A Calculus of Mobile Processes, Part II

Robin Milner, University of Edinburgh, Scotland
Joachim Parrow, Swedish Institute of Computer Science, Kista, Sweden
David Walker, University of Technology, Sydney, Australia

June 1989 (Revised October 1990)

Running title: Calculus of Mobile Processes, Part II

Address for proofs: Joachim Parrow, SICS, Box 1263, S-16428 Kista, Sweden.

Special symbols: Since this copy is mathematically type-set, only a few of the less obvious symbols are listed below.

$\mathbf{0}$	Boldface zero
$\emptyset$	Empty set
$\alpha\ \beta\ \pi\ \sigma\ \tau\ \omega$	Greek letters
$\equiv_\alpha\ \equiv\ =\ \stackrel{\text{def}}{=}\ \sim\ \ \dot{\sim}\ \ \dot{\not\sim}\ \ \sim_D\ \ \dot{\sim}'\ \ \sim'$	Various relation symbols
$\mathcal{N}$	The set of names
$\mathcal{S}\ \mathcal{S}^*\mathcal{S}_n\ \ldots$	Bisimulation symbols
$\xrightarrow{\alpha}\ \xrightarrow{\tau}\ \xrightarrow{\overline{x}y}\ \xrightarrow{x(y)}\ \xrightarrow{\overline{x}(y)}\ \ldots$	Transition arrows
$\Uparrow\ \Downarrow\ \Updownarrow$	Other arrows
$\bigcup_{n<\omega}$	Indexed union (note: Greek omega)
$\cup\ \cap\ \times\ \setminus\ \subseteq\ \upharpoonright$	Operations on sets etc.
$:=$	Replacement symbol
$::=$	Syntax definition symbol
A C0 . . . SGE D	Boldface names of axioms etc.
INPUT-ACT PAR CLOSE . . .	SMALL CAPS names of inference rules

Abstract

This is the second of two papers in which we present the π-calculus, a calculus of mobile processes. We provide a detailed presentation of some of the theory of the calculus developed to date, and in particular we establish most of the results stated in the companion paper.

Introduction

This is the second of two papers in which we present the π-calculus, a calculus of mobile processes. The companion paper [2] contains an introduction to the calculus through a sequence of examples, together with statements of many results about it. The purpose of the present paper is to provide a detailed presentation of some of the theory of the calculus developed to date, and in particular to establish most of the results stated in the companion paper. Once the motivation and intuition for the π-calculus are understood, with the help of [2], the present paper serves as a self-contained development of the theory. To achieve this we have found it necessary to repeat some material from the companion paper.

Section 1 contains a description of the syntax of agents and a discursive presentation of the transitional semantics. In Section 2 we present and motivate the definitions of strong bisimulation and strong bisimilarity, strong equivalence, and a useful family of indexed equivalences. Section 3 contains a series of properties of strong bisimilarity, while properties of strong equivalence and indexed equivalences are developed in Section 4. A complete axiomatization for finite agents is presented in Section 5.

There are many points of interest in the detailed development of the theory. However, in order to reduce the length of the paper and to avoid giving the impression that the theory generally is more complicated or surprising than it in fact is, we do not include complete proofs of all results. Instead, the Appendix contains extracts giving a taste of the techniques used. Complete proofs may be found in [3].

1 Agents and their transitional semantics

1.1 Agents

We first recapitulate some of the definitions and the notation from our companion paper. Assume an infinite set $\mathcal{N}$ of *names* and use x, y, z, w, v, u as metavariables over names. We assume also a set of *agent identifiers*. Each agent identifier A has a nonnegative *arity*.

Definition 1 The set of *agents* is defined as follows (we use P, Q, R as metavariables over agents):

$$
\begin{array}{rcl}
P & ::= & \mathbf{0} \\
 & \mid & \overline{x}y.\,P \\
 & \mid & x(y).\,P \\
 & \mid & \tau.\,P \\
 & \mid & (x)P \\
 & \mid & [x\!=\!y]P \\
 & \mid & P \mid Q \\
 & \mid & P + Q \\
 & \mid & A(y_1, \ldots, y_n)
\end{array}
$$

Here $\mathbf{0}$ is a nullary operator, $\overline{x}y.$, $x(y).$, $\tau.$, (x) and $[x\!=\!y]$ are unary operators, $\mid$ and $+$ are binary operators, and n is the arity of A. □

The order of precedence among the operators is the order listed above. For a description of the intended interpretation of agents see [2]. In that paper we also use a general summation operator Σ; in the present paper we will be satisfied with nullary and binary summation ($\mathbf{0}$ and $+$) and regard general summation as a derived operator.

Definition 2 In each agent of one of the forms $x(y).\,P$ and $(y)P$ the occurrence of y within parentheses is a *binding occurrence*, and in each case the *scope* of the occurrence is P. An occurrence of y in an agent is said to be *free* if it does not lie within the scope of a binding occurrence of y. The set of names occurring free in P is denoted $\mathrm{fn}(P)$. We sometimes write $\mathrm{fn}(P, Q, \ldots, x, y, \ldots)$ as an abbreviation for $\mathrm{fn}(P) \cup \mathrm{fn}(Q) \cup \ldots \cup \{x, y, \ldots\}$. □

Definition 3 A *defining equation* for an agent identifier A of arity n is of the form

$$A(x_1, \ldots, x_n) \stackrel{\mathrm{def}}{=} P$$

where the x_i are pairwise distinct and $\mathrm{fn}(P) \subseteq \{x_1, \ldots, x_n\}$. □

In the following we assume that each agent identifier A has a unique defining equation.

Definition 4 An occurrence of a name in an agent is said to be *bound* if it is not free. We assume that the set of *bound names of* P, $\mathrm{bn}(P)$, is defined in such a way that it contains all names which occur bound in P and that if $A(\tilde{x}) \stackrel{\mathrm{def}}{=} Q$ then $\mathrm{bn}(A(\tilde{x})) = \mathrm{bn}(Q)$, where $\tilde{x} = x_1, \ldots, x_n$. We write $\mathrm{n}(P)$ for the set $\mathrm{fn}(P) \cup \mathrm{bn}(P)$ of *names of* P. □

To avoid pathological technical difficulties we further assume that the family of defining equations of agent identifiers is such that for each identifier A, $\mathrm{bn}(A(\tilde{x}))$ is finite.

Definition 5 A *substitution* is a function σ from $\mathcal{N}$ to $\mathcal{N}$ which is almost everywhere the identity. If $x_i\sigma = y_i$ for all i with $1 \leq i \leq n$ (and $x\sigma = x$ for all other names x), we sometimes write $\{y_1/x_1, \ldots, y_n/x_n\}$ or $\{\tilde{y}/\tilde{x}\}$ for σ. □

Definition 6 $P\sigma$ denotes the agent obtained from P by simultaneously substituting $z\sigma$ for each free occurrence of z in P for each z, with change of bound names to avoid captures. In particular the following hold where $\equiv$ denotes syntactic identity:

$$\begin{array}{rcll} (x(y).\,P)\sigma & \equiv & x\sigma(y').\,P\{y'/y\}\sigma & \text{where } y' \notin \mathrm{fn}((y)P, P\sigma) \text{ and } y'\sigma = y' \\ ((y)P)\sigma & \equiv & (y')P\{y'/y\}\sigma & \text{where } y' \notin \mathrm{fn}((y)P, P\sigma) \text{ and } y'\sigma = y' \end{array}$$

□

Definition 7 The symbol $\equiv_\alpha$ denotes the relation of *alpha-convertibility* on agents defined in the standard way. (The subscript α here bears no relation to the actions α defined below.) □

1.2 Actions

Precisely as in CCS [1], a *transition* in the π-calculus is of the form

$$P \xrightarrow{\alpha} Q$$

Intuitively, this transition means that P can evolve into Q, and in doing so perform the *action* α. In our calculus there will be four kinds of action α as follows:

1. The *silent* action τ. As in CCS, $P \xrightarrow{\tau} Q$ means that P can evolve into Q, and in doing so requires no interaction with the environment. Silent actions can naturally arise from agents of form $\tau.P$, but also from communications within an agent.

2. A *free output* action $\overline{x}y$. The transition $P \xrightarrow{\overline{x}y} Q$ implies that P can emit the free name y on the port $\overline{x}$. Free output actions arise from the output prefix form $\overline{x}y.P$.

3. An *input* action $x(y)$. Intuitively, $P \xrightarrow{x(y)} Q$ means that P can receive *any* name w on the port x, and then evolve into $Q\{w/y\}$. Note that this departs slightly from CCS, where an input action contains the actual received value. Here, (y) instead represents a reference to the place where the received name will go; y is enclosed in brackets in order to stress this fact. Input actions arise from the input prefix form $x(y).P$.

α	Kind	Free/Bound	Polarity	$\mathrm{fn}(\alpha)$	$\mathrm{bn}(\alpha)$
τ	Silent	f	0	$\emptyset$	$\emptyset$
$\overline{x}y$	Free Output	f	$-$	$\{x, y\}$	$\emptyset$
$x(y)$	Input	b	$+$	$\{x\}$	$\{y\}$
$\overline{x}(y)$	Bound Output	b	$-$	$\{x\}$	$\{y\}$

Table 1: The actions.

4. A *bound output* action $\overline{x}(y)$. This kind of action has no counterpart in CCS. Intuitively, $P \xrightarrow{\overline{x}(y)} Q$ means that P emits a private name (i.e. a name bound in P) on the port $\overline{x}$, and (y) is a reference to where this private name occurs. As in the input action above, y is enclosed in brackets to emphasize that it is a reference and does not represent a free name. Bound output actions arise from free output actions which carry names out of their scope, as e.g. in the agent $(y)\overline{x}y.P$.

The silent action and free output actions will collectively be called *free* actions, while input actions and bound output actions will be called *bound* actions. Thus, the bound actions carry "references" rather than values; these references are in the form of names within brackets.

The free output and bound output actions will collectively be called *output* actions, or sometimes *negative* actions (actions of negative polarity). Similarly, the input actions will be called *positive* actions (actions of positive polarity). Two actions must be of opposite polarity in order to combine into an internal communication.

In the output and input actions mentioned above, x is the *subject* and y the *object* or *parameter*. The object is said to be *bound* in the bound actions and *free* in the free actions. The set of *bound names* $\mathrm{bn}(\alpha)$ of an action α is the empty set if α is a free action; otherwise it contains just the bound object of α. The set of *free names* $\mathrm{fn}(\alpha)$ of α contains the subject and free object (if any) of α, and the *names* $\mathrm{n}(\alpha)$ of α is the union of $\mathrm{bn}(\alpha)$ and $\mathrm{fn}(\alpha)$. Note that $\mathrm{n}(\tau) = \emptyset$. A summary of these definitions appears in Table 1.

1.3 Transitions

We now proceed to define the transition relations $\xrightarrow{\alpha}$ on agents.

Definition 8 The *transition relations* are the smallest relations satisfying the rules of action in Table 2. $\square$

This definition has the same structure as the corresponding definition in CCS. However, the details differ to a considerable extent. Briefly stated, the differences between CCS and the present calculus emanate from the restriction

$$\text{TAU-ACT}: \quad \frac{-}{\tau.P \xrightarrow{\tau} P} \qquad\qquad \text{OUTPUT-ACT}: \quad \frac{-}{\overline{x}y.P \xrightarrow{\overline{x}y} P}$$

$$\text{INPUT-ACT}: \quad \frac{-}{x(z).P \xrightarrow{x(w)} P\{w/z\}} \quad w \notin \mathrm{fn}((z)P)$$

$$\text{SUM}: \quad \frac{P \xrightarrow{\alpha} P'}{P+Q \xrightarrow{\alpha} P'} \qquad\qquad \text{MATCH}: \quad \frac{P \xrightarrow{\alpha} P'}{[x=x]P \xrightarrow{\alpha} P'}$$

$$\text{IDE}: \quad \frac{P\{\tilde{y}/\tilde{x}\} \xrightarrow{\alpha} P'}{A(\tilde{y}) \xrightarrow{\alpha} P'} \quad A(\tilde{x}) \stackrel{\mathrm{def}}{=} P$$

$$\text{PAR}: \quad \frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q} \quad \mathrm{bn}(\alpha) \cap \mathrm{fn}(Q) = \emptyset$$

$$\text{COM}: \quad \frac{P \xrightarrow{\overline{x}y} P' \quad Q \xrightarrow{x(z)} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'\{y/z\}} \qquad\qquad \text{CLOSE}: \quad \frac{P \xrightarrow{\overline{x}(w)} P' \quad Q \xrightarrow{x(w)} Q'}{P \mid Q \xrightarrow{\tau} (w)(P' \mid Q')}$$

$$\text{RES}: \quad \frac{P \xrightarrow{\alpha} P'}{(y)P \xrightarrow{\alpha} (y)P'} \quad y \notin \mathrm{n}(\alpha) \qquad \text{OPEN}: \quad \frac{P \xrightarrow{\overline{x}y} P'}{(y)P \xrightarrow{\overline{x}(w)} P'\{w/y\}} \quad \begin{array}{l} y \neq x \\ w \notin \mathrm{fn}((y)P') \end{array}$$

Table 2: Rules of Action. Rules involving the binary operators $+$ and $\mid$ additionally have symmetric forms.

operator (x), which in the present calculus restricts the scope of both action subjects and action objects. It is worth noting that the complication over CCS comes from the ability to restrict the scope of action objects, and not primarily from the fusion of "port names" with "data values". We will here explain this issue.

1.3.1 Communicating Free Names

To begin, consider the usual CCS rules for deriving an internal communication. These are:

$$\frac{-}{\overline{a}v.P \xrightarrow{\overline{a}v} P} \qquad \frac{-}{a(x).P \xrightarrow{av} P\{v/x\}} \qquad \frac{P \xrightarrow{\overline{a}v} P' \quad Q \xrightarrow{av} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$$

Thus, the CCS value variable x is instantiated to a value v when inferring an action from $a(x).P$; the rule admits an instantiation to any such value, and hence the agent $a(x).P$ can combine with any output transition in the communication rule. We call this scheme *early instantiation*, since variables are instantiated at the time of inferring the input transition.

Although rules representing early instantiation can be given for the π-calculus we instead adopt a scheme of *late instantiation*, where the input actions contain bound objects which become instantiated only when inferring an internal communication. Our reason is simply that this will admit a notion of equivalence for which the algebraic theory appears somewhat simpler; we defer the treatment of early instantiation to a forthcoming paper. The late instantiation scheme in the π-calculus is represented by the rules OUTPUT-ACT, INPUT-ACT and COM in Table 2. We have explored a number of alternative rules, but they all seem to be essentially equivalent. Notice that scope intrusions resulting from COM if y occurs bound in Q' are properly taken care of since a bound y is renamed in the substitution $Q'\{y/z\}$ (cf. Definition 6).

However, bound objects require careful treatment: a bound object is essentially a reference to locations within an agent, and it is important that such references are maintained in all rules of action. The problematic rule in this respect is one of the usual CCS rules for parallel composition

$$\frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q} \tag{1}$$

The corresponding rule in the π-calculus is PAR in Table 2; it is different only in that it has a side condition $\mathrm{bn}(\alpha) \cap \mathrm{fn}(Q) = \emptyset$. To see that this condition is needed consider a transition $P \xrightarrow{x(z)} P'$. Here z is a reference to locations in P'; the intuition is that in a subsequent communication a name will be received and substituted for the z:s in P'. But if z also occurs free in Q, then in the conclusion of (1) the bound object z will refer to additional locations within Q. A subsequent communication will then substitute not only the z:s in P' but also the free z:s in Q. For example, from INPUT-ACT, (1) and COM we can derive the obviously incorrect transition

$$(x(z).P \mid Q) \mid \overline{x}y.R \stackrel{\tau}{\longrightarrow} (P \mid Q)\{y/z\} \mid R \qquad (2)$$

This transition is incorrect since the free name z in Q is only accidentally the same as the bound name z in $x(z).P$. For this reason we require in PAR that (1) can only be applied when a name bound in α does not occur free in Q. This also explains why INPUT-ACT cannot be simplified to the following rule:

$$\frac{-}{x(z).P \stackrel{x(z)}{\longrightarrow} P}$$

With this simpler rule the side condition in PAR would prevent *all* input transitions from e.g. $x(z).P|\overline{z}y.Q$. The change of bound name in INPUT-ACT is harmless since bound names represent references to places within an agent. Clearly, if w does not occur free in P, then w refers to the same places in $P\{w/z\}$ as z refers to in P. So we allow any such w (and also z itself) to stand for z. Instead of the incorrect (2) we can now correctly infer

$$(x(z).P \mid Q) \mid \overline{x}y.R \stackrel{\tau}{\longrightarrow} (P\{w/z\} \mid Q)\{y/w\} \mid R$$

The side condition in INPUT-ACT ensures that $w = z$ or $w \notin \mathrm{fn}(P)$, and the side condition in PAR ensures that $w \notin \mathrm{fn}(Q)$. Hence the agent after $\stackrel{\tau}{\longrightarrow}$ can be simplified to

$$(P\{y/z\} \mid Q) \mid R$$

which is the expected result of the communication.

1.3.2 Communicating bound names

The rules in the π-calculus must accommodate scope extrusions, as for example in

$$(y)\overline{x}y.P \mid x(z).Q \stackrel{\tau}{\longrightarrow} (y)(P \mid Q\{y/z\}) \qquad (3)$$

Note that we expect this transition to be correct only if either y is z or y is not free in Q: otherwise the restriction (y) in $(y)(P \mid Q\{y/z\})$ will bind occurrences of names in Q which are only accidentally related to the extrusion. If this requirement is not fulfilled, we expect an alpha-conversion of the bound y in the resulting agent:

$$(y)\overline{x}y.P \mid x(z).Q \stackrel{\tau}{\longrightarrow} (y')(P\{y'/y\} \mid Q\{y'/z\}) \qquad (4)$$

where y' is a fresh name.

We achieve the desired effect with two additional rules of action, OPEN and CLOSE, which have no counterparts in CCS. The *scope opening* rule OPEN transforms a free output action to a bound output action, and removes one restriction operator. The fact that y was bound is now represented in the action, which contains a reference to the places where this bound y occurred. Since the objects of bound output actions represent references, they must also obey the side condition in the rule PAR: a bound object may not occur free in Q in that rule. Therefore we allow a renaming in OPEN just as in the input prefix rule: the particular name representing the reference is unimportant as long as it refers to the same locations in P'. Note that the side condition ensures $y \neq x$, so the subject in the output action cannot be the same as the restricted name.

In the *scope closing* rule CLOSE, a bound output action combines with an input action. Intuitively, the rule means that the bound object is received, and then the restriction of this bound name must reappear: that name is still private although its scope has grown. Note that since both INPUT-ACT and OPEN allow an almost arbitrary choice of bound names, the two premises of CLOSE can use the same bound name without any loss of generality.

As an example of deriving a scope extrusion, consider again (3). We have from OPEN that

$$(y)\overline{x}y.P \xrightarrow{\overline{x}(w)} P\{w/y\}$$

for all w such that $w = y$ or $w \notin \mathrm{fn}(P)$. From INPUT-ACT we have that

$$x(z).Q \xrightarrow{x(w)} Q\{w/z\}$$

for all w such that $w = z$ or $w \notin \mathrm{fn}(Q)$. Applying the scope closing rule we get

$$(y)\overline{x}y.P \mid x(z).Q \xrightarrow{\tau} (w)(P\{w/y\} \mid Q\{w/z\})$$

for all w satisfying both the side conditions. If additionally $y = z$ or y does not occur free in Q, then y itself satisfies the accumulated conditions on w. We can then choose y instead of w in this derivation, so the final agent becomes

$$(y)(P \mid Q\{y/z\})$$

which is precisely the agent in (3). If $y \neq z$ and y is free in Q, then the side condition in the closing rule prevents this derivation, but we can always choose a fresh name y' in place of w and obtain precisely the transition (4).

2 Strong bisimilarity and equivalence

2.1 Strong bisimilarity

We will here present and motivate the definition of strong bisimilarity in the π-calculus. It is helpful to first recapitulate ordinary CCS, where the strong equivalence may be defined through simulations: a binary relation $\mathcal{S}$ is a simulation if $P\mathcal{S}Q$ implies that

$$\text{If } P \xrightarrow{\alpha} P' \text{ then for some } Q',\ Q \xrightarrow{\alpha} Q' \quad \text{and} \quad P'\mathcal{S}Q' \tag{5}$$

In other words, any transition from P must be simulated by a transition from Q, such that the derivatives P' and Q' remain in the simulation. A binary relation $\mathcal{S}$ is a bisimulation if both $\mathcal{S}$ and its inverse are simulations. Strong equivalence on agents is defined as the largest bisimulation.

We will apply the same idea to the π-calculus. The main modification is that we must take special account of actions with bound objects. For example, if $z \notin \text{fn}(R, x)$ we obviously want the following agents

$$\begin{aligned} P &\equiv x(y).R \\ Q &\equiv (z)x(y).R \end{aligned}$$

to be bisimilar, even though P has an input transition $\xrightarrow{x(z)}$ which Q cannot simulate exactly. The reason that this difference between P and Q is unimportant is that Q (and P) have other transitions $\xrightarrow{x(w)}$ which only differ in the choice of the bound name w. A bound object is merely a reference to locations within an agent, and the particular name used for this reference is unimportant — an external observer cannot observe the identity of the bound name. So, for the purpose of defining bisimilarity, we will only consider bound objects which are completely fresh, i.e. do not occur in any of the agents to be compared. Recalling the rules of the previous section the limitation to use fresh bound objects is harmless: for any transition with a bound object there is a corresponding transition where the object is suitably fresh (cf. also Lemma 2 in Section 3.1 below).

Another important point is that in order to simulate an input action, it is *not* sufficient that the derivatives P' and Q' continue to simulate. Intuitively, an object in an input action is a placeholder for something to be received, and can become instantiated to an arbitrary name. We thus require that P' and Q' continue to simulate for all *instantiations* of the object in the input action. These considerations lead to the following definition:

Definition 9 A binary relation $\mathcal{S}$ on agents is a *(strong) simulation* if it satisfies the requirements in Table 3. The relation $\mathcal{S}$ is a *(strong) bisimulation*

$\mathcal{S}$ is a *simulation* if $P\mathcal{S}Q$ implies that 1. If $P \xrightarrow{\alpha} P'$ and α is a free action, then for some Q', $\quad Q \xrightarrow{\alpha} Q'$ and $P'\mathcal{S}Q'$ 2. If $P \xrightarrow{x(y)} P'$ and $y \notin \mathrm{n}(P,Q)$, then for some Q', $\quad Q \xrightarrow{x(y)} Q'$ and for all w, $P'\{w/y\}\mathcal{S}Q'\{w/y\}$ 3. If $P \xrightarrow{\overline{x}(y)} P'$ and $y \notin \mathrm{n}(P,Q)$, then for some Q', $\quad Q \xrightarrow{\overline{x}(y)} Q'$ and $P'\mathcal{S}Q'$

Table 3: Definition of (Strong) Simulation.

if both $\mathcal{S}$ and its inverse are simulations. The relation $\dot{\sim}$, *(strong) bisimilarity*, on agents is defined by $P \mathrel{\dot{\sim}} Q$ if and only if there exists a bisimulation $\mathcal{S}$ such that $P\mathcal{S}Q$. $\Box$

It is straightforward to verify that $\dot{\sim}$ is a bisimulation and hence the largest bisimulation.

Note that requirement (5) applies only to free actions α (clause 1), while other requirements are associated with the bound actions. Also, note that the clauses for input and bound output actions are different. In order to simulate an input transition, clause 2 requires Q to have a similar transition such that the derivatives P' and Q' continue to simulate for all instantiations w of the bound objects. On the other hand, the bound output transition in clause 3 intuitively means that P can emit a private name, and (y) refers to the places where this private name used to occur. In order to simulate such a transition Q should similarly emit a private name and continue to simulate P'. This is sufficient, since the bound object y cannot become instantiated through an interaction with the environment.

As an example consider the following equation, where we abbreviate $\overline{x}v$ to $\overline{x}$, and $y(u)$ to y, and omit a trailing $.\mathbf{0}$:

$$\overline{x} \mid y \quad \dot{\sim} \quad \overline{x}.y + y.\overline{x} \tag{6}$$

This equation is true when $x \neq y$ and $u \neq v$, since then any transition by the left hand side can be simulated by a transition of the right hand side, and vice versa. On the other hand,

$$\overline{x} \mid x \quad \dot{\not\sim} \quad \overline{x}.x + x.\overline{x} \tag{7}$$

since the left hand side has an additional τ-transition. It follows that $\dot{\sim}$ is not in general preserved by substitutions of names. (This is not surprising;

in CCS strong equivalence is also not in general preserved by substitution of port names, for the same reason.) It also follows that the equation

$$(y)\overline{z}y.(\overline{x} \mid y) \quad \dot{\sim} \quad (y)\overline{z}y.(\overline{x}.y + y.\overline{x})$$

is true, since a bound output transition of the left hand side can be simulated by a bound output transition of the right hand side, and vice versa. Note that the bound objects in these transitions cannot be x, since x occurs on both sides of the equation. In contrast,

$$z(y).(\overline{x} \mid y) \quad \dot{\not\sim} \quad z(y).(\overline{x}.y + y.\overline{x})$$

since clause 2 requires the derivatives of the leading input transitions to be similar for *all* instances of y, and they are not similar when y is instantiated to x. It follows that strong bisimilarity is not preserved by input prefix.

2.2 Strong equivalence and distinctions

Since strong bisimilarity is not preserved by substitution of free names we will sometimes refer to it as *(strong) ground equivalence*; this can be thought of as equivalence under the assumption that different names will not be identified, i.e. names behave as *constants*. It is then natural to consider the finer equivalence obtained as bisimilarity under all substitutions of names:

Definition 10 P and Q are *(strongly) equivalent*, written $P \sim Q$, if $P\sigma \,\dot{\sim}\, Q\sigma$ for all substitutions σ.

Thus (6) does not hold for strong equivalence; instead we have the more general

$$\overline{x} \mid y \quad \sim \quad \overline{x}.y + y.\overline{x} + [x\!=\!y]\tau$$

In a sense, for the purpose of strong equivalence names behave as *variables* in that equivalence must hold for all instantiations of free names. As pointed out in our companion paper there is a spectrum of equivalences between $\dot{\sim}$ and $\sim$ depending on which names may be assumed to be distinct:

Definition 11 A *distinction* is a symmetric irreflexive relation between names. We shall let D range over distinctions. A substitution σ *respects* a distinction D if, for all $(x, y) \in D$, $x\sigma \neq y\sigma$.

Definition 12 P and Q are *strongly D-equivalent*, written $P \sim_D Q$, if $P\sigma \,\dot{\sim}\, Q\sigma$ for all substitutions σ respecting D.

Note that an immediate consequence of this definition is that if $D \subseteq D'$ then $P \sim_D Q$ implies $P \sim_{D'} Q$. As a simple example, we have

$$\overline{x} \mid y \ \sim_{\{x,y\}} \ \overline{x}.y + y.\overline{x}$$

Here we have used a natural abbreviation, allowing ourselves to write a set $A \subseteq \mathcal{N}$ when we mean the distinction $A \times A - \mathrm{Id}_{\mathcal{N}}$, which keeps all members of A distinct from each other. Clearly, then, we have the two extreme cases

$$\dot{\sim} \ = \sim_{\mathcal{N}} \qquad \text{and} \qquad \sim \ = \sim_{\emptyset}$$

2.3 Late and early bisimilarity

We close this section with a discussion of an interesting alternative definition of bisimulation obtained by commuting the quantifiers in clause 2 in Table 3:

$2'$ If $P \xrightarrow{x(y)} P'$ and $y \notin \mathrm{n}(P,Q)$,
then for all w, there is Q' such that $Q \xrightarrow{x(y)} Q'$ and $P'\{{}^{w}\!/\!{}_{y}\}\mathcal{S}Q'\{{}^{w}\!/\!{}_{y}\}$

Write $\dot{\sim}'$ for the ground equivalence obtained with this modification. Now $\dot{\sim}'$ is strictly weaker than $\dot{\sim}$ (and the corresponding non-ground equivalence $\sim'$ is strictly weaker than $\sim$), i.e. more agents are equivalent when clause $2'$ is adopted. The reason is that clause 2 requires that there is one simulating input transition which is equipotent for *all* instances of the object. In contrast, clause $2'$ only requires that for each instance of the object there exists a simulating transition (and these simulating transitions may be different for different instances). Thus, for the purpose of $\dot{\sim}'$ the instantiation of the object can be regarded as happening simultaneously with (or even before) the input transition, and for $\dot{\sim}$ the instantiation may be regarded as happening after the transition. For this reason we will sometimes call $\dot{\sim}'$ *early* bisimilarity and $\dot{\sim}$ *late* bisimilarity.

As an example consider the following agents:

$$\begin{array}{rcl} P & = & x(u).R + x(u).\mathbf{0} \\ Q & = & P + x(u).[u = z]R \end{array}$$

It always holds that $P \,\dot{\sim}'\, Q$, but $P \,\dot{\sim}\, Q$ is not true in general. To see this consider the transition

$$Q \xrightarrow{x(u)} [u = z]R \tag{8}$$

P has no transition which simulates (8) for all instantiations of u. However, for each instantiation of u there is a simulating transition: for z it is

$$P \xrightarrow{x(u)} R$$

(since $([u=z]R)\{z/u\} \dot{\sim}' R\{z/u\}$) and for all other names it is

$$P \xrightarrow{x(u)} \mathbf{0}$$

(since $([u=z]R)\{z'/u\} \dot{\sim}' \mathbf{0} \equiv \mathbf{0}\{z'/u\}$ for all $z' \neq z$). A similar but slightly longer example not involving the matching operator also exists.

It is interesting to note that with the early instantiation scheme mentioned in Section 1.3.1 the natural concept of bisimilarity would coincide with early bisimilarity, while late bisimilarity would be hard to define. Our late instantiation scheme has thus the advantage that both versions of bisimilarity can easily be treated. Although early bisimilarity is closer to the original idea of equivalence as presented in CCS its equational theory is more complicated, and we defer a treatment of it to a forthcoming paper.

3 Properties of strong bisimilarity

The main contribution in this paper is to develop the properties of strong bisimilarity and equivalence. Even though equivalence is perhaps the more interesting of the two (since it turns out to be a congruence) it is necessary to first derive the properties of bisimilarity.

3.1 Transitions and alpha-conversion

In this subsection we give a series of fundamental lemmas which underpin many later results. None of the results is unexpected and their proofs are mostly straightforward, though they do require careful attention to detail. Moreover, care is also required in finding a correct order of presentation as the proofs of some of the lemmas rely on properties established earlier in the series.

The first lemma describes the relationships among the free names of an agent, the names of its possible actions, and the free names of its immediate derivatives.

Lemma 1 If $P \xrightarrow{\alpha} P'$ then (i) $\mathrm{fn}(\alpha) \subseteq \mathrm{fn}(P)$ and (ii) $\mathrm{fn}(P') \subseteq \mathrm{fn}(P) \cup \mathrm{bn}(\alpha)$.

Proof: By induction on depth of inference. See the Appendix. $\Box$

Definition 13 In the following lemmas the phrase:

$$\text{if } P \xrightarrow{\alpha} P' \text{ then } \textit{equally } Q \xrightarrow{\alpha'} Q'$$

means that if $P \xrightarrow{\alpha} P'$ may be inferred from the transition rules then so, by an inference of no greater depth, may be $Q \xrightarrow{\alpha'} Q'$. $\Box$

The reason for introducing this notion, and for including it in the statements of Lemmas 2–5 to follow, is that it facilitates the proofs of the properties of interest. It is not used anywhere other than in the present series of lemmas.

As discussed in the preceding sections the following lemma, whose content may be paraphrased by saying that the object of a bound action may be 'almost any' name, is of the utmost importance.

Lemma 2 Suppose that $P \xrightarrow{a(y)} P'$ where $a = x$ or $a = \overline{x}$ and that $z \notin \mathrm{n}(P)$. Then equally for some $P'' \equiv_\alpha P'\{z/y\}$, $P \xrightarrow{a(z)} P''$.

Proof: By induction on depth of inference. □

The following two lemmas are concerned with the relationship between action and substitution. First we define the result of applying a substitution to an action.

Definition 14 If α is an action and σ a substitution then $\alpha\sigma$ is defined as follows:

$$\begin{array}{rcl} (\overline{x}y)\sigma & = & \overline{x\sigma}y\sigma \\ \tau\sigma & = & \tau \\ (a(y))\sigma & = & a\sigma(y) \quad \text{if } a = x \text{ or } a = \overline{x} \end{array}$$

□

The next lemma asserts that if an agent P may perform an action α and thereby evolve into P', then up to alpha-equivalence $P\sigma$ may perform $\alpha\sigma$ and evolve into $P'\sigma$. In the case $\alpha = a(y)$ where $a = x$ or $a = \overline{x}$ a side condition is necessary. For in general, $P\sigma$ may not admit actions with y as bound object, and y may occur free in P'.

Lemma 3 If $P \xrightarrow{\alpha} P'$, $\mathrm{bn}(\alpha) \cap \mathrm{fn}(P'\sigma) = \emptyset$, and $\sigma \lceil \mathrm{bn}(\alpha) = \mathrm{id}$, then equally for some $P'' \equiv_\alpha P'\sigma$, $P\sigma \xrightarrow{\alpha\sigma} P''$.

Proof: By induction on depth of inference. □

The full converse of the preceding lemma does not hold. As a simple illustration of this point suppose that $P \equiv \overline{x}y.\mathbf{0} \mid w(z).\mathbf{0}$ and $\sigma = \{w/x\}$. Then $P\sigma \xrightarrow{\tau} (\mathbf{0} \mid \mathbf{0})$ but P cannot perform a τ-action. However the following partial converse does hold. A more general statement is possible but the one below suffices for the present development.

Lemma 4 If $P\{w/z\} \xrightarrow{\alpha} P'$ where $w \notin \mathrm{fn}(P)$ and $\mathrm{bn}(\alpha) \cap \mathrm{fn}(P, w) = \emptyset$, then equally for some Q and β with $Q\{w/z\} \equiv_\alpha P'$ and $\beta\sigma = \alpha$, $P \xrightarrow{\beta} Q$.

Proof: By induction on depth of inference. □

In stating the preceding three lemmas we have been careful in our use of the relation of alpha-convertibility of agents. The content of Theorem 1 below is that alpha-convertibility is a strong bisimulation and thus alpha-convertible agents are strongly bisimilar. To prove it we require the following lemma which describes the relationship between the actions of alpha-convertible agents.

Lemma 5 Suppose that $P \equiv_\alpha Q$.

(a) If α is a free action and $P \xrightarrow{\alpha} P'$ then equally for some Q' with $P' \equiv_\alpha Q'$, $Q \xrightarrow{\alpha} Q'$.

(b) If $P \xrightarrow{a(y)} P'$ where $a = x$ or $a = \overline{x}$ and $z \notin \mathrm{n}(Q)$ then equally for some Q' with $P'\{z/y\} \equiv_\alpha Q'$, $Q \xrightarrow{a(z)} Q'$.

Proof: By induction on depth of inference. □

Theorem 1 $\equiv_\alpha$ is a strong bisimulation.

Proof: Straightforward using the preceding lemma. □

Having established this theorem, in what follows we shall freely identify alpha-convertible agents writing $\equiv$ for $\equiv_\alpha$.

3.2 Bisimilarity as an equivalence

As we saw in Section 2.1 strong bisimilarity is not, in general, preserved by substitution. However the following important result holds.

Lemma 6 If $P \dot{\sim} Q$ and $w \notin \mathrm{fn}(P, Q)$, then $P\{w/z\} \dot{\sim} Q\{w/z\}$.

Proof: The relation $\mathcal{S} = \bigcup_{n<\omega} \mathcal{S}_n$ is a strong bisimulation where

$$\begin{array}{rcl} \mathcal{S}_0 & = & \dot{\sim} \\ \mathcal{S}_{n+1} & = & \{(P\{w/z\}, Q\{w/z\}) \mid P\mathcal{S}_nQ,\ w \notin \mathrm{fn}(P, Q)\} \end{array}$$

See the Appendix. □

The next objective is to establish that $\dot{\sim}$ is an equivalence relation preserved by many of the operators. To prove preservation in the case of the composition and scope restriction operators it is necessary to construct a suitable bisimulation. It turns out that this construction is useful in other contexts and thus we isolate it in a definition.

Definition 15 A relation $\mathcal{S}$ is a *strong simulation up to restriction* iff whenever $P\mathcal{S}Q$ then

1. if $w \notin \mathrm{fn}(P, Q)$ then $P\{w/z\}\mathcal{S}Q\{w/z\}$, and

2. (a) if $P \xrightarrow{\overline{x}y} P'$ then for some Q', $Q \xrightarrow{\overline{x}y} Q'$ and $P'\mathcal{S}Q'$,

   (b) if $y \notin \mathrm{n}(P, Q)$ and $P \xrightarrow{x(y)} P'$ then for some Q', $Q \xrightarrow{x(y)} Q'$ and for all v, $P'\{v/y\}\mathcal{S}Q'\{v/y\}$,

   (c) if $y \notin \mathrm{n}(P, Q)$ and $P \xrightarrow{\overline{x}(y)} P'$ then for some Q', $Q \xrightarrow{\overline{x}(y)} Q'$ and $P'\mathcal{S}Q'$,

   (d) if $P \xrightarrow{\tau} P'$ then for some Q', $Q \xrightarrow{\tau} Q'$ and either $P'\mathcal{S}Q'$ or for some P'', Q'' and w, $P' \equiv (w)P''$, $Q' \equiv (w)Q''$ and $P''\mathcal{S}Q''$.

A relation $\mathcal{S}$ is a *strong bisimulation up to restriction* iff both $\mathcal{S}$ and $\mathcal{S}^{-1}$ are strong simulations up to restriction. $\square$

The import of the next result is that in order to establish that $P \mathrel{\dot\sim} Q$ it suffices to find a strong bisimulation up to restriction containing (P, Q).

Lemma 7 If $\mathcal{S}$ is a strong bisimulation up to restriction then $\mathcal{S} \subseteq \dot\sim$.

Proof: We show that $\mathcal{S}^* = \bigcup_{n<\omega} \mathcal{S}_n$ is a strong bisimulation where

$$\begin{array}{rcl} \mathcal{S}_0 & = & \mathcal{S} \\ \mathcal{S}_{n+1} & = & \{((w)P, (w)Q) \mid P\mathcal{S}_nQ,\ w \in \mathcal{N}\} \end{array}$$

See the Appendix. $\square$

Combining the preceding results we can now prove the following.

Theorem 2 (a) $\dot\sim$ is an equivalence relation.

(b) If $P \mathrel{\dot\sim} Q$ then
$$\begin{array}{rcll} \alpha.\,P & \dot\sim & \alpha.\,Q, & \alpha \text{ a free action} \\ P + R & \dot\sim & Q + R, & \\ [x\!=\!y]P & \dot\sim & [x\!=\!y]Q, & \\ P|R & \dot\sim & Q|R, & \\ (w)P & \dot\sim & (w)Q. & \end{array}$$

(c) If for all $v \in \mathrm{fn}(P, Q, y)$, $P\{v/y\} \mathrel{\dot\sim} Q\{v/y\}$ then $x(y).\,P \mathrel{\dot\sim} x(y).\,Q$.

Proof: For details see the Appendix. The proof ideas are:

(a) Reflexivity and symmetry are obvious but transitivity is not. Indeed it is not in general the case that if $\mathcal{S}_1$ and $\mathcal{S}_2$ are strong bisimulations then so is $\mathcal{S}_1\mathcal{S}_2$. However it is the case that $\dot\sim\ \dot\sim$ is a strong bisimulation.

(b) The first three assertions are easily verified. The other two are proved by showing that $\{(P|R, Q|R) \mid P \mathrel{\dot\sim} Q\}$ is a strong bisimulation up to restriction, and by observing that by Lemma 6, $\dot\sim$ is a strong bisimulation up to restriction.

(c) This is straightforward using Lemma 6. $\square$

This theorem establishes that bisimilarity is almost a congruence; it is preserved by all operators but input prefix. Although $x(y).P \stackrel{.}{\sim} x(y).Q$ does *not* follow from $P \stackrel{.}{\sim} Q$ (as established in Section 2.1) it follows from the stronger assumption that P and Q are bisimilar for all instances of y.

3.3 Algebraic laws for bisimilarity

We proceed to investigate further the theory of $\stackrel{.}{\sim}$ by stating and proving a collection of algebraic laws. To begin, there are the obvious laws for summation from CCS, which establish that $\mathbf{0}$ is a zero for summation, and that summation is idempotent, commutative, and associative:

Theorem 3

$$\begin{array}{lrcl}
\text{(a)} & P + \mathbf{0} & \stackrel{.}{\sim} & P \\
\text{(b)} & P + P & \stackrel{.}{\sim} & P \\
\text{(c)} & P_1 + P_2 & \stackrel{.}{\sim} & P_2 + P_1 \\
\text{(d)} & P_1 + (P_2 + P_3) & \stackrel{.}{\sim} & (P_1 + P_2) + P_3
\end{array}$$

Proof: The following relations are easily seen to be strong bisimulations:

$$\begin{array}{lcl}
\mathcal{S}_a & = & \{(P_1 + \mathbf{0},\ \ P_1) \mid\ P_1 \text{ agent}\} \ \cup\ \mathbf{Id} \\
\mathcal{S}_b & = & \{(P_1 + P_1,\ \ P_1) \mid\ P_1 \text{ agent}\} \ \cup\ \mathbf{Id} \\
\mathcal{S}_c & = & \{(P_1 + P_2,\ \ P_2 + P_1) \mid\ P_1, P_2 \text{ agents}\} \ \cup\ \mathbf{Id} \\
\mathcal{S}_d & = & \{(P_1 + (P_2 + P_3),\ \ (P_1 + P_2) + P_3) \mid\ P_1, P_2, P_3 \text{ agents}\} \ \cup\ \mathbf{Id}
\end{array}$$

where **Id** is the identity on agents. $\square$

There are also the following simple laws for agent identifiers and matching:

Theorem 4 If $A(\tilde{x}) \stackrel{\text{def}}{=} P$ then $A(\tilde{y}) \ \stackrel{.}{\sim}\ P\{\tilde{y}/\tilde{x}\}$.

Proof: It is straightforward to show that the relation

$$\mathcal{S} = \{(A(\tilde{y}),\ \ P\{\tilde{y}/\tilde{x}\})\} \cup \mathbf{Id}$$

is a bisimulation $\square$

Theorem 5

$$\begin{array}{llcl}
\text{(a)} & [x = y]P & \stackrel{.}{\sim} & \mathbf{0} \quad \text{if } x \neq y \\
\text{(b)} & [x = x]P & \stackrel{.}{\sim} & P
\end{array}$$

Proof: We can prove the following relations to be strong bisimulations:

$$\begin{array}{lcl}
\mathcal{S}_a & = & \{([x = y]P_1,\ \ \mathbf{0}) \mid\ P_1 \text{ agent},\ x \neq y\} \\
\mathcal{S}_b & = & \{([x = x]P_1,\ \ P_1) \mid\ P_1 \text{ agent}\} \ \cup\ \mathbf{Id}
\end{array}$$

□

These are the only laws for $\dot{\sim}$ which correspond to the "dynamic" laws in CCS (of course matching is not present in CCS, but it qualifies as a "dynamic operator" since it disappears in the derivative of a transition). The "static" laws in CCS are related to the relabelling, restriction, and parallel operators. In the π-calculus there is no relabelling operator. Moreover, our restriction operator is perhaps not quite a static operator since it may disappear (through an application of the OPEN rule) and reappear in a different place (through the CLOSE rule). Nevertheless, it satisfies many natural laws:

Theorem 6

$$
\begin{array}{lrcll}
\text{(a)} & (y)P & \dot{\sim} & P & \text{if } y \notin \mathrm{fn}(P) \\
\text{(b)} & (y)(z)P & \dot{\sim} & (z)(y)P & \\
\text{(c)} & (y)(P+Q) & \dot{\sim} & (y)P + (y)Q & \\
\text{(d)} & (y)\alpha.P & \dot{\sim} & \alpha.(y)P & \text{if } y \notin \mathrm{n}(\alpha) \\
\text{(e)} & (y)\alpha.P & \dot{\sim} & \mathbf{0} & \text{if } y \text{ is the subject of } \alpha
\end{array}
$$

Proof: We prove the following relations to be strong bisimulations:

$$
\begin{array}{lcl}
\mathcal{S}_a & = & \{((y)P_1,\ \ P_1) \mid\ P_1 \text{ agent}, y \notin \mathrm{fn}(P_1)\} \\
\mathcal{S}_b & = & \{((y)(z)P_1,\ \ (z)(y)P_1) \mid\ P_1 \text{ agent}\} \cup \mathbf{Id} \\
\mathcal{S}_c & = & \{((y)(P_1+P_2),\ \ (y)P_1 + (y)P_2) \mid\ P_1, P_2 \text{ agents}\} \cup \mathbf{Id} \\
\mathcal{S}_d & = & \{((y)\alpha.P_1,\ \ \alpha.(y)P_1) \mid\ P_1 \text{ agent}, y \notin \mathrm{n}(\alpha)\} \cup \mathbf{Id} \\
\mathcal{S}_e & = & \{((y)\alpha.P_1,\ \ \mathbf{0}) \mid\ P_1 \text{ agent}, y \text{ subject of } \alpha\}
\end{array}
$$

We must include **Id** in $\mathcal{S}_b$ since one of the restrictions may disappear because of the OPEN rule. □

Theorem 6 (a) just says that vacuous restrictions can be removed, and Theorem 6 (b) that restrictions commute. Theorem 6 (c) implies that restriction distributes over summation, while the last two parts of Theorem 6 relate restriction and prefix. It is worth noting that neither Theorem 6 (d) nor Theorem 6 (e) is immediately applicable when y is the object in α. If y is a bound object, then an alpha-conversion will make an application of Theorem 6 (d) possible. But if y is a free object, i.e. $\alpha = \overline{x}y$, then the restriction cannot be propagated through the prefix operator. This is in contrast with the situation in CCS, where all restriction operators can be eliminated while preserving equivalence. In the π-calculus, agents of type $(y)\overline{x}y.P$ (when $x \neq y$) contain an irreducible restriction operator; this type of agent will be of importance for the completeness proof, so we define:

Definition 16 If $x \neq y$, then $\overline{x}(y).P$ means $(y)\overline{x}y.P$, and the prefix $\overline{x}(y)$ is called a *derived* prefix.

Thus, by Theorem 6 (d) and (e), any restriction operator can either be propagated through a prefix or form a derived prefix. It will often be useful to treat derived prefixes along with ordinary prefixes. In these situations it is important that Theorem 6 also holds for derived prefixes:

Theorem 7 Theorem 6 is valid also if α ranges over derived prefixes.

Proof: Directly from Theorem 6:

$$\begin{array}{llllllll}
\text{(d)} & (y)(z)\overline{x}z.P & \dot{\sim} & (z)(y)\overline{x}z.P & \dot{\sim} & (z)\overline{x}z.(y)P & & \text{if } z, x \neq y \\
\text{(e)} & (y)(z)\overline{y}z.P & \dot{\sim} & (z)(y)\overline{y}z.P & \dot{\sim} & (z)\mathbf{0} & \dot{\sim} & \mathbf{0}
\end{array}$$

□

We proceed with some expected laws for parallel composition.

Theorem 8

$$\begin{array}{lrcll}
\text{(a)} & P \mid \mathbf{0} & \dot{\sim} & P & \\
\text{(b)} & P_1 \mid P_2 & \dot{\sim} & P_2 \mid P_1 & \\
\text{(c)} & (y)P_1 \mid P_2 & \dot{\sim} & (y)(P_1 \mid P_2) & \text{if } y \notin \mathrm{fn}(P_2) \\
\text{(d)} & (P_1 \mid P_2) \mid P_3 & \dot{\sim} & P_1 \mid (P_2 \mid P_3) &
\end{array}$$

Proof: See the Appendix. Note that in order to prove (d) of the theorem, we must first establish (c), since a parallel composition may generate a restriction operator through the CLOSE rule. To prove (c) and (d) we show that certain relations are *strong bisimulations up to* $\dot{\sim}$ *and restriction*. The concept of a *strong bisimulation up to* $\dot{\sim}$ is the obvious analogue of a similar concept from CCS. It differs from a strong bisimulation in that any transition need be simulated only up to strong bisimilarity. To prove (c) and (d) we must combine this idea with that of bisimulation up to restriction introduced earlier. □

Theorem 8 (a), (b), and (d) assert that $\mathbf{0}$ is a unit for parallel, and that parallel is commutative and associative. Part (c) is the *scope extension* law: it says that a restriction can safely extend its scope to agents which do not contain free occurrences of the restricted name. This can be thought of as a generalization of Theorem 6 (a), which in fact is an easy consequence of Theorem 8 and $(y)\mathbf{0} \;\dot{\sim}\; \mathbf{0}$:

$$(y)P \quad \dot{\sim} \quad (y)(P \mid \mathbf{0}) \quad \dot{\sim} \quad P \mid (y)\mathbf{0} \quad \dot{\sim} \quad P \mid \mathbf{0} \quad \dot{\sim} \quad P \qquad \text{if } y \notin \mathrm{fn}(P)$$

The scope extension law is also related to the CCS law which says that a restriction distributes over parallel composition if the components cannot interact by means of the restricted port. In our calculus, two agents can communicate through a name if one agent has the name in positive subject position, and the other agent has the name in negative subject position. In the absence of a more refined notion of sort, we can at least say that both agents must have the name free, so our formulation of this law is:

Theorem 9

$$(y)(P_1 \mid P_2) \quad \dot{\sim} \quad (y)P_1 \mid (y)P_2 \quad \text{if } y \notin \mathrm{fn}(P_1) \cap \mathrm{fn}(P_2)$$

Proof: If $y \notin \mathrm{fn}(P_1) \cap \mathrm{fn}(P_2)$, then y cannot be free in both P_1 and P_2. Assume that y is not free in P_2. Then by Theorems 6 (a) and 8 (c):

$$(y)(P_1 \mid P_2) \quad \dot{\sim} \quad (y)P_1 \mid P_2 \quad \dot{\sim} \quad (y)P_1 \mid (y)P_2$$

The situation when y is not free in P_1 is similar. □

Conversely, Theorem 8 (c) is an easy consequence of Theorems 9 and 6 (a).

Finally, there is a counterpart to the expansion law in CCS. In our calculus the expansion law also covers derived prefixes, so in the following, α, β will range over ordinary and derived prefixes.

Theorem 10 Let $P \equiv \sum_i \alpha_i.P_i$ and $Q \equiv \sum_j \beta_j.Q_j$, where $\mathrm{bn}(\alpha_i) \cap \mathrm{fn}(Q) = \emptyset$ for all i, and $\mathrm{bn}(\beta_j) \cap \mathrm{fn}(P) = \emptyset$ for all j. Then

$$P \mid Q \quad \dot{\sim} \quad \sum_i \alpha_i.(P_i \mid Q) + \sum_j \beta_j.(P \mid Q_j) + \sum_{\alpha_i \,\mathbf{comp}\, \beta_j} \tau.R_{ij}$$

where the relation $\alpha_i \,\mathrm{comp}\, \beta_j$ (α_i *complements* β_j) holds in the following four cases, which also define R_{ij}:

1. α_i is $\overline{x}u$ and β_j is $x(v)$; then R_{ij} is $P_i \mid Q_j\{u/v\}$.
2. α_i is $\overline{x}(u)$ and β_j is $x(v)$; then R_{ij} is $(w)(P_i\{w/u\} \mid Q_j\{w/v\})$, where w is not free in $(u)P_i$ or in $(v)Q_j$.
3. α_i is $x(v)$ and β_j is $\overline{x}u$; then R_{ij} is $P_i\{u/v\} \mid Q_j$.
4. α_i is $x(v)$ and β_j is $\overline{x}(u)$; then R_{ij} is $(w)(P_i\{w/v\} \mid Q_j\{w/u\})$, where w is not free in $(v)P_i$ or in $(u)Q_j$.

Proof: Assume the premises of the lemma, and write R for the right hand side of the equation. Define the relation $\mathcal{S}$ by

$$\mathcal{S} \quad = \quad \{(P \mid Q, \;\; R)\} \;\cup\; \mathbf{Id}$$

We can show that $\mathcal{S}$ is a bisimulation. □

Note that the side conditions $\mathrm{bn}(\alpha_i) \cap \mathrm{fn}(Q) = \emptyset$ and $\mathrm{bn}(\beta_j) \cap \mathrm{fn}(P) = \emptyset$ are important, otherwise a bound object in α_i (or β_j) would bind names in Q (or P) in the right hand side but not in the left hand side.

4 Properties of strong (D-) equivalence

4.1 Algebraic properties of of D-equivalence

Most of the properties established for strong bisimilarity carry over to strong D-equivalence for any D:

Theorem 11 For any distinction D it holds that

(a) $\sim_D$ is an equivalence relation.

(b) If $P \sim_D Q$ then
$$\begin{array}{rcll} \alpha.\,P & \sim_D & \alpha.\,Q, & \alpha \text{ a free action} \\ P + R & \sim_D & Q + R, & \\ [x\!=\!y]P & \sim_D & [x\!=\!y]Q, & \\ P|R & \sim_D & Q|R, & \\ (w)P & \sim_D & (w)Q. & \end{array}$$

(c) If $P \sim_D Q$ and for all $v \in \mathrm{fn}(P,Q)$ such that $(v,y) \in D$ it holds that $P\{v/y\} \sim_D Q\{v/y\}$ then $x(y).\,P \sim_D x(y).\,Q$.

Proof: Directly from Definition 6 and Theorem 2. □

An immediate consequence is the following:

Theorem 12 Strong equivalence is a congruence.

Proof: Put $D = \emptyset$ in Theorem 11. □

So in particular $P \sim Q$ implies $x(y).P \sim x(y).Q$.

Theorem 13 All theorems in Section 3.3 except Theorems 5 (a) and 10 also hold for $\sim_D$ for all distinctions D.

Proof: Immediately from Definition 6 and the theorems in Section 3.3. □

To see that Theorem 5 (a) is invalid for strong equivalence note that

$$([x\!=\!y]P)\{x/y\} \;\dot{\not\sim}\; \mathbf{0}\{x/y\}$$

when $P \;\dot{\not\sim}\; \mathbf{0}$. The failure of the expansion law (Theorem 10) for strong equivalence was demonstrated in Section 2.2. Instead of these two theorems we have the following two:

Theorem 14

$$[x\!=\!y]P \sim_{\{x,y\}} \mathbf{0}$$

Proof: Immediately from Definition 12 and Theorem 5 (a). □

Theorem 15 Let $P \equiv \sum_i \alpha_i.P_i$ and $Q \equiv \sum_j \beta_j.Q_j$, where no α_i (resp. β_j) binds a name free in Q (resp. P); then

$$P \mid Q \;\sim\; \sum_i \alpha_i.(P_i \mid Q) + \sum_j \beta_j.(P \mid Q_j) + \sum_{\alpha_i\,\mathrm{opp}\,\beta_j} [x_i = y_j]\tau.R_{ij}$$

where the relation $\alpha_i\,\mathrm{opp}\,\beta_j$ (α_i *opposes* β_j) holds in four cases:

1. α_i is $\overline{x_i}u$ and β_j is $y_j(v)$; then R_{ij} is $P_i \mid Q_j\{u/v\}$.
2. α_i is $\overline{x_i}(u)$ and β_j is $y_j(v)$; then R_{ij} is $(w)(P_i\{w/u\} \mid Q_j\{w/v\})$, where w is not free in $(u)P_i$ or in $(v)Q_j$.
3. α_i is $x_i(v)$ and β_j is $\overline{y_j}u$; then R_{ij} is $P_i\{u/v\} \mid Q_j$.
4. α_i is $x_i(v)$ and β_j is $\overline{y_j}(u)$; then R_{ij} is $(w)(P_i\{w/v\} \mid Q_j\{w/u\})$, where w is not free in $(v)P_i$ or in $(u)Q_j$.

Proof: It is straightforward to check (using Theorems 5 (a) and 10) that applying a substitution σ to both sides of the equation yields strongly bisimilar agents. □

The last two theorems can be combined into expansion laws for D-equivalence for arbitrary D; we believe that these will be useful working laws. The following laws additionally relate D-equivalences for different D:s and throw light on the two forms of name binding in the calculus. We first define two operations on distinctions:

Definition 17

$$D\backslash x \;\stackrel{\mathrm{def}}{=}\; D - (\{x\} \times \mathcal{N} \cup \mathcal{N} \times \{x\})$$

□

This removes any constraint in D upon the substitution for x.

Definition 18 For any set $A \subseteq \mathcal{N}$ of names,

$$D{\restriction}A \;\stackrel{\mathrm{def}}{=}\; D \cap (A \times A)$$

□

Theorem 16 (a) If $P \sim_D Q$ then $(x)P \sim_{D\backslash x} (x)Q$

(b) If $P \sim_{D\backslash x} Q$ then $y(x).P \sim_D y(x).Q$

(c) If $P \sim_D Q$ and $A = \mathrm{fn}(P, Q)$ then $P \sim_{D{\restriction}A} Q$

Proof: For (a), if $P \sim_D Q$ and σ respects $D\backslash x$, then for some $x' \notin \mathrm{fn}((x)P, (x)Q, P\sigma, Q\sigma)$ with $x'\sigma = x'$, $((x)P)\sigma \equiv (x')P\sigma'$ and $((x)Q)\sigma \equiv (x')Q\sigma'$ where $\sigma' = \{x'/x\}\sigma$. Since σ' respects D, $P\sigma' \dot{\sim} Q\sigma'$ and hence $(x')P\sigma' \dot{\sim} (x')Q\sigma'$, i.e. $((x)P)\sigma \dot{\sim} ((x)Q)\sigma$. Hence $(x)P \sim_{D\backslash x} (x)Q$.

For (b), suppose that $P \sim_{D\backslash x} Q$ and σ respects D. Then for some $x' \notin \mathrm{fn}((x)P, (x)Q, P\sigma, Q\sigma)$ with $x'\sigma = x'$, $(y(x).\, P)\sigma \equiv y\sigma(x').\, P\{x'/x\}\sigma$ and $(y(x).\, Q)\sigma \equiv y\sigma(x').\, Q\{x'/x\}\sigma$. Then for any $w \in \mathrm{fn}(P\{x'/x\}\sigma, Q\{x'/x\}\sigma, x)$, since $\{x'/x\}\sigma\{w/x'\}$ respects $D\backslash x$, $P\{x'/x\}\sigma\{w/x'\} \dot{\sim} Q\{x'/x\}\sigma\{w/x'\}$. Hence $(y(x).\, P)\sigma \dot{\sim} (y(x).\, Q)\sigma$. So $y(x).\, P \sim_D y(x).\, Q$.

For (c), note that if σ respects $D\lceil A$ then there is σ' respecting D such that $\sigma\lceil A = \sigma'\lceil A$. $\square$

4.2 Strong equivalence and recursion

We record here the properties which we would expect of recursive definitions, by analogy with CCS [1]. First, if we transform the right-hand sides of definitions, respecting $\sim$, then the agent defined is the same up to $\sim$. Second, if two agents satisfy the same (recursive) equation, then they are the same up to $\sim$, provided the equation satisfies a standard condition. Both these properties hold for strong equivalence but fail for strong bisimilarity.

In order to state these results, we need a few preliminaries. We assume a set of *schematic identifiers*, each having a nonnegative *arity*. In the following, X and X_i will range over schematic identifiers. An *agent expression* is like an agent but may contain schematic identifiers in the same way as identifiers; we use E, F to range over agent expressions.

Definition 19 Let X have arity n, let $\tilde{x} = x_1, \ldots, x_n$ be distinct names, and assume that $\mathrm{fn}(P) \subseteq \{x_1, \ldots, x_n\}$. The *replacement of $X(\tilde{x})$ by P in E*, written $E\{X(\tilde{x}) := P\}$, means the result of replacing each subterm $X(\tilde{y})$ in E by $P\{\tilde{y}/\tilde{x}\}$. This extends in the obvious way to *simultaneous replacement* of several schematic identifiers, $E\{X_1(\tilde{x}_1) := P_1, \ldots, X_m(\tilde{x}_m) := P_m\}$. $\square$

As an example,

$$\Big(\overline{x}y.X(x,x) + (y)X(x,y)\Big)\{X(u,w) := \overline{u}w.\mathbf{0}\} \;\equiv\; \overline{x}y.\overline{x}x.\mathbf{0} + (y)\overline{x}y.\mathbf{0}$$

In what follows, we assume the indexing set I to be either $\{1, \ldots, m\}$ for some m, or else ω. We write $\widetilde{X}$ for a sequence $X_1, X_2, \ldots$ indexed by I; similarly $\widetilde{P}$, etc. We use i, j to range over I. When a sequence $\widetilde{X}$ of schematic identifiers is implied by context, each with an associated name sequence $\tilde{x}_i$, then it is convenient to write $E\{X_1(\tilde{x}_1) := P_1, \ldots, X_m(\tilde{x}_m) := P_m\}$ simply as $E(P_1, \ldots, P_m)$ or as $E(\widetilde{P})$. If each P_i is $A_i(\tilde{x}_i)$, we also write $E(A_1, A_2, \ldots)$ or $E(\widetilde{A})$.

It is natural to define strong equivalence between agent expressions as equivalence under all replacements of schematic identifiers by agents:

Definition 20 Let E and F be two agent expressions containing only the schematic identifiers $X_1, \ldots, X_m$, with associated name sequences $\widetilde{x}_1, \ldots, \widetilde{x}_m$. Then $E \sim F$ means that

$$E(\widetilde{P}) \sim F(\widetilde{P})$$

for all $\widetilde{P}$ such that $\mathrm{fn}(P_i) \subseteq \widetilde{x}_i$ for each i. □

We can now state our first result, that recursive definition preserves strong equivalence:

Theorem 17 Assume that $\widetilde{E}$ and $\widetilde{F}$ are agent expressions containing only the schematic identifiers X_i, each with associated name sequence $\widetilde{x}_i$. Assume that $\widetilde{A}$ and $\widetilde{B}$ are identifiers such that for each i the arities of A_i, B_i and X_i are equal. Assume that for all i:

$$\begin{array}{rcl} E_i & \sim & F_i \\ A_i(\widetilde{x}_i) & \stackrel{\mathrm{def}}{=} & E_i(\widetilde{A}) \\ B_i(\widetilde{x}_i) & \stackrel{\mathrm{def}}{=} & F_i(\widetilde{B}) \end{array}$$

Then $A_i(\widetilde{x}_i) \sim B_i(\widetilde{x}_i)$ for all i.

Proof: See the Appendix. □

Definition 21 A term or identifier is *weakly guarded* in P if it lies within some subterm $\alpha.Q$ of P. □

If A is weakly guarded in E then intuitively, from the definition $A \stackrel{\mathrm{def}}{=} E$, we can unfold the behaviour of A uniquely. The next result makes this precise in the general case:

Theorem 18 Assume that $\widetilde{E}$ are agent expressions containing only the schematic identifiers X_i, each with associated name sequence $\widetilde{x}_i$, and that each X_i is weakly guarded in each E_j. Assume that $\widetilde{P}$ and $\widetilde{Q}$ are agents such that $\mathrm{fn}(P_i) \subseteq \widetilde{x}_i$ and $\mathrm{fn}(Q_i) \subseteq \widetilde{x}_i$ for each i. Assume that for all i:

$$\begin{array}{rcl} P_i & \sim & E_i(\widetilde{P}) \\ Q_i & \sim & E_i(\widetilde{Q}) \end{array}$$

Then $P_i \sim Q_i$ for all i. □

Proof: The proof follows the lines of the proof of Proposition 14 (2) in [1]. It uses the idea of bisimulation up to $\dot{\sim}$ as defined in the appendix (Definition 25) below. We omit the details. □

5 Algebraic theory

In this section we establish an axiomatization of strong ground equivalence, and show how this axiomatization can easily be extended to non-ground equivalence and D-equivalences. These theories are complete over *finite* agents (i.e. agents not containing any agent identifiers), but incomplete over *all* agents (necessarily since $\dot{\sim}$ is not recursively enumerable).

We shall state the rules using the standard equality symbol $=$. We omit the usual rules for an equivalence relation. Note that $=$ is not assumed to stand for a congruence relation (since $\dot{\sim}$ is not a congruence); the substitutive properties of $=$ are therefore explicitly mentioned.

Definition 22 The theory **SGE** (for strong ground equivalence) consists of the following axioms and inference rules:

Alpha-conversion

A *From* $P \equiv Q$ *infer* $P = Q$

Congruence

C0 *From* $P = Q$ *infer*

$$\begin{array}{ll} \tau.P = \tau.Q & \overline{x}y.P = \overline{x}y.Q \\ P + R = Q + R & P \mid R = Q \mid R \\ (x)P = (x)Q & [x=y]P = [x=y]Q \end{array}$$

C1 *From* $P\{z/y\} = Q\{z/y\}$, for all names $z \in \mathrm{fn}(P, Q, y)$, *infer*

$$x(y).P = x(y).Q$$

Summation

S0	$P + \mathbf{0} = P$
S1	$P + P = P$
S2	$P + Q = Q + P$
S3	$P + (Q + R) = (P + Q) + R$

Restriction

R0	$(x)P = P$	if $x \notin \mathrm{fn}(P)$
R1	$(x)(y)P = (y)(x)P$	
R2	$(x)(P + Q) = (x)P + (x)Q$	
R3	$(x)\alpha.P = \alpha.(x)P$	if x is not in $\mathrm{n}(\alpha)$
R4	$(x)\alpha.P = \mathbf{0}$	if x is the subject of α

Match

$$\begin{array}{lll}\mathbf{M0} & [x=y]P = \mathbf{0} & \text{if } x \neq y \\ \mathbf{M1} & [x=x]P = P & \end{array}$$

Expansion

E *Assume* $P \equiv \sum_i \alpha_i.P_i$ and $Q \equiv \sum_j \beta_j.Q_j$, where no α_i (resp. β_j) binds a name free in Q (resp. P); then *infer*

$$P \mid Q \;=\; \sum_i \alpha_i.(P_i \mid Q) + \sum_j \beta_j.(P \mid Q_j) + \sum_{\alpha_i \,\mathbf{comp}\, \beta_j} \tau.R_{ij}$$

where the relation $\alpha_i \,\mathrm{comp}\, \beta_j$ (α_i *complements* β_j) holds in four cases:

1. α_i is $\overline{x}u$ and β_j is $x(v)$; then R_{ij} is $P_i \mid Q_j\{u/v\}$.
2. α_i is $\overline{x}(u)$ and β_j is $x(v)$; then R_{ij} is $(w)(P_i\{w/u\} \mid Q_j\{w/v\})$, where w is not free in $(u)P_i$ or in $(v)Q_j$.
3. α_i is $x(v)$ and β_j is $\overline{x}u$; then R_{ij} is $P_i\{u/v\} \mid Q_j$.
4. α_i is $x(v)$ and β_j is $\overline{x}(u)$; then R_{ij} is $(w)(P_i\{w/v\} \mid Q_j\{w/u\})$, where w is not free in $(v)P_i$ or in $(u)Q_j$.

Identifier

I *From* $A(\tilde{x}) \stackrel{\mathrm{def}}{=} P$ *infer* $A(\tilde{y}) = P\{\tilde{y}/\tilde{x}\}$

This completes the definition of **SGE**. □

If $P = Q$ can be proved in **SGE** we write

$$\mathbf{SGE} \vdash P = Q$$

or just $\vdash P = Q$

Theorem 19 (Soundness) If $\mathbf{SGE} \vdash P = Q$ then $P \stackrel{\cdot}{\sim} Q$

Proof: The soundness of all laws in **SGE** has been established in Section 3. □

We will next prove that **SGE** admits a natural head normal form, and that it is complete for finite agents.

Definition 23 The agent identifier A is *weakly-guardedly defined* if every agent identifier is weakly guarded in the right-hand side of the definition of A.

Definition 24 An agent P is in *head normal form* if it is a sum of prefixes:

$$P \equiv \sum_i \alpha_i.P_i \qquad \Box$$

The following shows the importance of head normal form:

Lemma 8 If every agent identifier is weakly-guardedly defined then, for any agent P, there is a head normal form H such that

$$\mathbf{SGE} \vdash P = H$$

Proof: By the assumption that every agent identifier is weakly-guardedly defined, we may work by induction on the structure of P. The case when P is an agent identifier follows from **I** above, while if P is a prefix form then P is in head normal form. If $P \equiv P_1 + P_2$ and H_1, H_2 are head normal forms such that $\vdash P_1 = H_1$ and $\vdash P_2 = H_2$, then $\vdash P = H$ where $H \equiv H_1 + H_2$. If $P \equiv [x = y]Q$ and $\vdash Q = H$, then since either $\vdash P = Q$ or $\vdash P = \mathbf{0}$, the result follows. If $P \equiv (y)Q$ and $\vdash Q = H$ then $\vdash P = (y)H$, so since using **R2**–**R4**, $\vdash (y)H = H'$ for some head normal form H', the result follows. If $P \equiv P_1 \mid P_2$ and $\vdash P_1 = H_1$ and $\vdash P_2 = H_2$ then $\vdash P = H_1 \mid H_2$, so since using **E**, $\vdash H_1 \mid H_2 = H$ for some head normal form H, the result follows. $\Box$

From this, it is a not hard to show that **SGE** is complete for strong ground equivalence of finite agents.

Theorem 20 (Completeness for finite agents) For all finite agents P and Q, if $P \mathrel{\dot\sim} Q$ then $\mathbf{SGE} \vdash P = Q$.

Proof: By the preceding two results it suffices to establish the claim when both P and Q are in head normal form. If $R \equiv \Sigma_{i=1}^{k} \alpha_i.\, R_i$ is in head normal form then the *depth*, $d(R)$, of R is 0 if $k = 0$ and $1 + \max\{d(R_i) \mid 1 \leq i \leq k\}$ otherwise. We prove the result by induction on $d = d(P) + d(Q)$. If $d = 0$ then $P \equiv \mathbf{0}$ and $Q \equiv \mathbf{0}$ and the result is immediate. Suppose $d > 0$.

If $\alpha.\, M$ is a summand of P with α a free action, then since $P \xrightarrow{\alpha} M$ and Q is in head normal form there is a summand $\alpha.\, N$ of Q such that $M \mathrel{\dot\sim} N$. By induction hypothesis, $\vdash M = N$, and so $\vdash \alpha.\, M = \alpha.\, N$.

Suppose that $x(y).\, M$ is a summand of P. Then choosing $z \notin \mathrm{n}(P, Q)$, $P \xrightarrow{x(z)} M' \equiv M\{z/y\}$. Hence there is a summand $x(w).\, N$ of Q such that for all v, $M'\{v/z\} \mathrel{\dot\sim} N'\{v/z\}$ where $N' \equiv N\{z/w\}$. Then by induction hypothesis for all v, $\vdash M'\{v/z\} = N'\{v/z\}$. So by **C1**, $\vdash x(z).\, M' = x(z).\, N'$, and hence by **A**, since $x(z).\,M' \equiv x(y).\,M$ and $x(z).\,N' \equiv x(w).\,N$, $\vdash x(y).\,M = x(w).\,N$.

Suppose that $\overline{x}(y).\,M$ is a summand of P. Then choosing $z \notin \mathrm{n}(P, Q)$, $P \xrightarrow{\overline{x}(z)} M' \equiv M\{z/y\}$. Hence there is a summand $\overline{x}(w).\,N$ of Q such that $M' \dot{\sim} N'$ where $N' \equiv N\{z/w\}$. Then by induction hypothesis $\vdash M' = N'$ and so $\vdash \overline{x}(z).\,M' = \overline{x}(z).\,N'$, and hence by **A**, since $\overline{x}(z).\,M' \equiv \overline{x}(y).\,M$ and $\overline{x}(z).\,N' \equiv \overline{x}(w).\,N$, $\vdash \overline{x}(y).\,M = \overline{x}(w).\,N$.

Similarly, for each summand $\alpha.\,N$ of Q there is a summand $\beta.\,M$ of P such that $\vdash \beta.\,M = \alpha.\,N$. The result follows by **S0**–**S3**. $\square$

With this result we easily obtain a complete axiomatization of strong D-equivalence by adding the following law:

D *From* $P\sigma = Q\sigma$, for all σ respecting D, *infer* $P =_D Q$

(A more refined formulation of rule **D** actually confines the hypothesis to finitely many distinct σ.)

Theorem 21 $\mathbf{SGE} \cup \{\mathbf{D}\}$ is sound, and complete over finite agents, when $=$ and $=_D$ are interpreted as $\dot{\sim}$ and $\sim_D$ respectively.

Proof: Directly from Definition 12 and Theorem 20. $\square$

Thus strong equivalence (with the pleasant property of being a congruence) is given an indirect axiomatization in terms of strong bisimilarity (which is not preserved by positive prefix). We leave the problem of axiomatizing strong equivalence directly as a topic of further investigation. At first it might appear that such a direct axiomatization can be obtained from **SGE** (omitting **M0** and **E** which are not valid for $\sim$) by adding appropriate laws from Section 4.1. Unfortunately this is not the case. There are equations involving matching, such as

$$[x=y][y=z]P \;\sim\; [x=y][x=z]P$$

which we are presently unable to derive without **D**.

References

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

[2] Milner, R., Parrow, J. and Walker, D.J., *A calculus of Mobile Processes, Part I*, Report ECS-LFCS-89-85, Laboratory for Foundations of Computer Science, Computer Science Department, Edinburgh University, 1989. To appear in *Information and Computation*.

[3] Milner, R., Parrow, J. and Walker, D.J., *A calculus of Mobile Processes, Part II*, Report ECS-LFCS-89-86, Laboratory for Foundations of Computer Science, Computer Science Department, Edinburgh University, 1989.

Appendix

In this Appendix we outline the proofs of some of the results stated in the text; most of the proofs are by case analysis, and we give the argument for a few crucial or typical cases. Full proofs may be found in [3].

Proof of Lemma 1: The proof is by induction on depth of inference. We consider in turn each transition rule as the last rule applied in the inference of the antecedent $P \xrightarrow{\alpha} P'$. We give two cases.

(INPUT-ACT) Then $\alpha = x(y)$ and $P \equiv x(z).P_1$ with $y \notin \mathrm{fn}((z)P_1)$ and $P' \equiv P_1\{y/z\}$, so (i) holds and (ii) $\mathrm{fn}(P') \subseteq (\mathrm{fn}(P_1) - \{z\}) \cup \{y\} \subseteq \mathrm{fn}(P) \cup \{y\}$.

(CLOSE) Then $\alpha = \tau$ and $P \equiv P_1 \mid P_2$ with $P_1 \xrightarrow{\overline{x}(y)} P_1'$, $P_2 \xrightarrow{x(y)} P_2'$ and $P' \equiv (y)(P_1' \mid P_2')$, so (i) holds, and $\mathrm{fn}(P_1') \subseteq \mathrm{fn}(P_1) \cup \{y\}$ and $\mathrm{fn}(P_2') \subseteq \mathrm{fn}(P_2) \cup \{y\}$, so $\mathrm{fn}(P') = (\mathrm{fn}(P_1') \cup \mathrm{fn}(P_2')) - \{y\} \subseteq \mathrm{fn}(P)$. □

Lemmas 2–5 are all similarly proved by induction on depth of inference. Theorem 1 follows easily from the lemmas.

Proof of Lemma 6: Let $\mathcal{S} = \bigcup_{n<\omega} \mathcal{S}_n$ where

$$\begin{array}{rcl} \mathcal{S}_0 & = & \dot{\sim} \\ \mathcal{S}_{n+1} & = & \{(P\{w/z\}, Q\{w/z\}) \mid P\mathcal{S}_nQ,\ w \notin \mathrm{fn}(P,Q)\} \end{array}$$

We show that $\mathcal{S}$ is a strong bisimulation by showing by induction on n that if $P\mathcal{S}_nQ$ then

1. if α is a free action and $P \xrightarrow{\alpha} P'$ then for some Q', $Q \xrightarrow{\alpha} Q'$ and $P'\mathcal{S}Q'$,

2. if $y \notin \mathrm{fn}(P,Q)$ and $P \xrightarrow{x(y)} P'$ then for some Q', $Q \xrightarrow{x(y)} Q'$ and for all v, $P'\{v/y\}\mathcal{S}Q'\{v/y\}$,

3. if $y \notin \mathrm{fn}(P,Q)$ and $P \xrightarrow{\overline{x}(y)} P'$ then for some Q', $Q \xrightarrow{\overline{x}(y)} Q'$ and $P'\mathcal{S}Q'$.

If $n = 0$ then 1, 2 and 3 hold since $\mathcal{S}_0 = \dot{\sim}$.

Suppose $n > 0$ and that $P\sigma\mathcal{S}_nQ\sigma$ where $P\mathcal{S}_{n-1}Q$ and $\sigma = \{w/z\}$ where $w \notin \mathrm{fn}(P,Q)$. We consider only 3.

Suppose that $P\sigma \xrightarrow{\overline{x}(y)} P'$ where $y \notin \mathrm{fn}(P\sigma, Q\sigma)$. Choose $y' \notin \mathrm{n}(P,Q,w,z)$. Then $P\sigma \xrightarrow{\overline{x}(y')} P'' \equiv P'\{y'/y\}$. Hence by Lemma 4 for some P'' and x' with

$P'''\sigma \equiv P''$ and $x'\sigma = x$, $P \xrightarrow{\overline{x'}(y')} P'''$. Since $P\mathcal{S}_{n-1}Q$ and $y' \notin \mathrm{n}(P,Q)$ for some Q''', $Q \xrightarrow{\overline{x'}(y')} Q'''$ and $P'''\mathcal{S}Q'''$. Hence $Q\sigma \xrightarrow{\overline{x}(y')} Q'' \equiv Q'\sigma$, and so $Q\sigma \xrightarrow{\overline{x}(y)} Q' \equiv Q''\{y/y'\}$. Then

$$\begin{array}{rcll} P' & \equiv & P'''\{w/z\}\{y/y'\} & \\ & \mathcal{S} & Q'''\{w/z\}\{y/y'\} & \text{since } y \notin \mathrm{fn}(P'''\{w/z\}, Q'''\{w/z\}) \\ & \equiv & Q' & \end{array}$$

□

Proof of Lemma 7: Let $\mathcal{S}^* = \bigcup_{n<\omega} \mathcal{S}_n$ where

$$\begin{array}{rcl} \mathcal{S}_0 & = & \mathcal{S} \\ \mathcal{S}_{n+1} & = & \{((w)P,(w)Q) \mid P\mathcal{S}_nQ,\ w \in \mathcal{N}\} \end{array}$$

The proof involves showing that $\mathcal{S}^*$ is a strong bisimulation. First we note that by induction on n, if $P\mathcal{S}_nQ$ and $w \notin \mathrm{fn}(P,Q)$, then $P\{w/z\}\mathcal{S}_nQ\{w/z\}$. For $n = 0$ this is immediate from the definition. Suppose $n > 0$ and $(v)P\mathcal{S}_n(v)Q$ where $P\mathcal{S}_{n-1}Q$ and $w \notin \mathrm{fn}((v)P,(v)Q)$. Then $((v)P)\{w/z\} \equiv (u)P\{u/v\}\{w/z\}$ and $((v)Q)\{w/z\} \equiv (u)Q\{u/v\}\{w/z\}$ where $u \notin \mathrm{fn}((v)P,(v)Q,w)$ and $u\{w/z\} = u$, so $(v)P\{w/z\}\mathcal{S}_n(v)Q\{w/z\}$.

Next we show by induction on n that if $P\mathcal{S}_nQ$ then

1. if α is a free action and $P \xrightarrow{\alpha} P'$ then for some Q', $Q \xrightarrow{\alpha} Q'$ and $P'\mathcal{S}^*Q'$,
2. if $y \notin \mathrm{n}(P,Q)$ and $P \xrightarrow{x(y)} P'$ then for some Q', $Q \xrightarrow{x(y)} Q'$ and for all v, $P'\{v/y\}\mathcal{S}^*Q'\{v/y\}$,
3. if $y \notin \mathrm{n}(P,Q)$ and $P \xrightarrow{\overline{x}(y)} P'$ then for some Q', $Q \xrightarrow{\overline{x}(y)} Q'$ and $P'\mathcal{S}^*Q'$.

For $n = 0$ this is immediate from the fact that $\mathcal{S}_0$ is a strong bisimulation up to restriction and the definition of $\mathcal{S}^*$. The remaining details are omitted.

□

Proof of Theorem 2:

(a) That $\dot{\sim}$ is both reflexive and symmetric is clear. For transitivity it suffices to show that $\dot{\sim}\,\dot{\sim}$ is a strong bisimulation. The proof uses Lemma 2. We give one case.

Suppose that $y \notin \mathrm{n}(P,R)$ and $P \xrightarrow{x(y)} P'$. Choose $z \notin \mathrm{n}(P,Q,R)$. Then $P \xrightarrow{x(z)} P'' \equiv P'\{z/y\}$, so for some Q', $Q \xrightarrow{x(z)} Q'$ and for all w, $P''\{w/z\} \dot{\sim} Q'\{w/z\}$. Hence for some R', $R \xrightarrow{x(z)} R'$ and for all w, $Q'\{w/z\} \dot{\sim} R'\{w/z\}$. Then $R \xrightarrow{x(y)} R'' \equiv R'\{y/z\}$ and for all w, $P'\{w/y\} \dot{\sim}\,\dot{\sim} R''\{w/y\}$.

(b) For the congruence properties note that:

(1) $\{(\alpha.\,P, \alpha.\,Q) \mid P \dot{\sim} Q\} \cup \dot{\sim}$ is a strong bisimulation.

(2) $\{(P + R, Q + R) \mid P \dot{\sim} Q\} \cup \dot{\sim}$ is a strong bisimulation.

(3) $\{([x\!=\!y]P, [x\!=\!y]Q) \mid P \dot{\sim} Q\} \cup \dot{\sim}$ is a strong bisimulation.

(4) Let $\mathcal{S} = \{(P|R, Q|R) \mid P \dot{\sim} Q\}$. It suffices by Lemma 7 to show that $\mathcal{S}$ is a strong bisimulation up to restriction. To see this note first that if $P \dot{\sim} Q$ and $w \notin \mathrm{fn}(P, Q)$ then by Lemma 6, $P\{w/z\} \dot{\sim} Q\{w/z\}$ and so $(P|R)\{w/z\}\mathcal{S}(Q|R)\{w/z\}$. It is routine to check that the clauses concerning transitions hold. The only rules applicable are PAR, COM and CLOSE.

(5) It follows from Lemma 6 that $\dot{\sim}$ is a strong bisimulation up to restriction. Hence by the proof of Lemma 7, if $P \dot{\sim} Q$ then $(w)P \dot{\sim} (w)Q$.

(c) Note that $\{(x(y).\,P, x(y).\,Q) \mid$ for all $w \in \mathrm{fn}(P, Q, y)$, $P\{w/y\} \dot{\sim} Q\{w/y\}\}$ is a strong bisimulation. This follows easily using Lemma 6. □

Proof of Theorem 8: The proofs of Theorem 8 (a) and Theorem 8 (b) are straightforward. In contrast, the proofs of Theorem 8 (c) and (d) are not short.

Proof of Theorem 8 (c): In the proof we make use of the idea of a *strong bisimulation up to $\dot{\sim}$ and restriction*. For completeness we introduce first the following concept.

Definition 25 A relation $\mathcal{S}$ is a *strong simulation up to $\dot{\sim}$* iff whenever $P\mathcal{S}Q$ then

1. if α is a free action and $P \xrightarrow{\alpha} P'$ then for some Q', $Q \xrightarrow{\alpha} Q'$ and $P' \dot{\sim} \mathcal{S} \dot{\sim} Q'$,

2. if $y \notin \mathrm{n}(P, Q)$ and $P \xrightarrow{x(y)} P'$ then for some Q', $Q \xrightarrow{x(y)} Q'$ and for all w, $P'\{w/y\} \dot{\sim} \mathcal{S} \dot{\sim} Q'\{w/y\}$,

3. if $y \notin \mathrm{n}(P, Q)$ and $P \xrightarrow{\overline{x}(y)} P'$ then for some Q', $Q \xrightarrow{\overline{x}(y)} Q'$ and $P' \dot{\sim} \mathcal{S} \dot{\sim} Q'$.

$\mathcal{S}$ is a *strong bisimulation up to $\dot{\sim}$* iff both $\mathcal{S}$ and $\mathcal{S}^{-1}$ are strong simulations up to $\dot{\sim}$.

Lemma 9 If $\mathcal{S}$ is a strong bisimulation up to $\dot{\sim}$ then $\mathcal{S} \subseteq \dot{\sim}$.

Proof: Let $\mathcal{S}^* = \bigcup_{n<\omega} \mathcal{S}_n$ where

$$\begin{aligned} \mathcal{S}_0 &= \dot{\sim}\, \mathcal{S}\, \dot{\sim} \\ \mathcal{S}_{n+1} &= \{(P\{w/z\}, Q\{w/z\}) \mid P\mathcal{S}_n Q,\ w \notin \mathrm{fn}(P, Q)\} \end{aligned}$$

Then by an argument very similar to that in the proof of Lemma 6 it can be shown that S^* is a strong bisimulation. We omit the details. $\square$

Combining this concept with that of a strong bisimulation up to restriction we obtain the following.

Definition 26 A relation $\mathcal{S}$ is a *strong simulation up to* $\dot\sim$ *and restriction* iff whenever $P\mathcal{S}Q$ then

1. if $w \notin \mathrm{fn}(P,Q)$ then $P\{w/z\}\mathcal{S}Q\{w/z\}$,
2. if $P \xrightarrow{\overline{x}y} P'$ then for some Q', $Q \xrightarrow{\overline{x}y} Q'$ and $P' \dot\sim \mathcal{S} \dot\sim Q'$,
3. if $y \notin \mathrm{n}(P,Q)$ and $P \xrightarrow{x(y)} P'$ then for some Q', $Q \xrightarrow{x(y)} Q'$ and for all w, $P'\{w/y\} \dot\sim \mathcal{S} \dot\sim Q'\{w/y\}$,
4. if $y \notin \mathrm{n}(P,Q)$ and $P \xrightarrow{\overline{x}(y)} P'$ then for some Q', $Q \xrightarrow{\overline{x}(y)} Q'$ and $P' \dot\sim \mathcal{S} \dot\sim Q'$,
5. if $P \xrightarrow{\tau} P'$ then for some Q', $Q \xrightarrow{\tau} Q'$ and either $P' \dot\sim \mathcal{S} \dot\sim Q'$ or for some P'', Q'' and w, $P' \dot\sim (w)P''$, $Q' \dot\sim (w)Q''$ and $P''\mathcal{S}Q''$.

$\mathcal{S}$ is a *strong bisimulation up to* $\dot\sim$ *and restriction* iff both $\mathcal{S}$ and $\mathcal{S}^{-1}$ are strong simulations up to $\dot\sim$ and restriction.

We have the following result.

Lemma 10 If $\mathcal{S}$ is a strong bisimulation up to $\dot\sim$ and restriction then $\mathcal{S} \subseteq \dot\sim$.

Proof: Let $\mathcal{S}^* = \bigcup_{n<\omega} \mathcal{S}_n$ where

$$\begin{array}{rcl} \mathcal{S}_0 & = & \dot\sim \mathcal{S} \dot\sim \\ \mathcal{S}_{n+1} & = & \dot\sim \{((w)P,(w)Q) \mid P\mathcal{S}_nQ,\, w \in \mathcal{N}\} \dot\sim \end{array}$$

Then by an argument similar to that in the proof of Lemma 7 it may be shown that $\mathcal{S}^*$ is a strong bisimulation. We omit the details. $\square$

Returning to the main proof of Theorem 8 (c), we prove that the relation

$$\mathcal{S} \;=\; \{((y)P_1 \mid P_2,\ (y)(P_1 \mid P_2)) \mid\ P_1, P_2 \text{ agents},\ y \notin \mathrm{fn}(P_2)\} \ \cup\ \mathbf{Id}$$

is a strong bisimulation up to $\dot\sim$ and restriction. Thus, for each P and Q such that $P\mathcal{S}Q$ and each transition $P \xrightarrow{\alpha} P'$, we must find a "simulating" transition $Q \xrightarrow{\alpha} Q'$ satisfying the requirements of a strong simulation up to restriction and equivalence, and vice versa. Clearly, if $P \equiv Q$ this is trivial, so we assume that $P \equiv (y)P_1 \mid P_2$, $Q \equiv (y)(P_1 \mid P_2)$, and $y \notin \mathrm{fn}(P_2)$.

The proof that there always exists an appropriate transition $Q \equiv (y)(P_1 \mid P_2) \xrightarrow{\alpha} Q'$ is by a case analysis on how the transition $P \equiv (y)P_1 \mid P_2 \xrightarrow{\alpha} P'$ is derived, and vice versa. There are sixteen cases in all from which we draw a sample of two.

For each case the derivations of transitions from P and Q are presented in the following way:

$$\boxed{\dfrac{\vdots}{(y)P_1 \mid P_2 \xrightarrow{\alpha} P'}}$$

$$\Updownarrow$$

$$\boxed{\dfrac{\vdots}{(y)(P_1 \mid P_2) \xrightarrow{\alpha} Q'}}$$

We then have to prove three things:

($\Downarrow$): that the premises of the upper derivation imply the premises of the lower derivation;

($\Uparrow$): conversely that the premises of the lower derivation imply the premises of the upper derivation;

($\mathcal{S}$): that the derivatives P' and Q' satisfy the requirement of a strong bisimulation up $\dot{\sim}$ and restriction.

Note that by the definition of strong simulation we only have to consider α such that $y \notin \mathrm{bn}(\alpha)$, since y occurs in the agents P and Q.

Case :

$$\boxed{\text{COM}: \dfrac{\text{RES}: \dfrac{P_1 \xrightarrow{x(z)} P_1' \qquad x, z \neq y}{(y)P_1 \xrightarrow{x(z)} (y)P_1'} \qquad P_2 \xrightarrow{\overline{x}v} P_2'}{(y)P_1 \mid P_2 \xrightarrow{\tau} ((y)P_1')\{v/z\} \mid P_2'}}$$

$$\Updownarrow$$

$$\boxed{\text{RES}: \dfrac{\text{COM}: \dfrac{P_1 \xrightarrow{x(z)} P_1' \qquad P_2 \xrightarrow{\overline{x}v} P_2'}{P_1 \mid P_2 \xrightarrow{\tau} P_1'\{v/z\} \mid P_2'}}{(y)(P_1 \mid P_2) \xrightarrow{\tau} (y)(P_1'\{v/z\} \mid P_2')}}$$

($\Downarrow$): Trivial.
($\Uparrow$): From $y \notin \mathrm{fn}(P_2)$ and Lemma 1 we get that $x \neq y$. We cannot prove that $z \neq y$, but if $z = y$ then we use a fresh z' instead of z to get a simulating transition as follows: from Lemma 2 we get that $P_1 \xrightarrow{x(z')} P_1'\{z'/y\}$. The simulating transition then is:

$$(y)P_1 \mid P_2 \xrightarrow{\tau} ((y)P_1'\{z'/y\})\{v/z'\} \mid P_2' \qquad (*)$$

($\mathcal{S}$): From $v, z \neq y$ it follows that $((y)P_1')\{v/z\} \equiv (y)P_1'\{v/z\}$, and Lemma 1 with $y \notin \mathrm{fn}(P_2)$ gives that $y \notin \mathrm{fn}(P_2')$, so

$$((y)P_1')\{v/z\} \mid P_2' \quad \mathcal{S} \quad (y)(P_1'\{v/z\} \mid P_2')$$

as required. For the simulating transition $(*)$ we know that $z = y$, so it holds (since $v \neq y$ and z' is chosen fresh) that

$$((y)P_1\{z'/y\})\{v/z'\} \mid P_2' \quad \equiv \quad (y)P_1'\{v/y\} \mid P_2' \quad \mathcal{S} \quad (y)(P_1'\{v/y\} \mid P_2')$$

Case :

$$\boxed{\text{COM}: \dfrac{\text{RES}: \dfrac{P_1 \xrightarrow{\overline{x}v} P_1' \qquad x, v \neq y}{(y)P_1 \xrightarrow{\overline{x}v} (y)P_1'} \qquad P_2 \xrightarrow{x(z)} P_2'}{(y)P_1 \mid P_2 \xrightarrow{\tau} (y)P_1' \mid P_2'\{v/z\}}}$$

$$\Updownarrow$$

$$\boxed{\text{RES}: \dfrac{\text{COM}: \dfrac{P_1 \xrightarrow{\overline{x}v} P_1' \qquad P_2 \xrightarrow{x(z)} P_2'}{P_1 \mid P_2 \xrightarrow{\tau} P_1' \mid P_2'\{v/z\}}}{(y)(P_1 \mid P_2) \xrightarrow{\tau} (y)(P_1' \mid P_2'\{v/z\})}}$$

($\Downarrow$): Trivial.
($\Uparrow$): From Lemma 1 and $y \notin \mathrm{fn}(P_2)$ we get $x \neq y$. The situation when $v = y$ is treated in another case (see [3]).
($\mathcal{S}$): From Lemma 1 and $y \notin \mathrm{fn}(P_2)$ we get that $y = z$ or $y \notin \mathrm{fn}(P_2')$, so from $v \neq y$ it follows $y \notin \mathrm{fn}(P_2'\{v/z\})$. This proves as required

$$(y)P_1' \mid P_2'\{v/z\} \quad \mathcal{S} \quad (y)(P_1' \mid P_2'\{v/z\})$$

$\square$

Proof of Theorem 8 (d): The proof involves showing that the relation

$$\mathcal{S} \quad = \quad \{((P_1 \mid P_2) \mid P_3,\ \ P_1 \mid (P_2 \mid P_3)) \mid \ P_1, P_2, P_3 \text{ agents}\}$$

is a strong bisimulation up to $\dot{\sim}$ and restriction. Thus, for each P and Q such that $P\mathcal{S}Q$ and each transition $P \xrightarrow{\alpha} P'$ we must find a simulating transition $Q \xrightarrow{\alpha} Q'$ satisfying the requirements of a strong simulation up to $\dot{\sim}$ and restriction, and vice versa.

The proof that there always exists an appropriate transition $Q \xrightarrow{\alpha} Q'$ is by a case analysis on how the transition $P \xrightarrow{\alpha} P'$ is derived, and vice versa. There are 30 cases in total. We present one sample case in the same style as in the proof of Theorem 8 (c).

Case :

$$\boxed{\begin{array}{ll} & \dfrac{\dfrac{P_1 \xrightarrow{\overline{x}(z)} P_1' \qquad P_2 \xrightarrow{x(z)} P_2'}{P_1 \mid P_2 \xrightarrow{\tau} (z)(P_1' \mid P_2')}\ \text{CLOSE:}}{(P_1 \mid P_2) \mid P_3 \xrightarrow{\tau} (z)(P_1' \mid P_2') \mid P_3}\ \text{PAR :} \end{array}}$$

$$\Downarrow$$

$$\boxed{\dfrac{\dfrac{P_2 \xrightarrow{x(z')} P_2'\{z'/z\} \qquad z' \notin \text{fn}(P_3)}{P_2 \mid P_3 \xrightarrow{x(z')} P_2'\{z'/z\} \mid P_3}\ \text{PAR:} \qquad P_1 \xrightarrow{\overline{x}(z')} P_1'\{z'/z\}}{P_1 \mid (P_2 \mid P_3) \xrightarrow{\tau} (z')(P_1'\{z'/z\} \mid (P_2'\{z'/z\} \mid P_3))}\ \text{CLOSE :}}$$

($\Downarrow$): By Lemma 2 there exists a fresh z' such that $P_1 \xrightarrow{\overline{x}(z')} P_1'\{z'/z\}$ and $P_2 \xrightarrow{x(z')} P_2'\{z'/z\}$.
($\mathcal{S}$): Note that z' is a fresh name. By alpha-converting z to z' and then applying Theorem 8 (c) we get that

$$(z)(P_1' \mid P_2') \mid P_3 \quad \equiv \quad (z')(P_1'\{z'/z\} \mid P_2'\{z'/z\}) \mid P_3 \quad \dot{\sim} \quad (z')((P_1'\{z'/z\} \mid P_2'\{z'/z\}) \mid P_3)$$

so the condition for a simulation up to $\dot{\sim}$ and restriction is satisfied:

$$(P_1'\{z'/z\} \mid P_2'\{z'/z\}) \mid P_3 \quad \mathcal{S} \quad P_1'\{z'/z\} \mid (P_2'\{z'/z\} \mid P_3)$$

$\square$

Proof of Theorem 17: We first state some immediate consequences of the definition of replacement. If E is an agent expression and σ a substitution of names, then $E\sigma$ is defined to be the agent expression obtained in the way analogous to Definition 3. Then substitutions of names as expected commute with replacements in the following way: $E(A_1, \ldots A_n)\sigma \equiv E\sigma(A_1, \ldots A_n)$. Also, since replacement clearly distributes over the operators we have that Theorem 2 generalizes to agent expressions. These facts will be used freely in what follows.

We will only prove the theorem for $I = \{1\}$. The proof of the general case is similar and only notationally more cumbersome. We write $E, F, A, B, X, \tilde{x}$ for $E_1, F_1, A_1, B_1, X_1, \tilde{x}_1$. Assuming the premises of the theorem, define the relation $\mathcal{S}$ by

$$\mathcal{S} \quad = \quad \{(G(A), G(B)) : G \text{ has only the schematic identifier } X\}$$

We show that $\mathcal{S}$ is a strong bisimulation up to $\dot{\sim}$. By Lemma 9 it follows that $\mathcal{S} \subseteq \dot{\sim}$. By choosing $G \equiv X(\tilde{y})$ we then get that $A(\tilde{y}) \dot{\sim} B(\tilde{y})$; since this holds for any names $\tilde{y}$ it implies that $A(\tilde{x})\sigma \dot{\sim} B(\tilde{x})\sigma$ for any σ, which amounts to $A(\tilde{x}) \sim B(\tilde{x})$.

To prove $\mathcal{S}$ a strong bisimulation up to $\dot{\sim}$ it is clearly enough to prove the following properties, which we will call $(*)$:

1. If $G(A) \xrightarrow{\alpha} P'$ and α is a free action or bound output action with $\mathrm{bn}(\alpha) \cap \mathrm{n}(G(A), G(B)) = \emptyset$, then $G(B) \xrightarrow{\alpha} Q''$ with $P' \mathcal{S} \dot{\sim} Q''$.

2. If $G(A) \xrightarrow{x(y)} P'$ and $y \notin \mathrm{n}(G(A), G(B))$ then $G(B) \xrightarrow{x(y)} Q''$ such that for all u, $P'\{u/y\} \mathcal{S} \dot{\sim} Q''\{u/y\}$.

So assume $G(A) \xrightarrow{\alpha} P'$; we will prove $(*)$ by induction on the depth of the inference of this transition. We argue by cases on how the last step in this transition is inferred. We give two sample cases.

Case : The transition $G(A) \xrightarrow{\alpha} P'$ is inferred with the rule IDE. Then $G(A) \equiv C(\tilde{y})$ for some identifier C. There are two subcases: either $G \equiv C(\tilde{y})$ or $G \equiv X(\tilde{y})$. In the first subcase $G(A) \equiv G(B)$, so $(*)$ is immediate. Consider the second subcase $G \equiv X(\tilde{y})$. Then $G(A) \equiv A(\tilde{y}) \xrightarrow{\alpha} P'$. Then by a shorter inference, $E(A)\{\tilde{y}/\tilde{x}\} \equiv E\{\tilde{y}/\tilde{x}\}(A) \xrightarrow{\alpha} P'$.

Consider first the subsubcase where α is a free action or a bound output action. We only have to consider α such that $\mathrm{bn}(\alpha) \cap \mathrm{n}(G(A), G(B)) = \emptyset$. By definition, then $\mathrm{bn}(\alpha) \cap \mathrm{n}(E\{\tilde{y}/\tilde{x}\}(A), E\{\tilde{y}/\tilde{x}\}(B)) = \emptyset$, so by induction, $E\{\tilde{y}/\tilde{x}\}(B) \xrightarrow{\alpha} Q''$ with $P' \mathcal{S} \dot{\sim} Q'$. Since $E \sim F$ it follows that $E\{\tilde{y}/\tilde{x}\}(B) \dot{\sim} F\{\tilde{y}/\tilde{x}\}(B)$; hence $F\{\tilde{y}/\tilde{x}\}(B) \xrightarrow{\alpha} Q''' \dot{\sim} Q''$. So by the IDE rule, $G(B) \equiv B(\tilde{y}) \xrightarrow{\alpha} Q'''$. Since $\dot{\sim}$ is transitive, $P' \mathcal{S} \dot{\sim} Q'''$ as required.

Consider next the subsubcase where $\alpha = x(y)$ is an input action. We only have to consider $y \notin \mathrm{n}(G(A), G(B))$. By definition, then $y \notin \mathrm{n}(E\{\tilde{y}/\tilde{x}\}(A), E\{\tilde{y}/\tilde{x}\}(B))$, so by induction, $E\{\widetilde{y}/\widetilde{x}\}(B) \xrightarrow{\alpha} Q''$ with $P'\{u/y\}\mathcal{S} \mathrel{\dot{\sim}} Q''\{u/y\}$ for all u. Since $E \sim F$ it follows that $E\{\widetilde{y}/\widetilde{x}\}(B) \mathrel{\dot{\sim}} F\{\widetilde{y}/\widetilde{x}\}(B)$; hence $F\{\widetilde{y}/\widetilde{x}\}(B) \xrightarrow{\alpha} Q'''$ such that for all u, $Q'''\{u/y\} \mathrel{\dot{\sim}} Q''\{u/y\}$. By the IDE rule, $G(B) \equiv B(\widetilde{y}) \xrightarrow{\alpha} Q'''$. Since $\dot{\sim}$ is transitive, $P'\{u/y\}\mathcal{S} \mathrel{\dot{\sim}} Q'''\{u/y\}$ as required.

Case : The transition $G(A) \xrightarrow{\alpha} P'$ is inferred with the rule PAR. Then $G \equiv G_1 \mid G_2$, and by a shorter inference, $G_i(A) \xrightarrow{\alpha} P_i'$ for $i = 1$ or $i = 2$; assume $i = 1$ (the case $i = 2$ is symmetric). So, $P' \equiv P_1' \mid G_2(A)$ and $\mathrm{bn}(\alpha) \cap \mathrm{fn}(G_2(A)) = \emptyset$.

Consider first the subcase where α is a free action or a bound output action. We only have to consider α such that $\mathrm{bn}(\alpha) \cap \mathrm{n}(G(A), G(B)) = \emptyset$. So by induction, $G_1(B) \xrightarrow{\alpha} Q_1''$ with $P_1'\mathcal{S} \mathrel{\dot{\sim}} Q_1''$. Hence there exists an H' such that $P_1' \equiv H'(A)$ and $Q_1'' \mathrel{\dot{\sim}} H'(B)$. By PAR (remember $\mathrm{fn}(G_2(A)) = \mathrm{fn}(G_2(B))$) we get that $G(B) \equiv G_1(B) \mid G_2(B) \xrightarrow{\alpha} Q_1'' \mid G_2(B)$. Let $H \equiv H' \mid G_2$. Then $P' \equiv H(A)$ and $Q_1'' \mid G_2(B) \mathrel{\dot{\sim}} H(B)$, so $P'\mathcal{S} \mathrel{\dot{\sim}} Q_1'' \mid G_2(B)$ as required.

Consider next the subcase where $\alpha = x(y)$ is an input action. We only have to consider y such that $y \notin \mathrm{n}(G(A), G(B))$. So by induction, $G_1(B) \xrightarrow{\alpha} Q_1''$ with $P_1'\{u/y\}\mathcal{S} \mathrel{\dot{\sim}} Q_1''\{u/y\}$ for all u. Hence there exist H_u' such that $P_1'\{u/y\} \equiv H_u'(A)$ and $Q_1''\{u/y\} \mathrel{\dot{\sim}} H_u'(B)$. By PAR (remember $\mathrm{fn}(G_2(A)) = \mathrm{fn}(G_2(B))$) we get that $G(B) \equiv G_1(B) \mid G_2(B) \xrightarrow{\alpha} Q_1'' \mid G_2(B)$. Let $H_u \equiv H_u' \mid G_2$. Then $P'\{u/y\} \equiv (P_1' \mid G_2(A))\{u/y\} \equiv P_1'\{u/y\} \mid G_2(A) \equiv H(A)$ and $Q_1'' \mid G_2(B)\{u/y\} \equiv Q_1''\{u/y\} \mid G_2(B) \mathrel{\dot{\sim}} H_u(B)$, so $P'\{u/y\}\mathcal{S} \mathrel{\dot{\sim}} (Q_1'' \mid G_2(B))\{u/y\}$ for all u as required. $\square$