'\,(Q_1 \mid \overline{c}_n^{\overline{a}} \oplus \nu a''\,(\overline{a}a'' \mid R(n-1,L \cup \{a''\}))) \stackrel{\tau}{\Rightarrow} Q''$$

As in the previous case we have two possibilities:

- $Q_1 \stackrel{\tau}{\Rightarrow} Q'$ and $Q'' \equiv \nu L' \cup \{a''\}\,(Q' \mid \overline{a}a'' \mid R(n-1, L \cup \{a''\}))$. Then $Q \stackrel{\tau}{\Rightarrow} Q_1 \stackrel{\tau}{\Rightarrow} Q'$ and $P' \approx_a^{n-1} Q' \mid \overline{a}a''$ by inductive hypothesis.
- $Q_1 \stackrel{aa''}{\Rightarrow} Q'$ and $Q'' \equiv \nu L' \cup \{a''\}\,(Q' \mid R(n-1, L \cup \{a''\}))$. Then $Q \stackrel{\tau}{\Rightarrow} Q_1 \stackrel{aa''}{\Rightarrow} Q'$ and $P' \approx_a^{n-1} Q'$ by inductive hypothesis.

$\alpha \equiv \overline{a}a'$ We may suppose $a' \in L$. Then:

$$\nu L'\,(P \mid R(n, L)) \stackrel{\tau}{\Rightarrow} \nu L'\,(P \mid \overline{c}_n^{aa'} \oplus a(a'').(\overline{c'}_n^{aa'} \oplus ([a'' = a']\overline{d}_n^{a'} \oplus R(n-1, L))))$$

This has to be matched by:

$$\nu L'\,(Q \mid R(n, L)) \stackrel{\tau}{\Rightarrow} \nu L'\,(Q_1 \mid \overline{c}_n^{aa'} \oplus a(a'').(\overline{c'}_n^{aa'} \oplus ([a'' = a']\overline{d}_n^{a'} \oplus R(n-1, L))))$$

We make a further move on the lhs:

$$\begin{array}{l}\nu L'\,(P \mid \overline{c}_n^{aa'} \oplus a(a'').(\overline{c'}_n^{aa'} \oplus ([a'' = a']\overline{d}_n^{a'} \oplus R(n-1, L)))) \stackrel{\tau}{\Rightarrow} \\ \nu L'\,(P' \mid ([a' = a']\overline{d}_n^{a'} \oplus R(n-1, L)))\end{array}$$

This has to be matched by (we have to lose the $\overline{c'}_n^{aa'}$ commitment while keeping the $\overline{d}_n^{a'}, \overline{b}_{n-1}, \overline{b'}_{n-1}$ commitments):

$$\begin{array}{l}\nu L'\,(Q_1 \mid \overline{c}_n^{aa'} \oplus a(a'').(\overline{c'}_n^{aa'} \oplus ([a'' = a']\overline{d}_n^{a'} \oplus R(n-1, L)))) \stackrel{\tau}{\Rightarrow} \\ \nu L'\,(Q_2 \mid ([a' = a']\overline{d}_n^{a'} \oplus R(n-1, L)))\end{array}$$

We note $Q_1 \stackrel{\overline{a}a'}{\Rightarrow} Q_2$. We take a further step on the lhs:

$$\nu L'\,(P' \mid ([a' = a']\overline{d}_n^{a'} \oplus R(n-1, L))) \stackrel{\tau}{\Rightarrow} \nu L'\,(P' \mid R(n-1, L))$$

This has to be matched by:

$$\nu L'\,(Q_2 \mid ([a' = a']\overline{d}_n^{a'} \oplus R(n-1, L))) \stackrel{\tau}{\Rightarrow} \nu L'\,(Q' \mid R(n-1, L))$$

Now we observe $Q \stackrel{\tau}{\Rightarrow} Q_1 \stackrel{\overline{a}a'}{\Rightarrow} Q_2 \stackrel{\tau}{\Rightarrow} Q'$ and we apply the inductive hypothesis to conclude $P' \approx_a^{n-1} Q'$.

$\alpha \equiv \overline{a}(a'')$ We may suppose a'' is the first element in $Ch'' \backslash L$ (otherwise we rename and use an injective substitution). Then:

$$\begin{array}{l}\nu L'\,(P \mid R(n, L)) \stackrel{\tau}{\Rightarrow} \\ \nu L'\,(P \mid \overline{c}_n^{a} \oplus a(a'').(\overline{c'}_n^{a} \oplus (\oplus\{[a'' = a']\overline{d}_n^{a'} \mid a' \in L\} \oplus \overline{e}_n \oplus R(n-1, L \cup \{a''\}))))\end{array}$$

This has to be matched by:

$$\begin{array}{l}\nu L'\,(Q \mid R(n, L)) \stackrel{\tau}{\Rightarrow} \\ \nu L'\,(Q_1 \mid \overline{c}_n^{a} \oplus a(a'').(\overline{c'}_n^{a} \oplus (\oplus\{[a'' = a']\overline{d}_n^{a'} \mid a' \in L\} \oplus \overline{e}_n \oplus R(n-1, L \cup \{a''\}))))\end{array}$$

We take a further step on the lhs:

$$\begin{array}{l}\nu L'\,(P \mid \overline{c}_n^{a} \oplus a(a'').(\overline{c'}_n^{a} \oplus (\oplus\{[a'' = a']\overline{d}_n^{a'} \mid a' \in L\} \oplus \overline{e}_n \oplus R(n-1, L \cup \{a''\})))) \stackrel{\tau}{\Rightarrow} \\ \nu L' \cup \{a''\}\,(P' \mid \oplus\{[a'' = a']\overline{d}_n^{a'} \mid a' \in L\} \oplus \overline{e}_n \oplus R(n-1, L \cup \{a''\}))\end{array}$$

This has to be matched by (we reason as in the previous case and note that the name sent by Q cannot be in L):

$$\nu L'\,(Q_1 \mid \overline{c}^{a}_{n} \oplus a(a'').(\overline{c'}^{a}_{n} \oplus (\oplus\{[a''=a']\overline{d}^{a'}_{n} \mid a' \in L\} \oplus \overline{e}_n \oplus R(n-1, L\cup\{a''\})))) \stackrel{\tau}{\Rightarrow}$$
$$\nu L' \cup \{a''\}\,(Q_2 \mid (\oplus\{[a''=a']\overline{d}^{a'}_{n} \mid a' \in L\} \oplus \overline{e}_n \oplus R(n-1, L\cup\{a''\})))$$

We note $Q_1 \stackrel{\overline{a}(a'')}{\Rightarrow} Q_2$. We take a last step on the lhs:

$$\nu L' \cup \{a''\}\,(P' \mid \oplus\{[a''=a']\overline{d}^{a'}_{n} \mid a' \in L\} \oplus \overline{e}_n \oplus R(n-1, L\cup\{a''\})) \stackrel{\tau}{\Rightarrow}$$
$$\nu L' \cup \{a''\}\,(P' \mid R(n-1, L\cup\{a''\}))$$

This has to be matched by:

$$\nu L' \cup \{a''\}\,(Q_2 \mid (\oplus\{[a''=a']\overline{d}^{a'}_{n} \mid a' \in L\} \oplus \overline{e}_n \oplus R(n-1, L\cup\{a''\}))) \stackrel{\tau}{\Rightarrow}$$
$$\nu L' \cup \{a''\}\,(Q' \mid R(n-1, L\cup\{a''\}))$$

We conclude by observing that $Q \stackrel{\tau}{\Rightarrow} Q_1 \stackrel{\overline{a}(a'')}{\Rightarrow} Q_2 \stackrel{\tau}{\Rightarrow} Q'$ and $P' \approx_a^{n-1} Q'$ by inductive hypothesis.

- In the strong case we can simulate matching with synchronization by replacing $[a''=a']\overline{d}^{a'}_{n}$ with $a''.\overline{f_n} \mid \overline{a'}.\overline{d}^{a'}_{n}$, where $\{f_n \mid n \in \omega\}$ is yet another sequence of names in Ch'. □

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399