

Contents

| | | |
|----------|---|-----------|
| 0.1 | Abstract | 3 |
| 1 | TODO | 7 |
| 2 | Π calculus | 9 |
| 2.1 | Syntax | 9 |
| 2.2 | Operational Semantic(without structural congruence) | 11 |
| 2.2.1 | Early operational semantic(without structural congruence) | 11 |
| 2.2.2 | Late operational semantic(without structural congruence) | 13 |
| 2.3 | Structural congruence | 13 |
| 2.4 | Early semantic with α conversion only | 19 |
| 2.5 | Operational semantic with structural congruence | 19 |
| 2.5.1 | Early semantic with structural congruence | 19 |
| 2.5.2 | Late semantic with structural congruence | 20 |
| 2.5.3 | Permutation semantic | 21 |
| 2.6 | Equivalence of the semantics | 22 |
| 2.6.1 | Equivalence of the early semantics | 22 |
| 2.6.2 | Equivalence of the late semantics | 32 |
| 2.7 | Normal form | 32 |
| 2.8 | Bisimilarity, congruence and equivalence | 33 |
| 2.8.1 | Late bisimilarity | 33 |
| 2.8.2 | Early bisimilarity | 34 |
| 2.8.3 | Congruence | 34 |
| 2.8.4 | Open bisimilarity | 34 |
| 3 | Multi π calculus with strong output | 35 |
| 3.1 | Syntax | 35 |
| 3.2 | Operational semantic | 35 |
| 3.2.1 | Early operational semantic with structural congruence | 35 |
| 3.2.2 | Low level semantic | 36 |
| 3.2.3 | Early operational semantic without structural congruence | 45 |
| 3.3 | Strong bisimilarity and equivalence | 50 |
| 3.3.1 | Strong bisimilarity | 50 |
| 3.3.2 | Properties of strong early bisimilarity | 50 |
| 3.3.3 | Strong D equivalence | 56 |
| 3.3.4 | Open bisimulation | 58 |
| 4 | Multi π calculus with strong input | 59 |
| 4.1 | Syntax | 59 |
| 4.2 | Operational semantic | 59 |
| 4.2.1 | Early operational semantic with structural congruence | 59 |
| 4.2.2 | Late operational semantic with structural congruence | 60 |
| 4.2.3 | Low level semantic | 64 |
| 4.2.4 | Semantic on normal forms | 68 |
| 4.3 | Strong bisimilarity and equivalence | 68 |

| | | |
|----------|---|-----------|
| 4.3.1 | Strong bisimilarity | 68 |
| 4.3.2 | Open bisimulation | 73 |
| 5 | Multi π calculus with strong input and output | 75 |
| 5.1 | Syntax | 75 |
| 5.2 | Operational semantic | 75 |
| 5.2.1 | Early operational semantic with structural congruence | 75 |
| 5.2.2 | Late operational semantic with structural congruence | 77 |
| 5.2.3 | Low level semantic | 80 |

Chapter 1

TODO

1. dimostrare(o negare) l'equivalenza del pi calcolo con e senza congruenza strutturale e con e senza alfa conversione. FATTO MA NON COME SPERATO.
2. nel multi pi calcolo con strong prefixing solo su input o solo su output: definire una semantica di basso livello sulla falsariga di quell'articolo. FATTO MA NON COME SPERATO. raggiungere un qualche risultato simile anche per multiInpOut
3. terminare la parte sulle bisimulazioni nel multiOut senza congruenza strutturale. fare una cosa simile anche per multiInp senza congruenza strutturale?
4. terminare la parte sulle bisimulazioni nel multiInp con congruenza strutturale. fare una cosa simile anche per multiOut con congruenza strutturale?
5. dare una semantica open step e provare a definire una bisimulazione open sulla semantica step. per multiOut con e senza congruenza strutturale e per multiInp con e senza congruenza strutturale
6. trovare la congruenza coarsest contenuta nella bisimulazione scelta in precedenza
7. ripetere i ragionamenti fatti in precedenza anche per multiInpOut

Chapter 2

Π calculus

The π calculus is a mathematical model of processes whose interconnections change as they interact. The basic computational step is the transfer of a communication link between two processes. The idea that the names of the links belong to the same category as the transferred objects is one of the cornerstone of the calculus. The π calculus allows channel names to be communicated along the channels themselves, and in this way it is able to describe concurrent computations whose network configuration may change during the computation.

A coverage of π calculus is on [4], [5] and [7]

2.1 Syntax

We suppose that we have a countable set of names \mathbb{N} , ranged over by lower case letters a, b, \dots, z . This names are used for communication channels and values. Furthermore we have a set of identifiers, ranged over by A . We represent the agents or processes by upper case letters P, Q, \dots . A process can perform the following actions:

$$\pi ::= \bar{x}y \mid x(z) \mid \tau$$

The process are defined by the following grammar:

$$P, Q ::= 0 \mid \pi.P \mid P|Q \mid P + Q \mid (\nu x)P \mid A(\tilde{x})$$

and they have the following intuitive meaning:

0 is the empty process which cannot perform any actions

$\pi.P$ is an action prefixing, this process can perform action π e then behave like P , the action can be:

$\bar{x}y$ is an output action, this sends the name y along the name x . We can think about x as a channel or a port, and about y as an output datum sent over the channel

$x(z)$ is an input action, this receives a name along the name x . z is a variable which stores the received data.

τ is a silent or invisible action, this means that a process can evolve to P without interaction with the environment

for any action which is not a τ , the first name that appears in the action is called subject of the action and the second name is called object of the action.

$P + Q$ is the sum, this process can enact either P or Q

$P|Q$ is the parallel composition, P and Q can execute concurrently and also synchronize with each other

| | |
|--|--|
| $B(0, I) = \emptyset$ | $B(Q + R, I) = B(Q, I) \cup B(R, I)$ |
| $B(\bar{x}y.Q, I) = B(Q, I)$ | $B(Q R, I) = B(Q, I) \cup B(R, I)$ |
| $B(x(y).Q, I) = \{y, \bar{y}\} \cup B(Q, I)$ | $B((\nu x)Q, I) = \{x, \bar{x}\} \cup B(Q, I)$ |
| $B(\tau.Q, I) = B(Q, I)$ | |
| $B(A, I) = \begin{cases} B(Q, I \cup \{A\}) & \text{where } A(\tilde{x}) \stackrel{def}{=} Q \text{ if } A \notin I \\ \emptyset & \text{if } A \in I \end{cases}$ | |

Table 2.1: Bound occurrences

| | | |
|---|--------------------------------|--|
| $fn(\bar{x}y.Q) = \{x, \bar{x}, y, \bar{y}\} \cup fn(Q)$ | $fn(Q + R) = fn(Q) \cup fn(R)$ | $fn(0) = \emptyset$ |
| $fn(x(y).Q) = \{x, \bar{x}\} \cup (fn(Q) - \{y, \bar{y}\})$ | $fn(Q R) = fn(Q) \cup fn(R)$ | |
| $fn((\nu x)Q) = fn(Q) - \{x, \bar{x}\}$ | $fn(\tau.Q) = fn(Q)$ | $\frac{A(\tilde{x}) \stackrel{def}{=} P}{fn(A) = \{\tilde{x}\}}$ |

Table 2.2: Free occurrences

$(\nu z)P$ is the scope restriction. This process behave as P but the name z is local. This process cannot use the name z to interact with other processes.

$A(\tilde{x})$ is an identifier. Every identifier has a definition

$$A(x_1, \dots, x_n) = P$$

the x_i s must be pairwise different. The intuition is that we can substitute for some of the x_i s in P to get a π calculus process. We can write \tilde{x} for x_1, \dots, x_n .

To resolve ambiguity we can use parenthesis and observe the conventions that prefixing and restriction bind more tightly than composition and prefixing binds more tightly than sum.

Definition 2.1.1. We say that the input prefix $x(z).P$ binds z in P or is a *binder* for z in P . We also say that P is the *scope* of the binder and that any occurrence of z in P are *bound* by the binder. Also the restriction operator $(\nu z)P$ is a binder for z in P .

Definition 2.1.2. $bn(P)$ is the set of names that have a bound occurrence in P and is defined as $B(P, \emptyset)$, where $B(P, I)$, with I a set of identifiers, is defined in table 2.1

Definition 2.1.3. We say that a name x is *free* in P if P contains a non bound occurrence of x . We write $fn(P)$ for the set of names with a free occurrence in P . $fn(P)$ is defined in table 2.2

Definition 2.1.4. $n(P)$ which is the set of all names in P and is defined in the following way:

$$n(P) = fn(P) \cup bn(P)$$

Definition 2.1.5. We say that τ and actions which does not have any binder, such as $xy, \bar{x}y$, are *free* actions. Whether the other actions are *bound* actions.

In a definition $A(\tilde{x}) = P$ the \tilde{x} are exactly the free names contained in P , specifically $fn(P) = \{\tilde{x}\}$. If we look at the definitions of bn and of fn we notice that if P contains another identifier whose definition is: $B(\tilde{z}) = Q$ then we have $fn(Q) \subseteq \{\tilde{x}\}$.

| | | |
|--|--|-----------------------------------|
| $0\{b/a\} = 0$ | $(\bar{x}y.Q)\{b/a\} = \bar{x}\{b/a\}y\{b/a\}.Q\{b/a\}$ | $(\tau.Q)\{b/a\} = \tau.Q\{b/a\}$ |
| $\frac{y \neq a \quad y \neq b}{(x(y).Q)\{b/a\} = x\{b/a\}(y).Q\{b/a\}}$ | $\frac{c \notin n(x(b).Q)}{(x(b).Q)\{b/a\} = x\{b/a\}(c).(Q\{c/b\})\{b/a\}}$ | |
| $(x(a).Q)\{b/a\} = x\{b/a\}(a).Q$ | | |
| $\frac{a \in \tilde{x} \quad A(\tilde{x}) \stackrel{def}{=} P}{A(\tilde{x})\{b/a\} = A(\tilde{x}\{b/a\})}$ | | |
| $(Q + R)\{b/a\} = Q\{b/a\} + R\{b/a\}$ | $(Q R)\{b/a\} = Q\{b/a\} R\{b/a\}$ | |
| $\frac{y \neq a \quad y \neq b}{((\nu y)Q)\{b/a\} = (\nu y)Q\{b/a\}}$ | $((\nu a)Q)\{b/a\} = (\nu a)Q$ | |
| $\frac{c \notin n((\nu b)Q) \quad a \in fn(Q)}{((\nu b)Q)\{b/a\} = (\nu c)((Q\{c/b\})\{b/a\})}$ | | |

Table 2.3: Syntactic substitution

Definition 2.1.6. $P\{b/a\}$ is the syntactic substitution of name b for a different name a inside a π calculus process and it consists in replacing every free occurrences of a with b . If b is a bound name in P , in order to avoid name capture we perform an appropriate α conversion. $P\{b/a\}$ is defined in table 2.3. We use the notation $\{\tilde{x}/\tilde{y}\}$ as a short for $\{x_1/y_1, \dots, x_n/y_n\}$ which is not the composition of the substitutions $\{x_1/y_1\} \circ \dots \circ \{x_n/y_n\}$

2.2 Operational Semantic(without structural congruence)

2.2.1 Early operational semantic(without structural congruence)

The semantic of a π calculus process is a labeled transition system such that:

- the nodes are π calculus process. The set of node is \mathbb{P}
- the actions can be:
 - unbound input xy
 - unbound output $\bar{x}y$
 - the silent action τ
 - bound output $\bar{x}(y)$

The set of actions is \mathbb{A} , we use α to range over the set of actions.

- the transition relations is $\rightarrow \subseteq \mathbb{P} \times \mathbb{A} \times \mathbb{P}$

Definition 2.2.1. The *early transition relation* $\rightarrow \subseteq \mathbb{P} \times \mathbb{A} \times \mathbb{P}$ is the smallest relation induced by the rules in table 2.4.

Example We show now an example of the so called scope extrusion, in particular we prove that

$$a(x).P \mid (\nu b)\bar{a}b.Q \xrightarrow{\tau} (\nu b)(P\{b/x\} \mid Q)$$

| | | |
|--|--|---|
| Out $\frac{}{\bar{x}y.P \xrightarrow{\bar{x}y} P}$ | EInp $\frac{}{x(y).P \xrightarrow{xz} P\{z/y\}}$ | Tau $\frac{}{\tau.P \xrightarrow{\tau} P}$ |
| SumL $\frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'}$ | SumR $\frac{Q \xrightarrow{\alpha} Q'}{P + Q \xrightarrow{\alpha} P'}$ | |
| ParL $\frac{P \xrightarrow{\alpha} P' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P Q \xrightarrow{\alpha} P' Q}$ | ParR $\frac{Q \xrightarrow{\alpha} Q' \quad bn(\alpha) \cap fn(P) = \emptyset}{P Q \xrightarrow{\alpha} P Q'}$ | |
| Res $\frac{P \xrightarrow{\alpha} P' \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha} (\nu z)P'}$ | | |
| ResAlp1 $\frac{(\nu w)P\{w/z\} \xrightarrow{\alpha} P' \quad w \notin n(P)}{(\nu z)P \xrightarrow{\alpha} P'}$ | ResAlp2 $\frac{P \xrightarrow{\alpha} P' \quad w \notin n(P)}{(\nu w)P\{w/z\} \xrightarrow{\alpha} (\nu w)P'}$ | |
| EComL $\frac{P \xrightarrow{xy} P' \quad Q \xrightarrow{\bar{x}y} Q'}{P Q \xrightarrow{\tau} P' Q'}$ | EComR $\frac{P \xrightarrow{\bar{x}y} P' \quad Q \xrightarrow{xy} Q'}{P Q \xrightarrow{\tau} P' Q'}$ | |
| ClsL $\frac{P \xrightarrow{\bar{x}(z)} P' \quad Q \xrightarrow{xz} Q' \quad z \notin fn(Q)}{P Q \xrightarrow{\tau} (\nu z)(P' Q')}$ | ClsR $\frac{P \xrightarrow{xz} P' \quad Q \xrightarrow{\bar{x}(z)} Q' \quad z \notin fn(P)}{P Q \xrightarrow{\tau} (\nu z)(P' Q')}$ | |
| Opn $\frac{P \xrightarrow{\bar{x}z} P' \quad z \neq x}{(\nu z)P \xrightarrow{\bar{x}(z)} P'}$ | OpnAlp $\frac{(\nu w)P\{w/z\} \xrightarrow{\bar{x}(w)} P' \quad w \notin n(P) \quad x \neq w \neq z}{(\nu z)P \xrightarrow{\bar{x}(w)} P'}$ | |
| Ide $\frac{A(\tilde{x}) \stackrel{def}{=} P \quad P\{\tilde{w}/\tilde{x}\} \xrightarrow{\alpha} P'}{A(\tilde{x})\{\tilde{w}/\tilde{x}\} \xrightarrow{\alpha} P'}$ | | |

Table 2.4: Early semantic without structural congruence and without explicit α conversion

where we suppose that $b \notin fn(P)$. In this example the scope of (νb) moves from the right hand component to the left hand.

$$\text{CLOSER} \frac{\text{EINP} \frac{}{a(x).P \xrightarrow{ab} P\{b/x\}} \quad \text{OPN} \frac{\text{OUT} \frac{}{\bar{a}b.Q \xrightarrow{\bar{a}b} Q} \quad a \neq b}{(\nu b)\bar{a}b.Q \xrightarrow{\bar{a}(b)} Q} \quad b \notin fn((\nu b)\bar{a}b.Q)}{a(x).P \mid (\nu b)\bar{a}b.Q \xrightarrow{\tau} (\nu b)(P\{b/x\} \mid Q)}$$

Example We want to prove now that:

$$((\nu b)a(x).P) \mid \bar{a}b.Q \xrightarrow{\tau} ((\nu c)(P\{c/b\}\{b/x\})) \mid Q$$

where $b \notin bn(P)$

$$\text{RESALP1} \frac{\text{RES} \frac{\text{EINP} \frac{}{(a(x).P)\{c/b\} \xrightarrow{ab} P\{c/b\}\{b/x\}} \quad c \notin n(a(b))}{(\nu c)((a(x).P)\{c/b\}) \xrightarrow{ab} (\nu c)(P\{c/b\}\{b/x\})} \quad b \notin n((a(x).P)\{c/b\})}{(\nu b)a(x).P \xrightarrow{ab} (\nu c)P\{c/b\}\{b/x\}}$$

$$\text{ECOML} \frac{(\nu b)a(x).P \xrightarrow{ab} (\nu c)P\{c/b\}\{b/x\} \quad \text{EOUT} \frac{}{\bar{a}b.Q \xrightarrow{\bar{a}b} Q}}{((\nu b)a(x).P) \mid \bar{a}b.Q \xrightarrow{\tau} ((\nu c)(P\{c/b\}\{b/x\})) \mid Q}$$

Example We have to spend some time to deal with the change of bound names in an identifier. Suppose we have

$$A(x) \stackrel{\text{def}}{=} \underbrace{x(y).x(a).0}_P$$

From the definition of substitution it follows that $A(x)\{y/x\} = A(y)$. The identifier $A(y)$ is expected to behave consistently with $P\{y/x\} = y(z).y(a).0$. So for example we have to prove that $A(y) \xrightarrow{yw} y(a).0$. We can prove this in the following way:

$$\text{Ide} \frac{A(x) \stackrel{\text{def}}{=} P \quad \text{EInp} \frac{}{P\{y/x\} \xrightarrow{yw} y(a).0}}{A(y) \xrightarrow{yw} y(a).0}$$

2.2.2 Late operational semantic(without structural congruence)

In this case the set of actions \mathbb{A} contains

- bound input $x(y)$
- unbound output $\bar{x}y$
- the silent action τ
- bound output $\bar{x}(y)$

Definition 2.2.2. The *late transition relation without structural congruence* $\rightarrow \subseteq \mathbb{P} \times \mathbb{A} \times \mathbb{P}$ is the smallest relation induced by the rules in table 2.5.

2.3 Structural congruence

Structural congruences are a set of equations defining equality and congruence relations on process. They can be used in combination with an SOS semantic for languages. In some cases structural congruences help simplifying the SOS rules: for example they can capture inherent properties of composition operators(e.g. commutativity, associativity and zero element). Also, in process calculi,

| | |
|--|---|
| LInp $\frac{z \notin fn(P)}{x(y).P \xrightarrow{x(z)} P\{z/y\}}$ | Res $\frac{P \xrightarrow{\alpha} P' \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha} (\nu z)P'}$ |
| SumL $\frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'}$ | SumR $\frac{Q \xrightarrow{\alpha} Q'}{P + Q \xrightarrow{\alpha} Q'}$ |
| ParL $\frac{P \xrightarrow{\alpha} P' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P Q \xrightarrow{\alpha} P' Q}$ | ParR $\frac{Q \xrightarrow{\alpha} Q' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P Q \xrightarrow{\alpha} P Q'}$ |
| ComL $\frac{P \xrightarrow{x(y)} P' \quad Q \xrightarrow{\bar{x}(z)} Q'}{P Q \xrightarrow{\tau} P'\{z/y\} Q'}$ | ComR $\frac{P \xrightarrow{\bar{x}(z)} P' \quad Q \xrightarrow{x(y)} Q'}{P Q \xrightarrow{\tau} P' Q'\{z/y\}}$ |
| Opn $\frac{P \xrightarrow{\bar{x}z} P' \quad z \neq x}{(\nu z)P \xrightarrow{\bar{x}(z)} P'}$ | Out $\frac{}{\bar{x}y.P \xrightarrow{\bar{x}y} P}$ |
| ClsL $\frac{P \xrightarrow{\bar{x}(z)} P' \quad Q \xrightarrow{xz} Q' \quad z \notin fn(Q)}{P Q \xrightarrow{\tau} (\nu z)(P' Q')}$ | ClsR $\frac{P \xrightarrow{xz} P' \quad Q \xrightarrow{\bar{x}(z)} Q' \quad z \notin fn(P)}{P Q \xrightarrow{\tau} (\nu z)(P' Q')}$ |
| Tau $\frac{}{\tau.P \xrightarrow{\tau} P}$ | Ide $\frac{A(\tilde{x}) \stackrel{def}{=} P \quad P\{\tilde{y}/\tilde{x}\} \xrightarrow{\alpha} P'}{A(\tilde{y}) \xrightarrow{\alpha} P'}$ |

Table 2.5: Late semantic without structural congruence and without explicit α conversion

structural congruences let processes interact even in case they are not adjacent in the syntax. There is a possible trade off between what to include in the structural congruence and what to include in the transition rules: for example in the case of the commutativity of the sum operator. It is worth noticing that in most process calculi every structurally congruent processes should never be distinguished and thus any semantic must assign them the same behaviour.

Definition 2.3.1. A *change of bound names* in a process P is the replacement of a subterm $x(z).Q$ of P by $x(w).Q\{w/z\}$ or the replacement of a subterm $(\nu z)Q$ of P by $(\nu w)Q\{w/z\}$ where in each case w does not occur in Q .

Definition 2.3.2. A *context* $C[\cdot]$ is a process with a placeholder. If $C[\cdot]$ is a context and we replace the placeholder with P , then we obtain $C[P]$. In doing so, we make no α conversions.

Definition 2.3.3. A *congruence* is a binary relation on processes such that:

- S is an equivalence relation
- S is preserved by substitution in contexts: for each pair of processes (P, Q) and for each context $C[\cdot]$

$$(P, Q) \in S \Rightarrow (C[P], C[Q]) \in S$$

Definition 2.3.4. Processes P and Q are α *convertible* or α *equivalent* if Q can be obtained from P by a finite number of changes of bound names. If P and Q are α equivalent then we write $P \equiv_{\alpha} Q$. Specifically the α equivalence is the smallest binary relation on processes that satisfies the laws in table 2.6. In a process P we can assume that all bound names are different.

It remains the problem of proving that α equivalence is well defined, i.e. if we change only some bound names in a process P then we get a process α equivalent to P .

According to [2] the following lemma holds:

| | | |
|---|---|---|
| $\text{ALPOUT} \frac{P \equiv_{\alpha} Q}{\bar{x}y.P \equiv_{\alpha} \bar{x}y.Q}$ | $\text{ALPTAU} \frac{P \equiv_{\alpha} Q}{\tau.P \equiv_{\alpha} \tau.Q}$ | $\text{ALPINP} \frac{P \equiv_{\alpha} Q}{x(y).P \equiv_{\alpha} x(y).Q}$ |
| $\text{ALPIDE} \frac{}{A(\tilde{x}) \equiv_{\alpha} A(\tilde{x})}$ | $\text{ALPZERO} \frac{}{0 \equiv_{\alpha} 0}$ | $\text{ALPRES} \frac{P \equiv_{\alpha} Q}{(\nu x)P \equiv_{\alpha} (\nu x)Q}$ |
| $\text{ALPPAR} \frac{P_1 \equiv_{\alpha} Q_1 \quad P_2 \equiv_{\alpha} Q_2}{P_1 P_2 \equiv_{\alpha} Q_1 Q_2}$ | $\text{ALPSUM} \frac{P_1 \equiv_{\alpha} Q_1 \quad P_2 \equiv_{\alpha} Q_2}{P_1 + P_2 \equiv_{\alpha} Q_1 + Q_2}$ | |
| $\text{ALPRES1} \frac{P \equiv_{\alpha} Q \quad x \neq y \quad y \notin n(Q) \quad x \in fn(Q)}{(\nu x)P \equiv_{\alpha} (\nu y)Q\{y/x\}}$ | | |
| $\text{ALPINP1} \frac{P \equiv_{\alpha} Q \quad x \neq y \quad y \notin n(Q) \quad x \in fn(Q)}{z(x).P \equiv_{\alpha} z(y).Q\{y/x\}}$ | | |
| $\text{ALPRES2} \frac{P \equiv_{\alpha} Q \quad x \neq y \quad x \notin n(P) \quad y \in fn(P)}{(\nu x)P\{x/y\} \equiv_{\alpha} (\nu y)Q}$ | | |
| $\text{ALPINP2} \frac{P \equiv_{\alpha} Q\{x/y\} \quad x \neq y \quad x \notin n(P) \quad y \in fn(P)}{z(x).P\{x/y\} \equiv_{\alpha} z(y).Q}$ | | |

Table 2.6: α equivalence laws

Lemma 2.3.1. Let P be a process and y, w, z names such that $w = z$ or $w \notin fn(P)$ then $P\{w/z\}\{y/w\} \equiv_{\alpha} P$.

Definition 2.3.5. *structural congruence* \equiv is the smallest relation on processes that satisfies the axioms in table 2.7

Proposition 2.3.2. \equiv as defined in table 2.7 is a congruence and an equivalence relation.

Proof. \equiv is a congruence thanks to rules *Cong1* and *Cong2*. Reflexivity holds for rule *Alp*. Symmetry holds because all the rules are symmetric or have a symmetric counterpart. Transitivity holds because of rule *Trans*. \square

We can make some clarification on the axioms of the structural congruence:

unfolding this just helps replace an identifier by its definition, with the appropriate parameter instantiation. The alternative is to use the rule *Cns* in table 2.4.

α *conversion* is the α conversion, i.e., the choice of bound names, it identifies agents like $x(y).\bar{z}y$ and $x(w).\bar{z}w$. In the semantic of π calculus we can use the structural congruence with the rule *Alp* or we can embed the α conversion in the SOS rules. In the early case, the rule for input and the rules *ResAlp1*, *OpnAlp*, *Cns* take care of α conversion, whether in the late case the rule for communication and the rules *ResAlp1*, *OpnAlp*, *Cns* are in charge for α conversion.

abelian monoidal properties of some operators We can deal with associativity and commutativity properties of sum and parallel composition by using SOS rules or by axiom of the structural congruence. For example the commutativity of the sum can be expressed by the following two rules:

$$\text{SumL} \frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'} \quad \text{SumR} \frac{Q \xrightarrow{\alpha} Q'}{P + Q \xrightarrow{\alpha} Q'}$$

| | |
|--|--|
| SumAsc1 $M_1 + (M_2 + M_3) \equiv (M_1 + M_2) + M_3$ | ParAsc1 $P_1 (P_2 P_3) \equiv (P_1 P_2) P_3$ |
| SumAsc2 $(M_1 + M_2) + M_3 \equiv M_1 + (M_2 + M_3)$ | ParAsc2 $(P_1 P_2) P_3 \equiv P_1 (P_2 P_3)$ |
| ParCom $P_1 P_2 \equiv P_2 P_1$ | ResCom $(\nu x)(\nu y)P \equiv (\nu y)(\nu x)P$ |
| SumCom $M_1 + M_2 \equiv M_2 + M_1$ | |
| ScpExtPar1 $\frac{z \notin fn(P_1)}{(\nu z)(P_1 P_2) \equiv P_1 (\nu z)P_2}$ | ScpExtPar2 $\frac{z \notin fn(P_1)}{P_1 (\nu z)P_2 \equiv (\nu z)(P_1 P_2)}$ |
| ScpExtSum1 $\frac{z \notin fn(P_1)}{(\nu z)(P_1 + P_2) \equiv P_1 + (\nu z)P_2}$ | ScpExtSum2 $\frac{z \notin fn(P_1)}{P_1 + (\nu z)P_2 \equiv (\nu z)(P_1 + P_2)}$ |
| Ide $\frac{A(\tilde{x}) \stackrel{def}{=} P}{A(\tilde{w}) \equiv P\{\tilde{w}/\tilde{x}\}}$ | Trans $\frac{P \equiv Q \quad Q \equiv R}{P \equiv R}$ |
| | Alp $\frac{P \equiv_\alpha Q}{P \equiv Q}$ |
| Cong1 $\frac{P \equiv Q}{C[P] \equiv C[Q]}$ | Cong2 $\frac{P_1 \equiv Q_1 \quad P_2 \equiv Q_2 \quad C[_, _] \in \{ _ + _, _ _ \}}{C[P_1, P_2] \equiv C[Q_1, Q_2]}$ |

Table 2.7: Structural congruence rules

or by the following rule and axiom:

$$\mathbf{Sum} \frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'} \quad \mathbf{SumCom} \quad P + Q \equiv Q + P$$

and the rule *Cong*

scope extension We can use the scope extension laws in table 2.7 or the rules *Opn* and *Cls* in table 2.4 to deal with the scope extension.

Lemma 2.3.3.

$$a \in fn(Q) \Rightarrow fn(Q\{b/a\}) = (fn(Q) - \{a\}) \cup \{b\}$$

Lemma 2.3.4. $P \equiv_\alpha Q \Rightarrow fn(P) = fn(Q)$

Proof. The proof goes by induction on rules

AlpZero the lemma holds because P and Q are the same process.

AlpTau :

$$\begin{array}{ll} P \equiv_\alpha Q & \text{rule premise} \\ \Rightarrow fn(P) = fn(Q) & \text{inductive hypothesis} \\ \Rightarrow fn(\tau.P) = fn(\tau.Q) & \text{definition of } fn \end{array}$$

AlpOut :

$$\begin{array}{ll} P \equiv_\alpha Q & \text{rule premise} \\ \Rightarrow fn(P) = fn(Q) & \text{inductive hypothesis} \\ \Rightarrow fn(P) \cup \{x, y\} = fn(Q) \cup \{x, y\} & \text{definition of } fn \\ \Rightarrow fn(\bar{x}y.P) = fn(\bar{x}y.Q) & \end{array}$$

AlpRes1 :

$$\begin{array}{ll}
P \equiv_{\alpha} Q & \text{rule premise} \\
\Rightarrow fn(P) = fn(Q) & \text{inductive hypothesis} \\
\Rightarrow fn(P) - \{y\} = fn(Q) - \{y\} & \\
\Rightarrow fn(P) - \{y\} = fn((\nu y)Q) & \text{definition of } fn \\
\Rightarrow ((fn(P) - \{y\}) \cup \{x\}) - \{x\} = fn((\nu y)Q) & \\
\Rightarrow fn(P\{x/y\}) - \{x\} = fn((\nu y)Q) & \\
\Rightarrow fn((\nu x)(P\{x/y\})) = fn((\nu y)Q) &
\end{array}$$

AlpRes2 : similar.

AlpInp1 :

$$\begin{aligned}
fn(a(x).(P\{x/y\})) &= (fn(P\{x/y\}) - \{x\}) \cup \{a\} = (((fn(P) - \{y\}) \cup \{x\}) - \{x\}) \cup \{a\} = \\
&= (fn(P) - \{y\}) \cup \{a\} = (fn(Q) - \{y\}) \cup \{a\} = fn(a(x).Q)
\end{aligned}$$

AlpInp2 : similar.

AlpSum :

$$\begin{array}{ll}
P_1 \equiv_{\alpha} Q_1 \text{ and } P_2 \equiv_{\alpha} Q_2 & \text{rule premises} \\
\Rightarrow fn(P_1) = fn(Q_1) \text{ and } fn(P_2) = fn(Q_2) & \text{inductive hypothesis} \\
\Rightarrow fn(P_1) \cup fn(P_2) = fn(Q_1) \cap fn(Q_2) & \text{definition of } fn \\
\Rightarrow fn(P_1 + P_2) = fn(Q_1 + Q_2) &
\end{array}$$

AlpPar :

$$\begin{array}{ll}
P_1 \equiv_{\alpha} Q_1 \text{ and } P_2 \equiv_{\alpha} Q_2 & \text{rule premises} \\
\Rightarrow fn(P_1) = fn(Q_1) \text{ and } fn(P_2) = fn(Q_2) & \text{inductive hypothesis} \\
\Rightarrow fn(P_1) \cup fn(P_2) = fn(Q_1) \cap fn(Q_2) & \text{definition of } fn \\
\Rightarrow fn(P_1|P_2) = fn(Q_1|Q_2) &
\end{array}$$

AlpRes :

$$\begin{array}{ll}
P \equiv_{\alpha} Q & \text{rule premise} \\
\Rightarrow fn(P) = fn(Q) & \text{inductive hypothesis} \\
\Rightarrow fn(P) - \{x\} = fn(Q) - \{x\} & \text{definition of } fn \\
\Rightarrow fn((\nu x)P) = fn((\nu x)Q) &
\end{array}$$

AlpInp :

$$\begin{array}{ll}
P \equiv_{\alpha} Q\{x/y\} & \text{rule premise} \\
\Rightarrow fn(P) = fn(Q) & \text{inductive hypothesis} \\
\Rightarrow (fn(P) - \{y\}) \cup \{x\} = (fn(Q) - \{y\}) \cup \{x\} & \text{definition of } fn \\
\Rightarrow fn(x(y).P) = fn(x(y).Q) &
\end{array}$$

AlpIde the lemma holds because P and Q are the same process.

□

Lemma 2.3.5. $P \equiv_{\alpha} P\{a/b\}\{b/a\}$

Lemma 2.3.6. α equivalence is invariant with respect to substitution. In other words

$$\begin{array}{l} P \equiv_{\alpha} Q \\ b \notin fn(P) \quad \Rightarrow \quad P\{b/a\} \equiv_{\alpha} Q\{b/a\} \\ b \notin fn(Q) \end{array}$$

Lemma 2.3.7.

$$P \equiv_{\alpha} P\{x/y\}\{y/x\}$$

In the proof of equivalence of the semantics in the next section we need the following lemmas

Lemma 2.3.8. The α equivalence is reflexive.

Proof. : We prove $P \equiv_{\alpha} P$ by structural induction on P :

0 :

$$\text{ALPZERO} \frac{}{0 \equiv_{\alpha} 0}$$

$\tau.P_1$: for induction $P_1 \equiv_{\alpha} P_1$ so

$$\text{ALPTAU} \frac{P_1 \equiv_{\alpha} P_1}{\tau.P_1 \equiv_{\alpha} \tau.P_1}$$

$x(y).P_1$: for induction $P_1 \equiv_{\alpha} P_1$ so

$$\text{ALPIMP} \frac{P_1 \equiv_{\alpha} P_1}{x(y).P_1 \equiv_{\alpha} x(y).P_1}$$

$\bar{x}y.P_1$: for induction $P_1 \equiv_{\alpha} P_1$ so

$$\text{ALPOUT} \frac{P_1 \equiv_{\alpha} P_1}{\bar{x}y.P_1 \equiv_{\alpha} \bar{x}y.P_1}$$

$P_1 + P_2$: for induction $P_1 \equiv_{\alpha} P_1$ and $P_2 \equiv_{\alpha} P_2$ so

$$\text{ALPSUM} \frac{P_1 \equiv_{\alpha} P_1 \quad P_2 \equiv_{\alpha} P_2}{P_1 + P_2 \equiv_{\alpha} P_1 + P_2}$$

$P_1|P_2$: for induction $P_1 \equiv_{\alpha} P_1$ and $P_2 \equiv_{\alpha} P_2$ so

$$\text{ALPPAR} \frac{P_1 \equiv_{\alpha} P_1 \quad P_2 \equiv_{\alpha} P_2}{P_1|P_2 \equiv_{\alpha} P_1|P_2}$$

$(\nu x)P_1$: for induction $P_1 \equiv_{\alpha} P_1$ so

$$\text{ALPRES} \frac{P_1 \equiv_{\alpha} P_1}{(\nu x)P_1 \equiv_{\alpha} (\nu x)P_1}$$

$A(\tilde{x})$:

$$\text{ALPIDE} \frac{}{A(\tilde{x}) \equiv_{\alpha} A(\tilde{x})}$$

□

Lemma 2.3.9. α equivalence is symmetric.

Proof. Every rule is symmetric or it has a symmetric counterpart.

□

Lemma 2.3.10. α equivalence is transitive.

Theorem 2.3.11. α equivalence is an equivalence relation.

Proof. Follows from lemmas 2.3.8, 2.3.9 and 2.3.10.

□

| | | |
|---|--|---|
| Out $\frac{}{\bar{x}y.P \xrightarrow{\bar{x}y} P}$ | EInp $\frac{}{x(y).P \xrightarrow{xz} P\{z/y\}}$ | Tau $\frac{}{\tau.P \xrightarrow{\tau} P}$ |
| ParL $\frac{P \xrightarrow{\alpha} P' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P Q \xrightarrow{\alpha} P' Q}$ | ParR $\frac{Q \xrightarrow{\alpha} Q' \quad bn(\alpha) \cap fn(P) = \emptyset}{P Q \xrightarrow{\alpha} P Q'}$ | |
| SumL $\frac{P \xrightarrow{\alpha} P' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P + Q \xrightarrow{\alpha} P'}$ | SumR $\frac{Q \xrightarrow{\alpha} Q' \quad bn(\alpha) \cap fn(P) = \emptyset}{P + Q \xrightarrow{\alpha} Q'}$ | |
| Res $\frac{P \xrightarrow{\alpha} P' \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha} (\nu z)P'}$ | Alp $\frac{P \equiv_{\alpha} Q \quad P \xrightarrow{\alpha} P'}{Q \xrightarrow{\alpha} P'}$ | |
| EComL $\frac{P \xrightarrow{xy} P' \quad Q \xrightarrow{\bar{x}y} Q'}{P Q \xrightarrow{\tau} P' Q'}$ | EComR $\frac{P \xrightarrow{\bar{x}y} P' \quad Q \xrightarrow{xy} Q'}{P Q \xrightarrow{\tau} P' Q'}$ | |
| ClsL $\frac{P \xrightarrow{\bar{x}(z)} P' \quad Q \xrightarrow{xz} Q' \quad z \notin fn(Q)}{P Q \xrightarrow{\tau} (\nu z)(P' Q')}$ | ClsR $\frac{P \xrightarrow{xz} P' \quad Q \xrightarrow{\bar{x}(z)} Q' \quad z \notin fn(P)}{P Q \xrightarrow{\tau} (\nu z)(P' Q')}$ | |
| Ide $\frac{A(\tilde{x}) \stackrel{def}{=} P \quad P\{\tilde{w}/\tilde{x}\} \xrightarrow{\alpha} P'}{A(\tilde{x}) \xrightarrow{\alpha} P'}$ | Opn $\frac{P \xrightarrow{\bar{x}z} P' \quad z \neq x}{(\nu z)P \xrightarrow{\bar{x}(z)} P'}$ | |

Table 2.8: Early transition relation with α conversion but without structural congruence

2.4 Early semantic with α conversion only

In this subsection we introduce the early operational semantic for π calculus with the use of a minimal structural congruence, specifically we exploit only the easy of α conversion.

Definition 2.4.1. The *early transition relation with α conversion* $\rightarrow_{\subseteq} \mathbb{P} \times \mathbb{A} \times \mathbb{P}$ is the smallest relation induced by the rules in table 2.8.

The following example shows why the condition $bn(\alpha) \cap fn(Q) = \emptyset$ in the rule *Sum* is desirable:

Example without the side condition we are able to prove:

$$\begin{array}{c}
 \text{Opn} \frac{(\nu y)\bar{x}y.0 \xrightarrow{\bar{x}y} (\nu y)0}{(\nu y)\bar{x}y.0 \xrightarrow{\bar{x}(y)} (\nu y)0} \\
 \text{Sum} \frac{(\nu y)\bar{x}y.0 \xrightarrow{\bar{x}(y)} (\nu y)0}{((\nu y)\bar{x}y.0) + \bar{y}x.0 \xrightarrow{\bar{x}(y)} (\nu y)0} \quad \text{EInp} \frac{}{x(z).0 \xrightarrow{xy} 0} \\
 \text{ClsL} \frac{}{(((\nu y)\bar{x}y.0) + \bar{y}x.0)|x(z).0 \xrightarrow{\tau} (\nu y)0}
 \end{array}$$

but $(((\nu y)\bar{x}y.0) + \bar{y}x.0)|x(z).0 \not\equiv (\nu y)(\bar{x}y.0 + \bar{y}x.0)|x(z).0$

2.5 Operational semantic with structural congruence

2.5.1 Early semantic with structural congruence

Definition 2.5.1. The *early transition relation with structural congruence* $\rightarrow_{\subseteq} \mathbb{P} \times \mathbb{A} \times \mathbb{P}$ is the smallest relation induced by the rules in table 2.9.

Example We prove now that

| | | |
|--|--|---|
| Out $\frac{}{\overline{xy}.P \xrightarrow{\overline{xy}} P}$ | EInp $\frac{}{x(y).P \xrightarrow{xz} P\{z/y\}}$ | Tau $\frac{}{\tau.P \xrightarrow{\tau} P}$ |
| Par $\frac{P \xrightarrow{\alpha} P' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P Q \xrightarrow{\alpha} P' Q}$ | Sum $\frac{P \xrightarrow{\alpha} P' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P + Q \xrightarrow{\alpha} P'}$ | |
| ECom $\frac{P \xrightarrow{xy} P' \quad Q \xrightarrow{\overline{xy}} Q'}{P Q \xrightarrow{\tau} P' Q'}$ | Cong $\frac{P \equiv P' \quad P' \xrightarrow{\alpha} Q}{P \xrightarrow{\alpha} Q}$ | |
| Opn $\frac{P \xrightarrow{\overline{xz}} P' \quad z \neq x}{(\nu z)P \xrightarrow{\overline{x(z)}} P'}$ | Res $\frac{P \xrightarrow{\alpha} P' \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha} (\nu z)P'}$ | |

Table 2.9: Early semantic with structural congruence

$$a(x).P \mid (\nu b)\overline{ab}.Q \xrightarrow{\tau} (\nu b)(P\{b/x\} \mid Q)$$

where $b \notin fn(P)$. This follows from

$$a(x).P \mid (\nu b)\overline{ab}.Q \equiv (\nu b)(a(x).P \mid \overline{ab}.Q)$$

and

$$(\nu b)(a(x).P \mid \overline{ab}.Q) \xrightarrow{\tau} (\nu b)(P\{b/x\} \mid Q)$$

with the rule *Cong*. We can prove the last transition in the following way:

$$\text{RES} \frac{\text{COM} \frac{\text{EINP} \frac{}{a(x).P \xrightarrow{ab} P\{b/x\}} \quad \text{OUT} \frac{}{\overline{ab}.Q \xrightarrow{\overline{ab}} Q}}{a(x).P \mid \overline{ab}.Q \xrightarrow{\tau} P\{b/x\} \mid Q}}{(\nu b)(a(x).P \mid \overline{ab}.Q) \xrightarrow{\tau} (\nu b)(P\{b/x\} \mid Q)}$$

Example We want to prove now that:

$$((\nu b)a(x).P) \mid \overline{ab}.Q \xrightarrow{\tau} (\nu c)(P\{c/b\}\{b/x\} \mid Q)$$

where the name c is not in the free names of Q . We can exploit the structural congruence and get that

$$((\nu b)a(x).P) \mid \overline{ab}.Q \equiv (\nu c)(a(x).(P\{c/b\}) \mid \overline{ab}.Q)$$

then we have

$$\text{RES} \frac{\text{COM} \frac{\text{EINP} \frac{}{a(x).P\{c/b\} \xrightarrow{ab} P\{c/b\}\{b/x\}} \quad \text{OUT} \frac{}{\overline{ab}.Q \xrightarrow{\overline{ab}} Q}}{a(x).(P\{c/b\}) \mid \overline{ab}.Q \xrightarrow{\tau} (P\{c/b\}\{b/x\} \mid Q)}}{(\nu c)(a(x).(P\{c/b\}) \mid \overline{ab}.Q) \xrightarrow{\tau} (\nu c)(P\{c/b\}\{b/x\} \mid Q)}$$

Now we just apply the rule *Cong* to prove the thesis.

2.5.2 Late semantic with structural congruence

Definition 2.5.2. The late transition relation with structural congruence $\rightarrow_{\subseteq} \mathbb{P} \times \mathbb{A} \times \mathbb{P}$ is the smallest relation induced by the rules in table 2.10.

Example We prove now that

$$a(x).P \mid (\nu b)\overline{ab}.Q \xrightarrow{\tau} P\{b/x\} \mid Q$$

| | | |
|---|--|---|
| Tau $\frac{}{\tau.P \xrightarrow{\tau} P}$ | LInp $\frac{}{x(y).P \xrightarrow{x(y)} P}$ | Out $\frac{}{\bar{x}y.P \xrightarrow{\bar{x}y} P}$ |
| Opn $\frac{P \xrightarrow{\bar{x}z} P' \quad z \neq x}{(\nu z)P \xrightarrow{\bar{x}(z)} P'}$ | Res $\frac{P \xrightarrow{\alpha} P' \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha} (\nu z)P'}$ | |
| LCom $\frac{P \xrightarrow{x(y)} P' \quad Q \xrightarrow{\bar{x}z} Q'}{P Q \xrightarrow{\tau} P'\{z/y\} Q'}$ | Par $\frac{P \xrightarrow{\alpha} P' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P Q \xrightarrow{\alpha} P' Q}$ | |
| Cong $\frac{P \equiv P' \quad P \xrightarrow{\alpha} Q}{P' \xrightarrow{\alpha} Q'}$ | Sum $\frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'}$ | |

Table 2.10: Late semantic with structural congruence

where $b \notin fn(P)$. This follows from

$$a(x).P \mid (\nu b)\bar{a}b.Q \equiv (\nu b)(a(x).P \mid \bar{a}b.Q)$$

and

$$(\nu b)(a(x).P \mid \bar{a}b.Q) \xrightarrow{\tau} (\nu b)(P\{b/x\} \mid Q)$$

with the rule *Cong*. We can prove the last transition in the following way:

$$\text{RES} \frac{\text{LInp} \frac{b \notin fn(P)}{a(x).P \xrightarrow{ab} P\{b/x\}} \quad \text{OUT} \frac{}{\bar{a}b.Q \xrightarrow{\bar{a}b} Q} \quad \text{LCom} \frac{}{a(x).P \mid \bar{a}b.Q \xrightarrow{\tau} P\{b/x\} \mid Q} \quad b \notin n(\tau)}{(\nu b)(a(x).P \mid \bar{a}b.Q) \xrightarrow{\tau} (\nu b)(P\{b/x\} \mid Q)}$$

Example We want to prove now that:

$$((\nu b)a(x).P) \mid \bar{a}b.Q \xrightarrow{\tau} (\nu c)(P\{c/b\}\{b/x\} \mid Q)$$

where the name c is not in the free names of Q and is not in the names of P . We can exploit the structural congruence and get that

$$((\nu b)a(x).P) \mid \bar{a}b.Q \equiv (\nu c)(a(x).(P\{c/b\}) \mid \bar{a}b.Q)$$

then we have

$$\text{RES} \frac{\text{LInp} \frac{b \notin fn(P\{c/b\})}{a(x).P\{c/b\} \xrightarrow{ab} P\{c/b\}\{b/x\}} \quad \text{OUT} \frac{}{\bar{a}b.Q \xrightarrow{\bar{a}b} Q} \quad \text{LCom} \frac{}{(a(x).(P\{c/b\}) \mid \bar{a}b.Q) \xrightarrow{\tau} (P\{c/b\}\{b/x\} \mid Q)} \quad c \notin n(\tau)}{(\nu c)(a(x).(P\{c/b\}) \mid \bar{a}b.Q) \xrightarrow{\tau} (\nu c)(P\{c/b\}\{b/x\} \mid Q)}$$

Now we just apply the rule *Cong* to prove the thesis.

2.5.3 Permutation semantic

We can define an early permutation semantic just as the semantic in table 2.9 but the rule *EInp* is replaced by the following rule

$$\text{EInpPerm} \frac{z \notin fn(P)}{x(y).P \xrightarrow{xz} P(zy)}$$

where (zy) is a permutation of names that swaps z with y but does not alter other names. If we modify the rule *Elnp* in table 2.9 by adding the premise $z \notin fn(P)$ then: $P \xrightarrow{\gamma} P'$ if and only if $P \xrightarrow{\gamma} P''$ in the permutation semantic and $P' \equiv_{\alpha} P''$.

2.6 Equivalence of the semantics

2.6.1 Equivalence of the early semantics

In this subsection we write \rightarrow_1 for the early semantic without structural congruence, \rightarrow_2 for the early semantic with just α conversion and \rightarrow_3 for the early semantic with the full structural congruence. We call R_1 the set of rules for \rightarrow_1 , R_2 the set of rules for \rightarrow_2 and R_3 the set of rules for \rightarrow_3 .

Lemma 2.6.1. Structurally equivalent process have the same free names:

$$P \equiv Q \Rightarrow fn(Q) = fn(P)$$

Proof. The proof is easy and is an induction on the rules of structural congruence. \square

We would like to prove that $P \xrightarrow{\alpha}_2 P' \Rightarrow P \xrightarrow{\alpha}_1 P'$ but this is false because

$$\text{ALP} \frac{\overline{xy}.x(y).0 \equiv_{\alpha} \overline{xy}.x(w).0 \quad \text{OUT} \frac{}{\overline{xy}.x(w).0 \xrightarrow{\overline{xy}}_2 x(w).0}}{\overline{xy}.x(y).0 \xrightarrow{\overline{xy}}_2 x(w).0}$$

so we want to prove

$$\overline{xy}.x(y).0 \xrightarrow{\overline{xy}}_1 x(w).0$$

The head of the transition has an output prefixing at the top level so the only rule we could use is *Out*, but the application of *Out* yields

$$\overline{xy}.x(y).0 \xrightarrow{\overline{xy}}_1 x(y).0$$

which is not what we want. But we prove some weaker results.

Lemma 2.6.2. Let \rightarrow_2 be the semantic of table 2.8 but without rule *Alp*. If $P \xrightarrow{\alpha}_2 P'$ then there exist a process Q such that $P \equiv_{\alpha} Q \xrightarrow{\alpha}_2 P'$.

Proof. We prove by cases that in a derivation of $P \xrightarrow{\alpha}_2 P'$ we can move downward to the end of the derivation any occurrence of the rule *Alp*:

ParL :

$$\text{ParL} \frac{\text{Alp} \frac{P \equiv_{\alpha} R \quad R \xrightarrow{\alpha} P'}{P \xrightarrow{\alpha} P'} \quad bn(\alpha) \cap fn(Q) = \emptyset}{P|Q \xrightarrow{\alpha} P'}$$

became

$$\text{Alp} \frac{\text{AlpPar} \frac{P \equiv_{\alpha} R}{P|Q \equiv_{\alpha} R|Q} \quad \text{ParL} \frac{R \xrightarrow{\alpha} P' \quad bn(\alpha) \cap fn(Q) = \emptyset}{R|Q \xrightarrow{\alpha} P'}}{P|Q \xrightarrow{\alpha} P'}$$

ParR, SumL, SumR similar.

Alp since α equivalence is transitive, we can merge any pair of consecutive instance of the rule *Alp*

Res :

$$\mathbf{Res} \frac{\mathbf{Alp} \frac{P \equiv_{\alpha} R \quad R \xrightarrow{\alpha} P'}{P \xrightarrow{\alpha} P'} \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha} (\nu z)P'}$$

became

$$\mathbf{Alp} \frac{\mathbf{AlpRes} \frac{P \equiv_{\alpha} R}{(\nu z)P \equiv_{\alpha} (\nu z)R} \quad \mathbf{Res} \frac{R \xrightarrow{\alpha} P' \quad z \notin n(\alpha)}{(\nu z)R \xrightarrow{\alpha} (\nu z)P'}}{(\nu z)P \xrightarrow{\alpha} (\nu z)P'}$$

Opn similar.

EComL :

$$\mathbf{EComL} \frac{\mathbf{Alp} \frac{P \equiv_{\alpha} R \quad R \xrightarrow{xy} P'}{P \xrightarrow{xy} P'} \quad \mathbf{Alp} \frac{Q \equiv_{\alpha} S \quad S \xrightarrow{\bar{x}y} S'}{Q \xrightarrow{\bar{x}y} S'}}{P|Q \xrightarrow{\tau} P'|S'}$$

became

$$\mathbf{Alp} \frac{\mathbf{AlpPar} \frac{P \equiv_{\alpha} R \quad Q \equiv_{\alpha} S}{P|Q \equiv_{\alpha} R|S} \quad \mathbf{EComL} \frac{R \xrightarrow{xy} P' \quad S \xrightarrow{\bar{x}y} S'}{R|S \xrightarrow{\tau} S'}}{P|Q \xrightarrow{\tau} P'|S'}$$

EComR similar.

ClsR :

$$\mathbf{ClsR} \frac{\mathbf{Alp} \frac{P \equiv_{\alpha} R \quad R \xrightarrow{xy} P'}{P \xrightarrow{xy} P'} \quad \mathbf{Alp} \frac{Q \equiv_{\alpha} S \quad S \xrightarrow{\bar{x}(y)} S'}{Q \xrightarrow{\bar{x}(y)} S'} \quad z \notin fn(P)}{P|Q \xrightarrow{\tau} (\nu z)(P'|S')}$$

became

$$\mathbf{Alp} \frac{\mathbf{AlpPar} \frac{P \equiv_{\alpha} R \quad Q \equiv_{\alpha} S}{P|Q \equiv_{\alpha} R|S} \quad \mathbf{ClsR} \frac{R \xrightarrow{xy} P' \quad S \xrightarrow{\bar{x}y} S' \quad z \notin fn(R)}{R|S \xrightarrow{\tau} (\nu y)(P'|S')}}{P|Q \xrightarrow{\tau} (\nu y)(P'|S')}$$

ClsL similar.

Ide :

$$\mathbf{Ide} \frac{A(\tilde{x}) \stackrel{def}{=} P \quad \mathbf{Alp} \frac{P\{\tilde{w}/\tilde{x}\} \equiv_{\alpha} R \quad R \xrightarrow{\alpha} P'}{P\{\tilde{w}/\tilde{x}\} \xrightarrow{\alpha} P'}}{A(\tilde{w}) \xrightarrow{\alpha} P'}$$

we can add a new definition $A(\tilde{w}) \stackrel{def}{=} R$ and this derivation became:

$$\mathbf{Ide} \frac{A(\tilde{w}) \stackrel{def}{=} R \quad R \xrightarrow{\alpha} P'}{A(\tilde{w}) \xrightarrow{\alpha} P'}$$

□

Theorem 2.6.3. If $P \xrightarrow{\alpha}_2 P'$ then there exists a process Q such that $P \equiv_{\alpha} Q \xrightarrow{\alpha}_1 P'$.

Proof. This result follows from lemma 2.6.2 observing that $\xrightarrow{\alpha}_2 \subseteq \xrightarrow{\alpha}_1$. □

Theorem 2.6.4. If $P \xrightarrow{\alpha}_2 P'$ then there exists a process P'' such that $P \xrightarrow{\alpha}_1 P''$ and $P'' \equiv_{\alpha} P'$

Proof. For lemma 2.6.2 there exists a process Q such that $P \equiv_{\alpha} Q \xrightarrow{\alpha}_2 P'$. The proof proceed by cases on the last rule used in the derivation of $P \equiv_{\alpha} Q$:

AlpIde in this case there is an identifier such that $P = A(\bar{x}) = Q$ so the conclusion holds.

AlpInp in this case $P = x(y).P_1$, $Q = x(y).Q_1$, $P_1 \equiv_{\alpha} Q_1$ and $\alpha = xz$. For rule *EInp*: $x(y).Q_1 \xrightarrow{xz} Q_1$.

AlpInp1 :

$$\text{Alp} \frac{\text{AlpInp1} \frac{P_1 \equiv_{\alpha} Q_1}{a(x).P_1 \equiv_{\alpha} a(y).(Q_1\{y/x\})} \quad \text{EInp} \frac{a(y).Q_1\{y/x\} \xrightarrow{az}_2 Q_1\{y/x\}\{z/y\}}{a(y).Q_1\{y/x\} \xrightarrow{az}_2 Q_1\{y/x\}\{z/y\}}}{a(x).P_1 \xrightarrow{az}_2 Q_1\{y/x\}\{z/y\}}$$

For rule *EInp*: $a(x).P_1 \xrightarrow{az}_1 P_1$. $P_1 \equiv_{\alpha} Q_1$ imply $P_1 \equiv_{\alpha} Q_1\{y/x\}\{z/y\}$

AlpInp2 :

$$\text{Alp} \frac{\text{AlpInp2} \frac{P_1 \equiv_{\alpha} Q_1}{a(x).(P_1\{x/y\}) \equiv_{\alpha} a(y).Q_1} \quad \text{EInp} \frac{a(y).Q_1 \xrightarrow{az}_2 Q_1\{z/y\}}{a(y).Q_1 \xrightarrow{az}_2 Q_1\{z/y\}}}{a(x).(P_1\{x/y\}) \xrightarrow{az}_2 Q_1\{z/y\}}$$

For rule *EInp*: $a(x).(P_1\{x/y\}) \xrightarrow{az}_1 P_1\{x/y\}\{z/y\}$. $P_1 \equiv_{\alpha} Q_1$ imply $P_1\{x/y\}\{z/y\} \equiv_{\alpha} Q_1\{z/y\}$

AlpOut in this case $P = \bar{a}x.P_1$, $Q = \bar{a}x.Q_1$, $P_1 \equiv_{\alpha} Q_1$ and $\alpha = \bar{a}x$. For rule *Out*: $\bar{a}x.P_1 \xrightarrow{\bar{a}x} P_1$ and $\bar{a}x.Q_1 \xrightarrow{\bar{a}x} Q_1$.

AlpTau similar.

AlpPar in this case $P = P_1|P_2$, $Q = Q_1|Q_2$, $P_1 \equiv_{\alpha} Q_1$ and $P_2 \equiv_{\alpha} Q_2$. The last rule used in the derivation of $Q_1|Q_2 \xrightarrow{\alpha}_2 P'|Q_2$ can be:

ParL in this case $P_1 \equiv_{\alpha} Q_1 \xrightarrow{\alpha}_2 P'$ and for inductive hypothesis $P_1 \xrightarrow{\alpha}_1 P'$. For rule *ParL*: $P_1|P_2 \xrightarrow{\alpha}_1 P'|P_2 \equiv_{\alpha} P'|Q_2$

ParR similar.

EComL in this case $P_1 \equiv_{\alpha} Q_1 \xrightarrow{xy}_2 Q'_1$ and $P_2 \equiv_{\alpha} Q_2 \xrightarrow{\bar{x}y}_2 Q'_2$, for inductive hypothesis $P_1 \xrightarrow{xy}_1 Q'_1$ and $P_2 \xrightarrow{\bar{x}y}_1 Q'_2$. For rule *EComL*: $P_1|P_2 \xrightarrow{\tau}_1 Q'_1|Q'_2$

EComR similar.

AlpSum similar.

AlpRes in this case $P = (\nu x)P_1$, $Q = (\nu x)Q_1$ and $P_1 \equiv_{\alpha} Q_1$. The last rule used in the derivation of $Q \xrightarrow{\alpha}_2 P'$ can be:

Res in this case $P_1 \equiv_{\alpha} Q_1 \xrightarrow{\alpha}_2 Q'_1$ and $x \notin n(\alpha)$. For inductive hypothesis $P_1 \xrightarrow{\alpha}_1 Q'_1$. For rule *Res*: $(\nu x)P_1 \xrightarrow{\alpha}_1 (\nu x)Q'_1$

Opn similar.

AlpRes1 The last rule used in the derivation of $Q \xrightarrow{\alpha}_2 P'$ can be:

Res :

$$\mathbf{Alp} \frac{\mathbf{AlpRes1} \frac{P_1 \equiv_{\alpha} Q_1}{(\nu x)P_1 \equiv_{\alpha} (\nu y)(Q_1\{y/x\})} \quad \mathbf{Res} \frac{Q_1\{y/x\} \xrightarrow{\alpha}_2 Q' \quad y \notin n(\alpha)}{(\nu y)(Q_1\{y/x\}) \xrightarrow{\alpha}_2 (\nu y)Q'} }{(\nu x)P_1 \xrightarrow{\alpha}_2 (\nu y)Q'}$$

$P_1 \equiv_{\alpha} Q_1$ imply $P_1\{y/x\} \equiv_{\alpha} Q_1\{y/x\}$, for inductive hypothesis $P_1\{y/x\} \xrightarrow{\alpha}_1 Q'$, now we can derive

$$\mathbf{ResAlp1} \frac{\mathbf{Res} \frac{P_1\{y/x\} \xrightarrow{\alpha}_1 Q' \quad y \notin n(\alpha)}{(\nu y)(P_1\{y/x\}) \xrightarrow{\alpha}_1 (\nu y)Q'}}{(\nu x)P_1 \xrightarrow{\alpha}_1 (\nu y)Q'}$$

Opn similar.

AlpRes2 The last rule used in the derivation of $Q \xrightarrow{\alpha}_2 P'$ can be:

Res :

$$\mathbf{Alp} \frac{\mathbf{AlpRes2} \frac{P_1 \equiv_{\alpha} Q_1}{(\nu x)(P_1\{x/y\}) \equiv_{\alpha} (\nu y)Q_1} \quad \mathbf{Res} \frac{Q_1 \xrightarrow{\alpha}_2 Q' \quad y \notin n(\alpha)}{(\nu y)Q_1 \xrightarrow{\alpha}_2 (\nu y)Q'}}{(\nu x)(P_1\{x/y\}) \xrightarrow{\alpha}_2 (\nu y)Q'}$$

$P_1 \equiv_{\alpha} Q_1 \xrightarrow{\alpha}_2 Q'$ imply for inductive hypothesis that $P_1 \xrightarrow{\alpha}_1 Q'$, now we can derive

$$\mathbf{ResAlp2} \frac{P_1 \xrightarrow{\alpha}_1 Q' \quad y \notin n(\alpha)}{(\nu x)P_1\{x/y\} \xrightarrow{\alpha}_1 (\nu y)Q'}$$

Opn similar.

AlpZero this case does not exist.

□

Theorem 2.6.5. $P \xrightarrow{\alpha}_1 P' \Rightarrow P \xrightarrow{\alpha}_2 P'$

Proof. The proof can go by induction on the length of the derivation of a transaction, and then both the base case and the inductive case proceed by cases on the last rule used in the derivation. However it is not necessary to show all the details of the proof because the rules in R_2 are almost the same as the rules in R_1 , the only difference is that in R_2 we have the rule *Alp* instead of *ResAlp1*, *ResAlp2* and *OpnAlp*. The rule *Alp* can mimic the rule *ResAlp1* in the following way:

$$\mathbf{Alp} \frac{\mathbf{AlpRes1} \frac{w \notin n(P)}{(\nu z)P \equiv_{\alpha} (\nu w)P\{w/z\}} \quad (\nu w)P\{w/z\} \xrightarrow{xz} P'}{(\nu z)P \xrightarrow{xz} P'}$$

The rule *Alp* can mimic the rule *ResAlp2* in the following way:

$$\mathbf{Alp} \frac{\mathbf{AlpRes2} \frac{w \notin n(P)}{(\nu w)P\{w/z\} \equiv_{\alpha} (\nu z)P} \quad \mathbf{Alp} \frac{P \xrightarrow{\alpha} P' \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha} (\nu z)P'}}{(\nu w)P\{w/z\} \xrightarrow{\alpha} (\nu z)P'}$$

The rule *Alp* can mimic the rule *OpnAlp* in the following way:

$$\mathbf{Alp} \frac{\mathbf{AlpRes1} \frac{w \notin n(P)}{(\nu z)P \equiv_{\alpha} (\nu w)P\{w/z\}} \quad (\nu w)P\{w/z\} \xrightarrow{\bar{x}(w)} P'}{(\nu z)P \xrightarrow{\bar{x}(w)} P'}$$

□

Lemma 2.6.6. If $P \xrightarrow{\bar{x}(y)}_2 P'$ then there is a process R such that $P \equiv R \xrightarrow{\bar{x}(y)}_2 P'$ and the last rule in this derivation is the instance of rule *Opn* used to open the scope of y .

Proof. The derivation of $P \xrightarrow{\bar{x}(y)}_2 P'$ must contain an instance of *Opn*. The proof consists in showing that we can move this instance of *Opn* downward in the inference tree of $P \xrightarrow{\bar{x}(y)}_2 P'$. The proof goes by induction on the depth of the derivation of $P \xrightarrow{\bar{x}(y)}_2 P'$ and then by cases on the last rule applied:

Opn if the derivation ends with *Opn* then the conclusion holds.

SumL :

$$\text{Opn} \frac{P_1 \xrightarrow{\bar{x}y}_2 P' \quad x \neq y}{(\nu y)P_1 \xrightarrow{\bar{x}(y)}_2 P'} \quad \text{SumL} \frac{\quad bn(\bar{x}(y)) \cap fn(R) = \emptyset}{P = (\nu y)P_1 + R \xrightarrow{\bar{x}(y)}_2 P'}$$

became:

$$\text{SumL} \frac{P_1 \xrightarrow{\bar{x}y}_2 P'}{P_1 + R \xrightarrow{\bar{x}y}_2 P'} \quad \text{Opn} \frac{\quad x \neq y}{(\nu y)(P_1 + R) \xrightarrow{\bar{x}(y)}_2 P'}$$

$bn(\bar{x}(y)) \cap fn(R) = \emptyset$ imply $y \notin fn(R)$ and so $(\nu y)(P_1 + R) \equiv (\nu y)P_1 + R$.

SumR symmetric to the previous case.

ParL :

$$\text{Opn} \frac{P_1 \xrightarrow{\bar{x}y}_2 P' \quad x \neq y}{(\nu y)P_1 \xrightarrow{\bar{x}(y)}_2 P'} \quad \text{ParL} \frac{\quad bn(\bar{x}(y)) \cap fn(R) = \emptyset}{P = (\nu y)P_1 | R \xrightarrow{\bar{x}(y)}_2 P' | R}$$

became:

$$\text{ParL} \frac{P_1 \xrightarrow{\bar{x}y}_2 P'}{P_1 | R \xrightarrow{\bar{x}y}_2 P' | R} \quad \text{Opn} \frac{\quad x \neq y}{(\nu y)(P_1 | R) \xrightarrow{\bar{x}(y)}_2 P' | R}$$

$bn(\bar{x}(y)) \cap fn(R) = \emptyset$ imply $y \notin fn(R)$ and so $(\nu y)(P_1 | R) \equiv (\nu y)P_1 | R$.

ParR symmetric to the previous case.

Res :

$$\text{Opn} \frac{P_1 \xrightarrow{\bar{x}y}_2 P' \quad x \neq y}{(\nu y)P_1 \xrightarrow{\bar{x}(y)}_2 P'} \quad \text{Res} \frac{\quad w \notin n(\bar{x}(y))}{P = (\nu w)(\nu y)P_1 \xrightarrow{\bar{x}(y)}_2 (\nu w)P'}$$

became:

$$\text{Opn} \frac{\text{Res} \frac{P_1 \xrightarrow{\bar{x}y}_2 P' \quad w \notin n(\bar{x}y)}{(\nu w)P_1 \xrightarrow{\bar{x}y}_2 (\nu w)P'} \quad x \neq y}{(\nu y)(\nu w)P_1 \xrightarrow{\bar{x}(y)}_2 (\nu w)P'}$$

$$(\nu y)(\nu w)P_1 \equiv (\nu w)(\nu y)P_1.$$

Alp(1) :

$$\text{Alp} \frac{\frac{P_1 \equiv_\alpha R_1}{(\nu y)P_1 \equiv_\alpha (\nu y)R_1} \quad \text{Opn} \frac{R_1 \xrightarrow{\bar{x}y}_2 R'_1 \quad x \neq y}{(\nu y)R_1 \xrightarrow{\bar{x}(y)}_2 R'_1}}{P = (\nu y)P_1 \xrightarrow{\bar{x}(y)}_2 R'_1 = P'}$$

became:

$$\text{Opn} \frac{\text{Alp} \frac{P_1 \equiv_\alpha R_1 \quad R_1 \xrightarrow{\bar{x}y}_2 R'_1}{P_1 \xrightarrow{\bar{x}y}_2 R'_1} \quad x \neq y}{(\nu y)P_1 \xrightarrow{\bar{x}(y)}_2 R'_1}$$

Alp(2) :

$$\text{Alp} \frac{\frac{P_1\{w/y\} \equiv_\alpha R_1 \quad w \notin n(P_1)}{(\nu w)P_1 \equiv_\alpha (\nu y)R_1} \quad \text{Opn} \frac{R_1 \xrightarrow{\bar{x}y}_2 R'_1 \quad x \neq y}{(\nu y)R_1 \xrightarrow{\bar{x}(y)}_2 R'_1}}{P = (\nu w)P_1 \xrightarrow{\bar{x}(y)}_2 R'_1 = P'}$$

became:

$$\text{Opn} \frac{\text{Alp} \frac{P_1\{y/w\} \equiv_\alpha R_1 \quad R_1 \xrightarrow{\bar{x}y}_2 R'_1}{P_1\{y/w\} \xrightarrow{\bar{x}y}_2 R'_1} \quad x \neq y}{(\nu y)P_1\{y/w\} \xrightarrow{\bar{x}(y)}_2 R'_1}$$

$$\text{and } (\nu y)P_1\{y/w\} \equiv (\nu w)P_1$$

□

Lemma 2.6.7. $P \xrightarrow{\alpha}_2 P'$ imply that there exist processes Q, Q' such that $P \equiv Q \xrightarrow{\alpha}_3 Q' \equiv P'$

Proof. The proof is by induction on the length of the derivation of $P \xrightarrow{\alpha}_2 P'$ and then both the base case and the inductive case proceed by cases on the last rule used:

Out, EInp, Tau in this case the rule used can be one of the following *Out, EInp, Tau* which are also in R_3 so a derivation of $P \xrightarrow{\alpha}_2 P'$ is also a derivation of $P \xrightarrow{\alpha}_3 P'$

Res, Opn the last rule used can be one in $R_2 \cap R_3 = \{\text{Res}, \text{Opn}\}$ and so for example we have

$$\text{Res} \frac{P \xrightarrow{\alpha}_2 P' \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha}_2 (\nu z)P'}$$

we apply the inductive hypothesis on $P \xrightarrow{\alpha}_2 P'$ and get that there exists a process P'' such that $P' \equiv P''$ and $P \xrightarrow{\alpha}_3 P''$. The proof we want is:

$$\text{Res} \frac{P \xrightarrow{\alpha}_3 P'' \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha}_3 (\nu z)P''}$$

$$\text{and } (\nu z)P'' \equiv (\nu z)P'$$

ParL, ParR, SumL, SumR, EComL, EComR In this cases we can proceed as in the previous case and if necessary add an application of *Cong* thus exploiting the commutativity of sum or parallel composition. For example

$$\mathbf{ParR} \frac{Q \xrightarrow{\alpha_2} Q' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P|Q \xrightarrow{\alpha_2} P|Q'}$$

now we apply the inductive hypothesis to $Q \xrightarrow{\alpha_2} Q'$ and get $Q \xrightarrow{\alpha_3} Q''$ for a Q'' such that $Q' \equiv Q''$. The proof we want is

$$\mathbf{Cong} \frac{P|Q \equiv Q|P \quad \mathbf{Par} \frac{Q \xrightarrow{\alpha_3} Q'' \quad bn(\alpha) \cap fn(Q) = \emptyset}{Q|P \xrightarrow{\alpha_3} Q''|P}}{P|Q \xrightarrow{\alpha_3} Q''|P}$$

and $Q''|P \equiv P|Q'$

Ide if the last rule used is *Ide*:

$$\mathbf{Ide} \frac{A(\tilde{x}) \stackrel{def}{=} P \quad P\{\tilde{y}/\tilde{x}\} \xrightarrow{\alpha_2} P'}{A(\tilde{y}) \xrightarrow{\alpha_2} P'}$$

we apply the inductive hypothesis on the premise and get that there exists a process P'' such that $P\{\tilde{y}/\tilde{x}\} \xrightarrow{\alpha_3} P''$ and $P'' \equiv P'$. Now the proof we want is

$$\mathbf{Ide} \frac{A(\tilde{y}) \equiv P\{\tilde{y}/\tilde{x}\} \quad P\{\tilde{y}/\tilde{x}\} \xrightarrow{\alpha_3} P''}{A(\tilde{y}) \xrightarrow{\alpha_3} P''}$$

Alp the rule *Alp* is a particular case of the rule *Cong*

ClsL if the last rule is *ClsL* then we have

$$\mathbf{ClsL} \frac{P \xrightarrow{\bar{x}(z)} P' \quad Q \xrightarrow{xz} Q' \quad z \notin fn(Q)}{P|Q \xrightarrow{\tau} (\nu z)(P'|Q')}$$

for lemma 2.6.6: $P \xrightarrow{\bar{x}(z)} P'$ imply that there exist a process $(\nu z)R$ such that $P \equiv (\nu z)R$, $(\nu z)R \xrightarrow{\bar{x}(z)} P'$ and the derivation of $(\nu z)R \xrightarrow{\tau} R'$ ends with the instance of *Opn* that opens the scope of z . So

$$\mathbf{Res} \frac{\mathbf{EComL} \frac{R \xrightarrow{\bar{x}z} P' \quad Q \xrightarrow{xz} Q'}{R|Q \xrightarrow{\tau} P'|Q'}}{(\nu z)(R|Q) \xrightarrow{\tau} (\nu z)(P'|Q')}$$

$P \equiv (\nu z)R$ and $z \notin fn(Q)$ imply $(\nu z)(R|Q) \equiv P|Q$. The conclusion follows after applying the inductive hypothesis on $(\nu z)(R|Q) \xrightarrow{\tau} (\nu z)(P'|Q')$ and the transitivity of structural congruence.

ClsR symmetric. □

Theorem 2.6.8. $P \xrightarrow{\alpha_2} P'$ imply that there exist processes Q' such that $P \xrightarrow{\alpha_3} Q' \equiv P'$.

Proof. For lemma 2.6.7 there exist processes Q, Q' such that $P \equiv Q \xrightarrow{\alpha_3} Q' \equiv P'$. So for rule *Cong*: $P \xrightarrow{\alpha_3} Q' \equiv P'$. □

Lemma 2.6.9. Let \twoheadrightarrow_3 be the semantic in table 2.9 but without rule *Cong*. $P \xrightarrow{\alpha_3} P'$ imply that there exist a process Q such that $P \equiv Q \twoheadrightarrow_3 P'$.

Proof. The proof needs to show that in any proof tree we can move downward any instance of a rule *Cong* until the proof tree has only one instance of the rule *Cong* and this is at the end. There are some cases to consider:

Sum :

$$\text{Sum} \frac{\text{Cong} \frac{P \equiv R \quad R \xrightarrow{\alpha} P'}{P \xrightarrow{\alpha} P'} \quad bn(\alpha) \cap fn(Q) = \emptyset}{P + Q \xrightarrow{\alpha} P'}$$

became:

$$\text{Cong} \frac{\frac{P \equiv R}{P + Q \equiv R + Q} \quad \text{Sum} \frac{R \xrightarrow{\alpha} P' \quad bn(\alpha) \cap fn(Q) = \emptyset}{R + Q \xrightarrow{\alpha} P'}}{P + Q \xrightarrow{\alpha} P'}$$

Par :

$$\text{Par} \frac{\text{Cong} \frac{P \equiv R \quad R \xrightarrow{\alpha} P'}{P \xrightarrow{\alpha} P'} \quad bn(\alpha) \cap fn(Q) = \emptyset}{P|Q \xrightarrow{\alpha} P'|Q}$$

became:

$$\text{Cong} \frac{\frac{P \equiv R}{P|Q \equiv R|Q} \quad \text{Par} \frac{R \xrightarrow{\alpha} P' \quad bn(\alpha) \cap fn(Q) = \emptyset}{R|Q \xrightarrow{\alpha} P'|Q}}{P + Q \xrightarrow{\alpha} P'|Q}$$

ECom :

$$\text{ECom} \frac{\text{Cong} \frac{P \equiv R \quad R \xrightarrow{xy} P'}{P \xrightarrow{xy} P'} \quad Q \xrightarrow{\bar{xy}} Q'}{P|Q \xrightarrow{\tau} P'|Q'}$$

became:

$$\text{Cong} \frac{\frac{P \equiv R}{P|Q \equiv R|Q} \quad \text{ECom} \frac{R \xrightarrow{xy} P' \quad Q \xrightarrow{\bar{xy}} Q'}{R|Q \xrightarrow{\tau} P'|Q'}}{P|Q \xrightarrow{\alpha} P'|Q'}$$

Res :

$$\text{Res} \frac{\text{Cong} \frac{P \equiv R \quad R \xrightarrow{\alpha} P'}{P \xrightarrow{\alpha} P'} \quad x \notin n(\alpha)}{(\nu x)P \xrightarrow{\alpha} (\nu x)P'}$$

became:

$$\text{Cong} \frac{\frac{P \equiv R}{(\nu x)P \equiv (\nu x)R} \quad \text{Res} \frac{R \xrightarrow{\alpha} P' \quad x \notin n(\alpha)}{(\nu x)R \xrightarrow{\alpha} (\nu x)P'}}{(\nu x)P \xrightarrow{\alpha} (\nu x)P'}$$

Opn :

$$\text{Opn} \frac{\text{Cong} \frac{P \equiv R \quad R \xrightarrow{\bar{y}x} P'}{P \xrightarrow{\bar{y}x} P'}}{(\nu x)P \xrightarrow{\bar{y}(x)} P'}$$

became:

$$\text{Cong} \frac{\frac{P \equiv R}{(\nu x)P \equiv (\nu x)R} \quad \text{Opn} \frac{R \xrightarrow{\bar{y}x} P'}{(\nu x)R \xrightarrow{\bar{y}(x)} P'}}{(\nu x)P \xrightarrow{\bar{y}(x)} P'}$$

Cong :

$$\text{Cong} \frac{P \equiv R \quad \text{Cong} \frac{R \equiv S \quad S \xrightarrow{\alpha} P'}{R \xrightarrow{\alpha} P'}}{P \xrightarrow{\alpha} P'}$$

became:

$$\text{Cong} \frac{\frac{P \equiv R \quad R \equiv S}{P \equiv S} \quad S \xrightarrow{\alpha} P'}{P \xrightarrow{\alpha} P'}$$

□

Theorem 2.6.10. $P \xrightarrow{\alpha}_3 P'$ imply that there exist a proces Q such that $P \equiv Q \xrightarrow{\alpha}_2 P'$.

Proof. This is a direct consequence of lemma 2.6.9 observing that $\rightarrow_3 \subseteq \rightarrow_2$. □

We would like to prove that $P \xrightarrow{\alpha}_3 P'$ imply that $P \xrightarrow{\alpha}_2 Q'$ and $Q' \equiv P'$ but this does not hold because:

$$\text{Cong} \frac{\text{ScpExtSum1} \frac{x \notin fn(\tau.a(b).0)}{(\nu x)(\tau.a(b).0 + Q) \equiv \tau.a(b).0 + (\nu x)Q} \quad \text{Sum} \frac{\tau.a(b).0 \xrightarrow{\tau}_3 a(b).0}{\tau.a(b).0 + (\nu x)Q \xrightarrow{\tau}_3 a(b).0}}{(\nu x)(\tau.a(b).0 + Q) \xrightarrow{\tau}_3 a(b).0}$$

but there is no way to prove that $(\nu x)(\tau.0 + Q) \xrightarrow{\tau}_2 a(b).0$ because: the only rules whose head's conclusion has a restriction at the top level are *Res* and *Opn*. Any instance of *Res* cannot match the transition we want to prove, because the head has (νx) at the top level whether the body has not. Any instance of *Opn* cannot match either because the transition's label is not an output. We can prove $(\nu x)(\tau.a(b).0 + Q) \xrightarrow{\tau}_2 (\nu x)a(b).0$ but $(\nu x)a(b).0 \not\equiv a(b).0$.

Lemma 2.6.11. Let \equiv_2 be the smallest congruence defined by only the rules *ScpExtPar1* and *ScpExtSum1*. If $P \equiv Q$ and $Q \xrightarrow{\alpha}_2 Q'$ then $P \equiv_2 R \xrightarrow{\alpha}_2 R'$ and $R' \equiv Q'$.

Proof. The proof is an induction on the sum of the size of the derivations of $P \equiv Q$ and of $Q \xrightarrow{\alpha}_2 Q'$. First we notice that we can get rid of the rules *ScpExtPar2* and *ScpExtSum2* because they can be mimicked by *ScpExtPar1*, *ScpExtSum1* and the commutativity. In a similar way we can ignore the rules *SumAsc2* and *ParAsc2*. The last pair of rules used can be:

$(\text{SumAsc1}, \text{SumL})(1)$:

$$\text{SumAsc1} \frac{P_1 \equiv Q_1 \quad P_2 \equiv Q_2 \quad P_3 \equiv Q_3}{P_1 + (P_2 + P_3) \equiv (Q_1 + Q_2) + Q_3} \quad \text{SumL} \frac{\text{SumL} \frac{Q_1 \xrightarrow{\gamma}_2 Q'}{Q_1 + Q_2 \xrightarrow{\gamma}_2 Q'}}{(Q_1 + Q_2) + Q_3 \xrightarrow{\gamma}_2 Q'}$$

for inductive hypothesis on $P_1 \equiv Q_1 \xrightarrow{\gamma}_2 Q'$: $P_1 \equiv_2 R_1$, $R_1 \xrightarrow{\gamma}_2 R'_1$ and $Q' \equiv R'_1$. Then $P_1 + (P_2 + P_3) \equiv_2 R_1 + (P_2 + P_3)$ and for rule *SumL*: $R_1 + (P_2 + P_3) \xrightarrow{\gamma}_2 R'_1$.

(*SumAsc1*, *SumL*)(2) :

$$\text{SumAsc1} \frac{P_1 \equiv Q_1 \quad P_2 \equiv Q_2 \quad P_3 \equiv Q_3}{P_1 + (P_2 + P_3) \equiv (Q_1 + Q_2) + Q_3} \quad \text{SumL} \frac{\text{SumR} \frac{Q_2 \xrightarrow{\gamma}_2 Q'_2}{Q_1 + Q_2 \xrightarrow{\gamma}_2 Q'_2}}{(Q_1 + Q_2) + Q_3 \xrightarrow{\gamma}_2 Q'_2}$$

similar.

(*SumAsc1*, *SumR*) similar to the cases (*SumAsc1*, *SumL*).

(*ParAsc1*, *ParL*) similar.

(*ParAsc1*, *ParR*) similar.

(*SumCom*, *SumL*) :

$$\text{SumCom} \frac{P_1 \equiv Q_1 \quad P_2 \equiv Q_2}{P_1 + P_2 \equiv Q_2 + Q_1} \quad \text{SumL} \frac{Q_2 \xrightarrow{\gamma}_2 Q'_2}{Q_2 + Q_1 \xrightarrow{\gamma}_2 Q'_2}$$

for inductive hypothesis on $P_2 \equiv Q_2 \xrightarrow{\gamma}_2 Q'_2$: $P_2 \equiv_2 R_2$ and $R_2 \xrightarrow{\gamma}_2 R'_2$ and $R'_2 \equiv Q'_2$. $P_1 + P_2 \equiv_2 P_1 + R_2$ Then for rule *SumR*: $P_1 + R_2 \xrightarrow{\gamma}_2 R'_2$

(*SumCom*, *SumR*) similar to the previous case.

(*ParCom*, *ParL*) similar to the previous case.

(*ParCom*, *ParR*) similar to the previous case.

(*ScpExtSum1*, $_$) in this case the conclusion holds because *ScpExtSum1* is also in \equiv_2 .

(*ScpExtPar1*, $_$) in this case the conclusion holds because *ScpExtPar1* is also in \equiv_2 .

(*CongSum*, $_$) in this case the conclusion holds because *CongSum* is also in \equiv_2 . Every other cases the first rule is congruence rule, hold for the same reason.

(*Alp*, $_$) in this case the conclusion holds thanks to rule *Alp* of \rightarrow_2 .

(*Ide*, *Ide*) similar to the previous case.

(*ParAsc1*, *EComL*)

(*ParAsc1*, *EComR*)

(*ParAsc1*, *ClsL*)

(*ParAsc1*, *ClsR*)

(*ParCom*, *EComL*)

(*ParCom*, *EComR*)

(*ParCom*, *ClsL*)

(*ParCom*, *ClsR*)

□

Lemma 2.6.12. If $P \equiv_2 Q$ and $Q \xrightarrow{\alpha}_2 Q'$ and we add the rules

$$\text{ScpExtPar} \frac{P \xrightarrow{\alpha}_2 P' \quad x \notin fn(P)}{P | (\nu x)Q \xrightarrow{\alpha}_2 (\nu x)(P | Q)} \quad \text{ScpExtSum} \frac{P \xrightarrow{\alpha}_2 P' \quad x \notin fn(P)}{P + (\nu x)Q \xrightarrow{\alpha}_2 (\nu x)(P + Q)}$$

then $P \xrightarrow{\alpha}_2 R'$ and $R' \equiv Q'$.

Proof.

□

Theorem 2.6.13. Let \rightarrow_2 be the extended semantic of lemma 2.6.12. If $P \xrightarrow{\alpha}_3 P'$ then $P \xrightarrow{\alpha}_3 Q'$ and $P' \equiv Q'$.

2.6.2 Equivalence of the late semantics

2.7 Normal form

Definition 2.7.1. Let $(\nu x)Q$ be an occurrence in a process P , i.e., there is a context $C[_]$ such that $C[(\nu x)Q] = P$. We say that this occurrence is *guarded* if it occurs right inside a prefix. Otherwise we say that the occurrence is *unguarded*. More formally the occurrence $(\nu x)Q$ is *guarded* in P if there is a context $C[_]$, an action prefixing α and names \tilde{y} such that $P = C[\alpha.(\nu \tilde{y})(\nu x)Q]$

Definition 2.7.2. We say that a process is in *normal form* if all bound names are distinct and all unguarded restrictions are at the top level, i.e., of the form $(\nu \tilde{x})P$ where P has no unguarded restrictions, note that \tilde{x} can eventually be empty. If a process P is in normal form, we write for short P n.f..

Lemma 2.7.1 (Existence of a normal form). Every process is structurally congruent to a process in normal form (but this process is not unique).

Proof. Let P be a process. We have to show that there exists a process N such that $P \equiv N$ and N is in normal form. We prove this by structural induction on P :

0 in this case $P = 0$ is already in normal form.

$\alpha.P_1$ for inductive hypothesis there exists a process N such that $P_1 \equiv N$ and N is in normal form. Then $\alpha.P_1 \equiv \alpha.N$ and $\alpha.N$ is in normal form.

$P_1 + P_2$ for inductive hypothesis there exist processes N_1 and N_2 such that $P_1 \equiv N_1$, $P_2 \equiv N_2$ and N_1, N_2 are in normal form. If N_1 or N_2 have unguarded restrictions at the top level then $N_1 + N_2$ is not in normal form but we can move the restrictions up to the top level using α equivalence and the rule

$$(\nu x)(P + Q) \equiv P + (\nu x)Q \quad \text{if } x \notin fn(P)$$

and we get something that is in normal form and structurally equivalent to $N_1 + N_2$ and so to $P_1 + P_2$.

$P_1 | P_2$ similar.

$(\nu x)P_1$ for inductive hypothesis there exists a process N such that $P_1 \equiv N$ and N is in normal form. $(\nu x)N$ is in normal form and it is structurally congruent to P .

□

Definition 2.7.3. The *late transition relation for normal forms* is the smallest relation induced by the rules in table 4.4, written \rightarrow_n . Every process in the head of transition in the premise of a rule in table 4.4 is assumed to be in normal form. Also when we write $(\nu \tilde{x})P$ is a normal form, it means that P has no restriction at the top level.

| | | | | | |
|--------------|---|------------|---|-------------|--|
| Out | $\frac{}{\bar{x}y.P \xrightarrow{\bar{x}y}_n P}$ | Tau | $\frac{}{\tau.P \xrightarrow{\tau}_n P}$ | LInp | $\frac{}{x(y).P \xrightarrow{x(y)}_n P}$ |
| LComL | $\frac{(\nu\tilde{a})P \xrightarrow{x(y)}_n (\nu\tilde{b})P' \quad (\nu\tilde{c})Q \xrightarrow{\bar{x}z}_n (\nu\tilde{d})Q'}{(\nu\tilde{a}\tilde{b})(P Q) \xrightarrow{\tau}_n (\nu\tilde{c}\tilde{d})(P'\{z/y\} Q')}$ | | | | |
| SumL | $\frac{(\nu\tilde{a})P \xrightarrow{\sigma}_n (\nu\tilde{b})P' \quad (\nu\tilde{c})Q \text{ n.f.} \quad \tilde{c} \cap n(\sigma) = \emptyset}{(\nu\tilde{a}\tilde{c})(P + Q) \xrightarrow{\sigma}_n (\nu\tilde{b}\tilde{c})P'}$ | | | | |
| Res | $\frac{P \xrightarrow{\sigma}_n P' \quad z \notin n(\sigma)}{(\nu z)P \xrightarrow{\sigma}_n (\nu z)P'}$ | Opn | $\frac{P \xrightarrow{\bar{x}z}_n P' \quad z \neq x}{(\nu z)P \xrightarrow{\bar{x}(z)}_n P'}$ | | |
| ParL | $\frac{(\nu\tilde{a})P \xrightarrow{\sigma}_n (\nu\tilde{b})P' \quad bn(\sigma) \cap fn(Q) = \emptyset \quad (\nu\tilde{c})Q \text{ n.f.} \quad \tilde{c} \cap n(\sigma) = \emptyset}{(\nu\tilde{a}\tilde{c})(P Q) \xrightarrow{\sigma}_n (\nu\tilde{b}\tilde{c})(P' Q)}$ | | | | |

Table 2.11: π calculus late semantic for normal forms. Every process in the head of a transition in the premise of a rule is in normal form. The restrictions can be empty. When we write $(\nu\tilde{x})P$ n.f. it means that P has no restriction at the top level. Commutative counterpart of rule *ParL*, *SumL*, *EComL* are omitted. Also in rules for prefix P is a normal form

2.8 Bisimilarity, congruence and equivalence

We present here some behavioural equivalences and some of their properties. In the following we will use the phrase $bn(\alpha)$ is fresh in a definition to mean that the name in $bn(\alpha)$, if any, is different from any free name occurring in any of the agents in the definition. We write \rightarrow_E for the early semantic and \rightarrow_L for the late semantic. It's not a concern which late semantic we are talking about because we have proved them equivalent.

2.8.1 Late bisimilarity

Definition 2.8.1. A *strong late bisimulation* (according to [4]) is a binary symmetric relation **S** on processes such that for each process P and Q , PSQ implies:

- if $P \xrightarrow{a(x)}_L P'$ and $x \notin fn(P) \cup fn(Q)$ then there exists a process Q' such that $Q \xrightarrow{a(x)}_L Q'$ and for all u $P'\{u/x\}SQ'\{u/x\}$
- if $P \xrightarrow{\alpha}_L P'$, α is not an input and $bn(\alpha) \cap (fn(P) \cup fn(Q)) = \emptyset$ then there exists a process Q' such that $Q \xrightarrow{\alpha}_L Q'$ and $P'SQ'$

P and Q are *late bisimilar* written $P \sim_L Q$ if there exists a strong late bisimulation **S** such that PSQ .

Example Strong late bisimulation is not closed under substitution in general:

$$a(u).0|\bar{b}v.0 \sim_L a(u).\bar{b}v.0 + \bar{b}v.a(u).0$$

and the bisimulation (without the symmetric part) is the following:

$$\{(a(u).0|\bar{b}v.0, a(u).\bar{b}v.0 + \bar{b}v.a(u).0), (a(u).0|0, a(u).0), (0|0, 0), (0|\bar{b}v.0, \bar{b}v.0)\}$$

If we apply the substitution $\{a/b\}$ to each process then they are not strongly bisimilar anymore because $(a(u).0|\bar{b}v.0)\{a/b\}$ is $a(u).0|\bar{a}v.0$ and this process can perform an invisible action whether $(a(u).\bar{b}v.0 + \bar{b}v.a(u).0)\{a/b\}$ cannot.

We refer to strong late bisimulation as strong *ground* late bisimulation, because it is not preserved by substitution.

Proposition 2.8.1. If $P \sim Q$ and σ is injective then $P\sigma \sim Q\sigma$

Proposition 2.8.2. \sim_L is an equivalence

Proposition 2.8.3. \sim_L is preserved by all operators except input prefix

Definition 2.8.2. Two processes P and Q are *strong late equivalent* written $P \sim_L Q$ if for each substitution σ $P\sigma \sim_L Q\sigma$

Example If $z \notin (fn(R) \cup \{x\})$ then $x(y).R \sim_L (z)x(y).R$

2.8.2 Early bisimilarity

Definition 2.8.3. A *strong early bisimulation* (according to [4]) is a symmetric binary relation \mathbf{S} on processes such that for each process P and Q : PSQ , $P \xrightarrow{\alpha}_E P'$ and $bn(\alpha) \cap (fn(P) \cup fn(Q)) = \emptyset$ implies that there exists Q' such that $Q \xrightarrow{\alpha}_E Q'$ and $P'SQ'$. P and Q are *early bisimilar* written $P \sim_E Q$ if there exists a strong early bisimulation \mathbf{S} such that PSQ

Definition 2.8.4. Two processes P and Q are *strong early equivalent* written $P \sim_E Q$ if for each substitution σ : $P\sigma \sim_E Q\sigma$

2.8.3 Congruence

Definition 2.8.5. We say that two agents P and Q are *strongly congruent*, written $P \sim Q$ if

$$P\sigma \sim Q\sigma \text{ for all substitution } \sigma$$

Proposition 2.8.4. Strong congruence is the largest congruence in bisimilarity.

2.8.4 Open bisimilarity

Definition 2.8.6. A *distinction* is a finite symmetric and irreflexive binary relation on names. A substitution σ *respects* a distinction D if for each name a, b aDb implies $\sigma(a) \neq \sigma(b)$. We write $D\sigma$ for the composition of the two relation.

Definition 2.8.7. An *strong open simulation* (according to [4]) is $\{S_D\}_{D \in \mathbb{D}}$ a family of binary relations on processes such that for each process P, Q , for each distinction $D \in \mathbb{D}$, for each name substitution σ which respects D if PS_DQ , $P\sigma \xrightarrow{\alpha} P'$ and $bn(\alpha) \cap (fn(P\sigma) \cup fn(Q\sigma)) = \emptyset$ then:

- if $\alpha = \bar{a}(x)$ then there exists Q' such that $Q\sigma \xrightarrow{\bar{a}(x)} Q'$ and $P'S_{D'}Q'$ where $D' = D\sigma \cup \{x\} \times (fn(P\sigma) \cup fn(Q\sigma)) \cup (fn(P\sigma) \cup fn(Q\sigma)) \times \{x\}$
- if α is not a bound output then there exists Q' such that $Q\sigma \xrightarrow{\alpha} Q'$ and $P'S_{D\sigma}Q'$

P and Q are *open D bisimilar*, written $P \sim_O^D Q$ if there exists a member S_D of an open bisimulation such that PS_DQ ; they are *open bisimilar* if they are open \emptyset bisimilar, written $P \sim_O Q$.

Chapter 3

Multi π calculus with strong output

3.1 Syntax

As we did with π calculus, we suppose that we have a countable set of names \mathbb{N} , ranged over by lower case letters a, b, \dots, z . These names are used for communication channels and values. Furthermore we have a set of identifiers, ranged over by A . We represent the agents or processes by upper case letters P, Q, \dots . A multi π process, in addition to the same actions of a π process, can perform also a strong prefix output:

$$\pi ::= \bar{x}y \mid x(z) \mid \underline{\bar{x}y} \mid \tau$$

The process are defined, just as original π calculus, by the following grammar:

$$P, Q ::= 0 \mid \pi.P \mid P|Q \mid P + Q \mid (\nu x)P \mid A(y_1, \dots, y_n)$$

and they have the same intuitive meaning as for the π calculus. The strong prefix output allows a process to make an atomic sequence of actions, so that more than one process can synchronize on this sequence. For the moment we allow the strong prefix to be on output names only. Also one can use the strong prefix only as an action prefixing for processes that can make at least a further action.

Multi π calculus is a conservative extension of the π calculus in the sense that: any π calculus process p is also a multi π calculus process and the semantic of p according to the SOS rules of π calculus is the same as the semantic of p according to the SOS rules of multi π calculus.

We have to extend the following definition to deal with the strong prefix:

$$B(\underline{\bar{x}y}.Q, I) = B(Q, I) \quad F(\underline{\bar{x}y}.Q, I) = \{x, \bar{x}, y, \bar{y}\} \cup F(Q, I)$$

3.2 Operational semantic

3.2.1 Early operational semantic with structural congruence

The semantic of a multi π process is labeled transition system such that

- the nodes are multi π calculus process. The set of node is \mathbb{P}_m
- the actions are multi π calculus actions. The set of actions is \mathbb{A}_m , we use $\alpha, \alpha_1, \alpha_2, \dots$ to range over the set of actions, we use $\sigma, \sigma_1, \sigma_2, \dots$ to range over the set $\mathbb{A}_m^+ \cup \{\tau\}$. Note that σ is a non empty sequence of actions.
- the transition relations is $\rightarrow \subseteq \mathbb{P}_m \times (\mathbb{A}_m^+ \cup \{\tau\}) \times \mathbb{P}_m$

In this case, a label can be a sequence of prefixes, whether in the original π calculus a label can be only a prefix. We use the symbol \cdot to denote the concatenation operator.

Definition 3.2.1. The *early transition relation* is the smallest relation induced by the rules in table 3.1.

| | | |
|---|--|---|
| Out $\frac{}{\bar{x}y.P \xrightarrow{\bar{x}y} P}$ | EInp $\frac{}{x(y).P \xrightarrow{xz} P\{z/y\}}$ | Tau $\frac{}{\tau.P \xrightarrow{\tau} P}$ |
| SOutSeq $\frac{P \xrightarrow{\sigma} Q \quad \sigma > 1}{\bar{x}y.P \xrightarrow{\bar{x}y \cdot \sigma} Q}$ | SOut $\frac{P \xrightarrow{\alpha} Q \quad \alpha \text{ output}}{\bar{x}y.P \xrightarrow{\bar{x}y \cdot \alpha} Q}$ | SOutTau $\frac{P \xrightarrow{\tau} Q}{\bar{x}y.P \xrightarrow{\bar{x}y} Q}$ |
| EComSng $\frac{P \xrightarrow{\bar{x}y} P' \quad Q \xrightarrow{xy} Q'}{P Q \xrightarrow{\tau} P' Q'}$ | EComSeq $\frac{P \xrightarrow{\bar{x}y \cdot \sigma} P' \quad Q \xrightarrow{xy} Q'}{P Q \xrightarrow{\sigma} P' Q'}$ | |
| Cong $\frac{P \equiv P' \quad P' \xrightarrow{\sigma} Q}{P \xrightarrow{\sigma} Q}$ | Par $\frac{P \xrightarrow{\sigma} P'}{P Q \xrightarrow{\sigma} P' Q}$ | |
| Sum $\frac{P \xrightarrow{\sigma} P'}{P + Q \xrightarrow{\sigma} P'}$ | Res $\frac{P \xrightarrow{\sigma} P' \quad z \notin n(\alpha)}{(\nu)zP \xrightarrow{\sigma} (\nu)zP'}$ | |

Table 3.1: Multi π early semantic with structural congruence

Lemma 3.2.1. If $P \xrightarrow{\sigma} Q$ then only one of the following cases hold:

- $|\sigma| = 1$
- $|\sigma| > 1$ and all the actions are output.

Example Multi-party synchronization. We show an example of a derivation of three processes that synchronize.

$$\begin{array}{c}
\text{Res} \frac{x \notin n(\tau) \quad \text{EComSeq} \frac{\bar{x}y.\bar{x}y.0|x(y).0 \xrightarrow{\bar{x}y} 0|0 \quad \text{Inp} \frac{}{x(y).0 \xrightarrow{xy} 0}}{((\bar{x}y.\bar{x}y.0|x(y).0)|x(y).0) \xrightarrow{\tau} ((0|0)|0)}}{(\nu x)((\bar{x}y.\bar{x}y.0|x(y).0)|x(y).0) \xrightarrow{\tau} (\nu x)((0|0)|0)} \\
\text{EComSng} \frac{\text{SOut} \frac{\text{Out} \frac{}{\bar{x}y.0 \xrightarrow{\bar{x}y} 0}}{\bar{x}y.\bar{x}y.0 \xrightarrow{\bar{x}y \cdot \bar{x}y} 0} \quad x(y).0 \xrightarrow{xy} 0}{\bar{x}y.\bar{x}y.0|x(y).0 \xrightarrow{\bar{x}y} 0|0}
\end{array}$$

Example Transactional synchronization In this setting two process cannot synchronize on a sequence of actions with length greater than one. This is because of the rules *EComSng* and *EComSeq*.

3.2.2 Low level semantic

This section contains the definition of an alternative semantic for multi π . First we define a low level version of the multi π calculus (here with strong prefixing on output only), we call this language low multi π . The low multi π is the multi π enriched with a marked or intermediate process $*P$:

$$P, Q ::= 0 \mid \pi.P \mid P|Q \mid P + Q \mid (\nu x)P \mid A \mid *P$$

$$\pi ::= \bar{x}y \mid x(y) \mid \underline{\bar{x}y} \mid \tau$$

Definition 3.2.2. The low level transition relation is the smallest relation induced by the rules in table 3.2 in which P stands for a process without mark, L stands for a process with mark and S can stand for both.

| | | |
|---|---|--|
| Out $\frac{}{\overline{xy}.P \vdash_{\rightarrow} P}$ | EInp $\frac{}{x(y).P \vdash_{\rightarrow} P\{z/y\}}$ | Tau $\frac{}{\tau.P \vdash_{\rightarrow} P}$ |
| SOutLow $\frac{}{\overline{xy}.P \vdash_{\rightarrow} *P}$ | StarEps $\frac{S \vdash_{\rightarrow} S'}{*S \vdash_{\rightarrow} S'}$ | StarOut $\frac{S \vdash_{\rightarrow} S'}{*S \vdash_{\rightarrow} S'}$ |
| Com1 $\frac{P \vdash_{\rightarrow} P' \quad Q \vdash_{\rightarrow} Q'}{P Q \vdash_{\rightarrow} P' Q'}$ | | |
| Com2L $\frac{L_1 \vdash_{\rightarrow} L'_1 \quad P \vdash_{\rightarrow} Q}{L_1 P \vdash_{\rightarrow} L'_1 Q}$ | Com2R $\frac{P \vdash_{\rightarrow} Q \quad L_1 \vdash_{\rightarrow} L'_1}{P L_1 \vdash_{\rightarrow} Q L'_1}$ | |
| Com3L $\frac{P \vdash_{\rightarrow} L \quad Q \vdash_{\rightarrow} Q'}{P Q \vdash_{\rightarrow} L Q'}$ | Com3R $\frac{P \vdash_{\rightarrow} P' \quad Q \vdash_{\rightarrow} L}{P Q \vdash_{\rightarrow} P' L}$ | |
| Com4L $\frac{L \vdash_{\rightarrow} Q \quad P \vdash_{\rightarrow} P'}{L P \vdash_{\rightarrow} Q P'}$ | Com4R $\frac{P \vdash_{\rightarrow} P' \quad L \vdash_{\rightarrow} Q}{P L \vdash_{\rightarrow} P' Q}$ | |
| Res $\frac{S \vdash_{\rightarrow} S' \quad y \notin n(\gamma)}{(\nu y)S \vdash_{\rightarrow} (\nu y)S'}$ | Sum $\frac{P \vdash_{\rightarrow} S}{P + Q \vdash_{\rightarrow} S}$ | Cong $\frac{P \equiv P' \quad P' \vdash_{\rightarrow} S}{P \vdash_{\rightarrow} S}$ |
| Par1L $\frac{S \vdash_{\rightarrow} S'}{S Q \vdash_{\rightarrow} S' Q}$ | Par1R $\frac{S \vdash_{\rightarrow} S'}{Q S \vdash_{\rightarrow} Q S'}$ | |

Table 3.2: Low multi π early semantic with structural congruence

Lemma 3.2.2. For all unmarked processes P, Q and marked processes L, L_1, L_2 .

- if $L_1 \xrightarrow{\alpha} L_2$ or $P \xrightarrow{\alpha} L$ then α can only be an output or an ϵ
- if $L \xrightarrow{\alpha} P$ then α can only be an output or a τ
- if $P \xrightarrow{\alpha} Q$ then α cannot be an ϵ

Definition 3.2.3. Let P, Q be unmarked processes and L_1, \dots, L_{k-1} marked processes. We define the derivation relation \rightarrow_s in the following way:

$$\mathbf{Low} \frac{P \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} Q \quad k \geq 1}{P \xrightarrow{\gamma_1 \cdots \gamma_k}_s Q}$$

We need to be precise about the concatenation operator \cdot since we have introduced the new label ϵ . Let a be an action such that $a \neq \tau$ and $a \neq \epsilon$ then the following rules hold:

$$\begin{aligned} \epsilon \cdot a &= a \cdot \epsilon = a & \epsilon \cdot \epsilon &= \epsilon & \tau \cdot \epsilon &= \epsilon \cdot \tau = \tau \\ \tau \cdot a &= a \cdot \tau = a & \tau \cdot \tau &= \tau \end{aligned}$$

Example Multi-parti synchronization

$$\begin{array}{c} \mathbf{SOutLow} \frac{}{\overline{xa}. \overline{xb}. \overline{xc}. P \xrightarrow{\overline{xa}} * \overline{xb}. \overline{xc}. P} \quad \mathbf{Inp} \frac{}{x(d). Q \xrightarrow{xa} Q\{a/d\}} \\ \mathbf{Com3L} \frac{}{\overline{xa}. \overline{xb}. \overline{xc}. P | x(d). Q \xrightarrow{\epsilon} * \overline{xb}. \overline{xc}. P | Q\{a/d\}} \\ \mathbf{Par1L} \frac{}{\overline{xa}. \overline{xb}. \overline{xc}. P | x(d). Q | x(e). R \xrightarrow{\epsilon} * \overline{xb}. \overline{xc}. P | Q\{a/d\} | x(e). R} \\ \mathbf{Par1L} \frac{}{\overline{xa}. \overline{xb}. \overline{xc}. P | x(d). Q | x(e). R | x(f). S \xrightarrow{\epsilon} * \overline{xb}. \overline{xc}. P | Q\{a/d\} | x(e). R | x(f). S} \\ \\ \mathbf{SOutLow} \frac{}{\overline{xb}. \overline{xc}. P \xrightarrow{\overline{xb}} * \overline{xc}. P} \\ \mathbf{StarOut} \frac{}{* \overline{xb}. \overline{xc}. P \xrightarrow{\overline{xb}} * \overline{xc}. P} \\ \mathbf{Par1L} \frac{}{* \overline{xb}. \overline{xc}. P | Q\{a/d\} \xrightarrow{\overline{xb}} * \overline{xc}. P | Q\{a/d\}} \quad \mathbf{EInp} \frac{}{x(e). R \xrightarrow{xb} R\{b/e\}} \\ \mathbf{Com2L} \frac{}{* \overline{xb}. \overline{xc}. P | Q\{a/d\} | x(e). R \xrightarrow{\epsilon} * \overline{xc}. P | Q\{a/d\} | R\{b/e\}} \\ \mathbf{Par1L} \frac{}{* \overline{xb}. \overline{xc}. P | Q\{a/d\} | x(e). R | x(f). S \xrightarrow{\epsilon} * \overline{xc}. P | Q\{a/d\} | R\{b/e\} | x(f). S} \\ \\ \mathbf{Out} \frac{}{\overline{xc}. P \xrightarrow{\overline{xc}} P} \\ \mathbf{StarOut} \frac{}{* \overline{xc}. P \xrightarrow{\overline{xc}} P} \\ \mathbf{Par1L} \frac{}{* \overline{xc}. P | Q\{a/d\} \xrightarrow{\overline{xc}} P | Q\{a/d\}} \\ \mathbf{Par1L} \frac{}{* \overline{xc}. P | Q\{a/d\} | R\{b/e\} \xrightarrow{\overline{xc}} P | Q\{a/d\} | R\{b/e\}} \quad \mathbf{EInp} \frac{}{x(f). S \xrightarrow{xc} R\{c/f\}} \\ \mathbf{Com4L} \frac{}{* \overline{xc}. P | Q\{a/d\} | R\{b/e\} | x(f). S \xrightarrow{\tau} P | Q\{a/d\} | R\{b/e\} | S\{c/f\}} \end{array}$$

Proposition 3.2.3. Let \rightarrow be the relation defined in table 3.1. If $P \xrightarrow{\sigma} Q$ then there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q \quad \text{and} \quad \gamma_1 \cdots \gamma_{k+1} = \sigma$$

Proof. The proof is by induction on the depth of the derivation tree of $P \xrightarrow{\sigma} Q$ and by cases on the last rule used in the derivation:

EInp, Out, Tau These rules are also in table 3.2 so we can derive $P \xrightarrow{\sigma} Q$.

SOutSeq : the last part of the derivation tree looks like this:

$$\mathbf{SOutSeq} \frac{P_1 \xrightarrow{\sigma} Q \quad |\sigma| > 1}{\overline{xy}. P_1 \xrightarrow{\overline{xy} \cdot \sigma} Q}$$

for inductive hypothesis there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q \quad \text{and} \quad \gamma_1 \cdot \dots \cdot \gamma_{k+1} = \sigma$$

then a proof of the conclusion follows from:

$$\mathbf{SOutLow} \frac{}{\overline{xy}.P_1 \xrightarrow{\overline{xy}} *P_1} \quad \mathbf{Star} \frac{P_1 \xrightarrow{\gamma_1} L_1}{*P_1 \xrightarrow{\gamma_1} L_1}$$

SOut : this case is similar to the previous.

SOutTau : this case is similar to the previous observing that $\overline{xy} \cdot \tau = \overline{xy}$.

Sum : the last part of the derivation tree looks like this:

$$\mathbf{Sum} \frac{P_1 \xrightarrow{\sigma} Q}{P_1 + P_2 \xrightarrow{\sigma} Q}$$

for the inductive hypothesis there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q \quad \text{and} \quad \gamma_1 \cdot \dots \cdot \gamma_{k+1} = \sigma$$

A proof of the conclusion is:

$$\mathbf{Sum} \frac{P_1 \xrightarrow{\gamma_1} L_1}{P_1 + P_2 \xrightarrow{\gamma_1} L_1}$$

Cong : this case is similar to the previous.

EComSng : the last part of the derivation tree looks like this:

$$\mathbf{Com} \frac{P_1 \xrightarrow{\overline{xy}} P'_1 \quad Q_1 \xrightarrow{xy} Q'_1}{P_1|Q_1 \xrightarrow{\tau} P'_1|Q'_1}$$

for inductive hypothesis there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} P'_1 \quad \text{and} \quad \gamma_1 \cdot \dots \cdot \gamma_{k+1} = \overline{xy}$$

and there exist R_1, \dots, R_h and $\delta_1, \dots, \delta_{h+1}$ with $h \geq 0$ such that

$$Q_1 \xrightarrow{\delta_1} R_1 \xrightarrow{\delta_2} R_2 \cdots R_{h-1} \xrightarrow{\delta_h} R_h \xrightarrow{\delta_{h+1}} Q'_1 \quad \text{and} \quad \delta_1 \cdot \dots \cdot \delta_{h+1} = xy$$

For lemma 3.2.2 there cannot be an input action in a transition involving marked processes so h must be 0 and $Q_1 \xrightarrow{\delta_1} Q'_1$ with $\delta_1 = xy$. Just one of the γ s is \overline{xy} and the others are ϵ or τ . We can have three different cases now:

$\gamma_1 = \overline{xy}$: A proof of the conclusion is:

$$P_1|Q_1 \xrightarrow{\tau} L_1|Q'_1 \xrightarrow{\epsilon} L_2|Q'_1 \cdots \xrightarrow{\epsilon} L_k|Q'_1 \xrightarrow{\tau} P'_1|Q'_1$$

we derive the first transition with rule *Com3L*, whether for the other transition we use the rule *Par1L*.

$\gamma_i = \overline{xy}$: A proof of the conclusion is:

$$P_1|Q_1 \xrightarrow{\epsilon} L_1|Q_1 \cdots \xrightarrow{\epsilon} L_{i-1}|Q_1 \xrightarrow{\epsilon} L_i|Q'_1 \xrightarrow{\epsilon} L_{i+1}|Q'_1 \cdots \xrightarrow{\epsilon} L_k|Q'_1 \xrightarrow{\tau} P'_1|Q'_1$$

we derive the transaction $L_{i-1}|Q_1 \xrightarrow{\epsilon} L_i|Q'_1$ with rule *Com2L*, whether for the other transactions we use the rule *Par1L*.

$\gamma_{k+1} = \bar{x}y$ similar.

Res : the last part of the derivation tree looks like this:

$$\mathbf{Res} \frac{P_1 \xrightarrow{\sigma} Q_1 \quad z \notin n(\sigma)}{(\nu z)P_1 \xrightarrow{\sigma} (\nu z)Q_1}$$

for the inductive hypothesis there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q_1 \quad \text{and} \quad \gamma_1 \cdot \dots \cdot \gamma_{k+1} = \sigma$$

We can apply the rule *Res* to each of the previous transitions because

$$z \notin n(\sigma) \text{ implies } z \notin n(\gamma_i) \text{ for each } i$$

and then get a proof of the conclusion:

$$(\nu z)P_1 \xrightarrow{\gamma_1} (\nu z)L_1 \xrightarrow{\gamma_2} (\nu z)L_2 \cdots (\nu z)L_{k-1} \xrightarrow{\gamma_k} (\nu z)L_k \xrightarrow{\gamma_{k+1}} (\nu z)Q_1$$

Par : this case is similar to the previous.

EComSeq : the last part of the derivation tree looks like this:

$$\mathbf{EComSeq} \frac{P_1 \xrightarrow{\bar{x}y \cdot \sigma} P'_1 \quad Q_1 \xrightarrow{xy} Q'_1}{P_1|Q_1 \xrightarrow{\sigma} P'_1|Q'_1}$$

for inductive hypothesis there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} P'_1 \quad \text{and} \quad \gamma_1 \cdot \dots \cdot \gamma_{k+1} = \bar{x}y \cdot \sigma$$

For inductive hypothesis and lemma 3.2.2 $Q_1 \xrightarrow{xy} Q'_1$. We can have two different cases now depending on where the first $\bar{x}y$ is:

$\gamma_1 = \bar{x}y$: A proof of the conclusion is:

$$P_1|Q_1 \xrightarrow{\epsilon} L_1|Q'_1 \xrightarrow{\gamma_2} L_2|Q'_1 \cdots \xrightarrow{\gamma_k} L_k|Q'_1 \xrightarrow{\gamma_{k+1}} P'_1|Q'_1$$

we derive the first transition with rule *Com3L*, whether for the other transactions we use the rule *Par1L*. Since $\gamma_1 \cdot \dots \cdot \gamma_{k+1} = \bar{x}y \cdot \sigma$ and $\gamma_1 = \bar{x}y$ then $\epsilon \cdot \gamma_2 \cdot \dots \cdot \gamma_{k+1} = \sigma$

$\gamma_i = \bar{x}y$: A proof of the conclusion is:

$$P_1|Q_1 \xrightarrow{\epsilon} L_1|Q_1 \cdots \xrightarrow{\epsilon} L_{i-1}|Q_1 \xrightarrow{\epsilon} L_i|Q'_1 \xrightarrow{\gamma_{i+1}} L_{i+1}|Q'_1 \cdots \xrightarrow{\gamma_k} L_k|Q'_1 \xrightarrow{\gamma_{k+1}} P'_1|Q'_1$$

we derive the transition $L_{i-1}|Q_1 \xrightarrow{\epsilon} L_i|Q'_1$ with rule *Com2L*, whether for the other transactions of the premises we use the rule *Par1L*.

$\gamma_{k+1} = \bar{x}y$: cannot happen because σ is not empty.

□

We would like to prove the converse of the previous proposition, namely: if there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q \quad \text{and} \quad \gamma_1 \cdot \dots \cdot \gamma_{k+1} = \sigma$$

then $P \xrightarrow{\sigma} Q$. But this is false as shown by these examples:

Example Interleaving

$$\begin{array}{c}
\text{SOutLow} \frac{}{\overline{xy}.\underline{ab}.\overline{xy}.0 \vdash^{\overline{xy}} * \underline{ab}.\overline{xy}.0} \quad \text{EInp} \frac{}{\overline{x(y)}.0 \vdash^{xy} 0} \\
\text{Com3L} \frac{}{\overline{xy}.\underline{ab}.\overline{xy}.0|x(y).0 \vdash^{\epsilon} * \underline{ab}.\overline{xy}.0|0} \\
\text{Par1L} \frac{}{\overline{xy}.\underline{ab}.\overline{xy}.0|x(y).0|x(y).0 \vdash^{\epsilon} * \underline{ab}.\overline{xy}.0|0|x(y).0} \\
\\
\text{SOutLow} \frac{}{\overline{ab}.\overline{xy}.0 \vdash^{\overline{ab}} * \overline{xy}.0} \\
\text{StarOut} \frac{}{* \underline{ab}.\overline{xy}.0 \vdash^{\overline{ab}} * \overline{xy}.0} \\
\text{Par1L} \frac{}{* \underline{ab}.\overline{xy}.0|0 \vdash^{\overline{ab}} * \overline{xy}.0|0} \\
\text{Par1L} \frac{}{* \underline{ab}.\overline{xy}.0|0|x(y).0 \vdash^{\overline{ab}} * \overline{xy}.0|0|x(y).0} \\
\\
\text{Out} \frac{}{\overline{xy}.0 \vdash^{\overline{xy}} 0} \\
\text{StarOut} \frac{}{* \overline{xy}.0 \vdash^{\overline{xy}} 0} \\
\text{Par1L} \frac{}{* \overline{xy}.0|0 \vdash^{\overline{xy}} 0|0} \quad \text{EInp} \frac{}{\overline{x(y)}.0 \vdash^{xy} 0} \\
\text{Com4L} \frac{}{* \overline{xy}.0|0|x(y).0 \vdash^{\tau} 0|0|0}
\end{array}$$

this prove:

$$\overline{xy}.\underline{ab}.\overline{xy}.0|x(y).0|x(y).0 \vdash^{\epsilon} * \underline{ab}.\overline{xy}.0|0|x(y).0 \vdash^{\overline{ab}} * \overline{xy}.0|0|x(y).0 \vdash^{\tau} 0|0|0$$

but there is no way to prove

$$\overline{xy}.\underline{ab}.\overline{xy}.0|x(y).0|x(y).0 \xrightarrow{\overline{ab}} 0|0|0$$

Example Transactional synchronization

$$\begin{array}{c}
\text{SOutLow} \frac{}{\overline{xy}.\overline{xy}.0 \vdash^{\overline{xy}} * \overline{xy}.0} \quad \text{EInp} \frac{}{\overline{x(y)}.x(y).0 \vdash^{xy} x(y).0} \\
\text{Com3L} \frac{}{\overline{xy}.\overline{xy}.0|x(y).x(y).0 \vdash^{\epsilon} * \overline{xy}.0|x(y).0} \\
\\
\text{Out} \frac{}{\overline{xy}.0 \vdash^{\overline{xy}} 0} \\
\text{StarOut} \frac{}{* \overline{xy}.0 \vdash^{\overline{xy}} 0} \quad \text{EInp} \frac{}{\overline{x(y)}.0 \vdash^{xy} 0} \\
\text{Com4L} \frac{}{* \overline{xy}.0|x(y).0 \vdash^{\tau} 0|0}
\end{array}$$

this prove:

$$\overline{xy}.\overline{xy}.0|x(y).x(y).0 \vdash^{\epsilon} * \overline{xy}.0|x(y).0 \vdash^{\tau} 0|0$$

but we cannot derive

$$\overline{xy}.\overline{xy}.0|x(y).x(y).0 \xrightarrow{\tau} 0|0$$

also we do not want to derive this transaction because the second process does not start with a strong prefix.

There is a much weaker propositions we can prove:

Proposition 3.2.4. Let \rightarrow be the relation defined in table 3.1. Let α be an action. If $P \vdash^{\alpha} Q$ then $P \xrightarrow{\alpha} Q$.

Proof. The proof is by induction the depth of the derivation of $P \vdash^{\alpha} Q$ and by cases on the last rule used in the derivation:

Out, EInp, Tau These rules are also in table 4.1 so we can derive $P \xrightarrow{\alpha} Q$.

Com1 :

$$\mathbf{Com1} \frac{P_1 \xrightarrow{\bar{x}y} Q_1 \quad P_2 \xrightarrow{xy} Q_2}{P_1|P_2 \xrightarrow{\tau} Q_1|Q_2}$$

for inductive hypothesis $P_1 \xrightarrow{\bar{x}y} Q_1$ and $P_2 \xrightarrow{xy} Q_2$ so for rule *Com* $P_1|P_2 \xrightarrow{\tau} Q_1|Q_2$

Sum :

$$\mathbf{Sum} \frac{P_1 \xrightarrow{\alpha} Q}{P_1 + P_2 \xrightarrow{\alpha} Q}$$

for inductive hypothesis $P_1 \xrightarrow{\alpha} Q$ and for rule *Sum* $P_1 + P_2 \xrightarrow{\alpha} Q$.

Res the first transition is:

$$\mathbf{Res} \frac{P_1 \xrightarrow{\alpha} Q_1 \quad z \notin n(\gamma_1)}{(\nu z)P_1 \xrightarrow{\alpha} (\nu z)Q_1}$$

for inductive hypothesis $P_1 \xrightarrow{\alpha} Q_1$ and for rule *Res* $(\nu z)P_1 \xrightarrow{\alpha} (\nu z)Q_1$.

others : other cases are similar.

□

Since it's important to give a low level semantic which is equivalent to the high level one, we can propose a change to the low level semantic that gets closer to our purpose. We replace the rule *Com3L*, *Com3R*, *Com2L* and *Com2R* with:

$$\begin{array}{ll} \mathbf{Com2LStop} \frac{L_1 \xrightarrow{\bar{x}y} L_2 \quad P \xrightarrow{xy} Q}{L_1|P \xrightarrow{\epsilon} L_2|stop(Q)} & \mathbf{Com2RStop} \frac{P \xrightarrow{xy} Q \quad L_1 \xrightarrow{\bar{x}y} L_2}{P|L_1 \xrightarrow{\epsilon} stop(Q)|L_2} \\ \mathbf{Com3LStop} \frac{P \xrightarrow{\bar{x}y} L \quad Q \xrightarrow{xy} Q'}{P|Q \xrightarrow{\epsilon} L|stop(Q')} & \mathbf{Com3RStop} \frac{P \xrightarrow{xy} P' \quad Q \xrightarrow{\bar{x}y} L}{P|Q \xrightarrow{\epsilon} stop(P')|L} \end{array}$$

where $stop(P)$ is a multi π process which cannot make any transition.

Definition 3.2.4. The *erase function* er is a function that eliminates the *stop* mark on processes. Its definition is straightforward.

Proposition 3.2.5. Let \rightarrow be the relation defined in table 3.1. If $P \xrightarrow{\sigma} Q$ then there exist L_1, \dots, L_k with $k \geq 0$ such that

$$P \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q' \quad \text{and} \quad \gamma_1 \cdots \gamma_{k+1} = \sigma \quad \text{and} \quad er(Q') = Q$$

Proof. The proof of the first part of this proposition is almost exactly as the proof of proposition 3.2.3. □

Proposition 3.2.6. Let \rightarrow be the relation defined in table 3.1. If there exist L_1, \dots, L_k with $k \geq 1$ such that

$$P \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q$$

and

- if all γ s are ϵ or τ then $P \xrightarrow{\tau} er(Q)$
- if there is an output $\bar{x}y$ in the γ s and all the other all γ s are ϵ or τ then $P \xrightarrow{\bar{x}y} er(Q)$

Proof. The proof is by induction on the depth of the derivation of $P \xrightarrow{\gamma_1} L_1$ and by cases on the last rule used in the derivation:

SOutLow :

$$\mathbf{SOutLow} \frac{}{\underbrace{\overline{xy}.P_1}_P \xrightarrow{\overline{xy}} \underbrace{*P_1}_{L_1}}$$

since $*P_1$ has a mark at the top level, the last rule used to derive $*P_1 \xrightarrow{\gamma_2}$ has to be *StarEps* so we have $P_1 \xrightarrow{\gamma_2} L_2$ or $P_1 \xrightarrow{\gamma_2} Q$ depending on k . We can build the following chain of transition:

$$P_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q$$

since γ_1 is an output, the other γ s are ϵ or τ , then we can apply the inductive hypothesis to get $P_1 \xrightarrow{\tau} er(Q)$. Now a proof of the conclusion is

$$\mathbf{SOutTau} \frac{P_1 \xrightarrow{\tau} er(Q)}{\overline{xy}.P_1 \xrightarrow{\overline{xy}} er(Q)}$$

Sum the first transition is:

$$\mathbf{Sum} \frac{P_1 \xrightarrow{\gamma_1} L_1}{P_1 + P_2 \xrightarrow{\gamma_1} L_1}$$

so we can build the following chain of transition:

$$P_1 \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q$$

apply the inductive hypothesis to get $P_1 \xrightarrow{\alpha} er(Q)$ where α is τ or an output. Now a proof of the conclusion is

$$\mathbf{Sum} \frac{P_1 \xrightarrow{\alpha} er(Q)}{P_1 + P_2 \xrightarrow{\alpha} er(Q)}$$

Res the first transition is:

$$\mathbf{Res} \frac{P_1 \xrightarrow{\gamma_1} L'_1 \quad z \notin n(\gamma_1)}{(\nu z)P_1 \xrightarrow{\gamma_1} (\nu z)L'_1}$$

given that L_1 has a restriction at the top level, all the other intermediate processes L_2, \dots, L_k and Q have the same restriction at the top level. This is because the only rule whose conclusion is a transition that start from a possibly marked process with a restriction at its top level is *Res*. So the last rule used to prove all transition is *Res*.

$$\mathbf{Res} \frac{L'_k \xrightarrow{\gamma_{k+1}} Q' \quad z \notin n(\tau)}{(\nu z)L'_k \xrightarrow{\gamma_{k+1}} (\nu z)Q'} \quad \mathbf{Res} \frac{L'_i \xrightarrow{\gamma_i} L'_{i+1} \quad z \notin n(\epsilon)}{(\nu z)L'_i \xrightarrow{\gamma_i} (\nu z)L'_{i+1}}$$

we can build the following chain of transitions:

$$P_1 \xrightarrow{\gamma_1} L'_1 \xrightarrow{\gamma_2} L'_2 \cdots L'_{k-1} \xrightarrow{\gamma_k} L'_k \xrightarrow{\gamma_{k+1}} Q'$$

then apply the inductive hypothesis to get $P_1 \xrightarrow{\alpha} er(Q')$. A proof of the conclusion can be

$$\mathbf{Res} \frac{P_1 \xrightarrow{\alpha} er(Q') \quad z \notin n(\tau)}{(\nu z)P_1 \xrightarrow{\alpha} (\nu z)er(Q') = er((\nu z)Q')}$$

Cong the last rule of the derivation of the first transition is:

$$\mathbf{Cong} \frac{P' \equiv P \quad \vdash^{\gamma_1} L_1}{P \vdash^{\gamma_1} L_1}$$

We derive the following chain of transition:

$$P' \vdash^{\gamma_1} L_1 \vdash^{\gamma_2} L_2 \cdots L_{k-1} \vdash^{\gamma_k} L_k \vdash^{\gamma_{k+1}} Q$$

for inductive hypothesis $P' \xrightarrow{\alpha} er(Q)$. A proof of the conclusion is

$$\mathbf{Cong} \frac{P' \equiv P \quad P' \xrightarrow{\alpha} er(Q)}{P \xrightarrow{\alpha} er(Q)}$$

Com3LStop : the last part of the derivation of the first transition is:

$$\mathbf{Com3LStop} \frac{P_1 \xrightarrow{\bar{x}y} L'_1 \quad P_2 \xrightarrow{xy} Q_2}{P_1|P_2 \xrightarrow{\epsilon} L'_1|stop(Q_2)}$$

the derivations of all other transitions can end only with an instance of *Par1L* so we have:

$$\mathbf{Par1L} \frac{L'_i \vdash^{\gamma_i} L'_{i+1}}{L'_i|stop(Q_2) \vdash^{\gamma_i} L'_{i+1}|stop(Q_2)} \quad \mathbf{Par1L} \frac{L'_k \vdash^{\gamma_{k+1}} Q_1}{L'_i|stop(Q_2) \vdash^{\gamma_{k+1}} Q_1|stop(Q_2)}$$

We derive the following chain of transition:

$$P_1 \xrightarrow{\bar{x}y} L'_1 \vdash^{\gamma_2} L'_2 \cdots L'_{k-1} \vdash^{\gamma_k} L'_k \vdash^{\gamma_{k+1}} Q_1$$

for inductive hypothesis $P_1 \xrightarrow{\bar{x}y} er(Q_1)$. A proof of the conclusion is

$$\mathbf{EComSeq} \frac{P_1 \xrightarrow{\bar{x}} er(Q_1) \quad P_2 \xrightarrow{xy} Q_2}{P_1|P_2 \xrightarrow{\tau} er(Q_1)|Q_2}$$

Par1L(1) : the last part of the derivation of the first transition is:

$$\mathbf{Par1L} \frac{P_1 \vdash^{\gamma_1} L'_1}{P_1|P_2 \vdash^{\gamma_1} L'_1|P_2}$$

the derivations of all the other transitions end with an instance of *Par1L*. We derive the following chain of transition:

$$P_1 \vdash^{\gamma_1} L'_1 \vdash^{\gamma_2} L'_2 \cdots L'_{k-1} \vdash^{\gamma_k} L'_k \vdash^{\gamma_k} Q_1$$

for inductive hypothesis $P_1 \xrightarrow{\alpha} er(Q_1)$. A proof of the conclusion is

$$\mathbf{Par} \frac{P_1 \xrightarrow{\alpha} er(Q_1)}{P_1|P_2 \xrightarrow{\alpha} er(Q_1)|P_2'}$$

Par1L(2) : the last part of the derivation of the first transition is:

$$\mathbf{Par1L} \frac{P_1 \vdash^{\gamma_1} L'_1}{P_1|P_2 \vdash^{\gamma_1} L'_1|P_2}$$

there is one derivation that ends with an instance of *Com2LStop* and the derivations of all the other transitions end with an instance of *Par1L*. We present here the case when the second transition ends with a *Com2LStop*, the other cases are similar. So

$$\mathbf{Com2LStop} \frac{L'_2 \xrightarrow{\bar{x}y} L'_2 \quad P_2 \xrightarrow{xy} P'_2}{L'_2|P_2 \xrightarrow{\epsilon} L'_2|stop(P'_2)}$$

We derive the following chain of transition:

$$P_1 \xrightarrow{\epsilon} L'_1 \xrightarrow{\bar{x}y} L'_2 \xrightarrow{\epsilon} \dots \xrightarrow{\epsilon} L'_{k-1} \xrightarrow{\epsilon} L'_k \xrightarrow{\tau} Q_1$$

for inductive hypothesis $P_1 \xrightarrow{\bar{x}y} er(Q_1)$. A proof of the conclusion is

$$\mathbf{EComSeq} \frac{P_1 \xrightarrow{\bar{x}y} er(Q_1) \quad P_2 \xrightarrow{xy} P'_2}{P_1|P_2 \xrightarrow{\tau} er(Q_1)|P'_2}$$

Par1L(3) : the last part of the derivation of the first transition is:

$$\mathbf{Par1L} \frac{P_1 \xrightarrow{\gamma_1} L'_1}{P_1|P_2 \xrightarrow{\gamma_1} L'_1|P_2}$$

the derivation of the last transition ends with an instance of *Com4L* and the derivations of all the other transitions end with an instance of *Par1L*. We derive the following chain of transition:

$$P_1 \xrightarrow{\epsilon} L'_1 \xrightarrow{\epsilon} L'_2 \dots L'_{k-1} \xrightarrow{\epsilon} L'_k \xrightarrow{\bar{x}y} Q_1$$

for inductive hypothesis $P_1 \xrightarrow{\bar{x}y} er(Q_1)$. A proof of the conclusion is

$$\mathbf{EComSeq} \frac{P_1 \xrightarrow{\bar{x}y} er(Q_1) \quad P_2 \xrightarrow{xy} P'_2}{P_1|P_2 \xrightarrow{\tau} er(Q_1)|P'_2}$$

□

3.2.3 Early operational semantic without structural congruence

Definition 3.2.5. The *late transition relation without structural congruence* is the smallest relation induced by the rules in table 3.3 and in table 3.4.

Example Scope extrusion with strong prefixing(1). $x \notin fn(y(z).Q|a(b).R|y(z).S)$. The following is the desired transition:

$$(\nu x)(\bar{y}x.\bar{a}b.\bar{y}x.\bar{a}b.P)|y(z).Q|a(b).R|y(z).S|a(b).T \xrightarrow{\tau} (\nu x)(P|Q\{x/z\}|S\{x/z\})|R|T$$

It is possible to infer this transition in the semantic with structural congruence. But without structural congruence and the following scope extrusion rules

$$\begin{aligned} \mathbf{Opn} \quad & \frac{P \xrightarrow{\sigma} P' \quad y \in obj(\sigma) \quad y \notin sbj(\sigma)}{(\nu y)P \xrightarrow{opn(\sigma,y)} P'} \\ \mathbf{Cls} \quad & \frac{P \xrightarrow{\bar{x}(z)} P' \quad Q \xrightarrow{xz} Q'}{P|Q \xrightarrow{\tau} (\nu z)(P'|Q')} \quad \mathbf{ClsSeq} \quad \frac{P \xrightarrow{\bar{x}(z).\sigma} P' \quad Q \xrightarrow{xz} Q'}{P|Q \xrightarrow{\sigma} (\nu z)(P'|Q')} \end{aligned}$$

| | | |
|--|---|---|
| SOut $\frac{n \geq 0}{\underline{\bar{x}_1 y_1} \dots \underline{\bar{x}_n y_n} \bar{x} y . P \xrightarrow{\bar{x} y \cdot \bar{x} y} P}$ | EInp $\frac{w \notin (fn(P) - \{z\})}{x(z) . P \xrightarrow{xw} P\{w/z\}}$ | Tau $\frac{}{\tau . P \xrightarrow{\tau} P}$ |
| EComSeq $\frac{P \xrightarrow{\bar{x} y \cdot \sigma} P' \quad Q \xrightarrow{xy} Q'}{P Q \xrightarrow{\sigma} P' Q'}$ | ECom $\frac{P \xrightarrow{\bar{x} y} P' \quad Q \xrightarrow{xy} Q'}{P Q \xrightarrow{\tau} P' Q'}$ | |
| ParL $\frac{P \xrightarrow{\sigma} P' \quad bn(\sigma) \cap fn(Q) = \emptyset}{P Q \xrightarrow{\sigma} P' Q}$ | ParR $\frac{Q \xrightarrow{\sigma} Q' \quad bn(\sigma) \cap fn(P) = \emptyset}{P Q \xrightarrow{\sigma} P Q'}$ | |
| Res $\frac{P \xrightarrow{\sigma} P' \quad z \notin n(\sigma)}{(\nu z)P \xrightarrow{\sigma} (\nu z)P'}$ | Ide $\frac{A(\tilde{x}) \stackrel{def}{=} P \quad P\{\tilde{y}/\tilde{x}\} \xrightarrow{\sigma} Q}{A(\tilde{y}) \xrightarrow{\sigma} Q}$ | |
| SumL $\frac{P \xrightarrow{\sigma} P'}{P + Q \xrightarrow{\sigma} P'}$ | SumR $\frac{Q \xrightarrow{\sigma} Q'}{P + Q \xrightarrow{\sigma} Q'}$ | |
| Alph $\frac{P \equiv_{\alpha} Q \quad Q \xrightarrow{\sigma} P'}{P \xrightarrow{\sigma} P'}$ | | |

Table 3.3: Multi π early semantic without structural congruence part 1

| | |
|---|--|
| Opn $\frac{P \xrightarrow{\sigma} P' \quad y \in obj(\sigma) \quad y \notin sbj(\sigma)}{(\nu y)P \xrightarrow{opn(\sigma, y)} P'}$ | Cls $\frac{P \xrightarrow{\bar{x}(y) \cdot (\nu y)} P' \quad Q \xrightarrow{xy} Q'}{P Q \xrightarrow{\tau} (\nu y)(P' Q')}$ |
| ClsSeq1 $\frac{P \xrightarrow{\bar{x}(y) \cdot (\nu y) \cdot \sigma} P' \quad Q \xrightarrow{xy} Q'}{P Q \xrightarrow{\sigma} (\nu y)(P' Q')}$ | ClsSeq2 $\frac{P \xrightarrow{\bar{x}(y) \cdot \gamma} P' \quad Q \xrightarrow{xy} Q'}{P Q \xrightarrow{\gamma} P' Q'}$ γ does not start with a ν |
| $ \begin{aligned} opn(\bar{x}y, y) &= \bar{x}(y) \cdot (\nu y) & opn(\bar{x}y \cdot \sigma, y) &= \begin{cases} \bar{x}(y) \cdot opn(\sigma, y) & \text{if } y \in obj(\sigma) \\ \bar{x}(y) \cdot (\nu y) \cdot opn(\sigma, y) & \text{if } y \notin obj(\sigma) \end{cases} \\ opn(\bar{x}z, y) &= \bar{x}z & opn(\bar{x}z \cdot \sigma, y) &= \bar{x}z \cdot opn(\sigma, y) \\ opn((\nu z), y) &= (\nu z) & opn((\nu z) \cdot \sigma, y) &= (\nu z) \cdot opn(\sigma, y) \end{aligned} $ | |
| $ \begin{aligned} sbj(\tau) &= \emptyset & sbj(\bar{x}y) &= \{x\} & sbj(x(y)) &= \{x\} & sbj((\nu y)) &= \emptyset & sbj(\alpha \cdot \sigma) &= sbj(\alpha) \cup sbj(\sigma) \\ obj(\tau) &= \emptyset & obj(\bar{x}y) &= \{y\} & obj(x(y)) &= \{y\} & obj((\nu y)) &= \emptyset & obj(\alpha \cdot \sigma) &= obj(\alpha) \cup obj(\sigma) \end{aligned} $ | |

Table 3.4: Multi π late semantic: scope extrusion rules

we can only infer

$$(\nu x)(\bar{y}x.\bar{a}b.\bar{y}x.\bar{a}b.P)|y(z).Q|a(b).R|y(z).S|a(b).T \xrightarrow{\tau} (\nu x)((\nu x)(P|Q\{x/z\})|R|S\{x/z\})|T$$

This transition is not what we want because now the scope of the inner νx hides in P the scope of the outer νx , so P and S cannot use x to communicate. But with the rules of table 3.4 the following transition can be inferred:

$$(\nu x)(\bar{y}x.\bar{a}b.\bar{y}x.\bar{a}b.P)|y(z).Q|a(b).R|y(z).S|a(b).T \xrightarrow{\tau} (\nu x)(P|Q\{x/z\})|R|S\{x/z\})|T$$

Example Scope intrusion without strong prefixing.

$$\bar{y}x.P|(\nu x)(y(z).Q) \xrightarrow{\tau} P|(\nu w)(Q\{w/x\}\{x/z\})$$

Example Scope extrusion without strong prefixing.

$$\text{CIs} \frac{\text{Opn} \frac{\text{Out} \frac{}{\bar{y}x.P \xrightarrow{\bar{y}x} P}}{(\nu x)(\bar{y}x.P) \xrightarrow{\bar{y}(x)(\nu x)} P} \quad \text{EInp} \frac{}{y(z).Q \xrightarrow{yx} Q\{x/z\}}}{(\nu x)(\bar{y}x.P)|y(z).Q \xrightarrow{\tau} (\nu x)(P|Q\{x/z\})}$$

Example Scope extrusion with strong prefixing(2). $x \in fn(y(z).Q|a(b).R|y(z).S)$ and $x' \notin fn(y(z).Q|a(b).R|y(z).S)$:

$$(\nu x)(\bar{y}x.\bar{a}b.\bar{y}x.\bar{a}b.P)|y(z).Q|a(b).R|y(z).S|a(b).T \xrightarrow{\tau} (\nu x')(P\{x'/x\}|Q\{x'/z\}|R|S\{x'/z\})|T$$

Example Scope intrusion with strong prefixing.

$$\bar{y}x.\bar{a}b.P|(\nu x)(y(z).Q)|(\nu b)(a(c).R) \xrightarrow{\tau} P|(\nu w)(Q\{w/x\}\{x/z\})|(\nu d)(R\{d/b\}\{b/c\})$$

In the following section we will try to prove that strong early bisimulation is preserved by some operators. Let \rightarrow be the smallest relation induced by the rules in table 3.3 excluding the rule Alp , and by the rules in table 3.4. We would like to use the fact that: $P \xrightarrow{\gamma} Q$ imply $P \xrightarrow{\gamma} S$ and $S \equiv_{\alpha} Q$. But this does not hold because for example:

$$(\nu x)z(a).0 \equiv_{\alpha} (\nu y)z(a).0 \xrightarrow{zx} (\nu y)0 \quad (\nu x)z(a).0 \not\xrightarrow{zx}$$

But we do have to following:

Lemma 3.2.7. If $P \xrightarrow{\gamma} Q$ then $P \equiv_{\alpha} R \xrightarrow{\gamma} S \equiv_{\alpha} Q$.

Proof. Similar to the proof of lemma 2.6.2 □

Lemma 3.2.8. Let \rightarrow be the semantic in table 3.3 and 3.4 but without rule Alp . $P \xrightarrow{\gamma} Q$ and $b \notin n(P)$ imply $P\{b/a\} \xrightarrow{\gamma\{b/a\}} S$ and $S \equiv_{\alpha} Q\{b/a\}$.

Proof. By induction on the derivation of $P \xrightarrow{\gamma} Q$ and by cases on the last rule used in the derivation:

Out : $\bar{x}y.P \xrightarrow{\bar{x}y} P$. For definition $(\bar{x}y.P)\{b/a\} = \bar{x}\{b/a\}y\{b/a\}.(P\{b/a\})$ and for rule *Out*:

$$\bar{x}\{b/a\}y\{b/a\}.(P\{b/a\}) \xrightarrow{(\bar{x}y)\{b/a\}} P\{b/a\}$$

Tau similar.

SOut suppose for induction hypothesis that $P\{b/a\} \xrightarrow{\gamma\{b/a\}} Q\{b/a\}$ and that γ is a non empty sequence of outputs. Then for rule *SOut*: $\bar{x}y\{b/a\}.P\{b/a\} \xrightarrow{\bar{x}y\{b/a\}.\gamma\{b/a\}} Q\{b/a\}$ which for definition of substitution imply

$$(\overline{xy}.P)\{b/a\} \xrightarrow{(\overline{xy}.\gamma)\{b/a\}} Q\{b/a\}$$

Einp(1) let $y \neq b$, $y \neq a$ and $z \neq a$: $x(y).P \xrightarrow{xz} P\{z/y\}$. For definition $(x(y).P)\{b/a\} = x\{b/a\}(y).(P\{b/a\})$ and for rule *Einp*:

$$x\{b/a\}(y).P\{b/a\} \xrightarrow{x\{b/a\}z} P\{b/a\}\{z/y\}$$

and $P\{b/a\}\{z/y\} \equiv_{\alpha} P\{z/y\}\{b/a\}$

Einp(2) let $y \neq b$, $y \neq a$: $x(y).P \xrightarrow{xa} P\{a/y\}$. For definition $(x(y).P)\{b/a\} = x\{b/a\}(y).(P\{b/a\})$ and for rule *Einp*:

$$x\{b/a\}(y).P\{b/a\} \xrightarrow{x\{b/a\}a} P\{b/a\}\{a/y\}$$

and $P\{b/a\}\{a/y\} \equiv_{\alpha} P\{a/y\}\{b/a\}$

Einp(3) let $a \neq z$: $x(a).P \xrightarrow{xz} P\{z/a\}$. For definition $(x(a).P)\{b/a\} = x\{b/a\}(a).P$ and for rule *Einp*:

$$x\{b/a\}(a).P \xrightarrow{x\{b/a\}z} P\{z/a\}$$

and $P\{z/a\}\{b/a\} \equiv_{\alpha} P\{z/a\}$

Einp(4) $x(a).P \xrightarrow{xa} P$. For definition $(x(a).P)\{b/a\} = x\{b/a\}(a).P$ and for rule *Einp*:

$$x\{b/a\}(a).P \xrightarrow{x\{b/a\}b} P\{b/a\}.$$

and $P\{b/a\} \equiv_{\alpha} P\{b/a\}$

ParL :

$$\mathbf{ParL} \frac{P \xrightarrow{\gamma} P' \quad bn(\gamma) \cap fn(P) = \emptyset}{P + Q \xrightarrow{\gamma} P'}$$

for induction $P\{b/a\} \xrightarrow{\gamma\{b/a\}} S$ and $S \equiv_{\alpha} P'\{b/a\}$. For rule *Sum* and definition of substitution: $(P + Q)\{b/a\} \xrightarrow{\gamma\{b/a\}} P'\{b/a\}$ and the conclusion holds.

ParR similar.

SumL similar.

SumR similar.

ECom :

$$\mathbf{ECom} \frac{P \xrightarrow{\overline{xy}} P' \quad Q \xrightarrow{xy} Q'}{P|Q \xrightarrow{\tau} P'|Q'}$$

for induction on the first premise $P\{b/a\} \xrightarrow{(\overline{xy})\{b/a\}} S_1$ and $S_1 \equiv_{\alpha} P'\{b/a\}$. For induction on the second premise $Q\{b/a\} \xrightarrow{(xy)\{b/a\}} S_2$ and $S_2 \equiv_{\alpha} Q'\{b/a\}$. For rule *ECom* and definition of substitution: $(P|Q)\{b/a\} \xrightarrow{\tau} S_1|S_2$ and $S_1|S_2 \equiv (P'|Q')\{b/a\}$.

EComSeq similar.

Res(1) $x \neq a$:

$$\mathbf{Res} \frac{P \xrightarrow{\gamma} P' \quad x \notin n(\gamma)}{(\nu x)P \xrightarrow{\gamma} (\nu x)P'}$$

for induction $P\{b/a\} \xrightarrow{\gamma\{b/a\}} S$ and $S \equiv_{\alpha} P'\{b/a\}$. For rule *Res* and definition of substitution: $(\nu x)P\{b/a\} \xrightarrow{\gamma\{b/a\}} (\nu x)S$ and $(\nu x)S \equiv_{\alpha} (\nu x)P'\{b/a\}$.

Res(2) :

$$\mathbf{Res} \frac{P \xrightarrow{\gamma} P' \quad a \notin n(\gamma)}{(\nu a)P \xrightarrow{\gamma} (\nu a)P'}$$

the conclusion holds because $((\nu a)P)\{b/a\} = (\nu a)P$ and $\gamma\{b/a\} = \gamma$.

Ide :

$$\mathbf{Ide} \frac{A(\tilde{x}) \stackrel{\text{def}}{=} P \quad P\{\tilde{w}/\tilde{x}\} \xrightarrow{\gamma} P'}{A(\tilde{x}) \xrightarrow{\gamma} P'}$$

for induction $P\{\tilde{w}/\tilde{x}\}\{b/a\} \xrightarrow{\gamma\{b/a\}} S$ and $S \equiv_{\alpha} P'\{b/a\}$

Opn(1) $y \neq a$:

$$\mathbf{Opn} \frac{P \xrightarrow{\bar{y}x} P' \quad y \neq x}{(\nu y)P \xrightarrow{\bar{y}(x)} P'}$$

for induction $P\{b/a\} \xrightarrow{\bar{y}x} S$ and $S \equiv_{\alpha} P'\{b/a\}$

Opn(2) :

$$\mathbf{Opn} \frac{P \xrightarrow{\bar{a}x} P' \quad a \neq x}{(\nu a)P \xrightarrow{\bar{a}(x)} P'}$$

for induction $P\{b/a\} \xrightarrow{\bar{b}x} S$ and $S \equiv_{\alpha} P'\{b/a\}$. So for rule *Opn*: $(\nu b)P\{b/a\} \xrightarrow{\bar{b}(x)} S$ and for rule *Alp*: $(\nu a)P \xrightarrow{\bar{b}(x)} S$

Cls :

$$\mathbf{Cls} \frac{P \xrightarrow{\bar{x}(y) \cdot (\nu y)} P' \quad Q \xrightarrow{xy} Q'}{P|Q \xrightarrow{\tau} (\nu y)(P'|Q')}$$

for induction on the first premise $P\{b/a\} \xrightarrow{\bar{x}\{b/a\}(y) \cdot (\nu y)} S$ and $S \equiv_{\alpha} P'\{b/a\}$. For induction on the second premise $Q\{b/a\} \xrightarrow{\bar{x}\{b/a\}(y) \cdot (\nu y)} T$ and $T \equiv_{\alpha} Q'\{b/a\}$. For rule *Cls* and definition of substitution: $(P|Q)\{b/a\} \xrightarrow{\tau} (\nu y)(S|T)$ and $(\nu y)S|T \equiv_{\alpha} (\nu y)(P'\{b/a\}|Q'\{b/a\})$.

ClsSeq1 similar.

ClsSeq2 similar.

□

3.3 Strong bisimilarity and equivalence

3.3.1 Strong bisimilarity

In the following section, \rightarrow is the transition relation defined in table 3.3.

Definition 3.3.1. A *strong early bisimulation* is a symmetric binary relation \mathbf{S} on multi π processes such that for all P and Q : PSQ , $P \xrightarrow{\gamma} P'$ and $bn(\gamma)$ is fresh imply that

$$\exists Q' : Q \xrightarrow{\gamma} Q' \text{ and } P' \mathbf{S} Q'$$

The *strong early bisimilarity*, written \sim_E , is the union of all strong early bisimulation. Two processes P, Q are *strong early bisimilar*, written $P \sim_E Q$, if they are related by the strong early bisimilarity. The strong early bisimilarity is a strong early bisimulation.

Definition 3.3.2. A *strong early bisimulation up to \sim_E* is a symmetric binary relation \mathbf{S} on multi π processes such that for all P and Q : PSQ , $P \xrightarrow{\gamma} P'$ and $bn(\gamma)$ is fresh imply that

$$\exists P'', Q', Q'' : Q \xrightarrow{\gamma} Q' \text{ and } P' \sim_E P'' \mathbf{S} Q'' \sim_E Q'$$

Two processes P, Q are *strong early bisimilar up to \sim_E* , written $P \sim_E^{up} Q$, if they are related by a strong early bisimulation up to \sim_E .

3.3.2 Properties of strong early bisimilarity

Proposition 3.3.1. \sim_E is an equivalence relation.

Proof. :

Reflexivity The identity relation on processes is a strong early bisimulation.

Simmetry It is in the definition.

Transitivity The composition $\sim_E \sim_E$ is a strong early bisimulation.

□

Proposition 3.3.2. $P \sim_E^{up} Q$ imply $P \sim_E Q$.

Proof. Let \mathbf{S} be a bisimulation up to \sim_E such that PSQ . It can be proved that $\sim_E \mathbf{S} \sim_E$ is a bisimulation: let $A \sim_E BSC \sim_E D$

$$\begin{aligned} A \xrightarrow{\gamma} A' \wedge A \sim_E B \wedge \text{definition 3.3.1} &\Rightarrow \exists B' : B \xrightarrow{\gamma} B' \wedge A' \sim_E B' \\ BSC \wedge \text{definition 3.3.2} &\Rightarrow \exists C' C'' B'' : C \xrightarrow{\gamma} C' \wedge B' \sim_E B'' \mathbf{S} C'' \sim_E C' \\ C \xrightarrow{\gamma} C' \wedge C \sim_E D \wedge \text{definition 3.3.1} &\Rightarrow \exists D' : D \xrightarrow{\gamma} D' \wedge C' \mathbf{S} D' \\ A' \sim_E B' \sim_E B'' \mathbf{S} C'' \sim_E C' \sim_E D' \wedge \text{transitivity of } \sim_E &\Rightarrow A' \sim_E B'' \mathbf{S} C'' \sim_E D' \end{aligned}$$

It is easy to see that the symmetric also holds.

□

Proposition 3.3.3. If \mathbf{S} is a strong early bisimulation up to restriction then $\mathbf{S} \subseteq \sim_E$.

Proof. Let \mathbf{S} be a strong early bisimulation up to restriction then we define

$$\begin{cases} \mathbf{S}_0 = \mathbf{S} \\ \mathbf{S}_{n+1} = \{((\nu w)P, (\nu w)Q) : P \mathbf{S}_n Q, w \in \mathbf{N}\} \\ \mathbf{S}^* = \bigcup_{n < \omega} \mathbf{S}_n \end{cases}$$

Clearly $\mathbf{S} \subseteq \mathbf{S}^*$. We have to prove that \mathbf{S}^* is a strong early bisimulation. The proof is an induction on n

□

Proposition 3.3.4. \equiv_α is a strong early bisimulation.

Proof. We prove by induction on the derivation of $P \equiv_\alpha Q$ and the derivation of $P \xrightarrow{\gamma} P'$ that $Q \xrightarrow{\gamma} Q'$ and $Q' \equiv_\alpha P'$. This imply that also the symmetric holds because α equivalence is symmetric. The last pair of rules used can be:

(*AlpOut, Out*) :

$$\mathbf{AlpOut} \frac{P \equiv_\alpha Q}{\overline{xy}.P \equiv_\alpha \overline{xy}.Q} \quad \mathbf{Out} \frac{}{\overline{xy}.P \xrightarrow{\overline{xy}} P} \quad \mathbf{Out} \frac{}{\overline{xy}.Q \xrightarrow{\overline{xy}} Q}$$

(*AlpSOut, SOut*) :

$$\mathbf{AlpSOut} \frac{P \equiv_\alpha Q}{\overline{xy}.P \equiv_\alpha \overline{xy}.Q} \quad \mathbf{SOut} \frac{P \xrightarrow{\gamma} P'}{\overline{xy}.P \xrightarrow{\overline{xy} \cdot \gamma} P'}$$

$Q \equiv_\alpha P \xrightarrow{\gamma} P'$ imply for inductive hypothesis that $Q \xrightarrow{\gamma} Q' \equiv P'$. So for rule *SOut*:
 $\overline{xy}.Q \xrightarrow{\overline{xy} \cdot \gamma} Q'$

(*AlpTau, Tau*) :

$$\mathbf{AlpTau} \frac{P \equiv_\alpha Q}{\tau.P \equiv_\alpha \tau.Q} \quad \mathbf{Tau} \frac{}{\tau.P \xrightarrow{\tau} P} \quad \mathbf{Tau} \frac{}{\tau.Q \xrightarrow{\tau} Q}$$

(*AlpInp, EInp*) :

$$\mathbf{AlpInp} \frac{P \equiv_\alpha Q}{x(y).P \equiv_\alpha x(y).Q} \quad \mathbf{EInp} \frac{}{x(y).P \xrightarrow{xz} P\{z/y\}} \quad \mathbf{EInp} \frac{}{x(y).Q \xrightarrow{xz} Q\{z/y\}}$$

$P \equiv_\alpha Q$ imply for lemma 2.3.6: $P\{z/y\} \equiv_\alpha Q\{z/y\}$

(*AlpInp1, EInp*) :

$$\mathbf{AlpInp1} \frac{P \equiv_\alpha Q}{x(y).P \equiv_\alpha x(z).Q\{z/y\}} \quad \mathbf{EInp} \frac{}{x(y).P \xrightarrow{xw} P\{w/y\}}$$

For rule *EInp*: $x(z).Q\{z/y\} \xrightarrow{xw} Q\{z/y\}\{w/z\}$. $P \equiv_\alpha Q$ imply for lemma 2.3.6: $P\{w/y\} \equiv_\alpha Q\{w/y\}$, for transitivity $P\{w/y\} \equiv_\alpha Q\{z/y\}\{w/z\}$

(*AlpInp2, EInp*) similar.

(*AlpIde, Ide*) in this case P and Q are both equal to some identifier.

(*AlpPar, Par*) :

$$\mathbf{AlpPar} \frac{P_1 \equiv_\alpha Q_1 \quad P_2 \equiv_\alpha Q_2}{P_1|P_2 \equiv_\alpha Q_1|Q_2} \quad \mathbf{Par} \frac{P_1 \xrightarrow{\gamma} P'_1}{P_1|P_2 \xrightarrow{\gamma} P'_1}$$

For induction $Q_1 \xrightarrow{\gamma} Q'_1$ and $P'_1 \equiv_\alpha Q'_1$. For rule *Par*: $Q_1|Q_2 \xrightarrow{\gamma} Q'_1$.

(*AlpPar, ECom*) :

$$\mathbf{AlpPar} \frac{P_1 \equiv_\alpha Q_1 \quad P_2 \equiv_\alpha Q_2}{P_1|P_2 \equiv_\alpha Q_1|Q_2} \quad \mathbf{ECom} \frac{P_1 \xrightarrow{\overline{xy}} P'_1 \quad P_2 \xrightarrow{xy} P'_2}{P_1|P_2 \xrightarrow{\tau} P'_1|P'_2}$$

For induction on the first premise of *ECom*: $Q_1 \xrightarrow{\overline{xy}} Q'_1$ and $P'_1 \equiv_\alpha Q'_1$. For induction on the second premise: $Q_2 \xrightarrow{xy} Q'_2$ and $P'_2 \equiv_\alpha Q'_2$. For rule *ECom*: $Q_1|Q_2 \xrightarrow{\tau} Q'_1|Q'_2$ and for rule *AlpPar*: $P'_1|P'_2 \equiv_\alpha Q'_1|Q'_2$.

(*AlpPar*, *EComSeq*) similar.

(*AlpPar*, *Cls*) :

$$\mathbf{AlpPar} \frac{P_1 \equiv_\alpha Q_1 \quad P_2 \equiv_\alpha Q_2}{P_1|P_2 \equiv_\alpha Q_1|Q_2} \quad \mathbf{Cls} \frac{P_1 \xrightarrow{\bar{x}(y) \cdot (\nu x)} P'_1 \quad P_2 \xrightarrow{xy} P'_2}{P_1|P_2 \xrightarrow{\tau} (\nu x)(P'_1|P'_2)}$$

For induction on the first premise of *ECom*: $Q_1 \xrightarrow{\bar{x}(y) \cdot (\nu x)} Q'_1$ and $P'_1 \equiv_\alpha Q'_1$. For induction on the second premise: $Q_2 \xrightarrow{xy} Q'_2$ and $P'_2 \equiv_\alpha Q'_2$. For rule *Cls*: $Q_1|Q_2 \xrightarrow{\tau} (\nu x)(Q'_1|Q'_2)$ and for rules *AlpPar* and *AlpRes*: $(\nu x)(P'_1|P'_2) \equiv_\alpha (\nu x)(Q'_1|Q'_2)$.

(*AlpPar*, *ClsSeq1*) similar.

(*AlpPar*, *ClsSeq2*) similar.

(*AlpRes*, *Res*) :

$$\mathbf{AlpRes} \frac{P \equiv_\alpha Q}{(\nu y)P \equiv_\alpha (\nu y)Q} \quad \mathbf{Res} \frac{P \xrightarrow{\gamma} P' \quad y \notin n(\gamma)}{(\nu y)P \xrightarrow{\gamma} (\nu y)P'}$$

$P \equiv_\alpha Q$ and $P \xrightarrow{\gamma} P'$ imply for inductive hypothesis that $Q \xrightarrow{\gamma} Q'$ and $Q' \equiv_\alpha P'$. For rule *Res* and *AlpRes*: $(\nu y)Q \xrightarrow{\gamma} (\nu y)Q'$ and $(\nu y)Q' \equiv_\alpha (\nu y)P'$.

(*AlpRes*, *Opn*) similar.

(*AlpRes1*, *Res*) :

$$\mathbf{AlpRes1} \frac{P \equiv_\alpha Q}{(\nu y) \equiv_\alpha (\nu z)Q\{z/y\}} \quad \mathbf{Res} \frac{P \xrightarrow{\gamma} P' \quad y \notin n(\gamma)}{(\nu y)P \xrightarrow{\gamma} (\nu y)P'}$$

$P \equiv_\alpha Q$ and $P \xrightarrow{\gamma} P'$ imply for inductive hypothesis that $Q \xrightarrow{\gamma} Q'$ and $Q' \equiv_\alpha P'$. So:

$$\mathbf{Alp} \frac{\mathbf{AlpRes2} \frac{Q \equiv_\alpha Q}{(\nu z)Q\{z/y\} \equiv_\alpha (\nu y)Q} \quad \mathbf{Res} \frac{Q \xrightarrow{\gamma} Q' \quad y \notin n(\gamma)}{(\nu y)Q \xrightarrow{\gamma} (\nu y)Q'}}{(\nu z)Q\{z/y\} \xrightarrow{\gamma} (\nu y)Q'}$$

and $Q' \equiv_\alpha P'$ imply for rule *Res* that $(\nu y)Q' \equiv_\alpha (\nu y)P'$.

(*AlpRes1*, *Opn*) similar.

(*AlpRes2*, *Res*) similar.

(*AlpRes2*, *Opn*) similar.

□

Lemma 3.3.5. \sim_E is preserved by all operators except input prefixing.

Proof. The proof goes by cases on operators:

Output prefixing

The relation $\{(\bar{x}y.P, \bar{x}y.Q) : P \sim_E Q\} \cup \sim_E$ is a strong early bisimulation. We can apply the following rules to $\bar{x}y$:

$$\text{Out } \bar{x}y.P \xrightarrow{\bar{x}y} P \sim_E Q \xleftarrow{\bar{x}y} \bar{x}y.Q$$

Alp :

$$\mathbf{Alp} \frac{\frac{P \equiv_{\alpha} R}{\overline{xy}.P \equiv_{\alpha} \overline{xy}.R} \quad \overline{xy}.R \xrightarrow{\overline{xy}} R}{\overline{xy}.P \xrightarrow{\overline{xy}} R}$$

$\overline{xy}.P \xrightarrow{\overline{xy}} R$, $\overline{xy}.Q \xrightarrow{\overline{xy}} Q$ and $R \equiv_{\alpha} P \sim_E Q$ imply $\overline{xy}.P$ and $\overline{xy}.Q$ are early bisimilar up to α equivalence. For proposition 3.3.2: $\overline{xy}.P$ and $\overline{xy}.Q$ are early bisimilar.

Strong output prefixing

The relation $\{(\overline{xy}.P, \overline{xy}.Q) : P \sim_E Q\} \cup \sim_E$ is a strong early bisimulation: there are three cases to consider:

- If there exists a transition $P \xrightarrow{\gamma} P'$ where γ is a non empty sequence of outputs then we can apply the rule *SOut*:

$$\frac{P \xrightarrow{\gamma} P'}{\overline{xy}.P \xrightarrow{\overline{xy} \cdot \gamma} P'}$$

$P \xrightarrow{\gamma} P'$ and $P \sim_E Q$ imply $Q \xrightarrow{\gamma} Q'$ and $P' \sim_E Q'$. For rule *SOut*: $\overline{xy}.Q \xrightarrow{\overline{xy} \cdot \gamma} Q'$ so the conclusion holds.

- There exists a process R α equivalent to P such that $R \xrightarrow{\gamma} R'$ where γ is a non empty sequence of outputs. We can apply the following rules:

$$\mathbf{Alp} \frac{\frac{P \equiv_{\alpha} R}{\overline{xy}.P \equiv_{\alpha} \overline{xy}.R} \quad \mathbf{SOut} \frac{R \xrightarrow{\gamma} R'}{\overline{xy}.R \xrightarrow{\overline{xy} \cdot \gamma} R'}}{\overline{xy}.P \xrightarrow{\overline{xy} \cdot \gamma} R'}$$

For rule *Alp*: $P \xrightarrow{\gamma} R'$ and so we are back to the previous case.

- Otherwise there is no transition starting from $\overline{xy}.P$ or from $\overline{xy}.Q$ so these processes are strongly bisimilar.

Tau prefixing

The relation $\{(\tau.P, \tau.Q) : P \sim_E Q\} \cup \sim_E$ is a strong early bisimulation. We have to consider in turn each rule that can be applied to $\tau.P$:

$$\mathbf{Tau} \quad \tau.P \xrightarrow{\tau} P \sim_E Q \xleftarrow{\tau} \tau.Q$$

Alp :

$$\mathbf{Alp} \frac{\frac{P \equiv_{\alpha} R}{\tau.P \equiv_{\alpha} \tau.R} \quad \tau.R \xrightarrow{\tau} R}{\tau.P \xrightarrow{\tau} R}$$

$\tau.P \xrightarrow{\tau} R \equiv_{\alpha} P \sim_E Q$ and $\tau.Q \xrightarrow{\tau} Q$ imply $\tau.P$ and $\tau.Q$ are early bisimilar up to α equivalence. For proposition 3.3.2: $\tau.P$ and $\tau.Q$ are early bisimilar.

Sum

The relation $\{(P + R, Q + R) : P \sim_E Q\} \cup \sim_E$ is a strong early bisimulation. The rules that can be applied to $P + Q$ are:

$$\mathbf{Sum} \quad P + R \xrightarrow{\gamma} P' \sim_E Q' \xleftarrow{\gamma} Q + R$$

Alp :

$$\mathbf{Alp} \frac{\frac{P \equiv_{\alpha} P_1 \quad R \equiv_{\alpha} R_1}{P + R \equiv_{\alpha} P_1 + R_1} \quad \mathbf{Sum} \frac{P_1 \xrightarrow{\gamma} P'_1}{P_1 + R_1 \xrightarrow{\gamma} P'_1}}{P + R \xrightarrow{\gamma} P'_1}$$

$P \equiv_{\alpha} P_1$ and $P_1 \xrightarrow{\gamma} P'_1$ imply for rule *Alp*: $P \xrightarrow{\gamma} P'_1$ which in turn imply $Q \xrightarrow{\gamma} Q'_1$ and $P'_1 \sim_E Q'_1$ since $P \sim_E Q$. Now an application of the rule *Sum* yields $Q + R \xrightarrow{\gamma} Q'_1$.

Restriction

We prove that the relation $Res(\sim_E) = \{((\nu x)P, (\nu x)Q) : P \sim_E Q\} \cup \sim_E$ is a strong early bisimulation up to α equivalence. This imply that $Res(\sim_E)$ is a strong early bisimulation. For lemma 3.2.7 and for reflexivity of α equivalence we can assume that every transition that starts from $(\nu x)P$ is in the form $(\nu x)P \equiv_{\alpha} R$ and $R \xrightarrow{\gamma} R'$. Where $\xrightarrow{\gamma}$ is the semantic in table 3.3 and 3.4 but without rule *Alp*. So we can proceed by induction on both the derivation of $(\nu x)P \equiv_{\alpha} R$ and of $R \xrightarrow{\gamma} R'$. There are some cases depending on the last pair of rule used in the derivation of $(\nu x)P \equiv_{\alpha} R$ and of $R \xrightarrow{\gamma} R'$:

(*AlpRes*, *Res*) :

$$\mathbf{Alp} \frac{\mathbf{AlpRes} \frac{P \equiv_{\alpha} R}{(\nu x)P \equiv_{\alpha} (\nu x)R} \quad \mathbf{Res} \frac{R \xrightarrow{\gamma} R' \quad x \notin n(\gamma)}{(\nu x)R \xrightarrow{\gamma} (\nu x)R'}}{(\nu x)P \xrightarrow{\gamma} (\nu x)R'}$$

$P \equiv_{\alpha} R \xrightarrow{\gamma} R'$ imply $P \xrightarrow{\gamma} R'$ which together with $P \sim_E Q$ imply $Q \xrightarrow{\gamma} Q'$ and $Q' \sim_E R'$. For rule *Res*: $(\nu x)Q \xrightarrow{\gamma} (\nu x)Q'$. Putting it all together:

$$(\nu x)P \xrightarrow{\gamma} (\nu x)P' \equiv_{\alpha} (\nu x)R' Res(\sim_E)(\nu x)Q' \xleftarrow{\gamma} (\nu x)Q$$

(*AlpRes*, *Res1*) : let $(\nu x)P \xrightarrow{\gamma} P'$, having in mind lemma 3.2.7

$$\mathbf{Alp} \frac{\mathbf{AlpRes1} \frac{P \equiv_{\alpha} R \quad y \notin n(R)}{(\nu x)P \equiv_{\alpha} (\nu y)R\{y/x\}} \quad \mathbf{Res} \frac{R\{y/x\} \xrightarrow{\gamma} R' \quad y \notin n(\gamma)}{(\nu y)R\{y/x\} \xrightarrow{\gamma} (\nu y)R'}}{(\nu x)P \xrightarrow{\gamma} (\nu y)R'}$$

$$\begin{array}{ll} R\{y/x\} \xrightarrow{\gamma} R' & \text{lemma 3.2.8} \\ \text{imply } R \xrightarrow{\gamma\{x/y\}} S \text{ and } S \equiv_{\alpha} R'\{x/y\} & P \equiv_{\alpha} R \text{ and rule } Alp \\ \text{imply } P \xrightarrow{\gamma\{x/y\}} S. & P \sim_E Q \\ \text{imply } Q \xrightarrow{\gamma\{x/y\}} Q' \text{ and } Q' \sim_E S. & \end{array}$$

$$\begin{array}{ll} Q \xrightarrow{\gamma\{x/y\}} Q' & \text{lemma 3.2.8} \\ \text{imply } Q\{y/x\} \xrightarrow{\gamma} Q'\{y/x\}. & \text{rule } Res \\ \text{imply } (\nu y)Q\{y/x\} \xrightarrow{\gamma} (\nu y)Q'\{y/x\} & \text{rule } Alp \text{ and } (\nu x)Q \equiv_{\alpha} (\nu y)Q\{y/x\} \\ \text{imply } (\nu x)Q \xrightarrow{\gamma} (\nu y)Q'\{y/x\} & \end{array}$$

$$\begin{array}{l} Q' \sim_E R'\{x/y\} \\ \text{imply } (\nu x)Q' Res(\sim_E)(\nu x)R'\{x/y\} \end{array}$$

Putting it all together:

$$(\nu x)P \xrightarrow{\gamma} (\nu y)R' \equiv_{\alpha} (\nu x)R' \{x/y\} \equiv_{\alpha} (\nu x)S \text{ Res}(\sim_E) (\nu x)Q' \equiv_{\alpha} (\nu y)Q' \{y/x\} \xleftarrow{\gamma} (\nu x)Q$$

(*AlpRes*, *Res2*) similar.

(*AlpRes*, *Opn*) similar.

(*AlpRes1*, *Opn*) similar.

(*AlpRes2*, *Opn*) similar.

Parallel composition

We prove that the relation $Par(\sim_E) = \{(P|R, Q|R) : P \sim_E Q\} \cup \sim_E$ is a strong early bisimulation up to α equivalence. This imply that $Par(\sim_E)$ is a strong early bisimulation. For lemma 3.2.7 and for reflexivity of α equivalence we can assume that every transition that starts from $P|R$ is in the form $P|R \equiv_{\alpha} S$ and $S \xrightarrow{\gamma} S'$. Where \rightarrow is the semantic in table 3.3 and 3.4 but without rule *Alp*. We can proceed by induction on both the derivation of $P|R \equiv_{\alpha} Q|R$ and of $S \xrightarrow{\gamma} S'$. The last rule used in the derivation of $P|R \equiv_{\alpha} Q|R$ can only be *AlpPar*, but there are some different cases on the last rule used to prove that $S \xrightarrow{\gamma} S'$:

ECom :

$$\text{Alp} \frac{\frac{P \equiv_{\alpha} P_2 \quad R \equiv_{\alpha} R_2}{P|R \equiv_{\alpha} P_2|R_2} \quad \text{ECom} \frac{P_2 \xrightarrow{\bar{x}y} P'_2 \quad R_2 \xrightarrow{xy} R'_2}{P_2|R_2 \xrightarrow{\tau} P'_2|R'_2}}{P|R \xrightarrow{\tau} P'_2|R'_2}$$

$P \equiv_{\alpha} P_2$ and $P_2 \xrightarrow{\bar{x}y} P'_2$ for rule *Alp* imply $P \xrightarrow{\bar{x}y} P'_2$. $P \sim_E Q$ imply that there exists a process Q' such that $Q \xrightarrow{\bar{x}y} Q'$ and $P'_2 \sim_E Q'$. $R \equiv_{\alpha} R_2$ and $R_2 \xrightarrow{xy} R'_2$ imply for rule *Alp* that $R \xrightarrow{xy} R'_2$. So for rule *ECom*: $Q|R \xrightarrow{\tau} Q'|R'_2$ and $P'_2|R'_2 \sim_E Q'|R'_2$

Cls :

$$\frac{P \xrightarrow{\bar{x}(y) \cdot (\nu y)} P' \quad R \xrightarrow{xy} R'}{P|R \xrightarrow{\tau} (\nu y)(P'|R')}$$

$P \xrightarrow{\bar{x}(y) \cdot (\nu y)} P'$ and $P \sim_E Q$ imply that there exists a process Q' such that $Q \xrightarrow{\bar{x}(y) \cdot (\nu y)} Q'$ and $P' \sim_E Q'$. So for rule *Cls*: $Q|R \xrightarrow{\tau} (\nu y)(Q'|R')$ and $(\nu y)(P'|R') \sim_E (\nu y)(Q'|R')$

ClsSeq1, *ClsSeq2*, *ParL*, *ParR* similar.

□

Example \sim_E is not in general preserved by input prefixing because:

$$a(x).0|\bar{b}y.0 \sim_E a(x).\bar{b}y.0 + \bar{b}y.a(x).0$$

but

$$c(a).(a(x).0|\bar{b}y.0) \not\sim_E c(a).(a(x).\bar{b}y.0 + \bar{b}y.a(x).0)$$

because

$$\begin{aligned} c(a).(a(x).0|\bar{b}y.0) &\xrightarrow{cb} b(x).0|\bar{b}y.0 \xrightarrow{\tau} 0|0 \\ c(a).(a(x).\bar{b}y.0 + \bar{b}y.a(x).0) &\xrightarrow{cb} b(x).\bar{b}y.0 + \bar{b}y.b(x).0 \not\xrightarrow{\tau} \end{aligned}$$

3.3.3 Strong D equivalence

Definition 3.3.3. A *distinction* is a finite symmetric and irreflexive binary relation on names. A substitution σ *respects* a pair (a, b) if

$$a\sigma \neq b\sigma$$

A substitution σ *respects* a distinction D if it respects every pair in the distinction:

$$\forall a, b. aDb \Rightarrow a\sigma \neq b\sigma$$

We write $D \cdot \sigma$ for the composition of the two relation.

Example The empty relation \emptyset is a distinction. Every substitution respects the empty distinction.

Definition 3.3.4. Let D be a distinction and A be a set of names

$$D - A \stackrel{def}{=} D - (A \times \mathbb{N} \cup \mathbb{N} \times A)$$

Definition 3.3.5. Let D be a distinction and σ be a substitution. The application of σ to D is defined as:

$$D\sigma \stackrel{def}{=} \{(a\sigma, b\sigma) : (a, b) \in D\}$$

Proposition 3.3.6. Let D, D' be distinctions and σ be a substitution. Then

$$D' \subseteq D \text{ and } \sigma \text{ respects } D \text{ imply } \sigma \text{ respects } D'$$

Lemma 3.3.7. Let σ be a substitution, D be a distinction and $c \notin n(D)$. If σ respects D then $\sigma\{c/x\}$ respects $D - \{x\}$.

Proof. : σ respects D and $D - \{x\} \subseteq D$ imply σ respects $D - \{x\}$. $(d_1, d_2) \in (D - \{x\})$ imply $d_1\sigma\{c/x\} = d_1\sigma$ and $d_2\sigma = d_2\sigma\{c/x\}$. σ respects $D - \{x\}$ and $(d_1, d_2) \in (D - \{x\})$ for definition 3.3.3 imply $d_1\sigma \neq d_2\sigma$. Putting it all together $\sigma\{c/x\}$ respects (d_1, d_2) . \square

According to [2] the following holds:

Lemma 3.3.8. Let σ be a substitution, D be a distinction and $y\sigma = y$. If σ respects $D - \{x\}$ then $\{y/x\}\sigma$ respects D .

Definition 3.3.6. P and Q are *strongly D equivalent*, written $P \sim^D Q$, if for all substitution σ respecting D : $P\sigma \sim_E Q\sigma$. In this definition we assume that the application of σ to P and Q does not change any bound name.

Lemma 3.3.9. For any distinction $D \sim^D$ is an equivalence relation

Proof. \sim^D is an equivalence relation because \sim_E is an equivalence relation.

Reflexivity Since \sim_E is reflexive, for all substitution σ respecting D : $P\sigma \sim_E Q\sigma$ so $P \sim^D P$

Symmetry Let $P \sim^D Q$ then for all substitution σ respecting D : $P\sigma \sim_E Q\sigma$. Since \sim_E is symmetric $Q\sigma \sim_E P\sigma$ so $Q \sim^D P$

Transitivity Let $P \sim^D Q$ and $Q \sim^D R$ then for all substitution σ respecting D : $P\sigma \sim_E Q\sigma$ and $Q\sigma \sim_E R\sigma$. Since \sim_E is transitive $P\sigma \sim_E R\sigma$ so $P \sim^D R$.

\square

NON HA SENSO QUESTO LEMMA IN QUESTO CONTESTO

Lemma 3.3.10. If $P \sim^D Q$ and for all $v \in fn(P, Q)$ 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.

\square

Lemma 3.3.11. If $P \sim^D Q$ then

- $\tau.P \sim^D \tau.Q$
- $\bar{x}y.P \sim^D \bar{x}y.Q$
- $\underline{\bar{x}y}.P \sim^D \underline{\bar{x}y}.Q$
- $P + R \sim^D Q + R$
- $P|R \sim^D Q|R$
- $(\nu x)P \sim^D (\nu x)Q$

Proof. \sim^D is preserved by every operator. Let $P \sim^D Q$ and let σ be a substitution respecting D so $P\sigma \sim_E Q\sigma$:

Output prefixing

$$\begin{aligned}
P \sim^D Q & \text{definition 3.3.6} \\
\Rightarrow \forall \sigma \text{ respecting } D. P\sigma \sim_E Q\sigma & \text{lemma 3.3.5} \\
\Rightarrow (\bar{x}y)\sigma.(P\sigma) \sim_E (\bar{x}y)\sigma.(Q\sigma) & \text{definition of substitution} \\
\Rightarrow (\bar{x}y.P)\sigma \sim_E (\bar{x}y.Q)\sigma & \text{definition 3.3.6} \\
\Rightarrow \bar{x}y.P \sim^D \bar{x}y.Q
\end{aligned}$$

Strong output prefixing similar.

Tau prefixing

$$\begin{aligned}
P \sim^D Q & \text{definition 3.3.6} \\
\Rightarrow \forall \sigma \text{ respecting } D. P\sigma \sim_E Q\sigma & \text{lemma 3.3.5} \\
\Rightarrow \tau.(P\sigma) \sim_E \tau.(Q\sigma) & \text{definition of substitution} \\
\Rightarrow (\tau.P)\sigma \sim_E (\tau.Q)\sigma & \text{definition 3.3.6} \\
\Rightarrow \tau.P \sim^D \tau.Q
\end{aligned}$$

Sum

$$\begin{aligned}
P \sim^D Q & \text{definition 3.3.6} \\
\Rightarrow \forall \sigma \text{ respecting } D. P\sigma \sim_E Q\sigma & \text{lemma 3.3.5} \\
\Rightarrow (P\sigma) + (R\sigma) \sim_E (Q\sigma) + (R\sigma) & \text{definition of substitution} \\
\Rightarrow (P + R)\sigma \sim_E (Q + R)\sigma & \text{definition 3.3.6} \\
\Rightarrow P + R \sim^D Q + R
\end{aligned}$$

Parallel composition

$$\begin{aligned}
P \sim^D Q & \text{definition 3.3.6} \\
\Rightarrow \forall \sigma \text{ respecting } D. P\sigma \sim_E Q\sigma & \text{lemma 3.3.5} \\
\Rightarrow (P\sigma)|(R\sigma) \sim_E (Q\sigma)|(R\sigma) & \text{definition of substitution} \\
\Rightarrow (P|R)\sigma \sim_E (Q|R)\sigma & \text{definition 3.3.6} \\
\Rightarrow P|R \sim^D Q|R
\end{aligned}$$

Restriction Note that in definition 3.3.6 we assume that the substitution does not change any bound name so $((\nu x)P)\sigma = (\nu x)(P\sigma)$:

$$\begin{aligned}
P \sim^D Q & \text{definition 3.3.6} \\
\Rightarrow \forall \sigma \text{ respecting } D. P\sigma \sim_E Q\sigma & \text{lemma 3.3.5} \\
\Rightarrow (\nu x)(P\sigma) \sim_E (\nu x)(Q\sigma) & \text{definition of substitution} \\
\Rightarrow ((\nu x)P)\sigma \sim_E ((\nu x)Q)\sigma & \text{definition 3.3.6} \\
\Rightarrow (\nu x)P \sim^D (\nu x)Q
\end{aligned}$$

□

Theorem 3.3.12. \sim^\emptyset is a congruence.

Proof. Lemma 3.3.9 and put $D = \emptyset$ in lemma 3.3.11 and in lemma 3.3.10

□

3.3.4 Open bisimulation

Let \rightarrow_L be the semantic defined in table 3.1 the following is an attempt to extend the definition of strong open bisimulation found in [4]:

Definition 3.3.7. An *strong open simulation* is $\{S_D\}_{D \in \mathbb{D}}$ a family of binary relations on processes such that for each process P, Q , for each distinction $D \in \mathbb{D}$, for each name substitution σ which respects D if PS_DQ , $P\sigma \xrightarrow{\gamma}_L P'$ and $bn(\gamma) \cap (fn(P\sigma) \cup fn(Q\sigma)) = \emptyset$ then:

- if γ is the sequence of bound outputs $\widetilde{a(x)}$ then there exists Q' such that $Q\sigma \xrightarrow{\widetilde{a(x)}}_L Q'$ and $P'S_{D'}Q'$ where $D' = D\sigma \cup \{\bar{x}\} \times (fn(P\sigma) \cup fn(Q\sigma)) \cup (fn(P\sigma) \cup fn(Q\sigma)) \times \{\bar{x}\}$
- if α is not a bound output then there exists Q' such that $Q\sigma \xrightarrow{\alpha}_L Q'$ and $P'S_{D\sigma}Q'$

P and Q are *open D bisimilar*, written $P \sim_O^D Q$ if there exists a member S_D of an open bisimulation such that PS_DQ ; they are *open bisimilar* if they are open \emptyset bisimilar, written $P \sim_O Q$.

Theorem 3.3.13. $P \sim_O^D Q$ is a congruence.

Proof.

□

Chapter 4

Multi π calculus with strong input

4.1 Syntax

As we did with π calculus, we suppose that we have a countable set of names \mathbf{N} , ranged over by lower case letters a, b, \dots, z . These names are used for communication channels and values. Furthermore we have a set of identifiers, ranged over by A . We represent the agents or processes by upper case letters P, Q, \dots . A multi π process, in addition to the same actions of a π process, can perform also a strong prefix input:

$$\pi ::= \bar{x}y \mid x(z) \mid \underline{x(y)} \mid \tau$$

The process are defined, just as original π calculus, by the following grammar:

$$P, Q ::= 0 \mid \pi.P \mid P|Q \mid P + Q \mid (\nu x)P \mid A(y_1, \dots, y_n)$$

and they have the same intuitive meaning as for the π calculus. The strong prefix input allows a process to make an atomic sequence of actions, so that more than one process can synchronize on this sequence. For the moment we allow the strong prefix to be on input names only. Also one can use the strong prefix only as an action prefixing for processes that can make at least a further action.

Multi π calculus is a conservative extension of the π calculus in the sense that: any π calculus process p is also a multi π calculus process and the semantic of p according to the SOS rules of π calculus is the same as the semantic of p according to the SOS rules of multi π calculus. We have to extend the following definition to deal with the strong prefix:

$$B(\underline{x(y)}.Q, I) = \{y, \bar{y}\} \cup B(Q, I) \quad F(\underline{x(y)}.Q, I) = \{x, \bar{x}\} \cup (F(Q, I) - \{y, \bar{y}\})$$

The scope of the object of a strong input is the process that follows the strong input. For example the scope of a name x in a process $\underline{y(x)}.x(b).P$ is $x(b).P$.

In this setting two process cannot synchronize on a sequence of actions with length greater than one so we cannot have transactional synchronization but we can have multi-party synchronization.

4.2 Operational semantic

4.2.1 Early operational semantic with structural congruence

The semantic of a multi π process is labeled transition system such that

- the nodes are multi π calculus process. The set of node is \mathbf{P}_m
- the actions are multi π calculus actions. The set of actions is \mathbf{A}_m , we use $\alpha, \alpha_1, \alpha_2, \dots$ to range over the set of actions, we use $\sigma, \sigma_1, \sigma_2, \dots$ to range over the set $\mathbf{A}_m^+ \cup \{\tau\}$.
- the transition relations is $\rightarrow \subseteq \mathbf{P}_m \times (\mathbf{A}_m^+ \cup \{\tau\}) \times \mathbf{P}_m$

| | | |
|--|--|---|
| Out $\frac{}{\bar{x}y.P \xrightarrow{\bar{x}y} P}$ | EInp $\frac{}{x(y).P \xrightarrow{xz} P\{z/y\}}$ | Tau $\frac{}{\tau.P \xrightarrow{\tau} P}$ |
| SInpTau $\frac{P\{y/z\} \xrightarrow{\tau} P'}{x(z).P \xrightarrow{xy} P'}$ | SInp $\frac{P\{y/z\} \xrightarrow{ab} P'}{x(z).P \xrightarrow{xy.ab} P'}$ | SInpSeq $\frac{P\{y/z\} \xrightarrow{\sigma} P' \quad \sigma > 1}{x(z).P \xrightarrow{xy.\sigma} P'}$ |
| Sum $\frac{P \xrightarrow{\sigma} P'}{P + Q \xrightarrow{\sigma} P'}$ | Cong $\frac{P \equiv P' \quad P' \xrightarrow{\alpha} Q}{P \xrightarrow{\alpha} Q}$ | Res $\frac{P \xrightarrow{\sigma} P' \quad z \notin n(\sigma)}{(\nu z)P \xrightarrow{\sigma} (\nu z)P'}$ |
| Par $\frac{P \xrightarrow{\sigma} P'}{P Q \xrightarrow{\sigma} P' Q}$ | Opn $\frac{P \xrightarrow{\bar{x}z} P' \quad z \neq x}{(\nu z)P \xrightarrow{\bar{x}(z)} P'}$ | |
| ECom $\frac{P \xrightarrow{xy} P' \quad Q \xrightarrow{\bar{x}y} Q'}{P Q \xrightarrow{\tau} P' Q'}$ | EComSeq $\frac{P \xrightarrow{xy.\sigma} P' \quad Q \xrightarrow{\bar{x}y} Q'}{P Q \xrightarrow{\sigma} P' Q'}$ | |

Table 4.1: Multi π early semantic with structural congruence

In this case, a label can be a sequence of prefixes, whether in the original π calculus a label can be only a prefix. We use the symbol \cdot to denote the concatenation operator.

Definition 4.2.1. The *early transition relation with structural congruence* is the smallest relation induced by the rules in table 4.1 where *inpSeq* is a non empty sequence of input actions and σ is a sequence of any action.

Example Multi-party synchronization We show an example of a derivation of three processes that synchronize.

$$\begin{array}{c}
\text{EInp} \frac{}{(x(b).P)\{y/a\} \xrightarrow{xz} P\{y/a\}\{z/b\}} \\
\text{SInp} \frac{}{x(a).(x(b).P) \xrightarrow{xy.xz} P\{y/a\}\{z/b\}} \\
\text{EComSeq} \frac{}{x(a).x(b).P|\bar{x}y.Q \xrightarrow{xz} P\{y/a\}\{z/b\}|Q} \\
\text{Out} \frac{}{\bar{x}y.Q \xrightarrow{\bar{x}y} Q} \\
\text{EComSng} \frac{x(a).x(b).P|\bar{x}y.Q \xrightarrow{xz} P\{y/a\}\{z/b\}|Q \quad \text{Out} \frac{}{\bar{x}z.R \xrightarrow{\bar{x}z} R}}{x(a).x(b).P|\bar{x}y.Q|\bar{x}z.R \xrightarrow{\tau} (P\{y/a\}\{z/b\}|Q)|R}
\end{array}$$

Lemma 4.2.1. If $P \xrightarrow{\sigma} Q$ then only one of the following cases hold:

- $|\sigma| = 1$
- $|\sigma| > 1$, the actions in σ are input.

4.2.2 Late operational semantic with structural congruence

Definition 4.2.2. The *late transition relation with structural congruence* is the smallest relation induced by the rules in table 4.2.

Example Multi-party synchronization We show an example of a derivation of three processes that synchronize with the late semantic. The three processes are $x(a).x(b).P$, $\bar{x}y.Q$ and $\bar{x}z.R$. We assume modulo α conversion that:

$$a \notin fn(x(b)) \cup fn(x(a).x(b).P)$$

| | | |
|--|--|--|
| Out $\frac{}{\bar{x}y.P \xrightarrow{\bar{x}y} P}$ | LInp $\frac{}{x(y).P \xrightarrow{x(y)} P}$ | Tau $\frac{}{\tau.P \xrightarrow{\tau} P}$ |
| SInp $\frac{P \xrightarrow{\gamma} P'}{x(z).P \xrightarrow{x(z) \cdot \gamma} P'}$ | γ is a non empty sequence of inputs | |
| LComSeq $\frac{P \xrightarrow{x(y) \cdot \sigma} P' \quad Q \xrightarrow{\bar{x}z} Q' \quad bn(\sigma) \cap fn(Q) = \emptyset}{P Q \xrightarrow{\sigma\{z/y\}} P'\{z/y\} Q'}$ | LCom $\frac{P \xrightarrow{x(y)} P' \quad Q \xrightarrow{\bar{x}z} Q'}{P Q \xrightarrow{\tau} P'\{z/y\} Q'}$ | |
| Sum $\frac{P \xrightarrow{\sigma} P'}{P + Q \xrightarrow{\sigma} P'}$ | Cong $\frac{P \equiv P' \quad P' \xrightarrow{\sigma} Q}{P \xrightarrow{\sigma} Q}$ | Opn $\frac{P \xrightarrow{\bar{x}z} P' \quad z \neq x}{(\nu z)P \xrightarrow{\bar{x}(z)} P'}$ |
| Res $\frac{P \xrightarrow{\sigma} P' \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\sigma} (\nu z)P'}$ | Par $\frac{P \xrightarrow{\sigma} P' \quad bn(\sigma) \cap fn(Q) = \emptyset}{P Q \xrightarrow{\sigma} P' Q}$ | |

Table 4.2: Multi π late semantic with structural congruence

and

$$\begin{array}{c}
c \notin fn(\bar{x}y.Q) \\
\textbf{LInp} \frac{}{x(b).P \xrightarrow{x(b)} P} \\
\textbf{SInp} \frac{}{x(a).x(b).P \xrightarrow{x(a) \cdot x(b)} P} \quad \textbf{Out} \frac{}{\bar{x}y.Q \xrightarrow{\bar{x}y} Q} \\
\textbf{LComSeq} \frac{}{x(a).x(b).P|\bar{x}y.Q \xrightarrow{x(b)} P\{y/a\}|Q} \\
\textbf{LCom} \frac{x(a).x(b).P|\bar{x}y.Q \xrightarrow{x(b)} P\{y/a\}|Q \quad \textbf{Out} \frac{}{\bar{x}z.R \xrightarrow{\bar{x}z} R}}{(x(a).x(b).P|\bar{x}y.Q)|\bar{x}z.R \xrightarrow{\tau} (P\{y/a\}|Q)\{z/b\}|R = (P\{y/a\}\{z/b\}|Q)|R}
\end{array}$$

Definition 4.2.3. \rightarrow is the smallest relation induced by the all the rules in table 4.2 except *Cong*.

Proposition 4.2.2. If $P \xrightarrow{\sigma} Q$ then there exists a process R such that: $R \xrightarrow{\sigma} Q$ and $P \equiv R$

Proof. We show that we can move the rule *Cong* down the inference tree of $P \xrightarrow{\sigma} Q$. So a derivation of $P \xrightarrow{\sigma} Q$ can translate into a derivation of $P \xrightarrow{\sigma} Q$ which uses the rule *Cong* only as its last rule.

SInp

$$\begin{array}{c}
\textbf{Cong} \frac{P \equiv R \quad R \xrightarrow{\gamma} Q}{P \xrightarrow{\gamma} Q} \\
\textbf{SInp} \frac{}{x(z).P \xrightarrow{x(z) \cdot \gamma} Q}
\end{array}$$

become

$$\textbf{Cong} \frac{\frac{P \equiv R}{x(z).P \equiv x(z).R} \quad \textbf{SInp} \frac{R \xrightarrow{\gamma} Q}{x(z).R \xrightarrow{x(z) \cdot \gamma} Q}}{x(z).P \xrightarrow{x(z) \cdot \gamma} Q}$$

Sum

$$\text{Sum} \frac{\text{Cong} \frac{P \equiv R \quad R \xrightarrow{\gamma} Q}{P \xrightarrow{\gamma} Q}}{P + S \xrightarrow{\gamma} Q}$$

become

$$\text{Cong} \frac{\frac{P \equiv R}{P + S \equiv R + S} \quad \text{Sum} \frac{R \xrightarrow{\gamma} Q}{R + S \xrightarrow{\gamma} Q}}{P + S \xrightarrow{\gamma} Q}$$

Cong

$$\text{Cong} \frac{P \equiv R \quad \text{Cong} \frac{R \equiv S \quad S \xrightarrow{\gamma} Q}{R \xrightarrow{\gamma} Q}}{P \xrightarrow{\gamma} Q}$$

become

$$\text{Cong} \frac{\frac{P \equiv R \quad R \equiv S}{P \equiv S} \quad S \xrightarrow{\gamma} Q}{P \xrightarrow{\gamma} Q}$$

Par

$$\text{Par} \frac{\text{Cong} \frac{P \equiv R \quad R \xrightarrow{\gamma} Q}{P \xrightarrow{\gamma} Q} \quad bn(\gamma) \cap fn(S) = \emptyset}{P|S \xrightarrow{\gamma} Q}$$

become

$$\text{Cong} \frac{\frac{P \equiv R}{P|S \equiv R|S} \quad \text{Par} \frac{R \xrightarrow{\gamma} Q \quad bn(\gamma) \cap fn(S) = \emptyset}{R|S \xrightarrow{\gamma} Q}}{P|S \xrightarrow{\gamma} Q}$$

LComSeq

$$\text{LComSeq} \frac{\text{Cong} \frac{P_1 \equiv R_1 \quad R_1 \xrightarrow{x(y) \cdot \sigma} Q_1}{P_1 \xrightarrow{x(y) \cdot \sigma} Q_1} \quad \text{Cong} \frac{P_2 \equiv R_2 \quad R_2 \xrightarrow{\bar{x}z} Q_2}{P_2 \xrightarrow{\bar{x}z} Q_2}}{P_1|P_2 \xrightarrow{\gamma\{z/y\}} Q_1\{z/y\}|Q_2}$$

become

$$\text{Cong} \frac{\frac{P_1 \equiv R_1 \quad P_2 \equiv R_2}{P_1|P_2 \equiv R_1|R_2} \quad \text{LComSeq} \frac{R_1 \xrightarrow{x(y) \cdot \sigma} Q_1 \quad R_2 \xrightarrow{\bar{x}z} Q_2}{R_1|R_2 \xrightarrow{\sigma\{z/y\}} Q_1\{z/y\}|Q_2}}{P_1|P_2 \xrightarrow{\gamma\{z/y\}} Q_1\{z/y\}|Q_2}$$

LCom

$$\mathbf{LCom} \frac{\mathbf{Cong} \frac{P_1 \equiv R_1 \quad R_1 \xrightarrow{x(y)} Q_1}{P_1 \xrightarrow{x(y)} Q_1} \quad \mathbf{Cong} \frac{P_2 \equiv R_2 \quad R_2 \xrightarrow{\bar{x}z} Q_2}{P_2 \xrightarrow{\bar{x}z} Q_2}}{P_1|P_2 \xrightarrow{\tau} Q_1\{z/y\}|Q_2}$$

become

$$\mathbf{Cong} \frac{\frac{P_1 \equiv R_1 \quad P_2 \equiv R_2}{P_1|P_2 \equiv R_1|R_2} \quad \mathbf{LCom} \frac{R_1 \xrightarrow{x(y)} Q_1 \quad R_2 \xrightarrow{\bar{x}z} Q_2}{R_1|R_2 \xrightarrow{\tau} Q_1\{z/y\}|Q_2}}{P_1|P_2 \xrightarrow{\tau} Q_1\{z/y\}|Q_2}$$

Res

$$\mathbf{Res} \frac{\mathbf{Cong} \frac{P \equiv R \quad R \xrightarrow{\gamma} Q}{P \xrightarrow{\gamma} Q} \quad z \notin n(\gamma)}{(\nu z)P \xrightarrow{\gamma} (\nu z)Q}$$

become

$$\mathbf{Cong} \frac{\frac{P \equiv R}{(\nu z)P \equiv (\nu z)R} \quad \mathbf{Res} \frac{R \xrightarrow{\gamma} Q \quad z \notin n(\gamma)}{(\nu z)R \xrightarrow{\gamma} (\nu z)Q}}{(\nu z)P \xrightarrow{\gamma} (\nu z)Q}$$

Opn

$$\mathbf{Opn} \frac{\mathbf{Cong} \frac{P \equiv R \quad R \xrightarrow{\bar{x}y} Q}{P \xrightarrow{\bar{x}y} Q} \quad y \neq x}{(\nu y)P \xrightarrow{\bar{x}(y)} Q}$$

become

$$\mathbf{Cong} \frac{\frac{P \equiv R}{(\nu y)P \equiv (\nu y)R} \quad \mathbf{Opn} \frac{R \xrightarrow{\bar{x}y} Q \quad y \neq x}{(\nu y)R \xrightarrow{\bar{x}(y)} Q}}{(\nu y)P \xrightarrow{\bar{x}(y)} Q}$$

□

In the following lemma we call \rightarrow_3 the semantic defined in table 4.2 and we call \rightarrow_2 the semantic defined in table 4.2 but without rule *Cong* and with rules: *Ide* and the commutativity counterpart for rules *Par*, *Sum*, *ECom*, *EComSeq*.

Lemma 4.2.3. If $P \xrightarrow{\gamma}_3 P'$ then there exist processes N, N' such that N is a normal form, $P \equiv N$, $P' \equiv N'$ and $N \xrightarrow{\gamma}_2 N'$ without using the rules *Cls*. Also we can assume that the rules *ScpExtPar1*, *ScpExtSum1* and the rules for commutativity are not used in the derivation of $P \equiv N$ because this rule can be emulated by *ScpExtPar2*, *ScpExtSum2* and commutativity in \rightarrow_2 .

Proof. The proof of this lemma is similar to those in section 2.6.1

□

In the following section the symbol \rightarrow will refer to the late semantic with structural congruence of multi π calculus with strong input which is illustrated in table 4.2. Also we consider a structural congruence without the rules $P|0 \equiv 0$ and $P + 0 \equiv 0$. For the purpose of clarity the rule of structural congruence are repeated in this section.

| | | |
|---|---|--|
| Out $\frac{}{\bar{x}y.P \xrightarrow{\bar{x}y} P}$ | EInp $\frac{}{x(y).P \xrightarrow{xz} P\{z/y\}}$ | Tau $\frac{}{\tau.P \xrightarrow{\tau} P}$ |
| StarInp $\frac{P \xrightarrow{xy} S'}{*P \xrightarrow{xy} S'}$ | SInpLow $\frac{}{\underline{x(z)}.P \xrightarrow{xy} *P\{y/z\}}$ | StarEps $\frac{P \xrightarrow{\epsilon} S'}{*P \xrightarrow{\epsilon} S'}$ |
| Com1 $\frac{P \xrightarrow{\bar{x}y} P' \quad Q \xrightarrow{xy} Q'}{P Q \xrightarrow{\tau} P' Q'}$ | | |
| Com2L $\frac{L_1 \xrightarrow{xy} L_2 \quad P \xrightarrow{\bar{x}y} Q}{L_1 P \xrightarrow{\epsilon} L_2 Q}$ | Com2R $\frac{P \xrightarrow{\bar{x}y} Q \quad L_1 \xrightarrow{xy} L_2}{P L_1 \xrightarrow{\epsilon} Q L_2}$ | |
| Com3L $\frac{P \xrightarrow{xy} L \quad Q \xrightarrow{\bar{x}y} Q'}{P Q \xrightarrow{\epsilon} L Q'}$ | Com3R $\frac{Q \xrightarrow{\bar{x}y} Q' \quad P \xrightarrow{xy} L}{Q P \xrightarrow{\epsilon} Q' L}$ | |
| Com4L $\frac{L \xrightarrow{xy} P \quad Q \xrightarrow{\bar{x}y} Q'}{L Q \xrightarrow{\tau} P Q'}$ | Com4R $\frac{Q \xrightarrow{\bar{x}y} Q' \quad L \xrightarrow{xy} P}{L Q \xrightarrow{\tau} P Q'}$ | |
| Res $\frac{S \xrightarrow{\gamma} S' \quad y \notin n(\gamma)}{(\nu y)S \xrightarrow{\gamma} (\nu y)S'}$ | Opn $\frac{P \xrightarrow{\bar{x}y} Q \quad y \neq x}{(\nu y)P \xrightarrow{\bar{x}(y)} Q}$ | Cong $\frac{P \equiv P' \quad P' \xrightarrow{\gamma} S}{P \xrightarrow{\gamma} S}$ |
| Par1L $\frac{S \xrightarrow{\gamma} S'}{S Q \xrightarrow{\gamma} S' Q}$ | Par1R $\frac{S \xrightarrow{\gamma} S'}{Q S \xrightarrow{\gamma} Q S'}$ | Sum $\frac{P \xrightarrow{\gamma} S}{P + Q \xrightarrow{\gamma} S}$ |

Table 4.3: Low multi π early semantic with structural congruence

4.2.3 Low level semantic

This section contains the definition of an alternative semantic for multi π . First we define a low level version of the multi π calculus (here with strong prefixing on input only), we call this language low multi π . The low multi π is the multi π enriched with a marked or intermediate process $*P$:

$$P, Q ::= 0 \mid \pi.P \mid P|Q \mid P + Q \mid (\nu x)P \mid A \mid *P$$

$$\pi ::= \bar{x}y \mid x(y) \mid \underline{x(y)} \mid \tau$$

Definition 4.2.4. The low level transition relation is the smallest relation induced by the rules in table 4.3 in which P stands for a process without mark, L stands for a process with mark and S can stand for both.

Lemma 4.2.4. For all unmarked processes P, Q and marked processes L_1, L_2 .

- if $P \xrightarrow{\alpha} L_1$ or $L_1 \xrightarrow{\alpha} L_2$ then α can only be an input or an ϵ
- if $L_1 \xrightarrow{\alpha} P$ then α is an input or a τ
- if $P \xrightarrow{\alpha} Q$ then α is not an ϵ

Definition 4.2.5. Let P, Q be unmarked processes and L_1, \dots, L_{k-1} marked processes. We define the derivation relation \rightarrow_s in the following way:

$$\text{Low} \frac{P \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} Q \quad k \geq 1}{P \xrightarrow{\gamma_1 \cdots \gamma_k}_s Q}$$

We need to be precise about the concatenation operator \cdot since we have introduced the new label ϵ . Let a be an action such that $a \neq \tau$ and $a \neq \epsilon$ then the following rules hold:

$$\begin{array}{lll} \epsilon \cdot a = a \cdot \epsilon = a & \epsilon \cdot \epsilon = \epsilon & \tau \cdot \epsilon = \epsilon \cdot \tau = \tau \\ \tau \cdot a = a \cdot \tau = a & \tau \cdot \tau = \tau & \end{array}$$

Example Multi-party synchronization We show an example of a derivation of three processes that synchronize.

$$\begin{array}{c} \text{SInpLow} \frac{}{x(a).x(b).P \xrightarrow{xy} *(x(b).P\{y/a\})} \quad \text{Out} \frac{}{\bar{x}y.Q \xrightarrow{\bar{x}y} Q} \\ \text{Com3L} \frac{}{x(a).x(b).P|\bar{x}y.Q \xrightarrow{\epsilon} *(x(b).P\{y/a\})|Q} \\ \text{Par1L} \frac{}{(x(a).x(b).P|\bar{x}y.Q)|\bar{x}z.R \xrightarrow{\epsilon} (*(x(b).P\{y/a\})|Q)|\bar{x}z.R} \\ \\ \text{EInp} \frac{}{x(b).P\{y/a\} \xrightarrow{xz} P\{y/a\}\{z/b\}} \\ \text{Star} \frac{}{*(x(b).P\{y/a\}) \xrightarrow{xz} P\{y/a\}\{z/b\}} \\ \text{Par1L} \frac{}{*(x(b).P\{y/a\})|Q \xrightarrow{xz} P\{y/a\}\{z/b\}|Q} \\ \\ \text{Com4L} \frac{*(x(b).P\{y/a\})|Q \xrightarrow{xz} P\{y/a\}\{z/b\}|Q \quad \text{Out} \frac{}{\bar{x}z.R \xrightarrow{\bar{x}z} R}}{(x(a).x(b).P|\bar{x}y.Q)|\bar{x}z.R \xrightarrow{\tau} (P\{y/a\}\{z/b\}|Q)|R} \end{array}$$

Proposition 4.2.5. Let \rightarrow be the relation defined in table 4.1. If $P \xrightarrow{\sigma} Q$ then there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q \quad \text{and} \quad \gamma_1 \cdots \gamma_{k+1} = \sigma$$

Proof. The proof is by induction on the depth of the derivation tree of $P \xrightarrow{\sigma} Q$ and by cases on the last rule used in the derivation:

EInp, Out, Tau These rules are also in table 4.3 so we can derive $P \xrightarrow{\sigma} Q$.

SInpSeq the last part of the derivation tree looks like this:

$$\text{SInpSeq} \frac{P_1\{y/z\} \xrightarrow{\sigma} Q \quad |\sigma| > 1}{x(z).P_1 \xrightarrow{xy \cdot \sigma} Q}$$

for inductive hypothesis there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1\{y/z\} \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q \quad \text{and} \quad \gamma_1 \cdots \gamma_{k+1} = \sigma$$

then a proof of the conclusion follows from:

$$\text{SInpLow} \frac{}{x(z).P_1 \xrightarrow{xy} *P_1\{y/z\}} \quad \text{Star} \frac{P_1\{y/z\} \xrightarrow{\gamma_1} L_1}{*P_1\{y/z\} \xrightarrow{\gamma_1} L_1}$$

where *Star* means *StarInp* or *StarEps*, note that γ_1 is an input or an *epsilon* because of 4.2.1.

SInp this case is similar to the previous.

SInpTau this case is similar to the previous observing that $xy \cdot \tau = xy$.

Sum the last part of the derivation tree looks like this:

$$\mathbf{Sum} \frac{P_1 \xrightarrow{\sigma} Q}{P_1 + P_2 \xrightarrow{\sigma} Q}$$

for the inductive hypothesis there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q \quad \text{and} \quad \gamma_1 \cdots \gamma_{k+1} = \sigma$$

A proof of the conclusion is:

$$\mathbf{Sum} \frac{P_1 \xrightarrow{\gamma_1} L_1}{P_1 + P_2 \xrightarrow{\gamma_1} L_1}$$

Cong this case is similar to the previous.

ECOM the last part of the derivation tree looks like this:

$$\mathbf{ECOM} \frac{P_1 \xrightarrow{xy} P'_1 \quad Q_1 \xrightarrow{\bar{xy}} Q'_1}{P_1 | Q_1 \xrightarrow{\tau} P'_1 | Q'_1}$$

for inductive hypothesis there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} P'_1 \quad \text{and} \quad \gamma_1 \cdots \gamma_{k+1} = xy$$

and there exist R_1, \dots, R_h and $\delta_1, \dots, \delta_{h+1}$ with $h \geq 0$ such that

$$Q_1 \xrightarrow{\delta_1} R_1 \xrightarrow{\delta_2} R_2 \cdots R_{h-1} \xrightarrow{\delta_h} R_h \xrightarrow{\delta_{h+1}} Q'_1 \quad \text{and} \quad \delta_1 \cdots \delta_{h+1} = \bar{xy}$$

For lemma 4.2.4 there cannot be an output action in a transition involving marked processes so h must be 0 and $Q_1 \xrightarrow{\delta_1} Q'_1$ with $\delta_1 = \bar{xy}$. We can have three different cases now:

$\gamma_1 = xy$ A proof of the conclusion is:

$$P_1 | Q_1 \xrightarrow{\epsilon} L_1 | Q'_1 \xrightarrow{\epsilon} L_2 | Q'_1 \cdots \xrightarrow{\epsilon} L_k | Q'_1 \xrightarrow{\tau} P'_1 | Q'_1$$

we derive the first transition with rule *Com3L*, whether for the other transition we use the rule *Par1L*.

$\gamma_i = xy$ A proof of the conclusion is:

$$P_1 | Q_1 \xrightarrow{\epsilon} L_1 | Q_1 \cdots \xrightarrow{\epsilon} L_{i-1} | Q_1 \xrightarrow{\epsilon} L_i | Q'_1 \xrightarrow{\epsilon} L_{i+1} | Q'_1 \cdots \xrightarrow{\epsilon} L_k | Q'_1 \xrightarrow{\tau} P'_1 | Q'_1$$

we derive the transaction $L_{i-1} | Q_1 \xrightarrow{\epsilon} L_i | Q'_1$ with rule *Com2L*, whether for the other transactions we use the rule *Par1L*.

$\gamma_{k+1} = xy$ similar.

Res the last part of the derivation tree looks like this:

$$\mathbf{Res} \frac{P_1 \xrightarrow{\sigma} Q_1 \quad z \notin n(\sigma)}{(\nu z)P_1 \xrightarrow{\sigma} (\nu z)Q_1}$$

for the inductive hypothesis there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q_1 \quad \text{and} \quad \gamma_1 \cdots \gamma_{k+1} = \sigma$$

We can apply the rule *Res* to each of the previous transitions because

$$z \notin n(\sigma) \text{ implies } z \notin n(\gamma_i) \text{ for each } i$$

and then get a proof of the conclusion:

$$(\nu z)P_1 \xrightarrow{\gamma_1} (\nu z)L_1 \xrightarrow{\gamma_2} (\nu z)L_2 \cdots (\nu z)L_{k-1} \xrightarrow{\gamma_k} (\nu z)L_k \xrightarrow{\gamma_{k+1}} (\nu z)Q_1$$

Par this case is similar to the previous.

EComSeq the last part of the derivation tree looks like this:

$$\mathbf{EComSeq} \frac{P_1 \xrightarrow{xy \cdot \sigma} P'_1 \quad Q_1 \xrightarrow{\bar{x}y} Q'_1}{P_1|Q_1 \xrightarrow{\sigma} P'_1|Q'_1}$$

for inductive hypothesis there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} P'_1 \quad \text{and} \quad \gamma_1 \cdot \dots \cdot \gamma_{k+1} = xy \cdot \sigma$$

For inductive hypothesis and lemma 4.2.4 $Q_1 \xrightarrow{\bar{x}y} Q'_1$. We can have two different cases now depending on where the first xy is:

$\gamma_1 = xy$ A proof of the conclusion is:

$$P_1|Q_1 \xrightarrow{\epsilon} L_1|Q'_1 \xrightarrow{\gamma_2} L_2|Q'_1 \cdots \xrightarrow{\gamma_k} L_k|Q'_1 \xrightarrow{\gamma_{k+1}} P'_1|Q'_1$$

we derive the first transition with rule *Com3L*, whether for the other transactions we use the rule *Par1L*. Since $\gamma_1 \cdot \dots \cdot \gamma_{k+1} = xy \cdot \sigma$ and $\gamma_1 = xy$ then $\epsilon \cdot \gamma_2 \cdot \dots \cdot \gamma_{k+1} = \sigma$

$\gamma_i = xy$ A proof of the conclusion is:

$$P_1|Q_1 \xrightarrow{\epsilon} L_1|Q_1 \cdots \xrightarrow{\epsilon} L_{i-1}|Q_1 \xrightarrow{\epsilon} L_i|Q'_1 \xrightarrow{\gamma_{i+1}} L_{i+1}|Q'_1 \cdots \xrightarrow{\gamma_k} L_k|Q'_1 \xrightarrow{\gamma_{k+1}} P'_1|Q'_1$$

we derive the transition $L_{i-1}|Q_1 \xrightarrow{\epsilon} L_i|Q'_1$ with rule *Com2L*, whether for the other transactions of the premises we use the rule *Par1L*.

$\gamma_{k+1} = xy$ cannot happen because σ is not empty.

□

Proposition 4.2.6. Let \rightarrow be the relation defined in table 4.1. Let α be an action. If $P \xrightarrow{\alpha} Q$ then $P \xrightarrow{\alpha} Q$.

Proof. The proof is by induction the depth of the derivation of $P \xrightarrow{\alpha} Q$:

Out, EInp, Tau These rules are also in table 4.1 so we can derive $P \xrightarrow{\alpha} Q$.

Com1

$$\mathbf{Com1} \frac{P_1 \xrightarrow{xy} Q_1 \quad P_2 \xrightarrow{\bar{x}y} Q_2}{P_1|P_2 \xrightarrow{\tau} Q_1|Q_2}$$

for inductive hypothesis $P_1 \xrightarrow{xy} Q_1$ and $P_2 \xrightarrow{\bar{x}y} Q_2$ so for rule *Com* $P_1|P_2 \xrightarrow{\tau} Q_1|Q_2$

Sum

$$\mathbf{Sum} \frac{P_1 \xrightarrow{\alpha} Q}{P_1 + P_2 \xrightarrow{\alpha} Q}$$

for inductive hypothesis $P_1 \xrightarrow{\alpha} Q$ and for rule *Sum* $P_1 + P_2 \xrightarrow{\alpha} Q$.

Res the first transition is:

| | | |
|---|--|---|
| Out $\frac{}{\overline{xy}.P \xrightarrow{\overline{xy}}_n P}$ | Tau $\frac{}{\tau.P \xrightarrow{\tau}_n P}$ | Inp $\frac{n \geq 0}{\underline{x_1(y_1)}. \dots \underline{x_n(y_n)}.z(w).P \xrightarrow{\widehat{x(y) \cdot z(w)}}_n P}$ |
| LComSeqL $\frac{(\nu \tilde{a})P \xrightarrow{x(y) \cdot \sigma}_n (\nu \tilde{b})P' \quad (\nu \tilde{c})Q \xrightarrow{\overline{xz}}_n (\nu \tilde{d})Q' \quad bn(\sigma\{z/y\}) \cap fn(Q) = \emptyset}{(\nu \tilde{a}\tilde{c})(P Q) \xrightarrow{\sigma\{z/y\}}_n (\nu \tilde{b}\tilde{d})(P'\{z/y\} Q')}$ | | |
| LComL $\frac{(\nu \tilde{a})P \xrightarrow{x(y)}_n (\nu \tilde{b})P' \quad (\nu \tilde{c})Q \xrightarrow{\overline{xz}}_n (\nu \tilde{d})Q'}{(\nu \tilde{a}\tilde{b})(P Q) \xrightarrow{\tau}_n (\nu \tilde{c}\tilde{d})(P'\{z/y\} Q')}$ | | |
| SumL $\frac{(\nu \tilde{a})P \xrightarrow{\sigma}_n (\nu \tilde{b})P' \quad (\nu \tilde{c})Q \text{ n.f.} \quad \tilde{c} \cap n(\sigma) = \emptyset}{(\nu \tilde{a}\tilde{c})(P + Q) \xrightarrow{\sigma}_n (\nu \tilde{b}\tilde{c})P'}$ | | |
| Res $\frac{P \xrightarrow{\sigma}_n P' \quad z \notin n(\sigma)}{(\nu z)P \xrightarrow{\sigma}_n (\nu z)P'}$ | Opn $\frac{P \xrightarrow{\overline{xz}}_n P' \quad z \neq x}{(\nu z)P \xrightarrow{\overline{x(z)}}_n P'}$ | |
| ParL $\frac{(\nu \tilde{a})P \xrightarrow{\sigma}_n (\nu \tilde{b})P' \quad bn(\sigma) \cap fn(Q) = \emptyset \quad (\nu \tilde{c})Q \text{ n.f.} \quad \tilde{c} \cap n(\sigma) = \emptyset}{(\nu \tilde{a}\tilde{c})(P Q) \xrightarrow{\sigma}_n (\nu \tilde{b}\tilde{c})(P' Q)}$ | | |

Table 4.4: Multi π calculus late semantic for normal forms. Every process in the head of a transition in the premise of a rule is in normal form. The restrictions can be empty. When we write $(\nu \tilde{x})P$ n.f. it means that P has no restriction at the top level. Commutative counterpart of rule *ParL*, *SumL*, *EComL* are omitted. Also in rules for prefix P is a normal form

$$\mathbf{Res} \frac{P_1 \xrightarrow{\alpha} Q_1 \quad z \notin n(\gamma_1)}{(\nu z)P_1 \xrightarrow{\alpha} (\nu z)Q_1}$$

for inductive hypothesis $P_1 \xrightarrow{\alpha} Q_1$ and for rule *Res* $(\nu z)P_1 \xrightarrow{\alpha} (\nu z)Q_1$.

others other cases are similar.

□

4.2.4 Semantic on normal forms

Definition 4.2.6. The *late transition relation for normal forms* is the smallest relation induced by the rules in table 4.4, written \rightarrow_n . Every process in the head of transition in the premise of a rule in table 4.4 is assumed to be in normal form. Also when we write $(\nu \tilde{x})P$ is a normal form, it means that P has no restriction at the top level.

4.3 Strong bisimilarity and equivalence

4.3.1 Strong bisimilarity

In the following $\widehat{x(y)} = x_1(y_1) \cdot \dots \cdot x_n(y_n)$ and $\tilde{x} = x_1 \cdot \dots \cdot x_n$. In the following section \rightarrow and \rightarrow_3 are the late semantic with structural congruence of multi π calculus with strong prefixing.

Definition 4.3.1. A *strong bisimulation* is a symmetric binary relation \mathbf{S} on multi π processes such that for all PSQ :

- $P \xrightarrow{\alpha} P'$, $bn(\alpha)$ is fresh and α is not an input nor a sequence of inputs then there exists some Q' such that $Q \xrightarrow{\alpha} Q'$ and $P' \mathbf{S} Q'$
- $P \xrightarrow{\widetilde{x(y)}} P'$ where γ is a possibly empty sequence of inputs and \tilde{y} is fresh then there exists some Q' such that $Q \xrightarrow{\widetilde{x(y)}} Q'$ and for all \tilde{w} , $P' \{ \tilde{w}/\tilde{y} \} \mathbf{S} Q' \{ \tilde{w}/\tilde{y} \}$

P and Q are strongly bisimilar, written $P \sim_L Q$, if they are related by a strong bisimulation.

Is this definition a proper extension of the one in [4]? The only way to tell is by showing some example of process that we intuitively want to be bisimilar.

Example :

$$P = \underline{a(u)}.b(v).0 \quad P \sim_L Q \quad \underline{a(x)}.b(v).(\nu y)\bar{y}u.0 = Q$$

This is because for all $u \in \mathbf{N} - \{b\}$ and for all $v \in \mathbf{N} - \{u\}$: $P \xrightarrow{a(u) \cdot b(v)} 0$. For all $x \in \mathbf{N} - \{b, u\}$ and for all $v \in \mathbf{N} - \{u, x, y\}$: $Q \xrightarrow{a(x) \cdot b(v)} 0$. Taking z, w fresh in P and Q means: $z, w \in \mathbf{N} - \{a, b, u\}$, so both P and Q can make the transition $\xrightarrow{a(z) \cdot b(w)}$ and arrive to 0.

Definition 4.3.2. Let \mathbf{R} be a strong late bisimulation. A *strong bisimulation up to \mathbf{R}* is a symmetric binary relation \mathbf{S} on multi π processes such that for all PSQ :

- $P \xrightarrow{\alpha} P'$, $bn(\alpha)$ is fresh and α is not an input nor a sequence of inputs then there exist processes Q', Q'', P'' such that $Q \xrightarrow{\alpha} Q'$ and $P' \mathbf{R} P'' \mathbf{S} Q'' \mathbf{R} Q'$
- $P \xrightarrow{x_1(y_1) \dots x_n(y_n)} P'$ where γ is a possibly empty sequence of inputs and $y_1 \dots y_n$ is fresh then there exists some Q' such that $Q \xrightarrow{x_1(y_1) \dots x_n(y_n)} Q'$ and for all $w_1 \dots w_n$ $P' \{ w_1/y_1, \dots, w_n/y_n \} \mathbf{R} \mathbf{S} \mathbf{R} Q' \{ w_1/y_1, \dots, w_n/y_n \}$

P and Q are strongly bisimilar up to \mathbf{R} , written $P \sim_L^{\mathbf{R}} Q$, if they are related by a strong bisimulation up to \mathbf{R} .

Proposition 4.3.1. $P \sim_L^{\mathbf{R}} Q$ imply $P \sim_L Q$.

Proof. Let \mathbf{S} be a bisimulation up to \mathbf{R} such that PSQ . It can be proved that $\mathbf{R} \mathbf{S} \mathbf{R}$ is a bisimulation: let $\mathbf{A} \mathbf{R} \mathbf{B} \mathbf{S} \mathbf{C} \mathbf{R} \mathbf{D}$ and let γ be a non input action

$$\begin{aligned} \mathbf{A} \xrightarrow{\gamma} \mathbf{A}' \wedge \mathbf{A} \mathbf{R} \mathbf{B} \wedge \text{definition 4.3.1} &\Rightarrow \exists \mathbf{B}' : \mathbf{B} \xrightarrow{\gamma} \mathbf{B}' \wedge \mathbf{A}' \mathbf{R} \mathbf{B}' \\ \mathbf{B} \mathbf{S} \mathbf{C} \wedge \text{definition 4.3.2} &\Rightarrow \exists \mathbf{C}' \mathbf{C}'' \mathbf{B}'' : \mathbf{C} \xrightarrow{\gamma} \mathbf{C}' \wedge \mathbf{B}' \mathbf{R} \mathbf{B}'' \mathbf{S} \mathbf{C}'' \mathbf{R} \mathbf{C}' \\ \mathbf{C} \xrightarrow{\gamma} \mathbf{C}' \wedge \mathbf{C} \mathbf{R} \mathbf{D} \wedge \text{definition 4.3.1} &\Rightarrow \exists \mathbf{D}' : \mathbf{D} \xrightarrow{\gamma} \mathbf{D}' \wedge \mathbf{C}' \mathbf{R} \mathbf{D}' \\ \mathbf{A}' \mathbf{R} \mathbf{B}' \mathbf{R} \mathbf{B}'' \mathbf{S} \mathbf{C}'' \mathbf{R} \mathbf{C}' \mathbf{R} \mathbf{D}' \wedge \text{transitivity of } \mathbf{R} &\Rightarrow \mathbf{A}' \mathbf{R} \mathbf{B}'' \mathbf{S} \mathbf{C}'' \mathbf{R} \mathbf{D}' \end{aligned}$$

It is easy to see that the symmetric also holds. For the other case: let $x_1(y_1) \dots x_n(y_n) = \tilde{x}(\tilde{y})$

$$\begin{aligned} \mathbf{A} \xrightarrow{\tilde{x}(\tilde{y})} \mathbf{A}' \wedge \mathbf{A} \mathbf{R} \mathbf{B} \wedge \text{definition 4.3.1} &\Rightarrow \exists \mathbf{B}' : \mathbf{B} \xrightarrow{\tilde{x}(\tilde{y})} \mathbf{B}' \text{ and for all } \tilde{w} : \mathbf{A}' \{ \tilde{w}/\tilde{y} \} \mathbf{R} \mathbf{B}' \{ \tilde{w}/\tilde{y} \} \\ \mathbf{B} \mathbf{S} \mathbf{C} \wedge \text{definition 4.3.2} &\Rightarrow \exists \mathbf{C}' : \mathbf{C} \xrightarrow{\tilde{x}(\tilde{y})} \mathbf{C}' \wedge \mathbf{B}' \{ \tilde{w}/\tilde{y} \} \mathbf{R} \mathbf{S} \mathbf{R} \mathbf{C}' \{ \tilde{w}/\tilde{y} \} \\ \mathbf{C} \xrightarrow{\tilde{x}(\tilde{y})} \mathbf{C}' \wedge \mathbf{C} \mathbf{R} \mathbf{D} \wedge \text{definition 4.3.1} &\Rightarrow \exists \mathbf{D}' : \mathbf{D} \xrightarrow{\tilde{x}(\tilde{y})} \mathbf{D}' \wedge \mathbf{C}' \{ \tilde{w}/\tilde{y} \} \mathbf{R} \mathbf{D}' \{ \tilde{w}/\tilde{y} \} \\ \mathbf{A}' \{ \tilde{w}/\tilde{y} \} \mathbf{R} \mathbf{B}' \{ \tilde{w}/\tilde{y} \} \mathbf{R} \mathbf{S} \mathbf{R} \mathbf{C}' \{ \tilde{w}/\tilde{y} \} \mathbf{R} \mathbf{D}' \{ \tilde{w}/\tilde{y} \} \wedge \text{transitivity of } \mathbf{R} &\Rightarrow \mathbf{A}' \{ \tilde{w}/\tilde{y} \} \mathbf{R} \mathbf{S} \mathbf{R} \mathbf{D}' \{ \tilde{w}/\tilde{y} \} \end{aligned}$$

It is easy to see that the symmetric also holds. □

Lemma 4.3.2. Structural congruence is a strong late bisimulation.

Proof. Let $P \equiv Q$. If $P \xrightarrow{\sigma} P'$ then for symmetry of \equiv and rule *Cong*: $Q \xrightarrow{\sigma} P'$. If $Q \xrightarrow{\sigma} Q'$ then for rule *Cong*: $P \xrightarrow{\sigma} Q'$ □

Lemma 4.3.3. $P \sim_L Q$ and $P \equiv R$ then $R \sim_L Q$.

Proof. $P \equiv R$ implies for lemma 4.3.2: $P \sim_L R$. $P \sim_L Q$ and $P \sim_L R$ imply for transitivity and symmetry that $R \sim_L Q$ \square

Definition 4.3.3. A *strong bisimulation up to restriction* is a symmetric binary relation \mathbf{S} on multi π processes such that for all P and Q : $\mathbf{P}\mathbf{S}Q$ imply

- for all $w \notin (fn(P) \cup fn(Q))$: $P\{w/z\} \sim_L Q\{w/z\}$
- if $P \xrightarrow{\bar{x}y} P'$ then there exists Q' such that $Q \xrightarrow{\bar{x}y} Q'$ and $P'\mathbf{S}Q'$
- if $y \notin n(P, Q)$ and $P \xrightarrow{x(y)} P'$ then there exists Q' such that $Q \xrightarrow{x(y)} Q'$ and for all v : $P'\{v/y\}\mathbf{S}Q'\{v/y\}$
- if $y \notin n(P, Q)$ and $P \xrightarrow{\bar{x}(y)} P'$ then there exists Q' such that $Q \xrightarrow{\bar{x}(y)} Q'$ and $P'\mathbf{S}Q'$
- if $P \xrightarrow{\tau} P'$ then for some Q' : $Q \xrightarrow{\tau} Q'$ and either $P'\mathbf{S}Q'$ or for some P'', Q'' and w : $P' \equiv (\nu w)P'', Q' \equiv (\nu w)Q''$ and $P''\mathbf{S}Q''$

Two processes P, Q are *strongly late bisimilar up to restriction*, written $P \sim_L^\nu Q$, if they are related by a strong early bisimulation up to restriction.

Lemma 4.3.4. A strong bisimulation up to restriction is a strong bisimulation.

Lemma 4.3.5. \sim_L is preserved by output prefix.

Proof. Let $P \sim_L Q$. There are two cases:

Out : $\bar{x}y.P \xrightarrow{\bar{x}y} P \sim_L Q \xleftarrow{\bar{x}y} \bar{x}y.Q$.

Cong :

$$\text{Cong} \frac{\text{CongOut} \frac{P \equiv R}{\bar{x}y.P \equiv \bar{x}y.R} \quad \bar{x}y.R \xrightarrow{\bar{x}y} R}{\bar{x}y.P \xrightarrow{\bar{x}y} R}$$

for rule *Out*: $\bar{x}y.P \xrightarrow{\bar{x}y} P$. $P \equiv R$ imply $P \sim_L R$.

\square

Lemma 4.3.6. \sim_L is preserved by τ prefix.

Proof. Let $P \sim_L Q$. There are two cases:

Tau : $\tau.P \xrightarrow{\tau} P \sim_L Q \xleftarrow{\tau} \tau.Q$.

Cong :

$$\text{Cong} \frac{\text{CongTau} \frac{P \equiv R}{\tau.P \equiv \tau.R} \quad \tau.R \xrightarrow{\tau} R}{\tau.P \xrightarrow{\tau} R}$$

for rule *Out*: $\tau.P \xrightarrow{\tau} P$. $P \equiv R$ imply $P \sim_L R$.

\square

Lemma 4.3.7. Let y be fresh in P, Q . If for all w : $P\{w/y\} \sim_L Q\{w/y\}$ then $((\nu x)P)\{x/y\} \equiv_\alpha \text{Res}(\sim_L) \equiv_\alpha ((\nu x)Q)\{x/y\}$.

Proof. for all w : $P\{w/y\} \sim_L Q\{w/y\}$ imply $P\{x/y\} \sim_L Q\{x/y\}$ which imply $P\{x/y\}(xy) \sim_L Q\{x/y\}(xy)$ where (xy) is the permutation that swaps x with y and does not change other names. $P\{x/y\}(xy) \sim_L Q\{x/y\}(xy)$ imply

$$P\{y/x\} \equiv_\alpha P\{x/y\}(xy) \sim_L Q\{x/y\}(xy) \equiv_\alpha Q\{y/x\}$$

which imply $P\{y/x\} \sim_L Q\{y/x\}$. So $P\{y/x\} \sim_L Q\{y/x\}$ and

$$(\nu y)(P\{y/x\}) \text{Res}(\sim_L)(\nu y)(Q\{y/x\})$$

which imply

$$((\nu y)P\{y/x\})\{x/y\} \text{Res}(\sim_L)((\nu y)Q\{y/x\})\{x/y\}$$

and

$$((\nu x)P)\{x/y\} \equiv_\alpha ((\nu y)P\{y/x\})\{x/y\} \text{Res}(\sim_L)((\nu y)Q\{y/x\})\{x/y\} \equiv_\alpha ((\nu x)Q)\{x/y\}$$

□

Lemma 4.3.8. Restriction preserves strong late bisimulation.

Proof. We prove that

$$\text{Res}(\sim_L) = \{((\nu x)P, (\nu x)Q) : P \sim_L Q\} \cup \sim_L$$

is a late bisimulation up to α equivalence. For lemma 4.2.3 and reflexivity of structural congruence we can assume that every transition that starts from $(\nu x)P$ is in the form: $(\nu x)P \equiv S$, $S \xrightarrow{\gamma} S$ and this derivation does not use the rule *Cong* but can use the commutative counterpart of the rules for sum, parallel and communication. So we can proceed by induction on the sum of the sizes of the derivations of $(\nu x)P \equiv S$ and $S \xrightarrow{\gamma} S$. Then by cases on the last pair of rules used:

(*CongRes*, *Res*) :

$$\text{CongRes} \frac{P \equiv P_1}{(\nu x)P \equiv (\nu x)P_1} \quad \text{Res} \frac{P_1 \xrightarrow{\gamma} P'_1 \quad x \notin n(\gamma)}{(\nu x)P_1 \xrightarrow{\gamma} (\nu x)P'_1}$$

$P \equiv P_1$ and $P_1 \xrightarrow{\gamma} P'_1$ imply $P \xrightarrow{\gamma} P'_1$. $P \sim_L$ and $P \xrightarrow{\gamma} P'_1$ imply $Q \xrightarrow{\gamma} Q'_1$ and the conditions of bisimulation on P'_1 and Q'_1 are met. For rule *Res*: $(\nu x)Q \xrightarrow{\gamma} (\nu x)Q'_1$. There are three cases now:

- γ is a sequence of bound outputs: $\widetilde{y}(z)$. In this case for all \tilde{w} such that $x \notin \tilde{w}$: $P'_1\{\tilde{w}/\tilde{z}\} \sim_L Q'_1\{\tilde{w}/\tilde{z}\}$ which imply $(\nu x)(P'_1\{\tilde{w}/\tilde{z}\}) \text{Res}(\sim_L)(\nu x)(Q'_1\{\tilde{w}/\tilde{z}\})$. For definition of substitution: $((\nu x)P'_1)\{\tilde{w}/\tilde{z}\} \text{Res}(\sim_L)((\nu x)Q'_1)\{\tilde{w}/\tilde{z}\}$.
- γ is a sequence of bound outputs: $\widetilde{y}(z)$. In this case for all \tilde{w} such that $x \in \tilde{w}$: $P'_1\{\tilde{w}/\tilde{z}\} \sim_L Q'_1\{\tilde{w}/\tilde{z}\}$ which imply $(\nu x)(P'_1\{\tilde{w}/\tilde{z}\}) \text{Res}(\sim_L)(\nu x)(Q'_1\{\tilde{w}/\tilde{z}\})$. For definition of substitution and lemma 4.3.7:

$$((\nu x')P'_1\{x'/x\})\{\tilde{w}/\tilde{z}\} \text{Res}(\sim_L) \equiv_\alpha ((\nu x')Q'_1\{x'/x\})\{\tilde{w}/\tilde{z}\}$$

- γ is an action and not a bound output. In this case $P'_1 \sim_L Q'_1$ and $(\nu x)(P'_1) \text{Res}(\sim_L)(\nu x)(Q'_1)$.

(*CongRes*, *Opn*) similar.

(*Alp*, *Res*)(1) :

$$\text{AlpRes} \frac{P \equiv_\alpha P_1}{(\nu x)P \equiv_\alpha (\nu x)P_1} \quad \text{Res} \frac{P_1 \xrightarrow{\gamma} P'_1 \quad x \notin n(\gamma)}{(\nu x)P_1 \xrightarrow{\gamma} (\nu x)P'_1}$$

$P \equiv_\alpha P_1$ imply $P \equiv P_1$ and this case is similar to (*CongRes*, *Res*).

(*Alp, Res*)(2) :

$$\mathbf{AlpRes} \frac{P \equiv_{\alpha} P_1 \quad y \notin fn(P_1, Q)}{(\nu x)P \equiv_{\alpha} (\nu y)(P_1\{y/x\})} \quad \mathbf{Res} \frac{P_1\{y/x\} \xrightarrow{\gamma} P'_1 \quad y \notin n(\gamma)}{(\nu y)P_1\{y/x\} \xrightarrow{\gamma} (\nu y)P'_1}$$

$y \notin fn(P_1)$ and (xy) is a permutation that swaps x with y imply that $P_1\{y/x\} \equiv_{\alpha} P_1(xy)$. $P \sim_E Q$ and $P \equiv_{\alpha} P_1$ imply $P_1 \sim_E Q$. $P_1 \sim_E Q$ imply $P_1(xy) \sim_E Q(xy)$. $P_1(xy) \sim_E Q(xy)$ and $P_1\{y/x\} \equiv_{\alpha} P_1(xy)$ imply $P_1\{y/x\} \sim_E Q(xy)$. $y \notin fn(Q)$ imply $Q \sim_E Q(xy)$. $Q \sim_E Q(xy)$ and $P_1\{y/x\} \sim_E Q(xy)$ imply $P_1\{y/x\} \sim_E Q$. So $Q \xrightarrow{\gamma} Q'_1$ and $Q'_1 \sim_E P'_1$. For rule *Res*: $(\nu x)Q \xrightarrow{\gamma} (\nu x)Q'_1$ and $(\nu x)Q'_1 \text{Res}(\sim_E)(\nu x)P'_1$.

(*Alp, Opn*) similar.

(*ScpExtPar1, _*) rule *ScpExtPar1* is emulated by *ScpExtPar2* and commutativity.

(*ScpExtSum1, _*) rule *ScpExtSum1* is emulated by *ScpExtSum2* and commutativity.

□

Lemma 4.3.9. \sim_L is preserved by sum.

Proof. For lemma 4.2.3 and reflexivity of structural congruence we can assume that every transition that starts from $P + R$ is in the form: $P + R \equiv S$, $S \xrightarrow{\gamma} S$ and this derivation does not used the rule *Cong* but can use the commutative counterpart of the rules for sum, parallel and communication. So we can proceed by induction on the sum of the sizes of the derivations of $P + R \equiv S$ and $S \xrightarrow{\gamma} S$. Then by cases on the last pair of rules used:

(*CongSum, SumL*) :

$$\mathbf{CongSum} \frac{P \equiv P_1 \quad R \equiv R_1}{P + R \equiv P_1 + R_1} \quad \mathbf{SumL} \frac{P_1 \xrightarrow{\gamma} P'_1}{P_1 + R_1 \xrightarrow{\gamma} P'_1}$$

$P \equiv P_1$ and $P_1 \xrightarrow{\gamma} P'_1$ imply $P \xrightarrow{\gamma} P'_1$. $P \sim_L Q$ and $P \xrightarrow{\gamma} P'_1$ imply $Q \xrightarrow{\gamma} Q'_1$ and the condition of late bisimulation on P'_1 and Q'_1 are met. For rule *SumL*: $Q + R \xrightarrow{\gamma} Q'_1$

(*CongSum, SumR*) :

$$\mathbf{CongSum} \frac{P \equiv P_1 \quad R \equiv R_1}{P + R \equiv P_1 + R_1} \quad \mathbf{SumR} \frac{R_1 \xrightarrow{\gamma} R'_1}{P_1 + R_1 \xrightarrow{\gamma} R'_1}$$

$R \equiv R_1$ and $R_1 \xrightarrow{\gamma} R'_1$ imply $R \xrightarrow{\gamma} R'_1$. For rule *SumR*: $Q + R \xrightarrow{\gamma} R'_1$

(*ScpExtSum2, Res*)(1) :

$$\mathbf{ScpExtSum2} \frac{P \equiv P_1 \quad R \equiv R_1}{P + (\nu x)R \equiv (\nu x)(P_1 + R_1)} \quad \mathbf{Res} \frac{\mathbf{SumL} \frac{P_1 \xrightarrow{\gamma} P'_1}{P_1 + R_1 \xrightarrow{\gamma} P'_1} \quad x \notin n(\gamma)}{(\nu x)(P_1 + R_1) \xrightarrow{\gamma} (\nu x)P'_1}$$

$P \equiv P_1$ and $P_1 \xrightarrow{\gamma} P'_1$ imply $P \xrightarrow{\gamma} P'_1$. $P \sim_L Q$ and $P \xrightarrow{\gamma} P'_1$ imply $Q \xrightarrow{\gamma} Q'_1$ and the conditions of late bisimulation on P'_1 and Q'_1 are met. For rule *SumL*: $Q + (\nu x)R \xrightarrow{\gamma} Q'_1$. For rules *Cong* and *ScpExtSum2*: $(\nu x)(Q + R) \xrightarrow{\gamma} (\nu x)Q'_1$. Also the conditions of late bisimulation on $(\nu x)P'_1$ and $(\nu x)Q'_1$ are met since for lemma 4.3.8 restriction preserves late bisimulation.

(*ScpExtSum2, Res*)(2) :

$$\text{ScpExtSum2} \frac{P \equiv P_1 \quad R \equiv R_1}{P + (\nu x)R \equiv (\nu x)(P_1 + R_1)} \quad \text{Res} \frac{\text{SumR} \frac{R_1 \xrightarrow{\gamma} R'_1}{P_1 + R_1 \xrightarrow{\gamma} R'_1} \quad x \notin n(\gamma)}{(\nu x)(P_1 + R_1) \xrightarrow{\gamma} (\nu x)R'_1}$$

For rules *SumR*, *Cong* and *ScpExtSum2*: $(\nu x)(Q + R) \xrightarrow{\gamma} (\nu x)Q'_1$ and the conditions of late bisimulation on $(\nu x)P'_1$ and $(\nu x)Q'_1$ are met.

(*ScpExtSum2*, *Opn*) similar.

(*ScpExtSum1*, $_$) rule *ScpExtSum1* is emulated by *ScpExtSum2* and commutativity.

(*Alp*, $_$)

□

Lemma 4.3.10. Parallel composition preserves late bisimilarity.

Proof.

□

Lemma 4.3.11. \sim_L is not preserved by input prefix and strong input prefix.

Proof.

□

Theorem 4.3.12. \sim_L is preserved by all operators except input prefix.

Proof.

□

4.3.2 Open bisimulation

Let \rightarrow_L be the semantic defined in table 4.2 the following is an attempt to extend the definition of strong open bisimulation found in [4]:

Definition 4.3.4. An *strong open \mathbb{D} bisimulation* is $\{S_D\}_{D \in \mathbb{D}}$ a family of symmetric binary relations on processes such that for each process P, Q , for each distinction $D \in \mathbb{D}$, for each name substitution σ which respects D if PS_DQ then

- if $P\sigma \xrightarrow{\bar{a}(x)}_L P'$ and x is fresh then there exists Q' such that $Q\sigma \xrightarrow{\bar{a}(x)}_L Q'$ and $P'S_{D'}Q'$ where $D' = D\sigma \cup \{x\} \times (fn(P\sigma) \cup fn(Q\sigma)) \cup (fn(P\sigma) \cup fn(Q\sigma)) \times \{x\}$
- if $P\sigma \xrightarrow{\gamma}_L P'$ and $bn(\gamma)$ is fresh then there exists Q' such that $Q\sigma \xrightarrow{\gamma}_L Q'$ and $P'S_{D\sigma}Q'$

P and Q are (*strongly*) *open \mathbb{D} bisimilar*, written $P \sim_O^{\mathbb{D}} Q$ if there exists a member S_D of an open bisimulation such that PS_DQ . They are *open bisimilar* if they are open \mathbb{D} bisimilar and $\emptyset \in \mathbb{D}$, written $P \sim_O D$.

Lemma 4.3.13. $P \sim_O Q$ imply $P \sim_O^{\mathbb{D}} Q$.

Proof. For definition $P \sim_O Q$ imply that there exists a \mathbb{D} such that $P \sim_O^{\mathbb{D}} Q$.

□

Lemma 4.3.14. Output prefixing preserves open \mathbb{D} bisimulation.

Proof. Let $\mathbb{S} = \{S_D\}_{D \in \mathbb{D}}$ be an open \mathbb{D} bisimulation and let PS_DQ for some $D \in \mathbb{D}$. Then the goal is to prove that: $(\bar{x}y.P)\sigma S_{D\sigma}(\bar{x}y.Q)\sigma$. There are two cases according to the last rule that can be applied to $(\bar{x}y.P)\sigma$:

$$\text{Out} : (\bar{x}y.P)\sigma \xrightarrow{(\bar{x}y)\sigma} P\sigma?Q\sigma \xleftarrow{(\bar{x}y)\sigma} (\bar{x}y.Q)\sigma \text{ E POI?}$$

Cong the last part of the derivation look like this:

$$\mathbf{Cong} \frac{\mathbf{CongOut} \frac{P \equiv R}{\bar{x}y.P \equiv \bar{x}y.R} \quad \mathbf{Out} \frac{}{\bar{x}y.R \xrightarrow{\bar{x}y} R}}{\bar{x}y.P \xrightarrow{\bar{x}y} R}$$

For rule *Out*: $\bar{x}y.Q \xrightarrow{\bar{x}y} Q$. $P \equiv R$ imply $P \sim_O$

□

Theorem 4.3.15. $P \sim_O^D Q$ is a congruence.

Proof.

□

Chapter 5

Multi π calculus with strong input and output

5.1 Syntax

As we did with multi π calculus, we suppose that we have a countable set of names \mathbb{N} , ranged over by lower case letters a, b, \dots, z . This names are used for communication channels and values. Furthermore we have a set of identifiers, ranged over by A . We represent the agents or processes by upper case letters P, Q, \dots . A multi π process, in addition to the same actions of a π process, can perform also a strong prefix:

$$\pi ::= \bar{x}y \mid x(z) \mid x(y) \mid \bar{x}y \mid \tau$$

The process are defined, just as original π calculus, by the following grammar:

$$P, Q ::= 0 \mid \pi.P \mid P|Q \mid P + Q \mid (\nu x)P \mid A(y_1, \dots, y_n)$$

and they have the same intuitive meaning as for the π calculus. The strong prefix input allows a process to make an atomic sequence of actions, so that more than one process can synchronize on this sequence.

We have to extend the following definition to deal with the strong prefix:

$$\begin{aligned} B(x(y).Q, I) &= \{y, \bar{y}\} \cup B(Q, I) & F(x(y).Q, I) &= \{x, \bar{x}\} \cup (F(Q, I) - \{y, \bar{y}\}) \\ B(\bar{x}y.Q, I) &= B(Q, I) & F(\bar{x}y.Q, I) &= \{x, \bar{x}, y, \bar{y}\} \cup F(Q, I) \end{aligned}$$

5.2 Operational semantic

5.2.1 Early operational semantic with structural congruence

Definition 5.2.1. The *early transition relation with structural congruence* is the smallest relation induced by the rules in table 5.1:

The names $\sigma, \sigma_1, \sigma_2, \sigma_3$ are non empty sequences of actions and are also not τ . The relation $ESync$ is defined by the axioms in table 5.2

Example *Transactional synchronization.* This is an example of two processes that synchronize over a sequence of actions of length two:

$$\bar{a}x.\bar{a}y.P|a(w).a(z).Q \xrightarrow{\tau} P|Q\{x/w\}\{y/z\}$$

We start first noticing that

$$\text{S4R} \frac{\text{S1R} \frac{}{Sync(\bar{a}y, ay, \tau)}}{Sync(\bar{a}x \cdot \bar{a}y, ax \cdot ay, \tau)}$$

and that

| | | |
|--|--|---|
| Inp $\frac{}{x(y).P \xrightarrow{xz} P\{z/x\}}$ | Tau $\frac{}{\tau.P \xrightarrow{\tau} P}$ | Out $\frac{}{\bar{x}y.P \xrightarrow{\bar{x}y} P}$ |
| SInp $\frac{P\{z/y\} \xrightarrow{\sigma} P' \quad \sigma \neq \tau}{\underline{x(y)}.P \xrightarrow{xz \cdot \sigma} P'}$ | SOut $\frac{P \xrightarrow{\sigma} P' \quad \sigma \neq \tau}{\underline{\bar{x}y}.P \xrightarrow{\bar{x}y \cdot \sigma} P'}$ | |
| ECom $\frac{P \xrightarrow{\sigma_1} P' \quad Q \xrightarrow{\sigma_2} Q' \quad ESync(\sigma_1, \sigma_2, \sigma_3)}{P Q \xrightarrow{\sigma_3} P' Q'}$ | | |
| Sum $\frac{P \xrightarrow{\sigma} P'}{P + Q \xrightarrow{\sigma} P'}$ | Cong $\frac{P \equiv P' \quad P' \xrightarrow{\alpha} Q}{P \xrightarrow{\alpha} Q}$ | |
| Res $\frac{P \xrightarrow{\sigma} P' \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\sigma} (\nu z)P'}$ | Par $\frac{P \xrightarrow{\sigma} P'}{P Q \xrightarrow{\sigma} P' Q}$ | |

Table 5.1: Multi π early semantic with structural congruence

| | |
|--|--|
| S1L $\frac{}{ESync(xy, \bar{x}y, \tau)}$ | S1R $\frac{}{ESync(\bar{x}y, xy, \tau)}$ |
| S2L $\frac{}{ESync(xy, \bar{x}y \cdot \sigma, \sigma)}$ | S2R $\frac{}{ESync(\bar{x}y \cdot \sigma, xy, \sigma)}$ |
| S3L $\frac{}{ESync(xy \cdot \sigma, \bar{x}y, \sigma)}$ | S3R $\frac{}{ESync(\bar{x}y, xy \cdot \sigma, \sigma)}$ |
| S4L $\frac{ESync(\sigma_1, \sigma_2, \sigma_3)}{ESync(xy \cdot \sigma_1, \bar{x}y \cdot \sigma_2, \sigma_3)}$ | S4R $\frac{ESync(\sigma_1, \sigma_2, \sigma_3)}{ESync(\bar{x}y \cdot \sigma_1, xy \cdot \sigma_2, \sigma_3)}$ |

Table 5.2: Synchronization relation

$$\text{SOUT} \frac{\text{OUT} \frac{}{\bar{a}y.P \xrightarrow{\bar{a}y} P}}{\bar{a}x.\bar{a}y.P \xrightarrow{\bar{a}x.\bar{a}y} P} \quad \text{SINP} \frac{\text{INP} \frac{}{(a(z).Q)\{x/w\} \xrightarrow{ay} Q\{x/w\}\{y/z\}}}{a(w).a(z).Q \xrightarrow{ax.ay} Q}$$

and in the end we just need to apply the rule **LCom**

Example Multi-party synchronization. In this example we have three processes that want to synchronize:

$$\begin{array}{c} \text{ECom} \frac{\bar{a}f.\bar{b}g.P|a(w).Q \xrightarrow{\bar{b}g} P|Q\{f/w\} \quad \text{Inp} \frac{}{b(y).R \xrightarrow{bg} R\{g/y\}} \quad \text{S1R} \frac{}{\text{Sync}(\bar{b}g, bg, \tau)}}{(\bar{a}f.\bar{b}g.P|a(w).Q)|b(y).R \xrightarrow{\tau} (P|Q\{f/w\})|R\{g/y\}} \\ \\ \text{LCom} \frac{\bar{a}f.\bar{b}g.P \xrightarrow{\bar{a}f.\bar{b}g} P \quad \text{Inp} \frac{}{a(w).Q \xrightarrow{af} Q\{f/w\}} \quad \text{S2R} \frac{}{\text{Sync}(\bar{a}f \cdot \bar{b}g, af, \bar{b}g)}}{\bar{a}f.\bar{b}g.P|a(w).Q \xrightarrow{\bar{b}g} P|Q\{f/w\}} \\ \\ \text{SOut} \frac{\text{Out} \frac{}{\bar{b}g.P \xrightarrow{\bar{b}g} P}}{\bar{a}f.\bar{b}g.P \xrightarrow{\bar{a}f.\bar{b}g} P} \end{array}$$

5.2.2 Late operational semantic with structural congruence

The semantic of a multi π process is labeled transition system such that

- the nodes are multi π calculus process. The set of node is \mathbb{P}_m
- The set of actions is \mathbb{A}_m and can contain
 - bound output $\bar{x}(y)$
 - unbound output $\bar{x}y$
 - bound input $x(z)$

We use $\alpha, \alpha_1, \alpha_2, \dots$ to range over the set of actions, we use $\sigma, \sigma_1, \sigma_2, \dots$ to range over the set $\mathbb{A}_m^+ \cup \{\tau\}$.

- the transition relations is $\rightarrow \subseteq \mathbb{P}_m \times (\mathbb{A}_m^+ \cup \{\tau\}) \times \mathbb{P}_m$

In this case, a label can be a sequence of prefixes, whether in the original π calculus a label can be only a prefix. We use the symbol \cdot to denote the concatenation operator.

Definition 5.2.2. The *late transition relation with structural congruence* is the smallest relation induced by the rules in table 5.3:

In what follows, the names $\delta, \delta_1, \delta_2$ represents substitutions, they can also be empty; the names $\sigma, \sigma_1, \sigma_2, \sigma_3$ are non empty sequences of actions. The relation *Sync* is defined by the axioms in table 5.4

Example Transactional synchronization. This is an example of two processes that synchronize over a sequence of actions of length two:

$$\bar{a}x.\bar{a}y.P|a(w).a(z).Q \xrightarrow{\tau} P|Q\{x/w\}\{y/z\}$$

We start first noticing that

$$\text{S4R} \frac{\text{S1R} \frac{}{\text{Sync}(\bar{a}y, a(z)\{x/w\}, \tau, \{\}, \{y/z\})}}{\text{Sync}(\bar{a}x \cdot \bar{a}y, a(w) \cdot a(z), \tau, \{\}, \{x/w\}\{y/z\})}$$

and that

| | |
|--|---|
| Pref $\frac{\alpha \text{ not a strong prefix}}{\alpha.P \xrightarrow{\alpha} P}$ | Par $\frac{P \xrightarrow{\sigma} P' \quad bn(\sigma) \cap fn(Q) = \emptyset}{P Q \xrightarrow{\sigma} P' Q}$ |
| SOut $\frac{P \xrightarrow{\sigma} P' \quad \sigma \neq \tau}{\overline{x}y.P \xrightarrow{\overline{x}y.\sigma} P'}$ | LCom $\frac{P \xrightarrow{\sigma_1} P' \quad Q \xrightarrow{\sigma_2} Q' \quad Sync(\sigma_1, \sigma_2, \sigma_3, \delta_1, \delta_2)}{P Q \xrightarrow{\sigma_3} P'\delta_1 Q'\delta_2}$ |
| Sum $\frac{P \xrightarrow{\sigma} P'}{P + Q \xrightarrow{\sigma} P'}$ | Cong $\frac{P \equiv P' \quad P' \xrightarrow{\alpha} Q}{P \xrightarrow{\alpha} Q}$ |
| Res $\frac{P \xrightarrow{\sigma} P' \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\sigma} (\nu z)P'}$ | SInp $\frac{P \xrightarrow{\sigma} P' \quad \sigma \neq \tau}{\underline{x}(y).P \xrightarrow{x(y).\sigma} P'}$ |

Table 5.3: Multi π late semantic with structural congruence

| | |
|---|---|
| S1L $\overline{Sync(x(y), \overline{x}z, \tau, \{z/y\}, \{\})}$ | S1R $\overline{Sync(\overline{x}z, x(y), \tau, \{\}, \{z/y\})}$ |
| S2L $\overline{Sync(x(y), \overline{x}z \cdot \sigma, \sigma, \{z/y\}, \{\})}$ | S2R $\overline{Sync(\overline{x}z \cdot \sigma, x(y), \sigma, \{\}, \{z/y\})}$ |
| S3L $\overline{Sync(x(y) \cdot \sigma, \overline{x}z, \sigma\{z/y\}, \{z/y\}, \{\})}$ | S3R $\overline{Sync(\overline{x}z, x(y) \cdot \sigma, \sigma\{z/y\}, \{\}, \{z/y\})}$ |
| S4L $\frac{Sync(\sigma_1, \sigma_2\{z/y\}, \sigma_3, \delta_1, \delta_2)}{Sync(x(y) \cdot \sigma_1, \overline{x}z \cdot \sigma_2, \sigma_3, \{z/y\}\delta_1, \delta_2)}$ | S4R $\frac{Sync(\sigma_1, \sigma_2\{z/y\}, \sigma_3, \delta_1, \delta_2)}{Sync(\overline{x}z \cdot \sigma_1, x(y) \cdot \sigma_2, \sigma_3, \delta_1, \{z/y\}\delta_2)}$ |

Table 5.4: Synchronization relation

$$\begin{array}{c}
\text{SOUT} \frac{\text{PREF} \frac{\overline{a}y.P \xrightarrow{\overline{a}y} P}}{\underline{a}x.\overline{a}y.P \xrightarrow{\overline{a}x.\overline{a}y} P} \quad \text{SINP} \frac{\text{PREF} \frac{\overline{a}(z).Q \xrightarrow{a(z)} Q}}{a(w).\overline{a}(z).Q \xrightarrow{a(w).\overline{a}(z)} Q}
\end{array}$$

and in the end we just need to apply the rule **LCom**

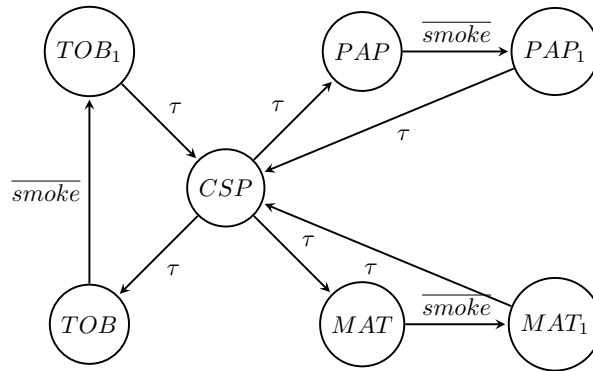
Example Multi-party synchronization. In this example we have three processes that want to synchronize:

$$\begin{array}{c}
\text{LCom} \frac{\overline{a}f.\overline{b}g.P|a(w).Q \xrightarrow{\overline{b}g} P|Q\{f/w\} \quad \text{Pref} \frac{\overline{b}(y).R \xrightarrow{b(y)} R} \quad \text{S1R} \frac{\overline{\text{Sync}}(\overline{b}g, b(y), \tau, \emptyset, \{g/y\})}{\text{Sync}(\overline{b}g, b(y), \tau, \emptyset, \{g/y\})}}{(\underline{a}f.\overline{b}g.P|a(w).Q)|b(y).R \xrightarrow{\tau} (P|Q\{f/w\})|R\{g/y\}} \\
\\
\text{LCom} \frac{\overline{a}f.\overline{b}g.P \xrightarrow{\overline{a}f.\overline{b}g} P \quad \text{Pref} \frac{\overline{a}(w).Q \xrightarrow{a(w)} Q} \quad \text{S2R} \frac{\overline{\text{Sync}}(\overline{a}f.\overline{b}g, a(w), \overline{b}g, \emptyset, \{f/w\})}{\text{Sync}(\overline{a}f.\overline{b}g, a(w), \overline{b}g, \emptyset, \{f/w\})}}{\underline{a}f.\overline{b}g.P|a(w).Q \xrightarrow{\overline{b}g} P|Q\{f/w\}} \\
\\
\text{SOut} \frac{\text{Out} \frac{\overline{b}g.P \xrightarrow{\overline{b}g} P}}{\underline{a}f.\overline{b}g.P \xrightarrow{\overline{a}f.\overline{b}g} P}
\end{array}$$

Example Cigarette smokers' problem. In this problem there are four processes: an agent and three smokers. Each smoker continuously makes a cigarette and smokes it. To make a cigarette each smoker needs three ingredients: tobacco, paper and matches. One of the smokers has paper, another tobacco and the third matches. The agent has an infinite supply of the ingredients. The agent places two of the ingredients on the table. The smoker who has the remaining ingredient take the others from the table, make a cigarette and smokes. Then the cycle repeats. A solution to the problem is the following:

$$\begin{aligned}
\text{Agent} &\stackrel{\text{def}}{=} \overline{\text{tob}}.\overline{\text{mat}}.\text{end}().\text{Agent} + \overline{\text{mat}}.\overline{\text{pap}}.\text{end}().\text{Agent} + \overline{\text{pap}}.\overline{\text{tob}}.\text{end}().\text{Agent} \\
S_{\text{pap}} &\stackrel{\text{def}}{=} \text{tob}().\text{mat}().\overline{\text{smoke}}.\text{end}.S_{\text{pap}} \\
S_{\text{tab}} &\stackrel{\text{def}}{=} \overline{\text{mat}}().\text{pap}().\overline{\text{smoke}}.\text{end}.S_{\text{tab}} \\
S_{\text{mat}} &\stackrel{\text{def}}{=} \overline{\text{pap}}().\text{tob}().\overline{\text{smoke}}.\text{end}.S_{\text{mat}} \\
\text{CSP} &\stackrel{\text{def}}{=} (\nu \text{tob}, \text{pap}, \text{mat}, \text{end})(\text{Agent}|S_{\text{tob}}|S_{\text{mat}}|S_{\text{pap}})
\end{aligned}$$

The semantic of *CSP* is the following graph:



where

$$\begin{aligned}
PAP &\stackrel{def}{=} (\nu tob, pap, mat, end)(end().Agent|S_{tob}|S_{mat}|\overline{smoke}.end.S_{pap}) \\
TOB &\stackrel{def}{=} (\nu tob, pap, mat, end)(end().Agent|\overline{smoke}.end.S_{tob}|S_{mat}|S_{pap}) \\
MAT &\stackrel{def}{=} (\nu tob, pap, mat, end)(end().Agent|S_{tob}|\overline{smoke}.end.S_{mat}|S_{pap}) \\
PAP_1 &\stackrel{def}{=} (\nu tob, pap, mat, end)(end().Agent|S_{tob}|S_{mat}|\overline{end}.S_{pap}) \\
TOB_1 &\stackrel{def}{=} (\nu tob, pap, mat, end)(end().Agent|\overline{end}.S_{tob}|S_{mat}|S_{pap}) \\
MAT_1 &\stackrel{def}{=} (\nu tob, pap, mat, end)(end().Agent|S_{tob}|\overline{end}.S_{mat}|S_{pap})
\end{aligned}$$

5.2.3 Low level semantic

This section contains the definition of an alternative semantic for multi π . First we define a low level version of the multi π calculus, we call this language low multi π . The low multi π is the multi π enriched with a marked or intermediate process $*P$:

$$P, Q ::= 0 \mid \pi.P \mid P|Q \mid P + Q \mid (\nu x)P \mid A(x_1, \dots, x_n) \mid *P$$

$$\pi ::= \overline{xy} \mid x(y) \mid \underline{\overline{xy}} \mid \underline{x(y)} \mid \tau$$

Definition 5.2.3. The low level transition relation is the smallest relation induced by the rules in table 5.5 in which P stands for a process without mark, L stands for a process with mark and S can stand for both.

Proposition 5.2.1. Let \rightarrow be the relation defined in table 5.1. If $P \xrightarrow{\sigma} Q$ then there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q \quad \text{and} \quad \gamma_1 \dots \gamma_{k+1} = \sigma$$

Proof. The proof is by induction on the depth of the derivation tree of $P \xrightarrow{\sigma} Q$:

base case

If the depth is one then the rule used have to be one of: *EInp*, *Out*, *Tau*. These rules are also in table 5.5 so we can derive $P \xrightarrow{\sigma} Q$.

inductive case

If the depth is greater than one then the last rule used in the derivation can be:

SOut : the last part of the derivation tree looks like this:

$$\mathbf{SOut} \frac{P_1 \xrightarrow{\sigma} Q \quad \sigma \neq \tau}{\underline{\overline{xy}}.P_1 \xrightarrow{\overline{xy} \cdot \sigma} Q}$$

for inductive hypothesis there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q \quad \text{and} \quad \gamma_1 \dots \gamma_{k+1} = \sigma$$

then a proof of the conclusion follows from:

$$\mathbf{SOutLow} \frac{}{\underline{\overline{xy}}.P_1 \xrightarrow{\overline{xy}} *P_1} \quad \mathbf{Star} \frac{P_1 \xrightarrow{\gamma_1} L_1}{*P_1 \xrightarrow{\gamma_1} L_1}$$

SInp : this case is similar to the previous.

Sum : the last part of the derivation tree looks like this:

$$\mathbf{Sum} \frac{P_1 \xrightarrow{\sigma} Q}{P_1 + P_2 \xrightarrow{\sigma} Q}$$

| | | |
|---|---|--|
| Out $\frac{}{\bar{x}y.P \mapsto P}$ | EInp $\frac{}{x(y).P \mapsto P\{z/y\}}$ | Tau $\frac{}{\tau.P \mapsto P}$ |
| SOutLow $\frac{}{\bar{x}y.P \mapsto *P}$ | SInpLow $\frac{}{\underline{x(y)}.P \mapsto *P\{z/y\}}$ | |
| StarEps $\frac{S \xrightarrow{\epsilon} S'}{*S \mapsto S'}$ | StarInp $\frac{S \xrightarrow{xy} S'}{*S \mapsto S'}$ | StarOut $\frac{S \xrightarrow{\bar{x}y} S'}{*S \mapsto S'}$ |
| Par1R $\frac{S \xrightarrow{\gamma} S'}{Q S \mapsto Q S'}$ | Par1L $\frac{S \xrightarrow{\gamma} S'}{S Q \mapsto S' Q}$ | |
| Sum $\frac{P \xrightarrow{\gamma} S}{P + Q \mapsto S}$ | Cong $\frac{P \equiv P' \quad P' \xrightarrow{\gamma} S}{P \mapsto S}$ | Res $\frac{S \xrightarrow{\gamma} S' \quad y \notin n(\gamma)}{(\nu y)S \mapsto (\nu y)S'}$ |
| Com1 $\frac{P \xrightarrow{\bar{x}y} P' \quad Q \xrightarrow{xy} Q'}{P Q \mapsto P' Q'}$ | | |
| Com2LOut $\frac{L_1 \xrightarrow{\bar{x}y} L'_1 \quad L_2 \xrightarrow{xy} S}{L_1 L_2 \mapsto L'_1 S}$ | Com2ROut $\frac{L_1 \xrightarrow{xy} S \quad L_2 \xrightarrow{\bar{x}y} L'_2}{L_1 L_2 \mapsto S L'_2}$ | |
| Com2LInp $\frac{L_1 \xrightarrow{\bar{x}y} S \quad L_2 \xrightarrow{xy} L'_2}{L_1 L_2 \mapsto S L'_2}$ | Com2RInp $\frac{L_1 \xrightarrow{xy} L'_1 \quad L_2 \xrightarrow{\bar{x}y} S}{L_1 L_2 \mapsto L'_1 S}$ | |
| Com3LOut $\frac{Q \xrightarrow{\bar{x}y} S \quad P \xrightarrow{xy} L}{Q P \mapsto S L}$ | Com3ROut $\frac{P \xrightarrow{xy} L \quad Q \xrightarrow{\bar{x}y} S}{P Q \mapsto L S}$ | |
| Com3LInp $\frac{Q \xrightarrow{xy} S \quad P \xrightarrow{\bar{x}y} L}{Q P \mapsto S L}$ | Com3RInp $\frac{P \xrightarrow{\bar{x}y} L \quad Q \xrightarrow{xy} S}{P Q \mapsto L S}$ | |
| Com4L $\frac{L_1 \xrightarrow{\bar{x}y} P \quad L_2 \xrightarrow{xy} Q}{L_1 L_2 \mapsto P Q}$ | Com4R $\frac{L_1 \xrightarrow{xy} P \quad L_2 \xrightarrow{\bar{x}y} Q}{L_1 L_2 \mapsto P Q}$ | |

Table 5.5: Low multi π early semantic with structural congruence

for the inductive hypothesis there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q \quad \text{and} \quad \gamma_1 \cdot \dots \cdot \gamma_{k+1} = \sigma$$

A proof of the conclusion is:

$$\mathbf{Sum} \frac{P_1 \xrightarrow{\gamma_1} L_1}{P_1 + P_2 \xrightarrow{\gamma_1} L_1}$$

Cong : this case is similar to the previous.

Res : the last part of the derivation tree looks like this:

$$\mathbf{Res} \frac{P_1 \xrightarrow{\sigma} Q_1 \quad z \notin n(\sigma)}{(\nu z)P_1 \xrightarrow{\sigma} (\nu z)Q_1}$$

for the inductive hypothesis there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} Q_1 \quad \text{and} \quad \gamma_1 \cdot \dots \cdot \gamma_{k+1} = \sigma$$

We can apply the rule *Res* to each of the previous transitions because

$$z \notin n(\sigma) \text{ implies } z \notin n(\gamma_i) \text{ for each } i$$

and then get a proof of the conclusion:

$$(\nu z)P_1 \xrightarrow{\gamma_1} (\nu z)L_1 \xrightarrow{\gamma_2} (\nu z)L_2 \cdots (\nu z)L_{k-1} \xrightarrow{\gamma_k} (\nu z)L_k \xrightarrow{\gamma_{k+1}} (\nu z)Q_1$$

Par : this case is similar to the previous.

ECom : the last part of the derivation tree looks like this:

$$\mathbf{ECom} \frac{P_1 \xrightarrow{\sigma_1} P'_1 \quad Q_1 \xrightarrow{\sigma_2} Q'_1 \quad ESync(\sigma_1, \sigma_2, \sigma_3)}{P_1|Q_1 \xrightarrow{\sigma_3} P'_1|Q'_1}$$

for inductive hypothesis there exist L_1, \dots, L_k and $\gamma_1, \dots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xrightarrow{\gamma_1} L_1 \xrightarrow{\gamma_2} L_2 \cdots L_{k-1} \xrightarrow{\gamma_k} L_k \xrightarrow{\gamma_{k+1}} P'_1 \quad \text{and} \quad \gamma_1 \cdot \dots \cdot \gamma_{k+1} = \sigma_1$$

and there exist R_1, \dots, R_h and $\delta_1, \dots, \delta_{h+1}$ with $h \geq 0$ such that

$$Q_1 \xrightarrow{\delta_1} R_1 \xrightarrow{\delta_2} R_2 \cdots R_{h-1} \xrightarrow{\delta_h} R_h \xrightarrow{\delta_{h+1}} Q'_1 \quad \text{and} \quad \delta_1 \cdot \dots \cdot \delta_{h+1} = \sigma_2$$

We proceed by cases on the derivation of $ESync(\sigma_1, \sigma_2, \sigma_3)$. We show just some cases because the others are similar.

S1L Suppose that δ_1 is $\bar{x}y$ (the other cases are similar), so the other δ s are ϵ or τ . We can have three different cases now each :

$\gamma_1 = xy$: The other γ s are ϵ or τ . A proof of the conclusion is:

$$P_1|Q_1 \xrightarrow{\epsilon} L_1|R_1 \xrightarrow{\epsilon} L_2|R_1 \cdots \xrightarrow{\epsilon} P'_1|R_1 \xrightarrow{\epsilon} P'_1|R_2 \cdots \xrightarrow{\epsilon} P'_1|Q'_1$$

we derive the first transition with rule *Com3ROut*, whether for the other transition we use the rules *Par1L*, *Par1R*, *Par3L* or *Par3R*.

$\gamma_i = xy$: A proof of the conclusion is:

$$P_1|Q_1 \xrightarrow{\epsilon} L_1|Q_1 \cdots \xrightarrow{\epsilon} L_{i-1}|Q_1 \xrightarrow{\tau} L_i|R_1 \xrightarrow{\epsilon} L_{i+1}|R_1 \cdots \xrightarrow{\epsilon} P'_1|R_1 \xrightarrow{\epsilon} P'_1|R_2 \cdots \xrightarrow{\epsilon} P'_1|Q'_1$$

we derive the transaction $L_{i-1}|Q_1 \xrightarrow{\tau} L_i|R_1$ with rule *Com5L*, whether for the other transactions we use some rule for parallel.

$\gamma_{k+1} = xy$ similar.

S2R : We suppose that $\delta_1 = xy$ and so other δ s are ϵ or τ , the other cases are similar. We can have two different cases now depending on where the first \bar{xy} is:

$\gamma_1 = \bar{xy}$: A proof of the conclusion is:

$$P_1|Q_1 \xrightarrow{\tau} L_1|R_1 \xrightarrow{\gamma_2} L_2|R_1 \cdots \xrightarrow{\gamma_{k+1}} P'_1|R_1 \xrightarrow{\delta_2} P'_1|R_2 \cdots \xrightarrow{\delta_{h+1}} P'_1|Q'_1$$

we derive the first transition with rule *Com3L*, whether for the other transactions we use some rule for parallel. Since $\gamma_1 \cdots \gamma_{k+1} = \bar{xy} \cdot \sigma$ and $\gamma_1 = \bar{xy}$ then $\tau \cdot \gamma_2 \cdots \gamma_{k+1} \cdot \epsilon \cdots \epsilon \cdot \tau = \sigma$

$\gamma_i = \bar{xy}$: A proof of the conclusion is:

$$P_1|Q_1 \xrightarrow{\epsilon} L_1|Q_1 \cdots \xrightarrow{\epsilon} L_{i-1}|Q_1 \xrightarrow{\tau} L_i|R_1 \xrightarrow{\gamma_{i+1}} L_{i+1}|R_1 \cdots \xrightarrow{\gamma_k} P'_1|R_1 \xrightarrow{\delta_2} P'_1|R_2 \cdots \xrightarrow{\delta_{h+1}} P'_1|Q'_1$$

we derive the transition $L_{i-1}|Q_1 \xrightarrow{\tau} L_i|Q'_1$ with rule *Com2L*, whether for the other transactions of the premises we use the rule *Par1L*.

$\gamma_{k+1} = \bar{xy}$: cannot happen because σ is not empty.

S4R We have three cases: $|\sigma_1| = |\sigma_2|$, $|\sigma_1| > |\sigma_2|$ or $|\sigma_2| > |\sigma_1|$. In the first case $|\sigma_3|$ must be τ and we can build a chain of transition as in the previous cases. In the second case there is a prefix of σ_1 which synchronize with σ_2 and σ_3 is the rest of σ_1 , in this case we can also build a chain of transition as in the previous cases. The third case is symmetric to the second.

□

The converse of lemma 5.2.1 does not hold because the low semantic allow to express interleaving behaviour. But there is the following weaker result:

Proposition 5.2.2. Let \rightarrow be the relation defined in table 5.1, let α be an action and P, Q be processes. If $P \xrightarrow{\alpha} Q$ then $P \xrightarrow{\alpha} Q$.

Proof. The proof is an easy induction on the proof tree of $P \xrightarrow{\alpha} Q$.

□

Bibliography

- [1] Roberto Gorrieri, Cristian Versari, *Multi π : a calculus for mobile multi-party and transactional communication*.
- [2] Robin Milner, Joachim Parrow, David Walker, *A calculus of mobile processes, part II*, 1990.
- [3] Roberto Gorrieri, *A fully-abstract semantics for atomicity*, Dipartimento di scienze dell'informazione, Università di Bologna.
- [4] Joachim Parrow, *An introduction to the π calculus*, Department Teleinformatics, Rotal Institute of Technology, Stockholm.
- [5] Davide Sangiorgi, David Walker, *The π -calculus*, Cambridge University Press.
- [6] Davide Sangiorgi, *A theory of bisimulation for the π -calculus*, Acta informatica, 33(1):69-97, 1996.
- [7] Milner, Robin, *Communicating and mobile systems: the π -calculus*, Cambridge University Press.
- [8] MohammedReza Mousavi, Michel A Reniers, *Congruence for structural congruences*, Department of Computer Science, Eindhoven University of Technology.