Universita' di Bologna

Facolta' di scienze matematiche fisiche e naturali

Corso di laurea magistrale in Scienze Informatiche

Tesi di laurea

Multi π calcolo

Candidato:
Federico Viscomi

Tutore
Prof. Roberto Gorrieri

. . . .

./pi.jpg

Anno accademico 20011/2012

0.1 Abstract

Il π calcolo e' un formalismo che descrive e analizza le proprieta' del calcolo concorrente. Nasce come proseguio del lavoro gia' svolto sul CCS (Calculus of Communicating Systems). L'aspetto appetibile del π calcolo rispetto ai formalismi precedenti e' l'essere in grado di descrivere la computazione concorrente in sistemi la cui configurazione puo' cambiare nel tempo. Nel CCS e nel π calcolo manca la possibilta' di modellare sequenze atomiche di azioni e di modellare la sincronizzazione multiparte. Il Multi CCS [5] estende il CCS con un'operatore di strong prefixing proprio per colmare tale vuoto. In questa tesi si cerca di trasportare per analogia le soluzioni introdotte dal Multi CCS verso il π calcolo. Il risultato finale e' un linguaggio chiamato Multi π calcolo.

In particolare il Multi π calcolo permette la sincronizzazione transazionale e la sincronizzazione multiparte. aggiungere una sintesi brevissima dei risultati ottenuti sul Multi π calcolo.

Contents

Chapter 1

TODO

- dimostrare(o negare) l'equivalenza del pi calcolo con e senza congruenza strutturale
- 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
- fare un quadro generale sulle equivalenze nel pi calcolo
- scegilere una equivalenza(forse la open va bene) per multi pi calcolo(quale?) che sia una congruenza per input(ma non lo sara' per il parallelo)
- trovare equivalenza che sia una congruenza(es: open step) per tutti gli operatori
- trovare la congruenza coarsest contenuta nella bisimulazione scelta in precedenza

perche' nella regola originale sull'input nel pi calcolo early c'e' quella premessa?

- stampare e rileggere
- riscrivere tutte le regole usando inferrule
- aggiungere la definizione precisa induttiva di alfa conversione anche per il multipi
- aggiungere a tutti i pi e multi pi la derivazione nella semantica late di
  - $(x(z).P|Q)|\overline{x}y.R \xrightarrow{\tau} (P\{w/z\}|Q)\{y/w\}|R$
  - $y = z$ or $y \notin fn(Q)$: $((\nu y)\overline{x}y.P)|x(z).Q \xrightarrow{\tau} (\nu y)(P|Q\{y/z\})$
  - $y \neq z$ and $y \in fn(Q)$ $y^{'}$ fresh: $((\nu y)\overline{x}y.P)|x(z).Q \xrightarrow{\tau} (\nu y^{'})(P\{y^{'}/y\}|Q\{y^{'}/z\})$
- le side condition delle par servono solo nella late? mi sembra di si
- sistemare index: o lo tolgo completamente o lo metto a tutte le definizioni.
- cercare di equilibrare l'importanza del contenuto con gli spazi usati
- qual'e' l'ortografia corretta di multi-parti?
- ci sono delle parti che sono uguali in multipioutput e multipiinput. fattorizzare.
- cosa succede nella regola opnseq di multiout se sigma contiene $\overline{z}u$? per adesso ho deciso di eliminarle perche' $(\nu z)(\underline{\overline{x}z}.\underline{\overline{x}y}.\overline{x}z.0) \xrightarrow{\overline{x}(z).\overline{x}y.\overline{x}(z)} 0$ non sembra essere derivabile in multioutlow. aggiungere un paragrafo alla fine che spiega quasta situazione ed eventualmente propone una soluzione con relative dimostrazioni.

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 communications 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 [6], [7] and [9]

2.1 Syntax

We suppose that we have a countable set of names $\mathbb{N}$, ranged over by lower case letters $a, b, \cdots, z$. 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 $P, Q, \cdots$. A process can perform the following actions:

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

The process are defined by the following grammar:

$$P, Q ::= 0 \mid \pi.P \mid P|Q \mid P + Q \mid (\nu x)P \mid A(x_1, \cdots, x_n)$$

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:

- $\overline{x}y$ 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

$$B(0,I) \;=\; \emptyset \qquad B(Q+R,I) \;=\; B(Q,I)\cup B(R,I)$$

$$B(\overline{x}y.Q,I) \;=\; B(Q,I) \qquad B(Q|R,I) \;=\; B(Q,I)\cup B(R,I)$$

$$B(x(y).Q,I) \;=\; \{y,\overline{y}\}\cup B(Q,I) \qquad B((\nu x)Q,I) \;=\; \{x,\overline{x}\}\cup B(Q,I)$$

$$B(\tau.Q,I) \;=\; B(Q,I)$$

$$B(A(\tilde{x}),I) = \begin{cases} B(Q,I\cup\{A\})\ where\ A(\tilde{x}) \stackrel{def}{=} Q & if\ A\notin I \\ \emptyset & if\ A\in I \end{cases}$$

Table 2.1: Bound occurrences

$$fn(\overline{x}y.Q) \;=\; \{x,\overline{x},y,\overline{y}\}\cup fn(Q) \qquad fn(Q+R) \;=\; fn(Q)\cup fn(R) \qquad fn(0) \;=\; \emptyset$$

$$fn(x(y).Q) \;=\; \{x,\overline{x}\}\cup(fn(Q)-\{y,\overline{y}\}) \qquad fn(Q|R) \;=\; fn(Q)\cup fn(R)$$

$$fn((\nu x)Q) \;=\; fn(Q)-\{x,\overline{x}\} \qquad fn(\tau.Q) \;=\; fn(Q) \qquad fn(A(\tilde{x})) = \{\tilde{x}\}$$

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.

$A(x_1,\cdots,x_n)$ is an identifier. Every identifier has a definition

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

the x_is must be pairwise different. The intuition is that we can substitute for some of the x_is in P to get a π calculus process.

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 $xy,\overline{x}y$ are *free* actions. Whether the other actions are *bound* actions.

In a definition

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

$$0\{b/a\} \;=\; 0$$

$$(\overline{x}y.Q)\{b/a\} \;=\; \overline{x}\{b/a\}y\{b/a\}.Q\{b/a\}$$

$$(x(y).Q)\{b/a\} \;=\; x\{b/a\}(y).Q\{b/a\} \text{ if } y \neq a \text{ and } y \neq b$$

$$(x(a).Q)\{b/a\} \;=\; x\{b/a\}(a).Q$$

$$(x(b).Q)\{b/a\} \;=\; x\{b/a\}(c).((Q\{c/b\})\{b/a\}) \text{ where } c \notin n(Q)$$

$$(\tau.Q)\{b/a\} \;=\; \tau.Q\{b/a\}$$

if $a \in \{x_1, \cdots, x_n\}$ then
$$(A(x_1, \cdots, x_n \mid y_1, \cdots, y_m))\{b/a\} =$$
$$\begin{cases} A(x_1\{b/a\}, \cdots, x_n\{b/a\} \mid y_1, \cdots, y_m) & if\ b \notin \{y_1, \cdots, y_m\} \\ A(x_1\{b/a\}, \cdots, x_n\{b/a\} \mid y_1, \cdots, y_{i-1}, c, y_{i+1}, \cdots, y_m) & if\ b = y_i \\ \quad where\ c\ is\ fresh & \end{cases}$$

if $a \notin \{x_1, \cdots, x_n\}$ then
$$(A(x_1, \cdots, x_n \mid y_1, \cdots, y_m))\{b/a\} = A(x_1, \cdots, x_n \mid y_1, \cdots, y_m)$$

$$(Q + R)\{b/a\} \;=\; Q\{b/a\} + R\{b/a\}$$

$$(Q|R)\{b/a\} \;=\; Q\{b/a\}|R\{b/a\}$$

$$((\nu y)Q)\{b/a\} \;=\; (\nu y)Q\{b/a\} \text{ if } y \neq a \text{ and } y \neq b$$

$$((\nu a)Q)\{b/a\} \;=\; (\nu a)Q$$

$$((\nu b)Q)\{b/a\} \;=\; (\nu c)((Q\{c/b\})\{b/a\}) \text{ where } c \notin n(Q) \text{ if } a \in fn(Q)$$

$$((\nu b)Q)\{b/a\} \;=\; (\nu b)Q \text{ if } a \notin fn(Q)$$

Table 2.3: Syntatic substitution

the $x_1, \cdots, x_n$ are all the free names contained in P, specifically

$$fn(P) \subseteq \{x_1, \cdots, x_n\}$$

If we look at the definitions of bn and of fn we notice that if P contains another identifier whose definition is:

$$B(z_1, \cdots, z_h) = Q$$

then we have

$$fn(Q) \subseteq \{x_1, \cdots, x_n\}$$

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. There is the following short notation

$$\{\tilde{x}/\tilde{y}\} \text{ means } \{x_1/y_1, \cdots, x_n/y_n\}$$

$$\textbf{Out}\ \frac{}{\overline{x}y.P \xrightarrow{\overline{x}y} P} \qquad \textbf{EInp}\ \frac{}{x(y).P \xrightarrow{xz} P\{z/y\}}$$

$$\textbf{SumR}\ \frac{Q \xrightarrow{\alpha} Q'}{P+Q \xrightarrow{\alpha} Q'} \qquad \textbf{ParR}\ \frac{Q \xrightarrow{\alpha} Q' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P|Q \xrightarrow{\alpha} P|Q'}$$

$$\textbf{SumL}\ \frac{P \xrightarrow{\alpha} P'}{P+Q \xrightarrow{\alpha} P'} \qquad \textbf{ParL}\ \frac{P \xrightarrow{\alpha} P' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P|Q \xrightarrow{\alpha} P'|Q}$$

$$\textbf{Res}\ \frac{P \xrightarrow{\alpha} P' \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha} (\nu z)P'} \qquad \textbf{ResAlp}\ \frac{(\nu w)P\{w/z\} \xrightarrow{xz} P' \quad w \notin n(P)}{(\nu z)P \xrightarrow{xz} P'}$$

$$\textbf{EComR}\ \frac{P \xrightarrow{\overline{x}y} P' \quad Q \xrightarrow{xy} Q'}{P|Q \xrightarrow{\tau} P'|Q'} \qquad \textbf{ClsL}\ \frac{P \xrightarrow{\overline{x}(z)} P' \quad Q \xrightarrow{xz} Q' \quad z \notin fn(Q)}{P|Q \xrightarrow{\tau} (\nu z)(P'|Q')}$$

$$\textbf{EComL}\ \frac{P \xrightarrow{xy} P' \quad Q \xrightarrow{\overline{x}y} Q'}{P|Q \xrightarrow{\tau} P'|Q'} \qquad \textbf{ClsR}\ \frac{P \xrightarrow{xz} P' \quad Q \xrightarrow{\overline{x}(z)} Q' \quad z \notin fn(P)}{P|Q \xrightarrow{\tau} (\nu z)(P'|Q')}$$

$$\textbf{Tau}\ \frac{}{\tau.P \xrightarrow{\tau} P} \qquad \textbf{Cns}\ \frac{A(\tilde{x}) \stackrel{def}{=} P \quad P\{\tilde{w}/\tilde{x}\} \xrightarrow{\alpha} P'}{A(\tilde{x})\{\tilde{w}/\tilde{x}\} \xrightarrow{\alpha} P'}$$

$$\textbf{Opn}\ \frac{P \xrightarrow{\overline{x}z} P' \quad z \neq x}{(\nu z)P \xrightarrow{\overline{x}(z)} P'} \qquad \textbf{OpnAlp}\ \frac{(\nu w)P\{w/z\} \xrightarrow{\overline{x}(w)} P' \quad w \notin n(P) \quad x \neq w \neq z.}{(\nu z)P \xrightarrow{\overline{x}(w)} P'}$$

Table 2.4: Early transition relation without structural congruence

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 $\overline{x}y$
  - the silent action τ
  - bound output $\overline{x}(y)$

  The set of actions is $\mathbb{A}$, we use α to range over the set of actions.
- the transition relations is $\rightarrow \subseteq \mathbb{P} \times \mathbb{A} \times \mathbb{P}$

In the following section we present the early semantic without structural congruence and without *alpha* conversion.

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. Where with $\tilde{x}$ we mean a sequence of names $x_1, \cdots, x_n$.

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

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

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.

$$\textsc{CloseR}\ \dfrac{\textsc{EInp}\ \dfrac{}{a(x).P \xrightarrow{ab} P\{b/x\}} \qquad \textsc{Opn}\ \dfrac{\textsc{Out}\ \dfrac{}{\overline{a}b.Q \xrightarrow{\overline{a}b} Q} \qquad a \neq b}{(\nu b)\overline{a}b.Q \xrightarrow{\overline{a}(b)} Q} \qquad b \notin fn((\nu b)\overline{a}b.Q)}{a(x).P \mid (\nu b)\overline{a}b.Q \xrightarrow{\tau} (\nu b)(P\{b/x\} \mid Q)}$$

Example We want to prove now that:

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

where $b \notin bn(P)$

$$\textsc{ResAlp}\ \dfrac{\textsc{Res}\ \dfrac{\textsc{EInp}\ \dfrac{}{(a(x).P)\{c/b\} \xrightarrow{ab} P\{c/b\}\{b/x\}} \qquad c \notin n(a(b))}{(\nu c)((a(x).P)\{c/b\}) \xrightarrow{ab} (\nu c)(P\{c/b\}\{b/x\})} \qquad b \notin n((a(x).P)\{c/b\})}{(\nu b)a(x).P \xrightarrow{ab} (\nu c)P\{c/b\}\{b/x\}}$$

$$\textsc{EComL}\ \dfrac{(\nu b)a(x).P \xrightarrow{ab} (\nu c)P\{c/b\}\{b/x\} \qquad \textsc{EOut}\ \dfrac{}{\overline{a}b.Q \xrightarrow{\overline{a}b} Q}}{((\nu b)a(x).P) \mid \overline{a}b.Q \xrightarrow{\tau} ((\nu c)(P\{c/b\}\{b/x\}))|Q}$$

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

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

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 we have to prove

$$A(y) \xrightarrow{yw} y(a).0$$

We can prove this in the following way:

$$\textsc{Cns}\ \dfrac{A(x) \stackrel{def}{=} P \qquad \textsc{EInp}\ \dfrac{}{P\{y/x\} \xrightarrow{yw} y(a).0}}{A(y) \xrightarrow{yw} y(a).0}$$

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 $\overline{x}y$
- the silent action τ
- bound output $\overline{x}(y)$

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. TUTTE LE SEMANTICHE LATE DEL PI CALCOLO SONO DA AGGIORNARE!!!! !!! !! !

LInp $\dfrac{z \notin fn(P)}{x(y).P \xrightarrow{x(z)} P\{z/y\}}$	**Res** $\dfrac{P \xrightarrow{\alpha} P' \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha} (\nu z)P'}$
SumL $\dfrac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'}$	**SumR** $\dfrac{Q \xrightarrow{\alpha} Q'}{P + Q \xrightarrow{\alpha} Q'}$
ParL $\dfrac{P \xrightarrow{\alpha} P' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P \vert Q \xrightarrow{\alpha} P' \vert Q}$	**ParR** $\dfrac{Q \xrightarrow{\alpha} Q' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P \vert Q \xrightarrow{\alpha} P \vert Q'}$
ComL $\dfrac{P \xrightarrow{x(y)} P' \quad Q \xrightarrow{\overline{x}(z)} Q'}{P \vert Q \xrightarrow{\tau} P'\{z/y\} \vert Q'}$	**ComR** $\dfrac{P \xrightarrow{\overline{x}(z)} P' \quad Q \xrightarrow{x(y)} Q'}{P \vert Q \xrightarrow{\tau} P' \vert Q'\{z/y\}}$
Opn $\dfrac{P \xrightarrow{\overline{x}z} P' \quad z \neq x}{(\nu z)P \xrightarrow{\overline{x}(z)} P'}$	**Out** $\dfrac{}{\overline{x}y.P \xrightarrow{\overline{x}y} P}$
ClsL $\dfrac{P \xrightarrow{\overline{x}(z)} P' \quad Q \xrightarrow{xz} Q' \quad z \notin fn(Q)}{P \vert Q \xrightarrow{\tau} (\nu z)(P' \vert Q')}$	**ClsR** $\dfrac{P \xrightarrow{xz} P' \quad Q \xrightarrow{\overline{x}(z)} Q' \quad z \notin fn(P)}{P \vert Q \xrightarrow{\tau} (\nu z)(P' \vert Q')}$
Tau $\dfrac{}{\tau.P \xrightarrow{\tau} P}$	**Cns** $\dfrac{A(\tilde{x}) \overset{def}{=} P \quad P\{\tilde{y}/\tilde{x}\} \xrightarrow{\alpha} P'}{A(\tilde{y}) \xrightarrow{\alpha} P'}$

Table 2.5: Late semantic without structural congruence

2.2.3 Distinction between late and early semantics

There are some differences between late and early semantics:

Communication da scrivere

Input da scrivere

Parallel composition the side condition in the rule Par for the late sematic is important because:
$(x(z).P|Q)|\overline{x}y.R \xrightarrow{\tau} (P\{w/z\}|Q)\{y/w\}|R$
da scrivere

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, 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, than we obtain $C[P]$. In doing so, we make no α conversions.

$$\textsc{AlpSum}\ \frac{P_1 \equiv_\alpha Q_1 \qquad P_2 \equiv_\alpha Q_2}{P_1 + P_2 \equiv_\alpha Q_1 + Q_2} \qquad \textsc{AlpTau}\ \frac{P \equiv_\alpha Q}{\tau.P \equiv_\alpha \tau.Q}$$

$$\textsc{AlpRes1}\ \frac{P\{y/x\} \equiv_\alpha Q \qquad x \neq y \qquad y \notin fn(P)}{(\nu x)P \equiv_\alpha (\nu y)Q} \qquad \textsc{AlpRes}\ \frac{P \equiv_\alpha Q}{(\nu x)P \equiv_\alpha (\nu x)Q}$$

$$\textsc{AlpInp1}\ \frac{P\{y/x\} \equiv_\alpha Q \qquad x \neq y \qquad y \notin fn(P)}{z(x).P \equiv_\alpha z(y).Q} \qquad \textsc{AlpInp}\ \frac{P \equiv_\alpha Q}{x(y).P \equiv_\alpha x(y).Q}$$

$$\textsc{AlpPar}\ \frac{P_1 \equiv_\alpha Q_1 \qquad P_2 \equiv_\alpha Q_2}{P_1|P_2 \equiv_\alpha Q_1|Q_2} \qquad \textsc{AlpOut}\ \frac{P \equiv_\alpha Q}{\overline{x}y.P \equiv_\alpha \overline{x}y.Q}$$

$$\textsc{AlpIde}\ \frac{}{A(\tilde{x}|\tilde{y}) \equiv_\alpha A(\tilde{x}|\tilde{y})} \qquad \textsc{AlpZero}\ \frac{}{0 \equiv_\alpha 0}$$

Table 2.6: α equivalence laws

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

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.

Lemma 2.3.1. Inversion lemma for α equivalence

- If $P \equiv_\alpha 0$ then P is also the null process 0
- If $P \equiv_\alpha \tau.Q_1$ then $P = \tau.P_1$ for some P_1 such that $P_1 \equiv_\alpha Q_1$
- If $P \equiv_\alpha \overline{x}y.Q_1$ then $P = \overline{x}y.P_1$ for some P_1 such that $P_1 \equiv_\alpha Q_1$
- If $P \equiv_\alpha z(y).Q_1$ then one and only one of the following cases holds:
  - $P = z(x).P_1$ for some P_1 such that $P_1\{y/x\} \equiv_\alpha Q_1$
  - $P = z(y).P_1$ for some P_1 such that $P_1 \equiv_\alpha Q_1$
- If $P \equiv_\alpha Q_1 + Q_2$ then $P = P_1 + P_2$ for some P_1 and P_2 such that $P_1 \equiv_\alpha Q_1$ and $P_2 \equiv_\alpha Q_2$.
- If $P \equiv_\alpha Q_1|Q_2$ then $P = P_1|P_2$ for some P_1 and P_2 such that $P_1 \equiv_\alpha Q_1$ and $P_2 \equiv_\alpha Q_2$.
- If $P \equiv_\alpha (\nu y)Q_1$ then one and only one of the following cases holds:
  - $P = (\nu x)P_1$ such that $P_1\{y/x\} \equiv_\alpha Q_1$
  - $P = (\nu y).P_1$ for some P_1 such that $P_1 \equiv_\alpha Q_1$
- If $P \equiv_\alpha A(\tilde{x})$ then P is Q.

SC-ALP	$\dfrac{P \equiv_\alpha Q}{P \equiv Q}$	α conversion
abelian monoid laws for sum:		
SC-SUM-ASC	$M_1 + (M_2 + M_3) \equiv (M_1 + M_2) + M_3$	associativity
SC-SUM-COM	$M_1 + M_2 \equiv M_2 + M_1$	commutativity
SC-SUM-INC	$M + 0 \equiv M$	zero element
abelian monoid laws for parallel:		
SC-COM-ASC	$P_1 \| (P_2 \| P_3) \equiv (P_1 \| P_2) \| P_3$	associativity
SC-COM-COM	$P_1 \| P_2 \equiv P_2 \| P_1$	commutativity
SC-COM-INC	$P \| 0 \equiv P$	zero element
scope extension laws:		
SC-RES	$(\nu z)(\nu w)P \equiv (\nu w)(\nu z)P$	
SC-RES-INC	$(\nu z)0 \equiv 0$	
SC-RES-COM	$(\nu z)(P_1 \| P_2) \equiv P_1 \| (\nu z)P_2$ if $z \notin fn(P_1)$	
SC-RES-SUM	$(\nu z)(P_1 + P_2) \equiv P_1 + (\nu z)P_2$ if $z \notin fn(P_1)$	
unfolding law:		
SC-IDE	$A(\tilde{w}) \equiv P\{\tilde{w}/\tilde{x}\}$	if $A(\tilde{x}) \stackrel{def}{=} P$

Table 2.7: Structural congruence axioms

Proof. This lemma works because given Q we know which rules must be at the end of any proof tree of $P \equiv_\alpha Q$. □

Lemma 2.3.2. Let P be a process and y, w, z names such that $w = z$ or $w \notin fn(P)$ then $P\{w/z\}\{y/w\} \equiv_\alpha P$ non ho una dimostrazione ma lo da per scontato in [4] paragrafo 1.3.1

Definition 2.3.5. We define a *structural congruence* $\equiv$ as the smallest congruence on processes that satisfies the axioms in table 2.7

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 SC-ALP or we can embed the α conversion in the SOS rules. In the early case, the rule for input and the rules $ResAlp, OpnAlp, Cns$ take care of α conversion, whether in the late case the rule for communication and the rules is $ResAlp, OpnAlp, Cns$ 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:

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

or by the following rule and axiom:

$$\textbf{Sum}\ \frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'} \quad \textbf{SC-SUM}\quad P + Q \equiv Q + P$$

and the rule Str

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.

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

Proof. □

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.

$AlpTau$:

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

$AlpOut$:

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

$AlpRes1$: we consider two cases:

$x \notin fn(P)$:

$$\begin{array}{ll} & \text{rule premises} \\ P\{y/x\} \equiv_\alpha Q \text{ and } y \notin fn(P) & \text{inductive hypothesis} \\ \Rightarrow fn(P\{y/x\}) = fn(Q) & x \notin fn(P) \text{ and def of substitution} \\ \Rightarrow fn(P) = fn(Q) & y \notin fn(P) \\ \Rightarrow fn(P) = fn(Q) \text{ and } y \notin fn(Q) & \end{array}$$

Since $x \notin fn(P)$ then $fn(P) = fn(P) - \{x\}$. Since $y \notin fn(Q)$ then $fn(Q) = fn(Q) - \{y\}$. From $fn(P) = fn(P) - \{x\}$, $fn(Q) = fn(Q) - \{y\}$, $fn(P) = fn(Q)$ and the definition of substitution it follows that $fn((\nu x)P) = fn((\nu y)Q)$

$x \in fn(P)$:

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

$AlpInp1$: we consider two cases:

$x \notin fn(P)$:

$$\begin{array}{ll} & \text{rule premises} \\ P\{y/x\} \equiv_\alpha Q \text{ and } y \notin fn(P) & \text{inductive hypothesis} \\ \Rightarrow fn(P\{y/x\}) = fn(Q) & x \notin fn(P) \text{ and def of substitution} \\ \Rightarrow fn(P) = fn(Q) & y \notin fn(P) \\ \Rightarrow fn(P) = fn(Q) \text{ and } y \notin fn(Q) & \end{array}$$

Since $x \notin fn(P)$ then $fn(P) = fn(P) - \{x\}$. Since $y \notin fn(Q)$ then $fn(Q) = fn(Q) - \{y\}$. From $fn(P) = fn(P) - \{x\}$, $fn(Q) = fn(Q) - \{y\}$ and $fn(P) = fn(Q)$ it follows that $fn(P) - \{x\} = fn(Q) - \{y\}$ and so $(fn(P) - \{x\}) \cup \{z\} = (fn(Q) - \{y\}) \cup \{z\}$ which gives $fn(z(x).P) = fn(z(y).Q)$.

$x \in fn(P)$:

$$
\begin{array}{ll}
 & \text{rule premise} \\
P\{y/x\} \equiv_\alpha Q & \text{inductive hypothesis} \\
\Rightarrow fn(P\{y/x\}) = fn(Q) & \\
\Rightarrow fn(P) - \{y\} = fn(Q) - \{y\} & \text{lemma 2.3.3} \\
\Rightarrow (fn(P) - \{x\} \cup \{y\}) - \{y\} = fn(Q) - \{y\} & \\
\Rightarrow fn(P) - \{x\} = fn(Q) - \{y\} & \\
\Rightarrow (fn(P) - \{x\}) \cup \{z\} = (fn(Q) - \{y\}) \cup \{z\} & \text{definition of } fn \\
\Rightarrow fn(z(x).P) = fn(z(y).Q) &
\end{array}
$$

$AlpSum$:

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

$AlpPar$:

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

$AlpRes$:

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

$AlpInp$:

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

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

□

Lemma 2.3.5. $x \notin fn(P) \Rightarrow P\{x/y\}\{b/a\} \equiv_\alpha P\{b/a\}\{x/y\}$

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

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

Proof. : If a and b are the same name then the substitution has no effect and the lemma holds. Otherwise:

$$\begin{array}{ll} & \text{lemma hypothesis} \\ P \equiv_\alpha Q & \text{lemma 2.3.4} \\ \Rightarrow fn(P) = fn(Q) & \\ \Rightarrow a \notin fn(P) \wedge a \notin fn(Q) \text{ or } a \in fn(P) \wedge a \in fn(Q) & \end{array}$$

In the former case a is not a free name in P and Q so the substitutions have no effects and the lemma holds. In the latter case a is a free names in both processes: the proof goes by induction on the length of the proof tree of $P \equiv_\alpha Q$ and then by cases on the last rule of the proof tree. Let x, y, a and b be pairwise different.

base case The length of the proof is one and the rule used can be only: *AlpZero* or *AlpIde*: the lemma holds because P and Q are syntacticly the same process.

inductive case :

AlpTau :

$$\begin{array}{ll} & \text{rule premise} \\ P_1 \equiv_\alpha Q_1 & \text{inductive hypothesis} \\ \Rightarrow P_1\{b/a\} \equiv_\alpha Q_1\{b/a\} & \text{rule } \mathit{AlpTau} \\ \Rightarrow \tau.(P_1\{b/a\}) \equiv_\alpha \tau.(Q_1\{b/a\}) & \text{definition of substitution} \\ \Rightarrow (\tau.P_1)\{b/a\} \equiv_\alpha (\tau.Q_1)\{b/a\} & \end{array}$$

AlpSum :

$$\begin{array}{ll} & \text{rule premises} \\ P_1 \equiv Q_1 \text{ and } P_2 \equiv Q_2 & \text{inductive hypothesis} \\ \Rightarrow P_1\{b/a\} \equiv Q_1\{b/a\} \text{ and } P_2\{b/a\} \equiv Q_2\{b/a\} & \text{rule } \mathit{AlpSum} \\ \Rightarrow P_1\{b/a\} + P_2\{b/a\} \equiv Q_1\{b/a\} + Q_2\{b/a\} & \text{definition of substitution} \\ \Rightarrow (P_1 + P_2)\{b/a\} \equiv_\alpha (Q_1 + Q_2)\{b/a\} & \end{array}$$

AlpPar : this case is very similar to the previous one.

AlpOut :

$$\begin{array}{ll} & \text{rule premise} \\ P_1 \equiv_\alpha Q_1 & \text{inductive hypothesis} \\ \Rightarrow P_1\{b/a\} \equiv_\alpha Q_1\{b/a\} & \text{rule } \mathit{AlpOut} \\ \Rightarrow \overline{x}\{b/a\}y\{b/a\}.P_1\{b/a\} \equiv_\alpha \overline{x}\{b/a\}y\{b/a\}.Q_1\{b/a\} & \text{definition of substitution} \\ \Rightarrow (\overline{x}y.P_1)\{b/a\} \equiv_\alpha (\overline{x}y.Q_1)\{b/a\} & \end{array}$$

AlpInp :

$$\begin{array}{ll} & \text{rule premise} \\ P_1 \equiv_\alpha Q_1 & \text{inductive hypothesis} \\ \Rightarrow P_1\{b/a\} \equiv_\alpha Q_1\{b/a\} & \text{rule } \mathit{AlpInp} \\ \Rightarrow x\{b/a\}(y).P_1\{b/a\} \equiv_\alpha x\{b/a\}(y).Q_1\{b/a\} & \text{definition of substitution} \\ \Rightarrow (x(y).P_1)\{b/a\} \equiv_\alpha (x(y).Q_1)\{b/a\} & \end{array}$$

$$\begin{array}{ll} & \text{rule premise} \\ P_1 \equiv_\alpha Q_1 & \text{rule } \mathit{AlpIn} \\ \Rightarrow b(a).P_1 \equiv_\alpha b(a).Q_1 & \text{definition of substitution} \\ \Rightarrow a\{b/a\}(a).P_1 \equiv_\alpha a\{b/a\}(a).Q_1 & \text{definition of substitution} \\ \Rightarrow (a(a).P_1)\{b/a\} \equiv_\alpha (a(a).Q_1)\{b/a\} & \end{array}$$

$$\begin{array}{ll} & \text{rule premise} \\ P_1 \equiv_\alpha Q_1 & \text{inductive hypothesis} \\ \Rightarrow P_1\{b/a\} \equiv_\alpha Q_1\{b/a\} & \text{rule } \mathit{AlpIn} \\ \Rightarrow b\{b/a\}(x).(P_1\{b/a\}) \equiv_\alpha b\{b/a\}(x).(Q_1\{b/a\}) & \text{definition of substitution} \\ \Rightarrow (b(x).P_1)\{b/a\} \equiv_\alpha (b(x).Q_1)\{b/a\} & \end{array}$$

$AlpInp1$: we have various cases:

- the last part of the proof tree of $P \equiv_\alpha Q$ is

$$\textsc{AlpInp1}\ \frac{P_1 \equiv_\alpha Q_1\{x/y\} \qquad x \neq y \qquad x \notin fn(Q_1)}{\underbrace{z(x).P_1}_{P} \equiv_\alpha \underbrace{z(y).Q_1}_{Q}}$$

	rule premise
$P_1 \equiv_\alpha Q_1\{x/y\}$ and $x \notin fn(Q_1)$	inductive hypothesis
$\Rightarrow P_1\{b/a\} \equiv_\alpha Q_1\{x/y\}\{b/a\}$	transitivity and lemma 2.3.5
$\Rightarrow P_1\{b/a\} \equiv_\alpha Q_1\{b/a\}\{x/y\}$	rule $AlpInp1$
$\Rightarrow z(x).(P_1\{b/a\}) \equiv_\alpha z(y).(Q_1\{b/a\})$	definition of substitution
$\Rightarrow (z(x).P_1)\{b/a\} \equiv_\alpha (z(y).Q_1)\{b/a\}$	

- the last part of the proof tree of $P \equiv_\alpha Q$ is

$$\textsc{AlpInp1}\ \frac{P_1 \equiv_\alpha Q_1\{x/y\} \qquad x \neq y \qquad x \notin fn(Q_1)}{\underbrace{b(x).P_1}_{P} \equiv_\alpha \underbrace{b(y).Q_1}_{Q}}$$

	rule premise
$P_1 \equiv_\alpha Q_1\{x/y\}$ and $x \notin fn(Q_1)$	inductive hypothesis
$\Rightarrow P_1\{b/a\} \equiv_\alpha Q_1\{x/y\}\{b/a\}$	transitivity and lemma 2.3.5
$\Rightarrow P_1\{b/a\} \equiv_\alpha Q_1\{b/a\}\{x/y\}$	rule $AlpInp1$
$\Rightarrow b(x).(P_1\{b/a\}) \equiv_\alpha b(y).(Q_1\{b/a\})$	definition of substitution
$\Rightarrow (b(x).P_1)\{b/a\} \equiv_\alpha (b(y).Q_1)\{b/a\}$	

- the last part of the proof tree of $P \equiv_\alpha Q$ is

$$\textsc{AlpInp1}\ \frac{P_1 \equiv_\alpha Q_1\{x/y\} \qquad x \neq y \qquad x \notin fn(Q_1)}{\underbrace{a(x).P_1}_{P} \equiv_\alpha \underbrace{a(y).Q_1}_{Q}}$$

	rule premise
$P_1 \equiv_\alpha Q_1\{x/y\}$ and $x \notin fn(Q_1)$	inductive hypothesis
$\Rightarrow P_1\{b/a\} \equiv_\alpha Q_1\{x/y\}\{b/a\}$	transitivity and lemma 2.3.5
$\Rightarrow P_1\{b/a\} \equiv_\alpha Q_1\{b/a\}\{x/y\}$	rule $AlpInp1$
$\Rightarrow a(x).(P_1\{b/a\}) \equiv_\alpha a(y).(Q_1\{b/a\})$	definition of substitution
$\Rightarrow (a(x).P_1)\{b/a\} \equiv_\alpha (a(y).Q_1)\{b/a\}$	

- the last part of the proof tree of $P \equiv_\alpha Q$ is

$$\textsc{AlpInp1}\ \frac{P_1 \equiv_\alpha Q_1\{a/y\} \qquad a \neq y \qquad a \notin fn(Q_1)}{\underbrace{a(a).P_1}_{P} \equiv_\alpha \underbrace{a(y).Q_1}_{Q}}$$

	rule premise
$P_1 \equiv_\alpha Q_1\{a/y\}$ and $x \notin fn(Q_1)$	inductive hypothesis
$\Rightarrow P_1\{b/a\} \equiv_\alpha Q_1\{a/y\}\{b/a\}$	transitivity and lemma 2.3.5
$\Rightarrow P_1\{b/a\} \equiv_\alpha Q_1\{b/a\}\{a/y\}$	rule $AlpInp1$
$\Rightarrow a(a).(P_1\{b/a\}) \equiv_\alpha a(y).(Q_1\{b/a\})$	definition of substitution
$\Rightarrow (a(a).P_1)\{b/a\} \equiv_\alpha (a(y).Q_1)\{b/a\}$	

- the last part of the proof tree of $P \equiv_\alpha Q$ is

$$\textsc{AlpInp1}\ \frac{P_1 \equiv_\alpha Q_1\{x/a\} \qquad x \neq a \qquad x \notin fn(Q_1)}{\underbrace{a(x).P_1}_{P} \equiv_\alpha \underbrace{a(a).Q_1}_{Q}}$$

$$
\begin{array}{ll}
 & \text{rule premise} \\
P_1 \equiv_\alpha Q_1\{x/a\} \text{ and } x \notin fn(Q_1) & \text{inductive hypothesis} \\
\Rightarrow P_1\{b/a\} \equiv_\alpha Q_1\{x/a\}\{b/a\} & \text{transitivity and lemma 2.3.5} \\
\Rightarrow P_1\{b/a\} \equiv_\alpha Q_1\{b/a\}\{x/a\} & \text{rule } AlpInp1 \\
\Rightarrow a(x).(P_1\{b/a\}) \equiv_\alpha a(a).(Q_1\{b/a\}) & \text{definition of substitution} \\
\Rightarrow (a(x).P_1)\{b/a\} \equiv_\alpha (a(a).Q_1)\{b/a\} &
\end{array}
$$

- mancano x x y e x y x

AlpRes :

$$
\begin{array}{ll}
 & \text{rule premise} \\
P_1 \equiv_\alpha Q_1 & \text{inductive hypothesis} \\
\Rightarrow P_1\{b/a\} \equiv_\alpha Q_1\{b/a\} & \text{rule } AlpRes \\
\Rightarrow (\nu x)(P_1\{b/a\}) \equiv_\alpha (\nu x)(Q_1\{b/a\}) & \text{definition of substitution} \\
\Rightarrow ((\nu x)P_1)\{b/a\} \equiv_\alpha ((\nu x)Q_1)\{b/a\} &
\end{array}
$$

*AlpRes*1 :

$$
\textsc{AlpRes1}\ \frac{P_1 \equiv_\alpha Q_1\{x/y\} \qquad x \neq y \qquad x \notin fn(Q_1)}{\underbrace{(\nu x)P_1}_{P} \equiv_\alpha \underbrace{(\nu y)Q_1}_{Q}}
$$

$$
\begin{array}{ll}
 & \text{rule premises} \\
P_1 \equiv_\alpha Q_1\{x/y\} \text{ and } x \neq y \text{ and } x \notin fn(Q_1) & \text{inductive hypothesis} \\
P_1\{b/a\} \equiv_\alpha Q_1\{x/y\}\{b/a\} & \text{lemma 2.3.5 and transitivity} \\
P_1\{b/a\} \equiv_\alpha Q_1\{b/a\}\{x/y\} & \text{rule } AlpRes1 \\
(\nu x)(P_1\{b/a\}) \equiv_\alpha (\nu y)(Q_1\{b/a\}) & \text{definition of substitution} \\
((\nu x)P_1)\{b/a\} \equiv_\alpha ((\nu y)Q_1)\{b/a\} &
\end{array}
$$

□

Lemma 2.3.7.

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

esistono delle precondizioni per le quali il lemma e' vero? esistono delle precondizioni per le quali si puo' addirittura avere l'uguaglianza sintattica?

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

Lemma 2.3.8. $P\{x/y\} \equiv_\alpha Q$ if and only if $P \equiv_\alpha Q\{y/x\}$.
NON FUNZIONA LA DIMOSTRAZIONE! staro' forse esagerando?

Proof. The proof is an induction on the length of the proof tree of $P\{x/y\} \equiv_\alpha Q$ and then by cases on the last rule:

base case the last rule can be

AlpZero in this case both P and Q are the null process 0 so the thesis holds.

AlpIde for this rule to apply $P\{x/y\}$ and Q must be some identifier A with the same variable. Suppose that $P = A(\tilde{a}|\tilde{b})$ There can be some different cases:

$y \in \tilde{a}$ we can suppose that $\tilde{a} = y, \tilde{c}$ then

$x \in \tilde{b}$ we can suppose that $\tilde{b} = x, \tilde{d}$, then

$$Q = P\{x/y\} = A(x, \tilde{c}|z, \tilde{d})$$

where z is a fresh name. We need now the identifier equal to $Q\{y/x\} = A(x, \tilde{c}|z, \tilde{d})\{y/x\}$ so we have to distinguish two cases:

$x \in tilded$

$x \notin tilded$

$$Q\{y/x\} = A(x, \tilde{c}|z, \tilde{d})\{y/x\} = A(y, \tilde{c}|z, \tilde{d})$$

$y \notin \tilde{y}$ in this case there is no need to change bound names so

$$Q\{y/x\} = A(y, \tilde{z}|\tilde{y})$$

$x \notin \tilde{x}$ then

$$Q\{y/x\} = Q = A(\tilde{x}|\tilde{y})$$

□

Lemma 2.3.9. The α equivalence is an equivalence relation.

Proof. :

reflexivity We prove $P \equiv_\alpha P$ by structural induction on P:

0 :

$$\textsc{AlpZero}\ \frac{}{0 \equiv_\alpha 0}$$

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

$$\textsc{AlpTau}\ \frac{P_1 \equiv_\alpha P_1}{\tau.P_1 \equiv_\alpha \tau.P_1}$$

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

$$\textsc{AlpInp}\ \frac{P_1 \equiv_\alpha P_1}{x(y).P_1 \equiv_\alpha x(y).P_1}$$

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

$$\textsc{AlpOut}\ \frac{P_1 \equiv_\alpha P_1}{\overline{x}y.P_1 \equiv_\alpha \overline{x}y.P_1}$$

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

$$\textsc{AlpSum}\ \frac{P_1 \equiv_\alpha P_1 \qquad P_2 \equiv_\alpha P_2}{P_1 + P_2 \equiv_\alpha P_1 + P_2}$$

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

$$\textsc{AlpPar}\ \frac{P_1 \equiv_\alpha P_1 \qquad P_2 \equiv_\alpha P_2}{P_1|P_2 \equiv_\alpha P_1|P_2}$$

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

$$\textsc{AlpRes}\ \frac{P_1 \equiv_\alpha P_1}{(\nu x)P_1 \equiv_\alpha (\nu x)P_1}$$

$A(\tilde{x}|\tilde{y})$:

$$\textsc{AlpIde}\ \frac{}{A(\tilde{x}|\tilde{y}) \equiv_\alpha A(\tilde{x}|\tilde{y})}$$

symmetry A proof of

$$P \equiv_\alpha Q \Rightarrow Q \equiv_\alpha P$$

can go by induction on the length of the proof tree of $P \equiv_\alpha Q$ and then by cases on the last rule used. Nevertheless we notice that the base case rules *AlpZero* and *AlpIde* are symmetric and the inductive case rules are symmetric except for *AlpRes*1 and *AlpInp*1. So we provide with the cases for those last two rules:

*AlpRes*1 the last part of the proof tree is

$$\text{AlpRes1}\ \frac{P\{y/x\} \equiv_\alpha Q \qquad x \neq y \qquad y \notin fn(P)}{(\nu x)P \equiv_\alpha (\nu y)Q}$$

we apply the inductive hypothesis on $P\{y/x\} \equiv_\alpha Q$ and get $Q \equiv_\alpha P\{y/x\}$ which implies $Q\{x/y\} \equiv_\alpha P$

DA DIMOSTRARE $Q \equiv_\alpha P\{y/x\}$ and $y \notin fn(P)$ implies $Q\{x/y\} \equiv_\alpha P$ and $x \notin fn(Q)$

so an application of the same rule yields:

$$\text{AlpRes1}\ \frac{Q\{x/y\} \equiv_\alpha P \qquad x \neq y \qquad x \notin fn(Q)}{(\nu y)QP \equiv_\alpha (\nu x)}$$

*AlpInp*1 this is very similar to the previous.

transitivity suppose

$$P \equiv_\alpha Q \text{ and } Q \equiv_\alpha R$$

we prove the thesis $P \equiv_\alpha R$ by induction on the length of the proof tree of $P \equiv_\alpha Q$. If the tree has only one node then the rule used must be *AlpZero* or *AlpIde*. In the former case both P and Q are 0 and so $0 \equiv_\alpha R$. For symmetry and the inversion lemma then R is also 0. In the latter case a similar argument applies. If the proof tree has more than one node then we proceed by cases on the last rule

AlpInp : In this case $P = x(y).P_1$, $Q = x(y).Q_1$ and $P_1 \equiv_\alpha Q_1$ and $x(y).Q_1 \equiv_\alpha R$ which implies for symmetry and the inversion lemma that one of the following cases holds:

- $R = x(y).R_1$ and $Q_1 \equiv_\alpha R_1$:

$$\begin{array}{ll} P_1 \equiv_\alpha Q_1 \text{ and } Q_1 \equiv_\alpha R_1 & \text{inductive hypothesis} \\ \Rightarrow P_1 \equiv_\alpha R_1 & \text{rule } AlpInp \\ \Rightarrow x(y).P_1 \equiv_\alpha x(y).R_1 & \end{array}$$

- $R = x(z).R_1$ and $Q_1\{y/z\} \equiv_\alpha R_1$:

$$\begin{array}{ll} P_1 \equiv_\alpha Q_1 & \text{lemma 2.3.6} \\ \Rightarrow P_1\{y/z\} \equiv_\alpha Q_1\{y/z\} & \text{inductive hypothesis} \\ \Rightarrow P_1\{y/z\} \equiv_\alpha R_1 & \text{rule } AlpInp1 \\ \Rightarrow x(y).P_1 \equiv_\alpha x(z).R_1 & \end{array}$$

AlpRes :

*AlpInp*1 :

*AlpRes*1 :

AlpSum :

AlpPar :

AlpSum :

AlpTau :

AlpOut :

□

Lemma 2.3.10. E' FALSO!!!!! !!!! !!! !! !:

- If $P \equiv \tau.Q$ then $P = \tau.P_1$ for some P_1 such that $P_1 \equiv Q$
- If $P \equiv \overline{x}y.Q$ then $P = \overline{x}y.P_1$ for some P_1 such that $P_1 \equiv Q$
- If $P \equiv x(y).Q$ then one and only one of the following cases holds:

Out $\dfrac{}{\overline{x}y.P \xrightarrow{\overline{x}y} P}$	**EInp** $\dfrac{}{x(y).P \xrightarrow{xz} P\{z/y\}}$
ParL $\dfrac{P \xrightarrow{\alpha} P^{'} \quad bn(\alpha) \cap fn(Q) = \emptyset}{P\vert Q \xrightarrow{\alpha} P^{'}\vert Q}$	**ParR** $\dfrac{Q \xrightarrow{\alpha} Q^{'} \quad bn(\alpha) \cap fn(Q) = \emptyset}{P\vert Q \xrightarrow{\alpha} P\vert Q^{'}}$
SumL $\dfrac{P \xrightarrow{\alpha} P^{'}}{P + Q \xrightarrow{\alpha} P^{'}}$	**SumR** $\dfrac{Q \xrightarrow{\alpha} Q^{'}}{P + Q \xrightarrow{\alpha} Q^{'}}$
Res $\dfrac{P \xrightarrow{\alpha} P^{'} \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha} (\nu z)P^{'}}$	**Alp** $\dfrac{P \equiv_{\alpha} Q \quad P \xrightarrow{\alpha} P^{'}}{Q \xrightarrow{\alpha} P^{'}}$
EComL $\dfrac{P \xrightarrow{xy} P^{'} \quad Q \xrightarrow{\overline{x}y} Q^{'}}{P\vert Q \xrightarrow{\tau} P^{'}\vert Q^{'}}$	**EComR** $\dfrac{P \xrightarrow{\overline{x}y} P^{'} \quad Q \xrightarrow{xy} Q^{'}}{P\vert Q \xrightarrow{\tau} P^{'}\vert Q^{'}}$
ClsL $\dfrac{P \xrightarrow{\overline{x}(z)} P^{'} \quad Q \xrightarrow{xz} Q^{'} \quad z \notin fn(Q)}{P\vert Q \xrightarrow{\tau} (\nu z)(P^{'}\vert Q^{'})}$	**ClsR** $\dfrac{P \xrightarrow{xz} P^{'} \quad Q \xrightarrow{\overline{x}(z)} Q^{'} \quad z \notin fn(P)}{P\vert Q \xrightarrow{\tau} (\nu z)(P^{'}\vert Q^{'})}$
Cns $\dfrac{A(\tilde{x}\vert\tilde{y}) \overset{def}{=} P \quad P\{\tilde{w}/\tilde{x}\} \xrightarrow{\alpha} P^{'}}{A(\tilde{x}\vert\tilde{y})\{\tilde{w}/\tilde{x}\} \xrightarrow{\alpha} P^{'}}$	**Opn** $\dfrac{P \xrightarrow{\overline{x}z} P^{'} \quad z \neq x}{(\nu z)P \xrightarrow{\overline{x}(z)} P^{'}}$
Tau $\dfrac{}{\tau.P \xrightarrow{\tau} P}$	

Table 2.8: Early transition relation with α conversion

- $P = x(z).P_1$ for some P_1 such that $P_1\{z/y\} \equiv Q$
- $P = x(y).P_1$ for some P_1 such that $P_1 \equiv Q$

- If $P \equiv Q_1 + Q_2$ then $P = P_1 + P_2$ for some P_1 and P_2 such that $P_1 \equiv Q_1$ and $P_2 \equiv Q_2$.
- If $P \equiv Q_1|Q_2$ then $P = P_1|P_2$ for some P_1 and P_2 such that $P_1 \equiv Q_1$ and $P_2 \equiv Q_2$.
- If $P \equiv (\nu y)Q$ then one and only one of the following cases holds:
  - $P = (\nu z)P_1$ such that $P_1\{z/y\} \equiv Q$
  - $P = (\nu y).P_1$ for some P_1 such that $P_1 \equiv Q$
- If $P \equiv A(\tilde{x}|\tilde{y})$ then ??? ?? ?

Proof. □

2.4 Operational semantic with structural congruence

2.4.1 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.

$$
\textbf{Out}\ \frac{}{\overline{x}y.P \xrightarrow{\overline{x}y} P}
\qquad
\textbf{EInp}\ \frac{}{x(z).P \xrightarrow{xy} P\{y/z\}}
\qquad
\textbf{Par}\ \frac{P \xrightarrow{\alpha} P' \quad bn(\alpha) \cap fn(Q) = \emptyset}{P|Q \xrightarrow{\alpha} P'|Q}
$$

$$
\textbf{Sum}\ \frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'}
\qquad
\textbf{ECom}\ \frac{P \xrightarrow{xy} P' \quad Q \xrightarrow{\overline{x}y} Q'}{P|Q \xrightarrow{\tau} P'|Q'}
\qquad
\textbf{Res}\ \frac{P \xrightarrow{\alpha} P' \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha} (\nu z)P'}
$$

$$
\textbf{Tau}\ \frac{}{\tau.P \xrightarrow{\tau} P}
\qquad
\textbf{Opn}\ \frac{P \xrightarrow{\overline{x}z} P' \quad z \neq x}{(\nu z)P \xrightarrow{\overline{x}(z)} P'}
\qquad
\textbf{Str}\ \frac{P \equiv P' \quad P \xrightarrow{\alpha} Q \quad Q \equiv Q'}{P' \xrightarrow{\alpha} Q'}
$$

Table 2.9: Early semantic with structural congruence

2.4.2 Early semantic with structural congruence

Definition 2.4.2. 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

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

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

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

and

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

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

$$
\textsc{Res}\ \frac{\textsc{Com}\ \dfrac{\textsc{EInp}\ \dfrac{}{a(x).P \xrightarrow{ab} P\{b/x\}} \qquad \textsc{Out}\ \dfrac{}{\overline{a}b.Q \xrightarrow{\overline{a}b} Q}}{a(x).P \mid \overline{a}b.Q \xrightarrow{\tau} P\{b/x\} \mid Q}}{(\nu b)(a(x).P \mid \overline{a}b.Q) \xrightarrow{\tau} (\nu b)(P\{b/x\} \mid Q)}
$$

Example We want to prove now that:

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

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

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

then we have

$$
\textsc{Res}\ \frac{\textsc{Com}\ \dfrac{\textsc{EInp}\ \dfrac{}{a(x).P\{c/b\} \xrightarrow{ab} P\{c/b\}\{b/x\}} \qquad \textsc{Out}\ \dfrac{}{\overline{a}b.Q \xrightarrow{\overline{a}b} Q}}{(a(x).(P\{c/b\})|\overline{a}b.Q) \xrightarrow{\tau} (P\{c/b\}\{b/x\}|Q)}}{(\nu c)(a(x).(P\{c/b\})|\overline{a}b.Q) \xrightarrow{\tau} (\nu c)(P\{c/b\}\{b/x\}|Q)}
$$

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

2.4.3 Late semantic with structural congruence

Definition 2.4.3. 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

$$\textbf{Prf}\ \frac{}{\alpha.P \xrightarrow{\alpha} P} \qquad \textbf{Sum}\ \frac{P \xrightarrow{\alpha} P^{'}}{P+Q \xrightarrow{\alpha} P^{'}}$$

$$\textbf{Par}\ \frac{P \xrightarrow{\alpha} P^{'} \quad bn(\alpha) \cap fn(Q) = \emptyset}{P|Q \xrightarrow{\alpha} P^{'}|Q} \qquad \textbf{Res}\ \frac{P \xrightarrow{\alpha} P^{'} \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha} (\nu z)P^{'}}$$

$$\textbf{LCom}\ \frac{P \xrightarrow{x(y)} P^{'} \quad Q \xrightarrow{\overline{x}z} Q^{'}}{P|Q \xrightarrow{\tau} P^{'}\{z/y\}|Q^{'}} \qquad \textbf{Str}\ \frac{P \equiv P^{'} \quad P \xrightarrow{\alpha} Q \quad Q \equiv Q^{'}}{P^{'} \xrightarrow{\alpha} Q^{'}}$$

$$\textbf{Opn}\ \frac{P \xrightarrow{\overline{x}z} P^{'} \quad z \neq x}{(\nu z)P \xrightarrow{\overline{x}(z)} P^{'}}$$

Table 2.10: Late semantic with structural congruence

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

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

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

and

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

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

$$\textsc{Res}\ \dfrac{\textsc{LCom}\ \dfrac{\textsc{LInp}\ \dfrac{b \notin fn(P)}{a(x).P \ \xrightarrow{ab}\ P\{b/x\}} \qquad \textsc{Out}\ \dfrac{}{\overline{a}b.Q \ \xrightarrow{\overline{a}b}\ Q}}{a(x).P \mid \overline{a}b.Q \ \xrightarrow{\tau}\ P\{b/x\} \mid Q} \qquad b \notin n(\tau)}{(\nu b)(a(x).P \mid \overline{a}b.Q) \ \xrightarrow{\tau}\ (\nu b)(P\{b/x\} \mid Q)}$$

Example We want to prove now that:

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

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

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

then we have

$$\textsc{Res}\ \dfrac{\textsc{LCom}\ \dfrac{\textsc{LInp}\ \dfrac{b \notin fn(P\{c/b\})}{a(x).P\{c/b\} \ \xrightarrow{ab}\ P\{c/b\}\{b/x\}} \qquad \textsc{Out}\ \dfrac{}{\overline{a}b.Q \ \xrightarrow{\overline{a}b}\ Q}}{(a(x).(P\{c/b\})|\overline{a}b.Q) \ \xrightarrow{\tau}\ (P\{c/b\}\{b/x\}|Q)} \qquad c \notin n(\tau)}{(\nu c)(a(x).(P\{c/b\})|\overline{a}b.Q) \ \xrightarrow{\tau}\ (\nu c)(P\{c/b\}\{b/x\}|Q)}$$

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

2.5 Equivalence of the semantics

2.5.1 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 $\rightarrow_3$. In the following section we will need:

Lemma 2.5.1.
$$P \equiv Q \;\Rightarrow\; fn(Q) = fn(P)$$

Proof. A proof can go by induction on the proof tree of $P \equiv Q$ and then by cases on the last rule used in the proof tree.

base case The last and only rule of the proof tree can be one of the following axioms:

SC-ALP $\dfrac{P \equiv_\alpha Q}{P \equiv Q}$

SC-SUM-ASC $M_1 + (M_2 + M_3) \equiv (M_1 + M_2) + M_3$

SC-SUM-COM $M_1 + M_2 \equiv M_2 + M_1$

SC-SUM-INC $M + 0 \equiv M$

SC-COM-ASC $P_1|(P_2|P_3) \equiv (P_1|P_2)|P_3$

SC-COM-COM $P_1|P_2 \equiv P_2|P_1$

SC-COM-INC $P|0 \equiv P$

SC-RES $(\nu z)(\nu w)P \equiv (\nu w)(\nu z)P$

SC-RES-INC $(\nu z)0 \equiv 0$

SC-RES-COM $(\nu z)(P_1|P_2) \equiv P_1|(\nu z)P_2$ if $z \notin fn(P_1)$

SC-RES-SUM $(\nu z)(P_1 + P_2) \equiv P_1 + (\nu z)P_2$ if $z \notin fn(P_1)$

SC-IDE $A(\tilde{w}|\tilde{y}) \equiv P\{\tilde{w}/\tilde{x}\}$

inductive case

SC-REFL $P \equiv P$

SC-SIMM $\dfrac{Q \equiv P}{P \equiv Q}$

SC-TRAN $\dfrac{P \equiv Q \qquad Q \equiv R}{P \equiv R}$

SC-CONG $\dfrac{P \equiv Q}{C[P] \equiv C[Q]}$

□

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

$$\textsc{Alp}\ \dfrac{\overline{x}y.x(y).0 \equiv_\alpha \overline{x}y.x(w).0 \qquad \textsc{Out}\ \dfrac{}{\overline{x}y.x(w).0 \xrightarrow{\overline{x}y}_2 x(w).0}}{\overline{x}y.x(y).0 \xrightarrow{\overline{x}y}_2 x(w).0}$$

so we want to prove
$$\overline{x}y.x(y).0 \xrightarrow{\overline{x}y}_1 x(w).0$$

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

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

which is not want we want. So we prove a weaker version

Theorem 2.5.2.
$$P \xrightarrow{\alpha}_2 P' \;\Rightarrow\; \exists P'' : P'' \equiv_\alpha P' \textit{ and } P \xrightarrow{\alpha}_1 P''$$

Proof. The proof goes by induction on the depth of the derivation tree of $P \xrightarrow{\alpha}_2 P^{'}$ and then by cases on the last rule used:

base case If the depth of the derivation tree is one, the rule used has to be a prefix rule

$$\{Out, EInp, Tau\} \subseteq R_1 \cap R_2$$

so a derivation tree of $P \xrightarrow{\alpha}_2 P^{'}$ is also a derivation tree of $P \xrightarrow{\alpha}_1 P^{'}$

inductive case If the depth of the derivation tree is more than one, then we proceed by cases on the last rule R. If the rule R is not a prefix rule and it is in common between the two semantics:

$$R \in \{ParL, ParR, SumL, SumR, Res, EComL, EComR, ClsL, ClsR, Cns, Opn\}$$

then we just apply the inductive hypothesis on the premises of R and then reapply R to get the desired derivation tree. We show just the case for $SumL$ when the end of the derivation tree is

$$\textsc{SumL}\ \frac{P_1 \xrightarrow{\alpha}_2 P_1^{'}}{\underbrace{P_1 + P_2}_{P} \xrightarrow{\alpha}_2 \underbrace{P_1^{'}}_{P'}}$$

$$\begin{array}{ll} & \text{rule premise} \\ P_1 \xrightarrow{\alpha}_2 P_1^{'} & \text{inductive hypothesis} \\ \Rightarrow P_1 \xrightarrow{\alpha}_1 P_1^{''} \text{ and } P_1^{'} \equiv_\alpha P_1^{''} & \text{rule } SumL \\ \Rightarrow P_1 + P_2 \xrightarrow{\alpha}_1 P_1^{''} & \end{array}$$

If the rule R is in

$$R_2 - R_1 = \{Alp\}$$

then the last part of the derivation tree of $P \xrightarrow{\alpha}_2 P^{'}$ is

$$\textsc{Alp}\ \frac{P \equiv_\alpha Q \qquad \mathrm{S}\ \dfrac{\cdots}{Q \xrightarrow{\alpha}_2 P^{'}}}{P \xrightarrow{\alpha}_2 P^{'}}$$

and the proof goes by cases on S the last rule in the proof tree of $Q \xrightarrow{\alpha}_2 P^{'}$:

Out : If $S = Out$ then there exists some names x, y and a process Q_1 such that

$$Q = \overline{x}y.Q_1$$

and $\alpha = \overline{x}y$.

$$\begin{array}{ll} P \equiv_\alpha \overline{x}y.Q_1 & \text{inversion lemma} \\ \Rightarrow P = \overline{x}y.P_1 \text{ and } P_1 \equiv_\alpha Q_1 & \text{rule } Out \\ \Rightarrow \overline{x}y.P_1 \xrightarrow{\overline{x}y}_1 P_1 & \end{array}$$

EInp If $S = EInp$ then there exists some names x, y, z and a process Q_1 such that $Q = x(y).Q_1$, $\alpha = xz$ and $P^{'} = Q_1\{z/y\}$. Since

$$P \equiv_\alpha x(y).Q_1$$

then for the inversion lemma we have two cases:

- :

$$\begin{array}{ll} P = x(y).P_1 \text{ and } P_1 \equiv_\alpha Q_1 & \text{rule } EInp \\ \Rightarrow x(y).P_1 \xrightarrow{xz}_1 P_1\{z/y\} & \end{array}$$

This is what we want because for lemma 2.3.6

$$P_1 \equiv_\alpha Q_1 \Rightarrow P_1\{z/y\} \equiv_\alpha Q_1\{z/y\}$$

- :

$$\begin{array}{ll} P = x(w).P_1 \text{ and } P_1\{y/w\} \equiv_\alpha Q_1 & \text{rule } EInp \\ \Rightarrow x(w).P_1 \xrightarrow{xz}_1 P_1\{z/w\} & \end{array}$$

This is what we want because

$$\begin{array}{ll} P_1\{y/w\} \equiv_\alpha Q_1 & \text{lemma 2.3.6} \\ \Rightarrow P_1\{y/w\}\{z/y\} \equiv_\alpha Q_1\{z/y\} & \\ \Rightarrow P_1\{z/w\} \equiv_\alpha Q_1\{z/y\} & \end{array}$$

Tau If $S = Tau$ then there exists a process Q_1 such that $Q = \tau.Q_1$ and $\alpha = \tau$ and $P' = Q_1$.

$$\begin{array}{ll} P \equiv_\alpha \tau.Q_1 & \text{inversion lemma} \\ \Rightarrow P = \tau.P_1 \text{ and } P_1 \equiv_\alpha Q_1 & \text{rule } Tau \\ \Rightarrow \tau.P_1 \xrightarrow{\tau}_1 P_1 & \end{array}$$

ParL If $S = ParL$ then there exists some processes Q_1, Q_2 such that

$$Q = Q_1|Q_2$$

Since

$$P \equiv_\alpha Q_1|Q_2$$

then for the inversion lemma there exists P_1, P_2 such that

$$P = P_1|P_2 \textit{ and } P_1 \equiv_\alpha Q_1 \textit{ and} P_2 \equiv_\alpha Q_2$$

and so the last part of the derivation tree of $P \xrightarrow{\alpha}_2 P'$ looks like this:

$$\textsc{Alp}\ \frac{P_1|P_2 \equiv_\alpha Q_1|Q_2 \qquad \textsc{ParL}\ \dfrac{Q_1 \xrightarrow{\alpha}_2 Q_1' \qquad bn(\alpha) \cap fn(Q_2) = \emptyset}{Q_1|Q_2 \xrightarrow{\alpha}_2 Q_1'|Q_2}}{\underbrace{P_1|P_2}_{P} \xrightarrow{\alpha}_2 \underbrace{Q_1'|Q_2}_{P'}}$$

from this hypothesis we can create the following proof tree of $P_1 \xrightarrow{\alpha}_2 Q_1'$:

$$\textsc{Alp}\ \frac{P_1 \equiv_\alpha Q_1 \qquad Q_1 \xrightarrow{\alpha}_2 Q_1'}{P_1 \xrightarrow{\alpha}_2 Q_1'}$$

this proof tree is smaller than the proof tree of $P_1|P_2 \xrightarrow{\alpha}_2 Q_1'|Q_2$ so we can apply the inductive hypothesis and get that there exists a process Q_1'' such that

$$Q_1' \equiv Q_1'' \textit{ and } P_1 \xrightarrow{\alpha}_1 Q_1''$$

then we apply again the rule $ParL$ and get

$$\textsc{ParL}\ \frac{P_1 \xrightarrow{\alpha}_1 Q_1'' \qquad bn(\alpha) \cap fn(P_2) = \emptyset}{\underbrace{P_1|P_2}_{P} \xrightarrow{\alpha}_1 \underbrace{Q_1''|P_2}_{P''}}$$

The second premise of the previous instance holds because:

$$bn(\alpha) \cap fn(Q_2) = \emptyset \textit{ and } P_2 \equiv_\alpha Q_2 \Rightarrow bn(\alpha) \cap fn(P_2) = \emptyset$$

ParR, SumL, SumR, EComL, EComR, ClsL, ClsR This cases are similar to the previous.

Res If $S = Res$ then there exists some name z and a process Q_1 such that

$$Q = (\nu z)Q_1$$

and $P^{'} = (\nu z)Q_1^{'}$. Since

$$P \equiv_\alpha (\nu z)Q_1$$

then for the inversion lemma we have two cases:

- there exists some P_1 such that

$$P = (\nu z)P_1 \textit{ and } P_1 \equiv_\alpha Q_1$$

and so the last part of the derivation tree of $P \xrightarrow{\alpha}_2 P^{'}$ looks like this:

$$\textsc{Alp}\ \dfrac{(\nu z)P_1 \equiv_\alpha (\nu z)Q_1 \qquad \textsc{Res}\ \dfrac{Q_1 \xrightarrow{\alpha}_2 Q_1^{'} \qquad z \notin n(\alpha)}{(\nu z)Q_1 \xrightarrow{\alpha}_2 (\nu z)Q_1^{'}}}{(\nu z)P_1 \xrightarrow{\alpha}_2 (\nu z)Q_1^{'}}$$

from this we create the following proof tree of $P_1 \xrightarrow{\alpha}_2 Q_1^{'}$:

$$\textsc{Alp}\ \dfrac{P_1 \equiv_\alpha Q_1 \qquad Q_1 \xrightarrow{\alpha}_2 Q_1^{'}}{P_1 \xrightarrow{\alpha}_2 Q_1^{'}}$$

to which we can apply the inductive hypothesis and get that there exists a process $Q_1^{''}$ such that

$$P_1 \xrightarrow{\alpha}_1 Q_1^{''} \textit{ and } Q_1^{''} \equiv_\alpha Q_1^{'}$$

then we apply the rule *Res* to get

$$\textsc{Res}\ \dfrac{P_1 \xrightarrow{\alpha}_1 Q_1^{''} \qquad z \notin n(\alpha)}{(\nu z)P_1 \xrightarrow{\alpha}_1 (\nu z)Q_1^{''}}$$

this satisfies the thesis of the theorem because

$$(\nu z)Q_1^{''} \equiv (\nu z)Q_1^{'}$$

- there exists some P_1 such that

$$P = (\nu y)P_1 \textit{ and } P_1\{z/y\} \equiv_\alpha Q_1$$

and so the last part of the derivation tree of $P \xrightarrow{\alpha}_2 P^{'}$ looks like this:

$$\textsc{Alp}\ \dfrac{(\nu y)P_1 \equiv_\alpha (\nu z)Q_1 \qquad \textsc{Res}\ \dfrac{Q_1 \xrightarrow{\alpha}_2 Q_1^{'} \qquad z \notin n(\alpha)}{(\nu z)Q_1 \xrightarrow{\alpha}_2 (\nu z)Q_1^{'}}}{(\nu y)P_1 \xrightarrow{\alpha}_2 (\nu z)Q_1^{'}}$$

from this we create the following proof tree of $P_1\{z/y\} \xrightarrow{\alpha}_2 Q_1^{'}$:

$$\textsc{Alp}\ \dfrac{P_1\{z/y\} \equiv_\alpha Q_1 \qquad Q_1 \xrightarrow{\alpha}_2 Q_1^{'}}{P_1\{z/y\} \xrightarrow{\alpha}_2 Q_1^{'}}$$

to which we can apply the inductive hypothesis and get that there exists a process $Q_1^{''}$ such that

$$P_1\{z/y\} \xrightarrow{\alpha}_1 Q_1^{''} \textit{ and } Q_1^{''} \equiv_\alpha Q_1^{'}$$

then we apply the rule *Res* and *ResAlp* to get

$$\textsc{ResAlp}\ \dfrac{\textsc{Res}\ \dfrac{P_1\{z/y\} \xrightarrow{\alpha}_1 Q_1^{''} \qquad z \notin n(\alpha)}{(\nu z)P_1\{z/y\} \xrightarrow{\alpha}_1 (\nu z)Q_1^{''}}}{(\nu y)P_1 \xrightarrow{\alpha}_1 (\nu z)Q_1^{''}}$$

this satisfies the thesis of the theorem because

$$(\nu z)Q_1^{''} \equiv (\nu z)Q_1^{'}$$

Alp we can assume that there are no two consecutive application of the rule *Alp* because we can merge them thanks to the transitivity of the alpha equivalence.

Opn If $S = Opn$ then there exists some names x, z and a process Q_1 such that

$$Q = (\nu z)Q_1$$

and $P^{'} = Q_1^{'}$ and $\alpha = \overline{x}(z)$. Since

$$P \equiv_\alpha (\nu z)Q_1$$

then for the inversion lemma we have two cases:

- there exists some P_1 such that

$$P = (\nu z)P_1 \textit{ and } P_1 \equiv_\alpha Q_1$$

  and so the last part of the derivation tree of $P \xrightarrow{\alpha}_2 P^{'}$ looks like this:

$$\textsc{Alp}\ \frac{(\nu z)P_1 \equiv_\alpha (\nu z)Q_1 \qquad \textsc{Opn}\ \dfrac{Q_1 \xrightarrow{\overline{x}z}_2 Q_1^{'} \qquad z \neq x}{(\nu z)Q_1 \xrightarrow{\overline{x}(z)}_2 Q_1^{'}}}{(\nu z)P_1 \xrightarrow{\overline{x}(z)}_2 Q_1^{'}}$$

  from this we create the following proof tree of $P_1 \xrightarrow{\overline{x}z}_2 Q_1^{'}$:

$$\textsc{Alp}\ \frac{P_1 \equiv_\alpha Q_1 \qquad Q_1 \xrightarrow{\overline{x}z}_2 Q_1^{'}}{P_1 \xrightarrow{\overline{x}z}_2 Q_1^{'}}$$

  to which we can apply the inductive hypothesis and get that there exists a process $Q_1^{''}$ such that

$$P_1 \xrightarrow{\overline{x}z}_1 Q_1^{''} \textit{ and } Q_1^{''} \equiv_\alpha Q_1^{'}$$

  then we apply the rule *Opn* to get

$$\textsc{Opn}\ \frac{P_1 \xrightarrow{\overline{x}z}_1 Q_1^{''} \qquad z \neq x}{(\nu z)P_1 \xrightarrow{\alpha}_1 Q_1^{''}}$$

- there exists some P_1 such that

$$P = (\nu y)P_1 \textit{ and } P_1\{z/y\} \equiv_\alpha Q_1$$

  and so the last part of the derivation tree of $P \xrightarrow{\alpha}_2 P^{'}$ looks like this:

$$\textsc{Alp}\ \frac{(\nu y)P_1 \equiv_\alpha (\nu z)Q_1 \qquad \textsc{Opn}\ \dfrac{Q_1 \xrightarrow{\overline{x}z}_2 Q_1^{'} \qquad z \neq x}{(\nu z)Q_1 \xrightarrow{\overline{x}z}_2 Q_1^{'}}}{(\nu y)P_1 \xrightarrow{\alpha}_2 Q_1^{'}}$$

  from this we create the following proof tree of $P_1\{z/y\} \xrightarrow{\alpha}_2 Q_1^{'}$:

$$\textsc{Alp}\ \frac{P_1\{z/y\} \equiv_\alpha Q_1 \qquad Q_1 \xrightarrow{\alpha}_2 Q_1^{'}}{P_1\{z/y\} \xrightarrow{\alpha}_2 Q_1^{'}}$$

  to which we can apply the inductive hypothesis and get that there exists a process $Q_1^{''}$ such that

$$P_1\{z/y\} \xrightarrow{\alpha}_1 Q_1^{''} \textit{ and } Q_1^{''} \equiv_\alpha Q_1^{'}$$

  then we apply the rule *Opn* and *OpnAlp* to get

$$\textsc{OpnAlp}\ \frac{\textsc{Opn}\ \dfrac{P_1\{z/y\} \xrightarrow{\overline{x}z}_1 Q_1^{''} \qquad z \neq x}{(\nu z)P_1\{z/y\} \xrightarrow{\overline{x}(z)}_1 Q_1^{''}} \qquad z \notin n(P) \qquad x \neq y \neq z}{(\nu y)P_1 \xrightarrow{\overline{x}(z)}_1 Q_1^{''}}$$

Cns Since there is no process α equivalent to an identifier except for the identifier itself, the last part of the derivation tree of $P \xrightarrow{\alpha}_2 P^{'}$ looks like this:

$$\textsc{Alp}\ \frac{A(\tilde{x}|\tilde{y})\{\tilde{w}/\tilde{x}\} \equiv_\alpha A(\tilde{x}|\tilde{y})\{\tilde{w}/\tilde{x}\} \qquad \textsc{Cns}\ \dfrac{A(\tilde{x}|\tilde{y}) \overset{def}{=} R \qquad R\{\tilde{w}/\tilde{x}\} \xrightarrow{\alpha}_2 P^{'}}{A(\tilde{x}|\tilde{y})\{\tilde{w}/\tilde{x}\} \xrightarrow{\alpha}_2 P^{'}}}{A(\tilde{x}|\tilde{y})\{\tilde{w}/\tilde{x}\} \xrightarrow{\alpha}_2 P^{'}}$$

here we can apply the inductive hypothesis on the conclusion of S and get that there exists a process $P^{''}$ such that $A(\tilde{x}|\tilde{y})\{\tilde{w}/\tilde{x}\} \xrightarrow{\alpha}_1 P^{''}$ and $P^{'} \equiv_\alpha P^{''}$

□

Theorem 2.5.3. $P \xrightarrow{\alpha}_1 P^{'} \Rightarrow P \xrightarrow{\alpha}_2 P^{'}$

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

$$\frac{(\nu z)P \equiv_\alpha (\nu w)P\{w/z\} \qquad w \notin n(P) \qquad (\nu w)P\{w/z\} \xrightarrow{xz} P^{'}}{(\nu z)P \xrightarrow{xz} P^{'}}$$

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

$$\frac{(\nu z)P \equiv_\alpha (\nu w)P\{w/z\} \qquad w \notin n(P) \qquad (\nu w)P\{w/z\} \xrightarrow{\overline{x}(w)} P^{'} \qquad x \neq w \neq z}{(\nu z)P \xrightarrow{\overline{x}(w)} P^{'}}$$

□

Theorem 2.5.4. $P \xrightarrow{\alpha}_2 P^{'} \Leftrightarrow \exists P^{''} : P^{'} \equiv P^{''}$ *and* $P \xrightarrow{\alpha}_3 P^{''}$

Proof. $\Rightarrow$ First we prove $P \xrightarrow{\alpha}_2 P^{'} \Rightarrow \exists P^{''} : P^{'} \equiv P^{''}$ *and* $P \xrightarrow{\alpha}_3 P^{''}$. 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.

base case 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 $P \xrightarrow{\alpha}_3 P^{'}$

inductive case :

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

$$\textsc{Res}\ \frac{P \xrightarrow{\alpha}_2 P^{'} \qquad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha}_2 (\nu z)P^{'}}$$

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

$$\textsc{Res}\ \frac{P \xrightarrow{\alpha}_3 P^{''} \qquad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\alpha}_3 (\nu z)P^{''}}$$

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

- the last rule used can be one in $\{ParL, ParR, SumL, SumR, EComL, EComR\}$, in this case we can proceed as in the previous case and if necessary add an application of *Str* thus exploiting the commutativity of sum or parallel composition. For example

$$\textsc{ParR}\ \frac{Q \xrightarrow{\alpha}_2 Q^{'} \qquad bn(\alpha) \cap fn(Q) = \emptyset}{P|Q \xrightarrow{\alpha}_2 P|Q^{'}}$$

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

$$\textsc{Str}\ \frac{P|Q \equiv Q|P \qquad \textsc{Par}\ \dfrac{Q \xrightarrow{\alpha}_3 Q^{''} \qquad bn(\alpha) \cap fn(Q) = \emptyset}{Q|P \xrightarrow{\alpha}_3 Q^{''}|P}}{P|Q \xrightarrow{\alpha}_3 Q^{''}|P}$$

and $Q^{''}|P \equiv P|Q^{'}$

- if the last rule used is Cns:

$$\textsc{Cns}\ \frac{A(\tilde{x}|\tilde{z}) \stackrel{def}{=} P \qquad P\{\tilde{y}/\tilde{x}\} \xrightarrow{\alpha}_2 P^{'}}{A(\tilde{y}|\tilde{z}) \xrightarrow{\alpha}_2 P^{'}}$$

we apply the inductive hypothesis on the premise and get $P\{\tilde{y}/\tilde{x}\} \xrightarrow{\alpha}_3 P^{''}$ such that $P^{''} \equiv P^{'}$. Now the proof we want is

$$\textsc{Str}\ \frac{A(\tilde{y}|\tilde{z}) \equiv P\{\tilde{y}/\tilde{x}\} \qquad P\{\tilde{y}/\tilde{x}\} \xrightarrow{\alpha}_3 P^{''}}{A(\tilde{y}|\tilde{z}) \xrightarrow{\alpha}_3 P^{''}}$$

- if the last rule is Alp, then we just notice that this rule is a particular case of Str
- if the last rule is $ClsL$(the case for $ClsR$ is simmetric) then we have

$$\textsc{ClsL}\ \frac{P \xrightarrow{\overline{x}(z)}_2 P^{'} \qquad Q \xrightarrow{xz}_2 Q^{'} \qquad z \notin fn(Q)}{P|Q \xrightarrow{\tau}_2 (\nu z)(P^{'}|Q^{'})}$$

there is no easy way to mimic this rule with the rules in R_3. But if in the derivation tree we have an introduction of the bound output $\overline{x}(z)$ followed directly by an elimination of the same bound output such as:

$$\textsc{ClsL}\ \frac{\textsc{Opn}\ \dfrac{P \xrightarrow{\overline{x}z}_2 P^{'} \qquad z \neq x}{(\nu z)P \xrightarrow{\overline{x}(z)}_2 P^{'}} \qquad Q \xrightarrow{xz}_2 Q^{'} \qquad z \notin fn(Q)}{((\nu z)P)|Q \xrightarrow{\tau}_2 (\nu z)(P^{'}|Q^{'})}$$

we can apply the inductive hypothesis and get that

$$P \xrightarrow{\overline{x}z}_3 P^{''} \textit{ and } Q \xrightarrow{xz}_3 Q^{''}$$

where $P^{'} \equiv P^{''}$ and $Q^{'} \equiv Q^{''}$, so we create the needed proof in the following way

$$\textsc{Str}\ \frac{(\nu z)(P|Q) \equiv ((\nu z)P)|Q \qquad \textsc{Res}\ \dfrac{\textsc{Com}\ \dfrac{P \xrightarrow{\overline{x}z}_3 P^{''} \qquad Q \xrightarrow{xz}_3 Q^{''}}{P|Q \xrightarrow{\tau}_3 P^{''}|Q^{''}}}{(\nu z)(P|Q) \xrightarrow{\tau}_3 (\nu z)(P^{''}|Q^{''})}}{((\nu z)P)|Q \xrightarrow{\tau}_3 (\nu z)(P^{''}|Q^{''})}$$

We can always take a derivation tree in R_2 and move downward each occurrence of Opn until we find the appropriate occurrence of $ClsL$. In this process we might need to use the structural congruence, in particular the scope extension axioms. We can attempt to prove that in the following way:

$$P \xrightarrow{\overline{x}(z)}_2 P^{'} \ \Rightarrow\ \exists R : (\nu z)R \equiv P$$

and if $(\nu z)R \xrightarrow{\overline{x}(z)}_2 P^{'}$ then there exists a derivation tree for this transition such that the last rule used is Opn

PRIMA DEVO DIMOSTRARE IL LEMMA DI INVERSIONE PER LA CONGRUENZA STRUTTURALE(SE E' VERO)

Secondly we prove $P \xrightarrow{\alpha}_3 P^{'} \;\Rightarrow\; \exists P^{''} : P^{'} \equiv P^{''}$ *and* $P \xrightarrow{\alpha}_2 P^{''}$. The proof is by induction on the length of the derivation of $P \xrightarrow{\alpha}_3 P^{'}$, and then both the base case and the inductive case proceed by cases on the last rule used.

$\Leftarrow$ **base case** in this case the rule used can be one of the following $Out, EInp, Tau$ which are also in R_2 so a derivation of $P \xrightarrow{\alpha}_3 P^{'}$ is also a derivation of $P \xrightarrow{\alpha}_2 P^{'}$

inductive case :

- the last rule used can be one in $R_2 \cap R_3 = \{Res, Opn\}$, this goes like in the previous proof for the opposite direction with the transition numbers swapped.
- the last rule used can be one of Par, Sum or $ECom$, in this case we apply the inductive hypothesis to the premises and the apply the appropriate rule: $ParL$, $SumL$ or $EComL$. For example

$$\textsc{Par}\ \frac{P \xrightarrow{\alpha}_3 P^{'} \qquad bn(\alpha) \cap fn(Q) = \emptyset}{P|Q \xrightarrow{\alpha}_3 P^{'}|Q}$$

now we apply the inductive hypothesis to $P \xrightarrow{\alpha}_3 P^{'}$ and get $P \xrightarrow{\alpha}_2 P^{''}$ for a $P^{''}$ such that $P^{'} \equiv P^{''}$. The proof we want is

$$\textsc{ParL}\ \frac{P \xrightarrow{\alpha}_2 P^{''} \qquad bn(\alpha) \cap fn(Q) = \emptyset}{P|Q \xrightarrow{\alpha}_2 P|Q^{''}}$$

and $Q^{''}|P \equiv P|Q^{'}$

- if the last rule is Str, then we have

$$\textsc{Str}\ \frac{P \equiv Q \qquad Q \xrightarrow{\alpha}_3 P^{'}}{P \xrightarrow{\alpha}_3 P^{'}}$$

we proceed by cases on the premise $Q \xrightarrow{\alpha}_3 P^{'}$. In the cases of prefix we can just use the appropriate prefix rule of R_2 and get rid of the Str. In the other cases we can move upward the occurrence of Str, after that we have one or two smaller derivation trees that are suitable to application of the inductive hypothesis and finally we apply some appropriate rules in R_2.

Out Since we are using the rule Out, $Q = \overline{x}y.Q_1$ for some Q_1. $Q \equiv P$ means for the inversion lemma for structural congruence that $P = \overline{x}y.P_1$ for some $P_1 \equiv Q_1$. The last part of the derivation tree is

$$\textsc{Str}\ \frac{\overline{x}y.P_1 \equiv \overline{x}y.Q_1 \qquad \textsc{Out}\ \dfrac{}{\overline{x}y.Q_1 \xrightarrow{\overline{x}y}_3 Q_1}}{\overline{x}y.P_1 \xrightarrow{\overline{x}y}_3 Q_1}$$

So we get

$$\textsc{Out}\ \frac{}{\overline{x}y.P_1 \xrightarrow{\overline{x}y}_2 P_1}$$

where $P_1 \equiv Q_1$

Tau this is very similar to the previous case

EInp Since we are using the rule $EInp$, $Q = x(y).Q_1$ for some Q_1. From $Q \equiv P$ using the inversion lemma for structural congruence we can have two cases:

- $P = x(y).P_1$ for some $P_1 \equiv Q_1$. The last part of the derivation tree is

$$\textsc{Str}\ \frac{x(y).P_1 \equiv x(y).Q_1 \qquad \textsc{EInp}\ \dfrac{}{x(y).Q_1 \xrightarrow{xw}_3 Q_1\{w/y\}}}{x(y).P_1 \xrightarrow{xw}_3 Q_1\{w/y\}}$$

So we get

$$\textsc{EInp}\ \frac{}{x(y).P_1 \xrightarrow{xw}_2 P_1\{w/y\}}$$

where $P_1 \equiv Q_1$ implies $P_1\{w/y\} \equiv Q_1\{w/y\}$

- $P = x(z).P_1$ for some $P_1 \equiv Q_1\{z/y\}$. The last part of the derivation tree is

$$\textsc{Str}\ \frac{x(z).P_1 \equiv x(y).Q_1 \qquad \textsc{EInp}\ \frac{}{x(y).Q_1 \xrightarrow{xw}_3 Q_1\{w/y\}}}{x(z).P_1 \xrightarrow{xw}_3 Q_1\{w/y\}}$$

  So we get

$$\textsc{EInp}\ \frac{}{x(z).P_1 \xrightarrow{xw}_2 P_1\{w/z\}}$$

  where $P_1 \equiv Q_1\{z/y\}$ implies $P_1\{w/z\} \equiv Q_1\{z/y\}\{w/z\} \equiv Q_1\{w/y\}$

Par Since we are using the rule *Par*, $Q = Q_1|Q_2$ for some Q_1, Q_2. $Q \equiv P$ means for the inversion lemma for structural congruence that $P = P_1|P_2$ for some P_1, P_2 such that $P_1 \equiv Q_1$ and $P_2 \equiv Q_2$. The last part of the derivation tree is

$$\textsc{Str}\ \frac{P_1|P_2 \equiv Q_1|Q_2 \qquad \textsc{Par}\ \frac{Q_1 \xrightarrow{\alpha}_3 Q_1^{'} \qquad bn(\alpha) \cap fn(Q_2) = \emptyset}{Q_1|Q_2 \xrightarrow{\alpha}_3 Q_1^{'}|Q_2}}{P_1|P_2 \xrightarrow{\alpha}_3 Q_1^{'}|Q_2}$$

the first step is the creation of this proof tree:

$$\textsc{Str}\ \frac{P_1 \equiv Q_1 \qquad Q_1 \xrightarrow{\alpha}_3 Q_1^{'}}{P_1 \xrightarrow{\alpha}_3 Q_1^{'}}$$

which is smaller then the inductive case, so we apply the inductive hypothesis and get $P_1 \xrightarrow{\alpha}_2 Q_1^{''}$ where $Q_1^{'} \equiv Q_1^{''}$. The last step is

$$\textsc{ParL}\ \frac{P_1 \xrightarrow{\alpha}_2 Q_1^{''} \qquad bn(\alpha) \cap fn(P_2) = \emptyset}{P_1|P_2 \xrightarrow{\alpha}_2 Q_1^{''}|P_2}$$

Sum this case is very similar to the previous.

ECom this case is also similar to the *Par* case.

Res Since we are using the rule *Res*, $Q = (\nu z)Q_1$ for some Q_1 and some z. $(\nu z)Q_1 \equiv P$ means thanks to the inversion lemma for structural congruence that one of the following cases holds:

- $P = (\nu z)P_1$ for some P_1 such that $P_1 \equiv Q_1$. The last part of the derivation tree is

$$\textsc{Str}\ \frac{(\nu z)P_1 \equiv (\nu z)Q_1 \qquad \textsc{Res}\ \frac{Q_1 \xrightarrow{\alpha}_3 Q_1^{'} \qquad z \notin n(\alpha)}{(\nu z)Q_1 \xrightarrow{\alpha}_3 (\nu z)Q_1^{'}}}{(\nu z)P_1 \xrightarrow{\alpha}_3 (\nu z)Q_1^{'}}$$

  first we create the following proof:

$$\textsc{Str}\ \frac{P_1 \equiv Q_1 \qquad Q_1 \xrightarrow{\alpha}_3 Q_1^{'}}{P_1 \xrightarrow{\alpha}_3 Q_1^{'}}$$

  now we can apply the inductive hypothesis and get $P_1 \xrightarrow{\alpha}_2 Q_1^{''}$ where $Q_1^{'} \equiv Q_1^{''}$. The last step is

$$\textsc{Res}\ \frac{P_1 \xrightarrow{\alpha}_2 Q_1^{''} \qquad z \notin n(\alpha)}{(\nu z)P_1 \xrightarrow{\alpha}_2 (\nu z)Q_1^{''}}$$

- $P = (\nu y)P_1$ for some P_1 such that $P_1\{z/y\} \equiv Q_1$. The last part of the derivation tree is

$$\textsc{Str}\ \dfrac{(\nu y)P_1 \equiv (\nu z)Q_1 \qquad \textsc{Res}\ \dfrac{Q_1 \xrightarrow{\alpha}_3 Q_1^{'} \qquad z \notin n(\alpha)}{(\nu z)Q_1 \xrightarrow{\alpha}_3 (\nu z)Q_1^{'}}}{(\nu y)P_1 \xrightarrow{\alpha}_3 (\nu z)Q_1^{'}}$$

we create the following proof of $P_1\{z/y\} \xrightarrow{\alpha}_3 Q_1^{'}$:

$$\textsc{Str}\ \dfrac{P_1\{z/y\} \equiv Q_1 \qquad Q_1 \xrightarrow{\alpha}_3 Q_1^{'}}{P_1\{z/y\} \xrightarrow{\alpha}_3 Q_1^{'}}$$

this proof tree is shorter then the one of $(\nu y)P_1 \xrightarrow{\alpha}_3 (\nu z)Q_1^{'}$ so we can apply the inductive hypothesis and get that there exists a process $Q_1^{''}$ such that

$$P_1\{z/y\} \xrightarrow{\alpha}_2 Q_1^{''} \textit{ and } Q_1^{''} \equiv Q_1^{'}$$

now we can apply the rules *Res* and *Alp* to get the desired proof tree:

$$\textsc{Alp}\ \dfrac{(\nu z)P_1\{z/y\} \equiv_\alpha (\nu y)P_1 \qquad \textsc{Res}\ \dfrac{P_1\{z/y\} \xrightarrow{\alpha}_2 Q_1^{''} \qquad z \notin (\alpha)}{(\nu z)P_1\{z/y\} \xrightarrow{\alpha}_2 (\nu z)Q_1^{''}}}{(\nu y)P_1 \xrightarrow{\alpha}_2 (\nu z)Q_1^{''}}$$

Opn Since we are using the rule *Opn*, $Q = (\nu z)Q_1$ for some Q_1. $(\nu z)Q_1 \equiv P$ means for the inversion lemma for structural congruence that

- $P = (\nu z)P_1$ for some P_1 such that $P_1 \equiv Q_1$. The last part of the derivation tree is

$$\textsc{Str}\ \dfrac{(\nu z)P_1 \equiv (\nu z)Q_1 \qquad \textsc{Opn}\ \dfrac{Q_1 \xrightarrow{\overline{x}z} Q_1^{'} \qquad z \neq x}{(\nu z)Q_1 \xrightarrow{\overline{x}(z)} Q_1^{'}}}{(\nu z)P_1 \xrightarrow{\overline{x}(z)} Q_1^{'}}$$

first:

$$\textsc{Str}\ \dfrac{P_1 \equiv Q_1 \qquad Q_1 \xrightarrow{\overline{x}z}_3 Q_1^{'}}{P_1 \xrightarrow{\overline{x}z}_3 Q_1^{'}}$$

then we apply the inductive hypothesis and get $P_1 \xrightarrow{\overline{x}z}_2 Q_1^{''}$ where $Q_1^{'} \equiv Q_1^{''}$. The last step is

$$\textsc{Res}\ \dfrac{P_1 \xrightarrow{\overline{x}z}_2 Q_1^{''} \qquad z \neq x}{(\nu z)P_1 \xrightarrow{\overline{x}z}_2 Q_1^{''}}$$

- $P = (\nu z)P_1$ for some P_1 such that $P_1 \equiv Q_1$. The last part of the derivation tree is

$$\textsc{Str}\ \dfrac{(\nu z)P_1 \equiv (\nu z)Q_1 \qquad \textsc{Opn}\ \dfrac{Q_1 \xrightarrow{\overline{x}z} Q_1^{'} \qquad z \neq x}{(\nu z)Q_1 \xrightarrow{\overline{x}(z)} Q_1^{'}}}{(\nu z)P_1 \xrightarrow{\overline{x}(z)} Q_1^{'}}$$

the first step is:

$$\textsc{Str}\ \dfrac{P_1 \equiv Q_1 \qquad Q_1 \xrightarrow{\overline{x}z}_3 Q_1^{'}}{P_1 \xrightarrow{\overline{x}z}_3 Q_1^{'}}$$

then we apply the inductive hypothesis and get $P_1 \xrightarrow{\overline{x}z}_2 Q_1^{''}$ where $Q_1^{'} \equiv Q_1^{''}$. The last step is

$$\textsc{Res}\ \dfrac{P_1 \xrightarrow{\overline{x}z}_2 Q_1^{''} \qquad z \neq x}{(\nu z)P_1 \xrightarrow{\overline{x}z}_2 Q_1^{''}}$$

- $P = (\nu y)P_1$ for some P_1 such that $P_1\{z/y\} \equiv Q_1$. The last part of the derivation tree is

$$\textsc{Str}\ \frac{(\nu y)P_1 \equiv (\nu z)Q_1 \qquad \textsc{Opn}\ \dfrac{Q_1 \xrightarrow{\overline{x}z}_3 Q_1^{'} \qquad z \neq x}{(\nu z)Q_1 \xrightarrow{\overline{x}(z)}_3 Q_1^{'}}}{(\nu y)P_1 \xrightarrow{\overline{x}(z)}_3 Q_1^{'}}$$

we can create the following proof of $P_1\{z/y\} \xrightarrow{\overline{x}z}_3 Q_1^{'}$:

$$\textsc{Str}\ \frac{P_1\{z/y\} \equiv Q_1 \qquad Q_1 \xrightarrow{\overline{x}z}_3 Q_1^{'}}{P_1\{z/y\} \xrightarrow{\overline{x}z}_3 Q_1^{'}}$$

this proof tree is shorter then the one of $(\nu y)P_1 \xrightarrow{\overline{x}(z)}_3 Q_1^{'}$ so we can apply the inductive hypothesis and get that there exists a process $Q_1^{''}$ such that

$$Q_1^{''} \equiv Q_1^{'} \textit{ and } P_1\{z/y\} \xrightarrow{\overline{x}z}_2 Q_1^{''}$$

so now we only need to apply the rules *Opn* and *Alp*:

$$\textsc{Alp}\ \frac{(\nu y)P_1 \equiv_\alpha (\nu z)P_1\{z/y\} \qquad \textsc{Opn}\ \dfrac{P_1\{z/y\} \xrightarrow{\overline{x}z}_2 Q_1^{''} \qquad z \neq x}{(\nu z)P_1\{z/y\} \xrightarrow{\overline{x}(z)}_3 Q_1^{''}}}{(\nu y)P_1 \xrightarrow{\overline{x}(z)}_2 Q_1^{''}}$$

□

2.5.2 Equivalence of the late semantics

2.6 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.

2.6.1 Late bisimilarity

Definition 2.6.1. A *strong late bisimulation*(according to [6]) is a binary simmetric relation $\mathbf{S}$ on processes such that for each process P and Q, $P\mathbf{S}Q$ implies:

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

P and Q are *late bisimilar* written $P \dot{\sim}_L Q$ if there exists a strong late bisimulation $\mathbf{S}$ such that $P\mathbf{S}Q$.

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

$$a(u).0|\bar{b}v.0 \dot{\sim}_L a(u).\bar{b}v.0 + \bar{b}v.a(u).0$$

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

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

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 2.6.1. If $P\dot{\sim}Q$ and σ is injective then $P\sigma\dot{\sim}Q\sigma$

Proposition 2.6.2. $\dot{\sim}_L$ is an equivalence

Proposition 2.6.3. $\dot{\sim}_L$ is preserved by all operators except input prefix

Definition 2.6.2. Two processes P and Q are *strong late equivalent* written $P \sim_L Q$ is for each substitution σ $P\sigma\dot{\sim}_L Q\sigma$

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

2.6.2 Early bisimilarity

Definition 2.6.3. A *strong early bisimulation*(according to [6])is a symmetric binary relation $\mathbf{S}$ on processes such that for each process P and Q: $P\mathbf{S}Q$, $P \xrightarrow{\alpha}_E P^{'}$ and $bn(\alpha) \cap (fn(P) \cup fn(Q)) = \emptyset$ implies that there exists $Q^{'}$ such that $Q \xrightarrow{\alpha}_E Q^{'}$ and $P^{'}\mathbf{S}Q^{'}$. P and Q are *early bisimilar* written $P\dot{\sim}_E Q$ if there exists a strong early bisimulation $\mathbf{S}$ such that $P\mathbf{S}Q$

Definition 2.6.4. Two processes P and Q are *strong early equivalent* written $P \sim_E Q$ if for each substitution σ $P\sigma\dot{\sim}_E Q\sigma$

2.6.3 Congruence

Definition 2.6.5. We say that two agents P and Q are *strongly congruent*, written $P \sim Q$ if

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

Proposition 2.6.4. Strong congruence is the largest congruence in bisimilarity.

2.6.4 Open bisimilarity

Definition 2.6.6. 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 2.6.7. An *strong open simulation*(according to [6]) 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 $P\dot{\sim}_O^D Q$ 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 $P\dot{\sim}_O D$.

Chapter 3

Multi π calculus with strong output

3.1 Syntax

As we did whit π calculus, we suppose that we have a countable set of names $\mathbb{N}$, ranged over by lower case letters $a, b, \cdots, z$. 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 $P, Q, \cdots$. A multi π process, in addiction to the same actions of a π process, can perform also a strong prefix output:

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

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

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

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:

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

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 $\alpha, \alpha_1, \alpha_2, \cdots$ to range over the set of actions, we use $\sigma, \sigma_1, \sigma_2, \cdots$ 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.

$$\textbf{Out}\ \frac{}{\overline{x}y.P \xrightarrow{\overline{x}y} P} \qquad \textbf{EInp}\ \frac{}{x(y).P \xrightarrow{xz} P\{z/y\}} \qquad \textbf{Tau}\ \frac{}{\tau.P \xrightarrow{\tau} P}$$

$$\textbf{SOutSeq}\ \frac{P \xrightarrow{\sigma} Q \qquad |\sigma| > 1}{\underline{\overline{x}y}.P \xrightarrow{\overline{x}y\cdot\sigma} Q} \qquad \textbf{SOut}\ \frac{P \xrightarrow{\alpha} Q \qquad \alpha\ output}{\underline{\overline{x}y}.P \xrightarrow{\overline{x}y\cdot\alpha} Q} \qquad \textbf{SOutTau}\ \frac{P \xrightarrow{\tau} Q}{\underline{\overline{x}y}.P \xrightarrow{\overline{x}y} Q}$$

$$\textbf{EComSng}\ \frac{P \xrightarrow{\overline{x}y} P' \qquad Q \xrightarrow{xy} Q'}{P|Q \xrightarrow{\tau} P'|Q'} \qquad \textbf{Cong}\ \frac{P \equiv P' \qquad P' \xrightarrow{\sigma} Q}{P \xrightarrow{\sigma} Q} \qquad \textbf{Par}\ \frac{P \xrightarrow{\sigma} P'}{P|Q \xrightarrow{\sigma} P'|Q}$$

$$\textbf{EComSeq}\ \frac{P \xrightarrow{\overline{x}y\cdot\sigma} P' \qquad Q \xrightarrow{xy} Q'}{P|Q \xrightarrow{\sigma} P'|Q'} \qquad \textbf{Sum}\ \frac{P \xrightarrow{\sigma} P'}{P+Q \xrightarrow{\sigma} P'}$$

$$\textbf{Res}\ \frac{P \xrightarrow{\sigma} P' \qquad z \notin n(\alpha)}{(\nu)zP \xrightarrow{\sigma} (\nu)zP'}$$

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.

$$\textbf{Res}\ \frac{x \notin n(\tau) \qquad \textbf{EComSeq}\ \dfrac{\underline{\overline{x}y}.\overline{x}y.0|x(y).0 \xrightarrow{\overline{x}y} 0|0 \qquad \textbf{Inp}\ \dfrac{}{x(y).0 \xrightarrow{xy} 0}}{((\underline{\overline{x}y}.\overline{x}y.0|x(y).0)|x(y).0) \xrightarrow{\tau} ((0|0)|0)}}{(\nu x)((\underline{\overline{x}y}.\overline{x}y.0|x(y).0)|x(y).0) \xrightarrow{\tau} (\nu x)((0|0)|0)}$$

$$\textbf{EComSng}\ \frac{\textbf{SOut}\ \dfrac{\textbf{Out}\ \dfrac{}{\overline{x}y.0 \xrightarrow{\overline{x}y} 0}}{\underline{\overline{x}y}.\overline{x}y.0 \xrightarrow{\overline{x}y\cdot\overline{x}y} 0} \qquad x(y).0 \xrightarrow{xy} 0}{\underline{\overline{x}y}.\overline{x}y.0|x(y).0 \xrightarrow{\overline{x}y} 0|0}$$

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 Late operational semantic with structural congruence

Definition 3.2.2. The *late transition relation with structural congruence* is the smallest relation induced by the rules in table 3.2.

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

$$\textbf{Res}\ \frac{x \notin n(\tau) \qquad \textbf{LComSeq}\ \dfrac{\underline{\overline{x}y}.\overline{x}y.0|x(y).0 \xrightarrow{\overline{x}y} 0|0 \qquad \textbf{Pref}\ \dfrac{}{x(y).0 \xrightarrow{x(y)} 0}}{((\underline{\overline{x}y}.\overline{x}y.0|x(y).0)|x(y).0) \xrightarrow{\tau} ((0|0)|0)}}{(\nu x)((\underline{\overline{x}y}.\overline{x}y.0|x(y).0)|x(y).0) \xrightarrow{\tau} (\nu x)((0|0)|0)}$$

$$\textbf{Pref}\ \frac{\alpha\ not\ a\ strong\ prefix}{\alpha.P \xrightarrow{\alpha} P} \qquad \textbf{Par}\ \frac{P \xrightarrow{\sigma} P' \qquad bn(\sigma) \cap fn(Q) = \emptyset}{P|Q \xrightarrow{\sigma} P'|Q}$$

$$\textbf{SOut}\ \frac{P \xrightarrow{\alpha} P' \qquad \alpha\ output}{\underline{\overline{x}y}.P \xrightarrow{\overline{x}y\cdot\alpha} P'} \qquad \textbf{LComSeq}\ \frac{P \xrightarrow{xy} P' \qquad Q \xrightarrow{\overline{x}z\cdot\sigma} Q' \qquad z \notin fn(P)}{P|Q \xrightarrow{\sigma} P'\{z/y\}|Q'}$$

$$\textbf{Sum}\ \frac{P \xrightarrow{\sigma} P'}{P + Q \xrightarrow{\sigma} P'} \qquad \textbf{Cong}\ \frac{P \equiv P' \qquad P' \xrightarrow{\sigma} Q}{P \xrightarrow{\sigma} Q}$$

$$\textbf{Res}\ \frac{P \xrightarrow{\sigma} P' z \notin n(\alpha)}{(\nu)zP \xrightarrow{\sigma} (\nu)zP'} \qquad \textbf{LComSng}\ \frac{P \xrightarrow{xy} P' \qquad Q \xrightarrow{\overline{x}z} Q' \qquad z \notin fn(P)}{P|Q \xrightarrow{\tau} P'\{z/y\}|Q'}$$

$$\textbf{SOutSeq}\ \frac{P \xrightarrow{\sigma} P' \qquad |\sigma| > 1}{\underline{\overline{x}y}.P \xrightarrow{\overline{x}y\cdot\sigma} P'}$$

Table 3.2: Multiπ late semantic with structural congruence

$$\textbf{LComSng}\ \frac{\textbf{SOut}\ \dfrac{\textbf{Pref}\ \dfrac{}{\overline{x}y.0 \xrightarrow{\overline{x}y} 0}}{\underline{\overline{x}y}.\overline{x}y.0 \xrightarrow{\overline{x}y\cdot\overline{x}y} 0} \qquad \textbf{Pref}\ \dfrac{}{x(y).0 \xrightarrow{x(y)} 0}}{\underline{\overline{x}y}.\overline{x}y.0|x(y).0 \xrightarrow{\overline{x}y} 0|0}$$

3.2.3 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 output only), we call this language low multi π. The low multi π is the multi π enriched with a marked or intermediate process $*P$:

$$P, Q ::= 0 \mid \pi.P \mid P|Q \mid P + Q \mid (\nu x)P \mid A(x_1, \cdots, x_n) \mid *P$$

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

Definition 3.2.3. The low level transition relation is the smallest relation induced by the rules in table 3.3 in which P stands for a process without mark, L stands for a process with mark and S can stand for both.

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

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

Proof. DA FARE □

Definition 3.2.4. Let P, Q be unmarked processes and $L_1, \cdots, L_{k-1}$ marked processes. We define the derivation relation $\rightarrow_s$ in the following way:

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

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:

$$\textbf{Out}\ \frac{}{\overline{x}y.P \xmapsto{\overline{x}y} P} \qquad \textbf{EInp}\ \frac{}{x(y).P \xmapsto{xz} P\{z/y\}} \qquad \textbf{SOutLow}\ \frac{}{\underline{\overline{x}y}.P \xmapsto{\overline{x}y} *P}$$

$$\textbf{Tau}\ \frac{}{\tau.P \xmapsto{\tau} P} \qquad \textbf{Res}\ \frac{S \xmapsto{\gamma} S^{'} \qquad y \notin n(\gamma)}{(\nu y)S \xmapsto{\gamma} (\nu y)S^{'}} \qquad \textbf{Star}\ \frac{S \xmapsto{\overline{x}y} S^{'}}{*S \xmapsto{\overline{x}y} S^{'}}$$

$$\textbf{Com1}\ \frac{P \xmapsto{\overline{x}y} P^{'} \qquad Q \xmapsto{xy} Q^{'}}{P|Q \xmapsto{\tau} P^{'}|Q^{'}} \qquad \textbf{Cong}\ \frac{P \equiv P^{'} \qquad P^{'} \xmapsto{\gamma} S}{P \xmapsto{\gamma} S} \qquad \textbf{Par1L}\ \frac{S \xmapsto{\gamma} S^{'}}{S|Q \xmapsto{\gamma} S^{'}|Q}$$

$$\textbf{Com3L}\ \frac{P \xmapsto{\overline{x}y} L \qquad Q \xmapsto{xy} Q^{'}}{P|Q \xmapsto{\epsilon} L|Q^{'}} \qquad \textbf{Com3R}\ \frac{P \xmapsto{xy} P^{'} \qquad Q \xmapsto{\overline{x}y} L}{P|Q \xmapsto{\epsilon} P^{'}|L} \qquad \textbf{Par1R}\ \frac{S \xmapsto{\gamma} S^{'}}{Q|S \xmapsto{\gamma} Q|S^{'}}$$

$$\textbf{Com4L}\ \frac{L \xmapsto{\overline{x}y} Q \qquad P \xmapsto{xy} P^{'}}{L|P \xmapsto{\tau} Q|P^{'}} \qquad \textbf{Com4R}\ \frac{P \xmapsto{xy} P^{'} \qquad L \xmapsto{\overline{x}y} Q}{P|L \xmapsto{\tau} P^{'}|Q} \qquad \textbf{Sum}\ \frac{P \xmapsto{\gamma} S}{P+Q \xmapsto{\gamma} S}$$

$$\textbf{Com2L}\ \frac{L_1 \xmapsto{\overline{x}y} L_1^{'} \qquad P \xmapsto{xy} Q}{L_1|P \xmapsto{\epsilon} L_1^{'}|Q} \qquad \textbf{Com2R}\ \frac{P \xmapsto{xy} Q \qquad L_1 \xmapsto{\overline{x}y} L_1^{'}}{P|L_1 \xmapsto{\epsilon} Q|L_1^{'}}$$

Table 3.3: Low multi π early semantic with structural congruence

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

Example Multi-parti synchronization

$$\textbf{Par1L}\ \dfrac{\textbf{Par1L}\ \dfrac{\textbf{Com2R}\ \dfrac{\textbf{SOut}\ \dfrac{}{\underline{\overline{x}a}.\overline{x}b.\overline{x}c.P \xmapsto{\overline{x}a} *\underline{\overline{x}b}.\overline{x}c.P} \qquad \textbf{Inp}\ \dfrac{}{x(d).Q \xrightarrow{xa} Q\{a/d\}}}{\underline{\overline{x}a}.\underline{\overline{x}b}.\overline{x}c.P|x(d).Q \xmapsto{\epsilon} *\underline{\overline{x}b}.\overline{x}c.P|Q\{a/d\}}}{\underline{\overline{x}a}.\underline{\overline{x}b}.\overline{x}c.P|x(d).Q|x(e).R \xmapsto{\epsilon} *\underline{\overline{x}b}.\overline{x}c.P|Q\{a/d\}|x(e).R}}{\underline{\overline{x}a}.\underline{\overline{x}b}.\overline{x}c.P|x(d).Q|x(e).R|x(f).S \xmapsto{\epsilon} *\underline{\overline{x}b}.\overline{x}c.P|Q\{a/d\}|x(e).R|x(f).S}$$

$$\textbf{Par1L}\ \dfrac{\textbf{Com2R}\ \dfrac{\textbf{Par1L}\ \dfrac{\textbf{Star}\ \dfrac{\textbf{SOut}\ \dfrac{}{\underline{\overline{x}b}.\overline{x}c.P \xmapsto{\overline{x}b} *\overline{x}c.P}}{*\underline{\overline{x}b}.\overline{x}c.P \xmapsto{\overline{x}b} *\overline{x}c.P}}{*\underline{\overline{x}b}.\overline{x}c.P|Q\{a/d\} \xmapsto{\overline{x}b} *\overline{x}c.P|Q\{a/d\}} \qquad \textbf{EInp}\ \dfrac{}{x(e).R \xmapsto{xb} R\{b/e\}}}{*\underline{\overline{x}b}.\overline{x}c.P|Q\{a/d\}|x(e).R \xmapsto{\epsilon} *\overline{x}c.P|Q\{a/d\}|R\{b/e\}}}{*\underline{\overline{x}b}.\overline{x}c.P|Q\{a/d\}|x(e).R|x(f).S \xmapsto{\epsilon} *\overline{x}c.P|Q\{a/d\}|R\{b/e\}|x(f).S}$$

$$\textbf{Com4R}\ \dfrac{\textbf{Par1L}\ \dfrac{\textbf{Par1L}\ \dfrac{\textbf{Star}\ \dfrac{\textbf{Out}\ \dfrac{}{\overline{x}c.P \xmapsto{\overline{x}c} P}}{*\overline{x}c.P \xmapsto{\overline{x}c} P}}{*\overline{x}c.P|Q\{a/d\} \xmapsto{\overline{x}c} P|Q\{a/d\}}}{*\overline{x}c.P|Q\{a/d\}|R\{b/e\} \xmapsto{\overline{x}c} P|Q\{a/d\}|R\{b/e\}} \qquad \textbf{EInp}\ \dfrac{}{x(f).S \xmapsto{xc} R\{c/f\}}}{*\overline{x}c.P|Q\{a/d\}|R\{b/e\}|x(f).S \xmapsto{\tau} P|Q\{a/d\}|R\{b/e\}|S\{c/f\}}$$

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

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

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 4.3 so we can derive $P \overset{\sigma}{\longmapsto} Q$.

inductive case

If the depth is greater than one then the last rule used in the derivation can be:

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

$$\textbf{SOutSeq}\ \frac{P_1 \xrightarrow{\sigma} Q \qquad |\sigma| > 1}{\underline{\overline{x}y}.P_1 \xrightarrow{\overline{x}y\cdot\sigma} Q}$$

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

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

then a proof of the conclusion follows from:

$$\textbf{SOutLow}\ \frac{}{\underline{\overline{x}y}.P_1 \overset{\overline{x}y}{\longmapsto} *P_1} \qquad \textbf{Star}\ \frac{P_1 \overset{\gamma_1}{\longmapsto} L_1}{*P_1 \overset{\gamma_1}{\longmapsto} L_1}$$

$SOut$: this case is similar to the previous.

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

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

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

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

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

A proof of the conclusion is:

$$\textbf{Sum}\ \frac{P_1 \overset{\gamma_1}{\longmapsto} L_1}{P_1 + P_2 \overset{\gamma_1}{\longmapsto} L_1}$$

$Cong$: this case is similar to the previous.

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

$$\textbf{Com}\ \frac{P_1 \xrightarrow{\overline{x}y} P_1' \qquad Q_1 \xrightarrow{xy} Q_1'}{P_1|Q_1 \xrightarrow{\tau} P_1'|Q_1'}$$

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

$$P_1 \overset{\gamma_1}{\longmapsto} L_1 \overset{\gamma_2}{\longmapsto} L_2 \cdots L_{k-1} \overset{\gamma_k}{\longmapsto} L_k \overset{\gamma_{k+1}}{\longmapsto} P_1' \quad \text{and} \quad \gamma_1 \cdot \ldots \cdot \gamma_{k+1} = \overline{x}y$$

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

$$Q_1 \xmapsto{\delta_1} R_1 \xmapsto{\delta_2} R_2 \cdots R_{h-1} \xmapsto{\delta_h} R_h \xmapsto{\delta_{h+1}} Q_1^{'} \quad \text{and} \quad \delta_1 \cdot \ldots \cdot \delta_{h+1} = xy$$

For lemma 3.2.2 there cannot be an input action in a transition involving marked processes so h must be 0 and $Q_1 \xmapsto{\delta_1} Q_1^{'}$ with $\delta_1 = xy$. For the same lemma just one of the γs is $\overline{x}y$ and the others are ϵ. We can have three different cases now:

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

$$P_1|Q_1 \xmapsto{\tau} L_1|Q_1^{'} \xmapsto{\epsilon} L_2|Q_1^{'} \cdots \xmapsto{\epsilon} L_k|Q_1^{'} \xmapsto{\epsilon} P_1^{'}|Q_1^{'}$$

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

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

$$P_1|Q_1 \xmapsto{\epsilon} L_1|Q_1 \cdots \xmapsto{\epsilon} L_{i-1}|Q_1 \xmapsto{\tau} L_i|Q_1^{'} \xmapsto{\epsilon} L_{i+1}|Q_1^{'} \cdots \xmapsto{\epsilon} L_k|Q_1^{'} \xmapsto{\epsilon} P_1^{'}|Q_1^{'}$$

we derive the transaction $L_{i-1}|Q_1 \xmapsto{\tau} L_i|Q_1^{'}$ with rule $Com3L$, whether for the other transactions we use the rule $Par1L$.

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

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

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

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

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

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:

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

Par : this case is similar to the previous.

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

$$\textbf{EComSeq}\ \frac{P_1 \xrightarrow{\overline{x}y\cdot\sigma} P_1^{'} \qquad Q_1 \xrightarrow{xy} Q_1^{'}}{P_1|Q_1 \xrightarrow{\sigma} P_1^{'}|Q_1^{'}}$$

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

$$P_1 \xmapsto{\gamma_1} L_1 \xmapsto{\gamma_2} L_2 \cdots L_{k-1} \xmapsto{\gamma_k} L_k \xmapsto{\gamma_{k+1}} P_1^{'} \quad \text{and} \quad \gamma_1 \cdot \ldots \cdot \gamma_{k+1} = \overline{x}y \cdot \sigma$$

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

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

$$P_1|Q_1 \xmapsto{\tau} L_1|Q_1^{'} \xmapsto{\gamma_2} L_2|Q_1^{'} \cdots \xmapsto{\gamma_k} L_k|Q_1^{'} \xmapsto{\gamma_{k+1}} P_1^{'}|Q_1^{'}$$

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

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

$$P_1|Q_1 \xmapsto{\epsilon} L_1|Q_1 \cdots \xmapsto{\epsilon} L_{i-1}|Q_1 \xmapsto{\tau} L_i|Q_1^{'} \xmapsto{\gamma_{i+1}} L_{i+1}|Q_1^{'} \cdots \xmapsto{\gamma_k} L_k|Q_1^{'} \xmapsto{\gamma_{k+1}} P_1^{'}|Q_1^{'}$$

we derive the transition $L_{i-1}|Q_1 \xmapsto{\tau} L_i|Q_1^{'}$ with rule $Com3L$, whether for the other transactions of the premises we use the rule $Par1L$.

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

$\square$

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

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

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

Example Interleaving

$$\textbf{Par1L}\ \dfrac{\textbf{Com3L}\ \dfrac{\textbf{SOut}\ \dfrac{}{\underline{\overline{x}y}.\underline{\overline{a}b}.\overline{x}y.0 \xmapsto{\overline{x}y} \ast\underline{\overline{a}b}.\overline{x}y.0} \qquad \textbf{EInp}\ \dfrac{}{x(y).0 \xmapsto{xy} 0}}{\underline{\overline{x}y}.\underline{\overline{a}b}.\overline{x}y.0|x(y).0 \xmapsto{\epsilon} \ast\underline{\overline{a}b}.\overline{x}y.0|0}}{\underline{\overline{x}y}.\underline{\overline{a}b}.\overline{x}y.0|x(y).0|x(y).0 \xmapsto{\epsilon} \ast\underline{\overline{a}b}.\overline{x}y.0|0|x(y).0}$$

$$\textbf{Par1L}\ \dfrac{\textbf{Par1L}\ \dfrac{\textbf{Star}\ \dfrac{\textbf{SOut}\ \dfrac{}{\underline{\overline{a}b}.\overline{x}y.0 \xmapsto{\overline{a}b} \ast\overline{x}y.0}}{\ast\underline{\overline{a}b}.\overline{x}y.0 \xmapsto{\overline{a}b} \ast\overline{x}y.0}}{\ast\underline{\overline{a}b}.\overline{x}y.0|0 \xmapsto{\overline{a}b} \ast\overline{x}y.0|0}}{\ast\underline{\overline{a}b}.\overline{x}y.0|0|x(y).0 \xmapsto{\overline{a}b} \ast\overline{x}y.0|0|x(y).0}$$

$$\textbf{Com3L}\ \dfrac{\textbf{Par1L}\ \dfrac{\textbf{Star}\ \dfrac{\textbf{Out}\ \dfrac{}{\overline{x}y.0 \xmapsto{\overline{x}y} 0}}{\ast\overline{x}y.0 \xmapsto{\overline{x}y} 0}}{\ast\overline{x}y.0|0 \xmapsto{\overline{x}y} 0|0} \qquad \textbf{EInp}\ \dfrac{}{x(y).0 \xmapsto{xy} 0}}{\ast\overline{x}y.0|0|x(y).0 \xmapsto{\tau} 0|0|0}$$

this prove:

$$\underline{\overline{x}y}.\underline{\overline{a}b}.\overline{x}y.0|x(y).0|x(y).0 \xmapsto{\epsilon} \ast\underline{\overline{a}b}.\overline{x}y.0|0|x(y).0 \xmapsto{\overline{a}b} \ast\overline{x}y.0|0|x(y).0 \xmapsto{\tau} 0|0|0$$

but there is no way to prove

$$\underline{\overline{x}y}.\underline{\overline{a}b}.\overline{x}y.0|x(y).0|x(y).0 \xrightarrow{\overline{a}b} 0|0|0$$

Example Transactional synchronization

$$\textbf{Com?}\ \dfrac{\textbf{Sout}\ \dfrac{}{\underline{\overline{x}y}.\overline{x}y.0 \xmapsto{\overline{x}y} \ast\overline{x}y.0} \qquad \textbf{EInp}\ \dfrac{}{x(y).x(y).0 \xmapsto{xy} x(y).0}}{\underline{\overline{x}y}.\overline{x}y.0|x(y).x(y).0 \xmapsto{\epsilon} \ast\overline{x}y.0|x(y).0}$$

$$\textbf{Com?}\ \dfrac{\textbf{Star}\ \dfrac{\textbf{Out}\ \dfrac{}{\overline{x}y.0 \xmapsto{\overline{x}y} 0}}{\ast\overline{x}y.0 \xmapsto{\overline{x}y} 0} \qquad \textbf{EInp}\ \dfrac{}{x(y).0 \xmapsto{xy} 0}}{\ast\overline{x}y.0|x(y).0 \xmapsto{\epsilon} 0|0}$$

this prove:

$$\underline{\overline{x}y}.\overline{x}y.0|x(y).x(y).0 \xmapsto{\epsilon} \ast\overline{x}y.0|x(y).0 \xmapsto{\tau} 0|0$$

but we cannot derive

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

There is a much weaker propositions we can prove:

Proposition 3.2.4. Let $\rightarrow$ be the relation defined in table 3.3. Let α be an action. If $P \xmapsto{\alpha} Q$ then $P \xrightarrow{\alpha} Q$.

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

base case in this case the derivation of this transition has depth one. The last(and only) rule used can be: *Out*, *EInp* or *Tau*; these rules are also in table 4.1 so we can derive $P \xrightarrow{\alpha} Q$.

inductive case in this case the last rule in the derivation can be: *Sum*, *Com*1, *Res*, *Par*1*L*, *Par*1*R*, *Cong*:

*Com*1 :

$$\textbf{Com1}\ \frac{P_1 \xmapsto{\overline{x}y} Q_1 \qquad P_2 \xmapsto{xy} Q_2}{P_1|P_2 \xmapsto{\tau} Q_1|Q_2}$$

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

Sum :

$$\textbf{Sum}\ \frac{P_1 \xmapsto{\alpha} Q}{P_1 + P_2 \xmapsto{\alpha} Q}$$

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

Res the first transition is:

$$\textbf{Res}\ \frac{P_1 \xmapsto{\alpha} Q_1 \qquad z \notin n(\gamma_1)}{(\nu z)P_1 \xmapsto{\alpha} (\nu z)Q_1}$$

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 fits our need. We replace the rule *Com*3*L*, *Com*3*R*, *Com*2*L* and *Com*2*R* with:

$$\textbf{Com3LStop}\ \frac{P \xmapsto{\overline{x}y} L \qquad Q \xmapsto{xy} Q'}{P|Q \xmapsto{\epsilon} L|stop(Q')} \qquad \textbf{Com3RStop}\ \frac{P \xmapsto{xy} P' \qquad Q \xmapsto{\overline{x}y} L}{P|Q \xmapsto{\epsilon} stop(P')|L}$$

$$\textbf{Com2LStop}\ \frac{L_1 \xmapsto{\overline{x}y} L_2 \qquad P \xmapsto{xy} Q}{L_1|P \xmapsto{\tau} L_2|stop(Q)} \qquad \textbf{Com2RStop}\ \frac{P \xmapsto{xy} Q \qquad L_1 \xmapsto{\overline{x}y} L_2}{P|L_1 \xmapsto{\tau} stop(Q)|L_2}$$

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

Definition 3.2.5. 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 $L_1, \cdots, L_k$ with $k \geq 0$ such that

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

- If there exist $L_1, \cdots, L_k$ with $k \geq 1$ such that

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

where at most one γ is an output whether all the other γs are ϵ or τ then $P \xrightarrow{\tau} er(Q)$ or if there is an output $\overline{x}y$ in the γs then $P \xrightarrow{\overline{x}y} er(Q)$.

Proof. The proof of the first part of this proposition is almost exactly as the proof of 3.2.3 DA VERIFICARE. The proof of the second part is by induction on the depth of the derivation of the first transition:

base case The last rule in the derivation of $P \xmapsto{\gamma_1} L_1$ can be only *SOutLow*:

$$\textbf{SOutLow}\ \frac{}{\underbrace{\underline{\overline{x}y}.P_1}_{P} \xmapsto{\overline{x}y} \underbrace{*P_1}_{L_1}}$$

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

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

since γ_1 is an output, the other γs 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

$$\textbf{SOutTau}\ \frac{P_1 \xrightarrow{\tau} er(Q)}{\underline{\overline{x}y}.P_1 \xrightarrow{\overline{x}y} er(Q)}$$

inductive case The last rule in the derivation of $P \xmapsto{\gamma_1} L_1$ can be:

Sum the first transition is:

$$\textbf{Sum}\ \frac{P_1 \xmapsto{\gamma_1} L_1}{P_1 + P_2 \xmapsto{\gamma_1} L_1}$$

so we can build the following chain of transition:

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

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

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

Res the first transition is:

$$\textbf{Res}\ \frac{P_1 \xmapsto{\gamma_1} L_1^{'} \qquad z \notin n(\gamma_1)}{(\nu z)P_1 \xmapsto{\gamma_1} (\nu z)L_1^{'}}$$

given that L_1 has a restriction at the top level, all the other intermediate processes $L_2, \cdots, L_k$ 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*.

$$\textbf{Res}\ \frac{L_k^{'} \xmapsto{\gamma_{k+1}} Q^{'} \qquad z \notin n(\tau)}{(\nu z)L_k^{'} \xmapsto{\gamma_{k+1}} (\nu z)Q^{'}} \qquad \textbf{Res}\ \frac{L_i^{'} \xmapsto{\gamma_i} L_{i+1}^{'} \qquad z \notin n(\epsilon)}{(\nu z)L_i^{'} \xmapsto{\gamma_i} (\nu z)L_{i+1}^{'}}$$

we can build the following chain of transitions:

$$P_1 \overset{\gamma_1}{\longmapsto} L_1^{'} \overset{\gamma_2}{\longmapsto} L_2^{'} \cdots L_{k-1}^{'} \overset{\gamma_k}{\longmapsto} L_k^{'} \overset{\gamma_{k+1}}{\longmapsto} Q^{'}$$

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

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

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

$$\textbf{Cong}\ \frac{P^{'} \equiv P \qquad \overset{\gamma_1}{\longmapsto} L_1}{P \overset{\gamma_1}{\longmapsto} L_1}$$

We derive the following chain of transition:

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

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

$$\textbf{Cong}\ \frac{P^{'} \equiv P \qquad P^{'} \xrightarrow{\alpha} er(Q)}{P \xrightarrow{\alpha} er(Q)}$$

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

$$\textbf{Com3L}\ \frac{P_1 \overset{\overline{x}y}{\longmapsto} L_1^{'} \qquad P_2 \overset{xy}{\longmapsto} Q_2}{P_1|P_2 \overset{\epsilon}{\longmapsto} L_1^{'}|stop(Q_2)}$$

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

$$\textbf{Par1L}\ \frac{L_i^{'} \overset{\gamma_i}{\longmapsto} L_{i+1}^{'}}{L_i^{'}|stop(Q_2) \overset{\gamma_i}{\longmapsto} L_{i+1}^{'}|stop(Q_2)} \qquad \textbf{Par1L}\ \frac{L_k^{'} \overset{\gamma_{k+1}}{\longmapsto} Q_1}{L_i^{'}|stop(Q_2) \overset{\gamma_{k+1}}{\longmapsto} Q_1|stop(Q_2)}$$

We derive the following chain of transition:

$$P_1 \overset{\overline{x}y}{\longmapsto} L_1^{'} \overset{\gamma_2}{\longmapsto} L_2^{'} \cdots L_{k-1}^{'} \overset{\gamma_k}{\longmapsto} L_k^{'} \overset{\gamma_{k+1}}{\longmapsto} Q_1$$

for inductive hypothesis $P_1 \xrightarrow{\overline{x}y} Q_1$. A proof of the conclusion is

$$\textbf{EComSeq}\ \frac{P_1 \xrightarrow{\overline{x}} Q_1 \qquad P_2 \xrightarrow{\overline{x}y} Q_2}{P_1|P_2 \xrightarrow{\tau} Q_1|Q_2}$$

Par1L : the last part of the derivation of the first transition is:

$$\textbf{Par1L}\ \frac{P_1 \overset{\gamma_1}{\longmapsto} L_1^{'}}{P_1|P_2 \overset{\gamma_1}{\longmapsto} L_1^{'}|P_2}$$

there can be three cases:

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

$$P_1 \overset{\gamma_1}{\longmapsto} L_1^{'} \overset{\gamma_2}{\longmapsto} L_2^{'} \cdots L_{k-1}^{'} \overset{\gamma_k}{\longmapsto} L_k^{'} \overset{\gamma_k}{\longmapsto} Q_1$$

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

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

- 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

$$\textbf{Com2LStop}\ \frac{L_2^{'} \xrightarrow{\overline{x}y} L_2^{'} \qquad P_2 \xrightarrow{xy} P_2^{'}}{L_2^{'}|P_2 \xrightarrow{\epsilon} L_2^{'}|stop(P_2^{'})}$$

We derive the following chain of transition:

$$P_1 \overset{\epsilon}{\longmapsto} L_1^{'} \overset{\overline{x}y}{\longmapsto} L_2^{'} \overset{\epsilon}{\longmapsto} \cdots \overset{\epsilon}{\longmapsto} L_{k-1}^{'} \overset{\epsilon}{\longmapsto} L_k^{'} \overset{\tau}{\longmapsto} Q_1$$

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

$$\textbf{EComSeq}\ \frac{P_1 \xrightarrow{\overline{x}y} er(Q_1) \qquad P_2 \xrightarrow{xy} P_2^{'}}{P_1|P_2 \xrightarrow{\tau} er(Q_1)|P_2^{'}}$$

- 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:

$$P_1 \overset{\epsilon}{\longmapsto} L_1^{'} \overset{\epsilon}{\longmapsto} L_2^{'} \cdots L_{k-1}^{'} \overset{\epsilon}{\longmapsto} L_k^{'} \overset{\overline{x}y}{\longmapsto} Q_1$$

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

$$\textbf{EComSeq}\ \frac{P_1 \xrightarrow{\overline{x}y} er(Q_1) \qquad P_2 \xrightarrow{xy} P_2^{'}}{P_1|P_2 \xrightarrow{\tau} er(Q_1)|P_2^{'}}$$

$\square$

Chapter 4

Multi π calculus with strong input

4.1 Syntax

As we did whit π calculus, we suppose that we have a countable set of names $\mathbb{N}$, ranged over by lower case letters $a, b, \cdots, z$. 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 $P, Q, \cdots$. A multi π process, in addiction to the same actions of a π process, can perform also a strong prefix input:

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

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

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

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. Since the strong prefix can be on input names only, the only synchronization possible is between a process that executes a sequence of n actions(only the last action can be an output) with $n \geq 1$ and n other processes each executing one single action(at least $n-1$ process execute an output and at most one executes an input).

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:

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

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

$$\textbf{Out}\ \frac{}{\overline{x}y.P \xrightarrow{\overline{x}y} P} \qquad \textbf{EInp}\ \frac{}{x(y).P \xrightarrow{xz} P\{z/y\}}$$

$$\textbf{Tau}\ \frac{}{\tau.P \xrightarrow{\tau} P} \qquad \textbf{SInp}\ \frac{P\{y/z\} \xrightarrow{\sigma} P' \qquad \sigma \neq \tau}{\underline{x(z)}.P \xrightarrow{xy\cdot\sigma} P'}$$

$$\textbf{Sum}\ \frac{P \xrightarrow{\sigma} P'}{P + Q \xrightarrow{\sigma} P'} \qquad \textbf{Str}\ \frac{P \equiv P' \qquad P' \xrightarrow{\alpha} Q}{P \xrightarrow{\alpha} Q}$$

$$\textbf{Com}\ \frac{P \xrightarrow{\overline{x}y} P' \qquad Q \xrightarrow{xy} Q'}{P|Q \xrightarrow{\tau} P'|Q'} \qquad \textbf{ComSeq}\ \frac{P \xrightarrow{xy\cdot\sigma} P' \qquad Q \xrightarrow{\overline{x}y} Q'}{P|Q \xrightarrow{\sigma} P'|Q'}$$

$$\textbf{Res}\ \frac{P \xrightarrow{\sigma} P' \qquad z \notin n(\sigma)}{(\nu z)P \xrightarrow{\sigma} (\nu z)P'} \qquad \textbf{SInpTau}\ \frac{P\{y/z\} \xrightarrow{\tau} P'}{\underline{x(z)}.P \xrightarrow{xy} P'}$$

$$\textbf{Par}\ \frac{P \xrightarrow{\sigma} P' \qquad bn(\sigma) \cap fn(Q) = \emptyset}{P|Q \xrightarrow{\sigma} P'|Q} \qquad \textbf{Opn}\ \frac{P \xrightarrow{\overline{x}z} P' \qquad z \neq x}{(\nu z)P \xrightarrow{\overline{x}(z)} P'}$$

$$\textbf{OpnSeq}\ \frac{P \xrightarrow{inpSeq\cdot\overline{x}z} P' \qquad z \neq x}{(\nu z)P \xrightarrow{inpSeq\cdot\overline{x}(z)} P'}$$

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.

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

$$\textbf{EComSng}\ \frac{\underline{x(a)}.x(b).P|\overline{x}y.Q) \xrightarrow{xz} P\{y/a\}\{z/b\}|Q \qquad \textbf{Out}\ \dfrac{}{\overline{x}z.R \xrightarrow{\overline{x}z} R}}{(\underline{x(a)}.x(b).P|\overline{x}y.Q)|\overline{x}z.R \xrightarrow{\tau} (P\{y/a\}\{z/b\}|Q)|R}$$

$$\textbf{EComSeq}\ \frac{\textbf{SInp}\ \dfrac{\textbf{EInp}\ \dfrac{}{(x(b).P)\{y/a\} \xrightarrow{xz} P\{y/a\}\{z/b\}}}{\underline{x(a)}.(x(b).P) \xrightarrow{xy\cdot xz} P\{y/a\}\{z/b\}} \qquad \textbf{Out}\ \dfrac{}{\overline{x}y.Q \xrightarrow{\overline{x}y} Q}}{\underline{x(a)}.x(b).P|\overline{x}y.Q) \xrightarrow{xz} P\{y/a\}\{z/b\}|Q}$$

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

- $|\sigma| = 1$
- $|\sigma| > 1$, the first $|\sigma| - 1$ actions are input and the last actions can be an input or an output.

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.

$$\textbf{Pref}\ \frac{\alpha \textit{ not a strong prefix}}{\alpha.P \xrightarrow{\alpha} P} \qquad \textbf{LComSeq}\ \frac{P \xrightarrow{x(y)\cdot\sigma} P^{'} \qquad Q \xrightarrow{\overline{x}z} Q^{'} \qquad z \notin fn(\sigma) \cup fn(P)}{P|Q \xrightarrow{\sigma\{z/y\}} P^{'}\{z/y\}|Q^{'}}$$

$$\textbf{SInp}\ \frac{P \xrightarrow{\sigma} P^{'} \qquad \sigma \neq \tau}{\underline{x(y)}.P \xrightarrow{x(y)\cdot\sigma} P^{'}} \qquad \textbf{LComSng}\ \frac{P \xrightarrow{x(y)} P^{'} \qquad Q \xrightarrow{\overline{x}z} Q^{'} \qquad z \notin fn(P)}{P|Q \xrightarrow{\tau} P^{'}\{z/y\}|Q^{'}}$$

$$\textbf{Sum}\ \frac{P \xrightarrow{\sigma} P^{'}}{P+Q \xrightarrow{\sigma} P^{'}} \qquad \textbf{Str}\ \frac{P \equiv P^{'} \qquad P^{'} \xrightarrow{\alpha} Q^{'} \qquad Q \equiv Q^{'}}{P \xrightarrow{\alpha} Q}$$

$$\textbf{Res}\ \frac{P \xrightarrow{\sigma} P^{'} \qquad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\sigma} (\nu z)P^{'}} \qquad \textbf{Par}\ \frac{P \xrightarrow{\sigma} P^{'} \qquad bn(\sigma) \cup fn(Q) = \emptyset}{P|Q \xrightarrow{\sigma} P^{'}|Q}$$

$$\textbf{Opn}\ \frac{P \xrightarrow{\overline{x}z} P^{'} \qquad z \neq x}{(\nu z)P \xrightarrow{\overline{x}(z)} P^{'}} \qquad \textbf{OpnSeq}\ \frac{P \xrightarrow{inpSeq\cdot\overline{x}z} P^{'} \qquad z \neq x}{(\nu z)P \xrightarrow{inpSeq\cdot\overline{x}(z)} P^{'}}$$

Table 4.2: Multi π late semantic with structural congruence

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

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

and

$$b \notin fn(\underline{x(a)}.x(b).P|\overline{x}y.Q)$$

$$\textbf{LComSng}\ \frac{\underline{x(a)}.x(b).P|\overline{x}y.Q \xrightarrow{x(b)} P\{y/a\}|Q \qquad \textbf{Pref}\ \dfrac{}{\overline{x}z.R \xrightarrow{\overline{x}z} R}}{(\underline{x(a)}.x(b).P|\overline{x}y.Q)|\overline{x}z.R \xrightarrow{\tau} (P\{y/a\}|Q)\{z/b\}|R}$$

$$\textbf{LComSeq}\ \frac{\textbf{SInp}\ \dfrac{\textbf{Pref}\ \dfrac{}{x(b).P \xrightarrow{x(b)} P}}{\underline{x(a)}.x(b).P \xrightarrow{x(a)\cdot x(b)} P} \qquad \textbf{Out}\ \dfrac{}{\overline{x}y.Q \xrightarrow{\overline{x}y} Q}}{\underline{x(a)}.x(b).P|\overline{x}y.Q) \xrightarrow{x(b)} P\{y/a\}|Q}$$

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 $*P$:

$$P, Q ::= 0 \mid \pi.P \mid P|Q \mid P+Q \mid (\nu x)P \mid A(x_1, \cdots, x_n) \mid *P$$

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

Definition 4.2.3. 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 4.2.2. For all unmarked processes P, Q and marked processes L_1, L_2.

- if $P \overset{\alpha}{\longmapsto} L_1$ or $L_1 \overset{\alpha}{\longmapsto} L_2$ then α can only be a free input or an ϵ

$$\textbf{Out}\ \frac{}{\overline{x}y.P \xmapsto{\overline{x}y} P} \qquad \textbf{EInp}\ \frac{}{x(y).P \xmapsto{xz} P\{z/y\}}$$

$$\textbf{Star}\ \frac{S \xmapsto{\gamma} S^{'}}{*S \xmapsto{\gamma} S^{'}} \qquad \textbf{Tau}\ \frac{}{\tau.P \xmapsto{\tau} P}$$

$$\textbf{Sum}\ \frac{P \xmapsto{\gamma} S}{P + Q \xmapsto{\gamma} S} \qquad \textbf{SInpLow}\ \frac{}{\underline{x(z)}.P \xmapsto{xy} *P\{y/z\}}$$

$$\textbf{Com1}\ \frac{P \xmapsto{\overline{x}y} P^{'} \qquad Q \xmapsto{xy} Q^{'}}{P|Q \xmapsto{\tau} P^{'}|Q^{'}}$$

$$\textbf{Com2}\ \frac{L_1 \xmapsto{xy} L_1^{'} \qquad L_2 \xmapsto{\overline{x}y} P}{L_1|L_2 \xmapsto{\epsilon} L_1^{'}|P} \qquad \textbf{Com2R}\ \frac{L_1 \xmapsto{xy} L_1^{'} \qquad L_2 \xmapsto{\overline{x}y} P}{L_2|L_1 \xmapsto{\epsilon} P|L_1^{'}}$$

$$\textbf{Com3L}\ \frac{P \xmapsto{xy} L \qquad Q \xmapsto{\overline{x}y} Q^{'}}{P|Q \xmapsto{\epsilon} L|Q^{'}} \qquad \textbf{Com3R}\ \frac{P \xmapsto{xy} L \qquad Q \xmapsto{\overline{x}y} Q^{'}}{Q|P \xmapsto{\epsilon} Q^{'}|L}$$

$$\textbf{Com4L}\ \frac{L_1 \xmapsto{\overline{x}y} P \qquad L_2 \xmapsto{xy} Q}{L_1|L_2 \xmapsto{\tau} P|Q} \qquad \textbf{Com4R}\ \frac{L_1 \xmapsto{xy} P \qquad L_2 \xmapsto{\overline{x}y} Q}{L_1|L_2 \xmapsto{\tau} P|Q}$$

$$\textbf{Res}\ \frac{S \xmapsto{\gamma} S^{'} \qquad y \notin n(\gamma)}{(\nu y)S \xmapsto{\gamma} (\nu y)S^{'}} \qquad \textbf{Opn}\ \frac{S \xmapsto{\overline{x}y} P \qquad y \neq x}{(\nu y)S \xmapsto{\overline{x}(y)} P}$$

$$\textbf{Par1L}\ \frac{S \xmapsto{\gamma} S^{'}}{S|Q \xmapsto{\gamma} S^{'}|Q} \qquad \textbf{Par1R}\ \frac{S \xmapsto{\gamma} S^{'}}{Q|S \xmapsto{\gamma} Q|S^{'}}$$

$$\textbf{Par2L}\ \frac{P \xmapsto{\gamma} L}{P|Q \xmapsto{\gamma} L|*Q} \qquad \textbf{Par2R}\ \frac{P \xmapsto{\gamma} L}{Q|P \xmapsto{\gamma} *Q|L}$$

$$\textbf{Par3L}\ \frac{L_1 \xmapsto{\gamma} L_1^{'}}{L_1|L_2 \xmapsto{\gamma} L_1^{'}|L_2} \qquad \textbf{Par3R}\ \frac{L_2 \xmapsto{\gamma} L_2^{'}}{L_1|L_2 \xmapsto{\gamma} L_1|L_2^{'}}$$

$$\textbf{Cong}\ \frac{P \equiv P^{'} \qquad P^{'} \xmapsto{\gamma} S}{P \xmapsto{\gamma} S}$$

Table 4.3: Low multi π early semantic with structural congruence

- if $L_1 \overset{\alpha}{\longmapsto} P$ of $P \overset{\alpha}{\longmapsto} Q$ then α is not an ϵ

Proof. DA FARE ☐

Lemma 4.2.3. If $P \overset{\gamma_1}{\longmapsto} L_1 \overset{\gamma_2}{\longmapsto} \cdots \overset{\gamma_k}{\longmapsto} L_k \overset{\gamma_{k+1}}{\longmapsto} Q$ and $\sigma = \gamma_1 \cdot \ldots \cdot \gamma_{k+1}$ with $k \geq 1$ then σ is a sequence of input and the last action in σ is an input or an output.

Proof. DA FARE ☐

Definition 4.2.4. Let P, Q be unmarked processes and $L_1, \cdots, L_{k-1}$ marked processes. We define the derivation relation $\rightarrow_s$ in the following way:

$$\textbf{Low}\ \frac{P \overset{\gamma_1}{\longmapsto} L_1 \overset{\gamma_2}{\longmapsto} L_2 \cdots L_{k-1} \overset{\gamma_k}{\longmapsto} Q \qquad k \geq 1}{P \xrightarrow{\gamma_1 \cdots \gamma_k}_s Q}$$

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:

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

HA PIU' SENSO DIRE $a \cdot \tau \neq a$ e $\tau \cdot \tau \neq \tau$?

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

$$\textbf{Par2}\ \frac{\textbf{Com3}\ \dfrac{\textbf{SInpLow}\ \dfrac{}{\underline{x(a)}.x(b).P \overset{xy}{\longmapsto} *(x(b).P\{y/a\})} \qquad \textbf{Out}\ \dfrac{}{\overline{x}y.Q \overset{\overline{x}y}{\longmapsto} Q}}{\underline{x(a)}.x(b).P|\overline{x}y.Q \overset{\epsilon}{\longmapsto} *(x(b).P\{y/a\})|Q}}{(\underline{x(a)}.x(b).P|\overline{x}y.Q)|\overline{x}z.R \overset{\epsilon}{\longmapsto} (*(x(b).P\{y/a\})|Q)| * (\overline{x}z.R)}$$

$$\textbf{Par1}\ \frac{\textbf{Star}\ \dfrac{\textbf{EInp}\ \dfrac{}{x(b).P\{y/a\} \overset{xz}{\longmapsto} P\{y/a\}\{z/b\}}}{*(x(b).P\{y/a\}) \overset{xz}{\longmapsto} P\{y/a\}\{z/b\}}}{*(x(b).P\{y/a\})|Q \overset{xz}{\longmapsto} P\{y/a\}\{z/b\}|Q}$$

$$\textbf{Com4}\ \frac{*(x(b).P\{y/a\})|Q \overset{xz}{\longmapsto} P\{y/a\}\{z/b\}|Q \qquad \textbf{Star}\ \dfrac{\textbf{Out}\ \dfrac{}{\overline{x}z.R \overset{\overline{x}z}{\longmapsto} R}}{*\overline{x}z.R \overset{\overline{x}z}{\longmapsto} R}}{(\underline{x(a)}.x(b).P|\overline{x}y.Q)| * (\overline{x}z.R) \overset{\tau}{\longmapsto} (P\{y/a\}\{z/b\}|Q)|R}$$

$$\textbf{Low}\ \frac{(\underline{x(a)}.x(b).P|\overline{x}y.Q)|\overline{x}z.R \overset{\epsilon}{\longmapsto} (*(x(b).P\{y/a\})|Q)| * (\overline{x}z.R) \xrightarrow{\tau} (P\{y/a\}\{z/b\}|Q)|R}{(\underline{x(a)}.x(b).P|\overline{x}y.Q)|\overline{x}z.R \xrightarrow{\tau}_s (P\{y/a\}\{z/b\}|Q)|R}$$

Proposition 4.2.4. Let $\rightarrow$ be the relation defined in table 4.1 and let P be a multi π process. Then

$$P \xrightarrow{\sigma} Q \Rightarrow P \xrightarrow{\sigma}_s Q$$

Proof. DA MODIFICARE PERCHE' LE REGOLE COM2 E COM3 SONO CAMBIATE 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 4.3 so we can derive $P \overset{\sigma}{\longmapsto} Q$ and then apply *Low* to get the result $P \xrightarrow{\sigma}_s Q$.

inductive case

If the depth is greater than one then the last rule used in the derivation can be:

$SInp$: the last part of the derivation tree looks like this:

$$\textbf{SInp}\ \frac{P_1\{y/z\} \xrightarrow{\sigma} Q \qquad \sigma \neq \tau}{\underline{x(z)}.P_1 \xrightarrow{xy\cdot\sigma} Q}$$

for inductive hypothesis $P_1\{y/z\} \xrightarrow{\sigma}_s Q$ which implies that there exist $L_1, \cdots, L_k$ and $\gamma_1, \cdots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1\{y/z\} \xmapsto{\gamma_1} L_1 \xmapsto{\gamma_2} L_2 \cdots L_{k-1} \xmapsto{\gamma_k} L_k \xmapsto{\gamma_{k+1}} Q$$

and

$$\gamma_1 \cdot \ldots \cdot \gamma_{k+1} = \sigma$$

since

$$\textbf{SInpLow}\ \frac{}{\underline{x(z)}.P_1 \xmapsto{xy} *P_1\{y/z\}} \qquad \textbf{Star}\ \frac{P_1\{y/z\} \xmapsto{\gamma_1} L_1}{*P_1\{y/z\} \xmapsto{\gamma_1} L_1}$$

then a proof of $P \xrightarrow{\sigma}_s Q$ is:

$$\textbf{Low}\ \frac{\underline{\underline{x(z)}}.P_1 \xmapsto{xy} *P_1\{y/z\} \xmapsto{\gamma_1} L_1 \xmapsto{\gamma_2} L_2 \cdots L_{k-1} \xmapsto{\gamma_k} L_k \xmapsto{\gamma_{k+1}} Q}{\underline{x(z)}.P_1 \xrightarrow{xy\cdot\gamma_1\cdot\ldots\cdot\gamma_{k+1}}_s Q}$$

and $xy \cdot \gamma_1 \cdot \ldots \cdot \gamma_{k+1} = xy \cdot \sigma$

$SInpTau$: this case is similar to the previous.

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

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

for the inductive hypothesis $P_1 \xrightarrow{\sigma}_s Q$ which implies that there exist $L_1, \cdots, L_k$ and $\gamma_1, \cdots, \gamma_{k+1}$ with $k \geq 0$ such that

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

and

$$\gamma_1 \cdot \ldots \cdot \gamma_{k+1} = \sigma$$

A proof of the conclusion is:

$$\textbf{Low}\ \frac{\textbf{Sum}\ \dfrac{P_1 \xmapsto{\gamma_1} L_1}{P_1 + P_2 \xmapsto{\gamma_1} L_1} \qquad L_1 \xmapsto{\gamma_2} L_2 \cdots L_{k-1} \xmapsto{\gamma_k} L_k \xmapsto{\gamma_{k+1}} Q}{P + R \xrightarrow{\sigma}_s Q}$$

Str : this case is similar to the previous.

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

$$\textbf{Com}\ \frac{P_1 \xrightarrow{\overline{x}y} P_1^{'} \qquad Q_1 \xrightarrow{xy} Q_1^{'}}{P_1|Q_1 \xrightarrow{\tau} P_1^{'}|Q_1^{'}}$$

for inductive hypothesis $P_1 \xrightarrow{\overline{x}y}_s P_1^{'}$ and $Q_1 \xrightarrow{xy}_s Q_1^{'}$ which imply that there exist $L_1, \cdots, L_k$ and $\gamma_1, \cdots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xmapsto{\gamma_1} L_1 \xmapsto{\gamma_2} L_2 \cdots L_{k-1} \xmapsto{\gamma_k} L_k \xmapsto{\gamma_{k+1}} P_1^{'}$$

and

$$\gamma_1 \cdot \ldots \cdot \gamma_{k+1} = \overline{x}y$$

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

$$Q_1 \xmapsto{\delta_1} R_1 \xmapsto{\delta_2} R_2 \cdots R_{h-1} \xmapsto{\delta_h} R_h \xmapsto{\delta_{h+1}} Q_1^{'}$$

and

$$\delta_1 \cdot \ldots \cdot \delta_{k+1} = xy$$

we can have nine different cases now:

$$\{\gamma_1 = \overline{x}y,\ \gamma_i = \overline{x}y\ 1 < i \leq k, \gamma_{k+1} = \overline{x}y\} \times \{\delta_1 = xy,\ \delta_j = xy\ 1 < j \leq h, \delta_{h+1} = xy\}$$

We show just the first three since the others are similar:

$\gamma_1 = \overline{x}y$ *and* $\delta_1 = xy$: in this case: $\gamma_{k+1} = \tau$ and the other γs are ϵ; $\delta_{h+1} = \tau$ and the other δs are ϵ. A proof of the conclusion is:

$$\textbf{Low}\ \frac{\begin{array}{c} P_1|Q_1 \xmapsto{\tau} L_1|R_1 \xmapsto{\epsilon} L_2|R_1 \cdots \\ L_{k-1}|R_1 \xmapsto{\epsilon} L_k|R_1 \xmapsto{\epsilon} L_k|R_2 \cdots \xmapsto{\epsilon} L_k|R_h \xmapsto{\tau} P_1^{'}|R_h \xmapsto{\tau} P_1^{'}|Q_1^{'} \end{array}}{P_1|Q_1 \xrightarrow{\tau}_s P_1^{'}|Q_1^{'}}$$

we derive the first transaction of the premises with rule $Com3$, whether for the other transactions we use the rule $Par1$ and when necessary $Cong1$.

$\gamma_i = \overline{x}y$ *and* $\delta_1 = xy$: in this case: $\gamma_1 = \gamma_{k+1} = \tau$ and the other γs are ϵ; $\delta_{h+1} = \tau$ and the other δs are ϵ. A proof of the conclusion is:

$$\textbf{Low}\ \frac{\begin{array}{c} P_1|Q_1 \xmapsto{\tau} L_1|Q_1 \xmapsto{\epsilon} L_2|Q_1 \cdots \xmapsto{\epsilon} L_{i-1}|Q_1 \xmapsto{\tau} L_i|R_1 \xmapsto{\epsilon} L_{i+1}|R_1 \\ \cdots \xmapsto{\epsilon} L_k|R_1 \xmapsto{\epsilon} L_k|R_2 \cdots \xmapsto{\epsilon} L_k|R_h \xmapsto{\tau} P_1^{'}|R_h \xmapsto{\tau} P_1^{'}|Q_1^{'} \end{array}}{P_1|Q_1 \xrightarrow{\tau}_s P_1^{'}|Q_1^{'}}$$

we derive the transaction $L_{i-1}|Q_1 \xmapsto{\tau} L_i|R_1$ with rule $Com3$, whether for the other transactions of the premises we use the rule $Par1$ and when necessary $Cong1$.

$\gamma_{k+1} = \overline{x}y$ *and* $\delta_1 = xy$: in this case: $\gamma_1 = \tau$ and the other γs are ϵ; $\delta_{h+1} = \tau$ and the other δs are ϵ. A proof of the conclusion is:

$$\textbf{Low}\ \frac{\begin{array}{c} P_1|Q_1 \xmapsto{\tau} L_1|Q_1 \xmapsto{\epsilon} L_2|Q_1 \cdots \xmapsto{\epsilon} L_k|Q_1 \\ \xmapsto{\tau} P_1^{'}|R_1 \xmapsto{\epsilon} P_1^{'}|R_2 \cdots \xmapsto{\epsilon} P_1^{'}|R_h \xmapsto{\tau} P_1^{'}|Q_1^{'} \end{array}}{P_1|Q_1 \xrightarrow{\tau}_s P_1^{'}|Q_1^{'}}$$

we derive the transaction $L_k|Q_1 \xmapsto{\tau} P_1^{'}|R_1$ with rule $Com3$, whether for the other transactions of the premises we use the rule $Par1$ and when necessary $Cong1$.

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

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

for the inductive hypothesis $P_1 \xrightarrow{\sigma}_s Q_1$ which implies that there exist $L_1, \cdots, L_k$ and $\gamma_1, \cdots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xmapsto{\gamma_1} L_1 \xmapsto{\gamma_2} L_2 \cdots L_{k-1} \xmapsto{\gamma_k} L_k \xmapsto{\gamma_{k+1}} Q_1$$

and

$$\gamma_1 \cdot \ldots \cdot \gamma_{k+1} = \sigma$$

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 apply the rule *Low* to get a proof of the conclusion:

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

Par : this case is similar to the previous.

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

$$\textbf{Com}\ \frac{P_1 \xrightarrow{xy\cdot\sigma} P_1^{'} \qquad Q_1 \xrightarrow{\overline{x}y} Q_1^{'}}{P_1|Q_1 \xrightarrow{\sigma} P_1^{'}|Q_1^{'}}$$

for inductive hypothesis $P_1 \xrightarrow{xy\cdot\sigma}_s P_1^{'}$ which implies that there exist $L_1, \cdots, L_k$ and $\gamma_1, \cdots, \gamma_{k+1}$ with $k \geq 0$ such that

$$P_1 \xmapsto{\gamma_1} L_1 \xmapsto{\gamma_2} L_2 \cdots L_{k-1} \xmapsto{\gamma_k} L_k \xmapsto{\gamma_{k+1}} P_1^{'}$$

and

$$\gamma_1 \cdot \ldots \cdot \gamma_{k+1} = xy \cdot \sigma$$

Again for inductive hypothesis $Q_1 \xrightarrow{xy}_s Q_1^{'}$ which implies that there exist $R_1, \cdots, R_h$ and $\delta_1, \cdots, \delta_{h+1}$ with $h \geq 0$ such that

$$Q_1 \xmapsto{\delta_1} R_1 \xmapsto{\delta_2} R_2 \cdots R_{h-1} \xmapsto{\delta_h} R_h \xmapsto{\delta_{h+1}} Q_1^{'}$$

and

$$\delta_1 \cdot \ldots \cdot \delta_{k+1} = \overline{x}y$$

We assume that σ is not τ otherwise we can replace this instance of the rule *ComSeq* with an instance of the rule *Com* and proceed as in the previous case. We can have six different cases now:

$$\{\gamma_1 = xy,\ \gamma_i = \overline{x}y\ 1 < i \leq k\} \times \{\delta_1 = \overline{x}y,\ \delta_j = \overline{x}y\ 1 < j \leq h, \delta_{h+1} = \overline{x}y\}$$

We show just the first two since the others are similar:

$\gamma_1 = xy$ *and* $\delta_1 = \overline{x}y$: in this case: $\delta_{h+1} = \tau$ and the other δs are ϵ. A proof of the conclusion is:

$$\mathbf{Low}\ \frac{\begin{array}{c}P_1|Q_1 \xmapsto{\tau} L_1|R_1 \xmapsto{\gamma_2} L_2|R_1 \cdots \xmapsto{\gamma_k} L_k|R_1\\ \xmapsto{\gamma_{k+1}} P_1^{'}|R_1 \xmapsto{\epsilon} P_1^{'}|R_2 \cdots \xmapsto{\epsilon} P_1^{'}|R_h \xmapsto{\tau} P_1^{'}|Q_1^{'}\end{array}}{P_1|Q_1 \xrightarrow{\tau\cdot\gamma_2\cdot\ldots\cdot\gamma_{k+1}\cdot\epsilon\cdot\ldots\epsilon\cdot\tau}_s P_1^{'}|Q_1^{'}}$$

we derive the first transaction of the premises with rule $Com3$, whether for the other transactions we use the rule $Par1$ and when necessary $Cong1$. Since $\gamma_1\cdot\ldots\cdot\gamma_{k+1} = xy\cdot\sigma$ and $\gamma_1 = xy$ then $\tau\cdot\gamma_2\cdot\ldots\cdot\gamma_{k+1}\cdot\epsilon\cdot\ldots\epsilon\cdot\tau = \sigma$

$\gamma_i = xy$ *and* $\delta_1 = \overline{x}y$: in this case: $\gamma_1 = \gamma_{k+1} = \tau$, $\gamma_j = \epsilon$ if $1 < j < i$ and $\gamma_{i+1}\cdot\ldots\cdot\gamma_{k+1} = \sigma$; $\delta_{h+1} = \tau$ and the other δs are ϵ. A proof of the conclusion is:

$$\mathbf{Low}\ \frac{\begin{array}{c}P_1|Q_1 \xmapsto{\tau} L_1|Q_1 \xmapsto{\epsilon} L_2|Q_1 \cdots \xmapsto{\epsilon} L_{i-1}|Q_1 \xmapsto{\tau} L_i|R_1 \xmapsto{\gamma_{i+1}} L_{i+1}|R_1\\ \cdots \xmapsto{\gamma_k} L_k|R_1 \xmapsto{\gamma_{k+1}} P_1^{'}|R_1 \xmapsto{\epsilon} P_1^{'}|R_2 \cdots \xmapsto{\epsilon} P_1^{'}|R_h \xmapsto{\tau} P_1^{'}|Q_1^{'}\end{array}}{P_1|Q_1 \xrightarrow{\tau\cdot\epsilon\cdot\ldots\cdot\epsilon\cdot\tau\cdot\gamma_{i+1}\cdot\ldots\cdot\gamma_{k+1}\cdot\epsilon\cdot\ldots\cdot\epsilon\cdot\tau}_s P_1^{'}|Q_1^{'}}$$

we derive the transaction $L_{i-1}|Q_1 \xmapsto{\tau} L_i|R_1$ with rule $Com3$ and $Cong1$, whether for the other transactions of the premises we use the rule $Par1$ and when necessary $Cong1$.

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

$$\mathbf{Opn}\ \frac{P_1 \xrightarrow{\overline{x}z} Q \qquad z \neq x}{(\nu z)P \xrightarrow{\overline{x}(z)} Q}$$

for the inductive hypothesis $P_1 \xrightarrow{\sigma}_s Q$ which implies that there exist $L_1, \cdots, L_k$ and $\gamma_1, \cdots, \gamma_{k+1}$ with $k \geq 0$ such that

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

and

$$\gamma_1\cdot\ldots\cdot\gamma_{k+1} = \overline{x}z$$

We can have three different cases now:

$\gamma_1 = \overline{x}z$: in this case $\gamma_{h+1} = \tau$ and the other γs are ϵ. A proof of the conclusion is:

$$\mathbf{Low}\ \frac{\mathbf{Open}\ \dfrac{P_1 \xmapsto{\overline{x}(z)} L_1}{(\nu z)P_1 \xmapsto{\overline{x}(z)} L_1} \qquad L_1 \xmapsto{\epsilon} L_2 \cdots \xmapsto{\epsilon} L_k \xmapsto{\epsilon} Q}{(\nu z)P_1 \xrightarrow{\overline{x}(z)\cdot\epsilon\cdot\ldots\cdot\epsilon}_s Q}$$

$\gamma_i = \overline{x}z$: in this case $\gamma_1 = \gamma_{k+1} = \tau$ and the other γs are ϵ. A proof of the conclusion is:

$$\mathbf{Low}\ \frac{\begin{array}{c}(\nu z)P_1 \xmapsto{\tau} (\nu z)L_1 \xmapsto{\epsilon} (\nu z)L_2 \cdots \xmapsto{\epsilon} (\nu z)L_{i-1}\\ (\nu z)L_{i-1} \xmapsto{\overline{x}(z)} L_i \qquad L_i \xmapsto{\epsilon} L_{i+1} \cdots \xmapsto{\epsilon} L_k \xmapsto{\tau} Q\end{array}}{(\nu z)P_1 \xrightarrow{\tau\cdot\epsilon\cdot\ldots\cdot\epsilon\cdot\tau\cdot\gamma_{i+1}\cdot\ldots\cdot\gamma_{k+1}\cdot\epsilon\cdot\ldots\cdot\epsilon\cdot\tau}_s Q}$$

we derive the transition $(\nu z)L_{i-1} \xmapsto{\overline{x}(z)} L_i$ with rule *Open* and the transitions $(\nu z)P_1 \xmapsto{\tau} (\nu z)L_1 \xmapsto{\epsilon} (\nu z)L_2 \cdots \xmapsto{\epsilon} (\nu z)L_{i-1}$ with rule *Res*.

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

$$\mathbf{OpnSeq}\ \frac{P_1 \xrightarrow{inpSeq\cdot\overline{x}z} Q \qquad z \neq x}{(\nu z)P_1 \xrightarrow{inpSeq\cdot\overline{x}(z)} Q}$$

for the inductive hypothesis $P_1 \xrightarrow{\sigma}_s Q$ which implies that there exist $L_1, \cdots, L_k$ and $\gamma_1, \cdots, \gamma_{k+1}$ with $k \geq 0$ such that

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

and

$$\gamma_1 \cdot \ldots \cdot \gamma_{k+1} = inpSeq \cdot \overline{x}z$$

We can have three different cases now:

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

$$\textbf{Low}\ \dfrac{\textbf{Opn}\ \dfrac{P_1 \overset{\overline{x}z}{\longmapsto} L_1}{(\nu z)P_1 \overset{\overline{x}(z)}{\longmapsto} L_1} \qquad L_1 \overset{\gamma_2}{\longmapsto} L_2 \cdots \overset{\gamma_k}{\longmapsto} L_k \overset{\gamma_{k+1}}{\longmapsto} Q}{(\nu z)P_1 \xrightarrow{\overline{x}(z)\cdot\gamma_2\cdot\ldots\cdot\gamma_{k+1}}_s Q}$$

$\gamma_i = \overline{x}z$: with $1 < i \leq k$ in this case $\gamma_{i+1} = \cdot = \gamma_k = \epsilon$, $\gamma_{k+1} = \tau$ and $\gamma_1 \cdot \ldots \cdot \gamma_{i-1} = inpSeq$. A proof of the conclusion is:

$$\textbf{Low}\ \dfrac{(\nu z)P_1 \overset{\gamma_1}{\longmapsto} (\nu z)L_1 \overset{\gamma_2}{\longmapsto} (\nu z)L_2 \cdots \overset{\gamma_{i-1}}{\longmapsto} (\nu z)L_{i-1} \overset{\overline{x}(z)}{\longmapsto} L_i \overset{\epsilon}{\longmapsto} L_{i+1} \cdots \overset{\epsilon}{\longmapsto} L_k \overset{\tau}{\longmapsto} Q}{(\nu z)P_1 \xrightarrow{\gamma_1\cdot\ldots\cdot\gamma_{i-1}\cdot\overline{x}(z)\cdot\epsilon\cdot\ldots\cdot\epsilon\cdot\cdot\tau}_s Q}$$

we derive the transition $(\nu z)L_{i-1} \overset{\overline{x}(z)}{\longmapsto} L_i$ with rule *Open* and the transitions $(\nu z)P_1 \overset{\gamma_1}{\longmapsto} (\nu z)L_1 \overset{\gamma_2}{\longmapsto} (\nu z)L_2 \cdots \overset{\gamma_{i-1}}{\longmapsto} (\nu z)L_{i-1}$ with rule *Res*.

$\gamma_{k+1} = \overline{x}z$: in this case $\gamma_1 \cdot \ldots \cdot \gamma_k = inpSeq$. A proof of the conclusion is:

$$\textbf{Low}\ \dfrac{(\nu z)P_1 \overset{\gamma_1}{\longmapsto} (\nu z)L_1 \overset{\gamma_2}{\longmapsto} (\nu z)L_2 \cdots \overset{\gamma_k}{\longmapsto} (\nu z)L_k \overset{\overline{x}(z)}{\longmapsto} Q}{(\nu z)P_1 \xrightarrow{\gamma_1\cdot\ldots\cdot\gamma_k\cdot\overline{x}(z)}_s Q}$$

we derive the transition $(\nu z)L_k \overset{\overline{x}(z)}{\longmapsto} L_i$ with rule *Open* and the others with rule *Res*.

□

Proposition 4.2.5. Let $\rightarrow$ be the relation defined in table 4.1 and let P be a multi π process. Then

$$P \xrightarrow{\sigma}_s Q \Rightarrow P \xrightarrow{\sigma} Q$$

Proof. DA MODIFICARE PERCHE' LE REGOLE COM2 E COM3 SONO CAMBIATE The proof is by induction on the sum of the length of the sequence of transitions used to derive $P \xrightarrow{\sigma}_s Q$ and on the depth of the proof tree of the first transition in such a sequence:

base case

in this case $P \overset{\sigma}{\longmapsto} Q$ and the derivation of this transition has depth one. The last(and only) rule used to derive $P \overset{\sigma}{\longmapsto} Q$ can be: *Out*, *EInp* or *Tau*; these rules are also in table 4.1 so we can derive $P \xrightarrow{\sigma} Q$.

inductive case

we have two cases: the first is $P \overset{\sigma}{\longmapsto} Q$ and the depth of the derivation is greater than one. In this case the last rule in the derivation can be: *Sum*, *Com*1, *Res*, *Opn*, *Par*1, *Cong*1:

Sum :

$$\textbf{Sum}\ \dfrac{P_1 \overset{\gamma_1}{\longmapsto} Q}{P_1 + P_2 \overset{\gamma_1}{\longmapsto} Q}$$

for rule *Low* $P_1 \xrightarrow{\gamma_1}_s Q$, for inductive hypothesis $P_1 \xrightarrow{\gamma_1} Q$ and finally for rule *Sum* $P_1 + P_2 \xrightarrow{\gamma_1} Q$.

others the other cases are similar to the previous since these rules are in common.

The second case is that there exist $L_1, \cdots, L_k$ and $\gamma_1, \cdots, \gamma_{k+1}$ with $k \geq 1$ such that

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

and

$$\gamma_1 \cdot \ldots \cdot \gamma_{k+1} = \sigma \text{ and } k \geq 1$$

we proceed by induction on the depth of the derivation of $P \overset{\gamma_1}{\longmapsto} L_1$. Let us call R the last instance of a rule used to derive $P \overset{\gamma_1}{\longmapsto} L_1$. R cannot be *Out*, *EInp*, *Tau*, *Com*1, *Cong*1 because L_1 is a marked process but these rules lead to a non marked process. Also R cannot be *Star*, *Com*2, *Com*4, *Cong*3 because P is not marked whether the head of the conclusion of these rules is marked.

base case

R can be only *SInpLow*, so

$$\underline{x(z)}.P_1 \overset{xy}{\longmapsto} *P_1\{y/z\} \overset{\gamma_2}{\longmapsto} L_2 \cdots L_{k-1} \overset{\gamma_k}{\longmapsto} L_k \overset{\gamma_{k+1}}{\longmapsto} Q$$

Since $*P_1\{y/z\}$ has a mark on top, the last rule used in a derivation of $*P_1\{y/z\} \overset{\gamma_2}{\longmapsto} L_2$ has to be *Star*. This means that $P_1\{y/z\} \overset{\gamma_2}{\longmapsto} L_2$ and so

$$P_1\{y/z\} \overset{\gamma_2}{\longmapsto} L_2 \cdots L_{k-1} \overset{\gamma_k}{\longmapsto} L_k \overset{\gamma_{k+1}}{\longmapsto} Q$$

which for rule *Low* and inductive hypothesis implies $P_1\{y/z\} \xrightarrow{\gamma_2 \cdot \ldots \cdot \gamma_{k+1}} Q$. A proof of the conclusion is:

$$\textbf{SInp}\ \frac{P_1\{y/z\} \xrightarrow{\gamma_2 \cdot \ldots \cdot \gamma_{k+1}} Q \qquad \gamma_2 \cdot \ldots \cdot \gamma_{k+1} \neq \tau}{\underline{x(z)}.P_1 \xrightarrow{xy \cdot \gamma_2 \cdot \ldots \cdot \gamma_{k+1}} Q}$$

or

$$\textbf{SInpTau}\ \frac{P_1\{y/z\} \xrightarrow{\gamma_2 \cdot \ldots \cdot \gamma_{k+1}} Q \qquad \gamma_2 \cdot \ldots \cdot \gamma_{k+1} = \tau}{\underline{x(z)}.P_1 \xrightarrow{xy} Q}$$

inductive case

R can be:

Sum the first transition is:

$$\textbf{Sum}\ \frac{P_1 \overset{\gamma_1}{\longmapsto} L_1}{P_1 + P_2 \overset{\gamma_1}{\longmapsto} L_1}$$

so we can build the following chain of transition:

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

apply the rule *Low* and the inductive hypothesis to get $P_1 \xrightarrow{\gamma_1 \cdot \ldots \cdot \gamma_{k+1}} Q$. Now a proof of the conclusion can be

$$\textbf{Sum}\ \frac{P_1 \xrightarrow{\gamma_1 \cdot \ldots \cdot \gamma_{k+1}} Q}{P_1 + P_2 \xrightarrow{\gamma_1 \cdot \ldots \cdot \gamma_{k+1}} Q}$$

Res the first transition is:

$$\textbf{Res}\ \frac{P_1 \overset{\gamma_1}{\longmapsto} L_1^{'} \qquad z \notin n(\gamma_1)}{(\nu z)P_1 \overset{\gamma_1}{\longmapsto} (\nu z)L_1^{'}}$$

given that L_1 has a restriction at the top level, all the other intermediate processes $L_2, \cdots, L_k$ have the same restriction at the top level. So the last rule used to prove $L_i \overset{\gamma_{i+1}}{\longmapsto} L_{i+1}$ for each $1 \le i \le k-1$ is *Res*. The last rule used to derive $L_k \overset{\gamma_{k+1}}{\longmapsto} Q$ can be one of:

Res :

$$\textbf{Res}\ \frac{L_k^{'} \overset{\gamma_{k+1}}{\longmapsto} Q^{'} \qquad z \notin n(Q^{'})}{(\nu z)L_k^{'} \overset{\gamma_{k+1}}{\longmapsto} (\nu z)Q^{'}}$$

we can build the following chain of transitions:

$$P_1 \overset{\gamma_1}{\longmapsto} L_1^{'} \overset{\gamma_2}{\longmapsto} L_2^{'} \cdots L_{k-1}^{'} \overset{\gamma_k}{\longmapsto} L_k^{'} \overset{\gamma_{k+1}}{\longmapsto} Q^{'}$$

then apply the rule *Low* and the inductive hypothesis to get $P_1 \xrightarrow{\gamma_1 \cdot \ldots \cdot \gamma_{k+1}} Q^{'}$. A proof of the conclusion can be

$$\textbf{Res}\ \frac{P_1 \xrightarrow{\gamma_1 \cdot \ldots \cdot \gamma_{k+1}} Q^{'} \qquad z \notin n(\gamma_1 \cdot \ldots \cdot \gamma_{k+1})}{(\nu z)P_1 \xrightarrow{\gamma_1 \cdot \ldots \cdot \gamma_{k+1}} (\nu z)Q^{'}}$$

*Cong*3 :

$$\textbf{Cong3}\ \frac{(\nu z)L_k^{'} \overset{\gamma_{k+1}}{\longmapsto} Q^{'} \qquad Q^{'} \equiv Q}{(\nu z)L_k^{'} \overset{\gamma_{k+1}}{\longmapsto} Q}$$

for transitivity of $\equiv$ we can assume that there is a derivation of $(\nu z)L_k^{'} \overset{\gamma_{k+1}}{\longmapsto} Q^{'}$ that does not end with an instance of *Cong*3 and so it does end with *Res* or *Opn*:

Res :

$$\textbf{Cong3}\ \frac{\textbf{Res}\ \dfrac{L_k^{'} \overset{\gamma_{k+1}}{\longmapsto} Q^{''}}{(\nu z)L_k^{'} \overset{\gamma_{k+1}}{\longmapsto} (\nu z)Q^{''}} \qquad (\nu z)Q^{''} \equiv Q}{(\nu z)L_k^{'} \overset{\gamma_{k+1}}{\longmapsto} Q}$$

we can derive the following chain of transition:

$$P_1 \overset{\gamma_1}{\longmapsto} L_1^{'} \overset{\gamma_2}{\longmapsto} L_2^{'} \cdots L_{k-1}^{'} \overset{\gamma_k}{\longmapsto} L_k^{'} \overset{\gamma_{k+1}}{\longmapsto} Q^{''}$$

and then apply the rule *Low* and the inductive hypothesis to get

$$P_1 \xrightarrow{\gamma_1 \cdot \cdots \cdot \gamma_{k+1}} Q^{''}$$

A proof of the conclusion is

$$\textbf{Cong}\ \frac{\textbf{Res}\ \dfrac{P_1 \xrightarrow{\gamma_1 \cdot \cdots \cdot \gamma_{k+1}} Q^{''}}{(\nu z)P_1 \xrightarrow{\gamma_1 \cdot \ldots \cdot \gamma_{k+1}} (\nu z)Q^{''}} \qquad (\nu z)Q^{''} \equiv Q}{(\nu z)P_1 \xrightarrow{\gamma_1 \cdot \ldots \cdot \gamma_{k+1}} Q}$$

Opn :

$$\textbf{Cong3}\ \frac{\textbf{Opn}\ \dfrac{L_k^{'} \overset{\overline{x}z}{\longmapsto} Q^{'}}{(\nu z)L_k^{'} \overset{\overline{x}(z)}{\longmapsto} Q^{'}} \qquad Q^{'} \equiv Q}{(\nu z)L_k^{'} \overset{\overline{x}(z)}{\longmapsto} Q}$$

we can derive the following chain of transition:

$$P_1 \overset{\gamma_1}{\longmapsto} L_1^{'} \overset{\gamma_2}{\longmapsto} L_2^{'} \cdots L_{k-1}^{'} \overset{\gamma_k}{\longmapsto} L_k^{'} \overset{\overline{x}z}{\longmapsto} Q^{'}$$

and then apply the rule *Low* and the inductive hypothesis to get

$$P_1 \xrightarrow{\gamma_1 \cdot \cdots \cdot \gamma_k \cdot \overline{x}z} Q^{'}$$

A proof of the conclusion is

$$\textbf{Cong}\ \dfrac{\textbf{OpnSeq}\ \dfrac{P_1 \xrightarrow{\gamma_1\cdot\ldots\cdot\gamma_k\cdot\overline{x}z} Q^{'}}{(\nu z)P_1 \xrightarrow{\gamma_1\cdot\ldots\cdot\gamma_k\cdot\overline{x}(z)} Q^{'}} \qquad Q^{'} \equiv Q}{(\nu z)P_1 \xrightarrow{\gamma_1\cdot\ldots\cdot\gamma_k\cdot\overline{x}(z)} Q}$$

Opn :

$$\textbf{Opn}\ \dfrac{L_k^{'} \overset{\overline{x}z}{\longmapsto} Q \qquad z \notin n(Q)}{(\nu z)L_k^{'} \overset{\overline{x}(z)}{\longmapsto} Q}$$

we can build the following chain of transitions:

$$P_1 \overset{\gamma_1}{\longmapsto} L_1^{'} \overset{\gamma_2}{\longmapsto} L_2^{'} \cdots L_{k-1}^{'} \overset{\gamma_k}{\longmapsto} L_k^{'} \overset{\overline{x}z}{\longmapsto} Q$$

then apply the rule *Low* and the inductive hypothesis to get $P_1 \xrightarrow{\gamma_1\cdot\ldots\cdot\gamma_k\cdot\overline{x}z} Q$. A proof of the conclusion can be

$$\textbf{OpnSeq}\ \dfrac{P_1 \xrightarrow{\gamma_1\cdot\ldots\cdot\gamma_k\cdot\overline{x}z} Q \qquad z \notin n(\gamma_1\cdot\ldots\cdot\gamma_k)}{(\nu z)P_1 \xrightarrow{\gamma_1\cdot\ldots\cdot\gamma_k\cdot\overline{x}(z)} Q}$$

Opn the first transition is:

$$\textbf{Opn}\ \dfrac{P_1 \overset{\overline{x}z}{\longmapsto} L_1 \qquad z \notin n(\overline{x}(z))}{(\nu z)P_1 \overset{\overline{x}(z)}{\longmapsto} L_1}$$

since γ_1 is an output for lemma 4.2.1 $\gamma_2\cdot\ldots\cdot\gamma_{k+1} = \tau$ and $\sigma = \overline{x}(z)$. We derive the following chain of transition:

$$P_1 \overset{\overline{x}z}{\longmapsto} L_1 \overset{\gamma_2}{\longmapsto} L_2 \cdots L_{k-1} \overset{\gamma_k}{\longmapsto} L_k \overset{\gamma_{k+1}}{\longmapsto} Q$$

for rule *Low* and inductive hypothesis

$$P_1 \xrightarrow{\overline{x}z} Q$$

A proof of the conclusion is

$$\textbf{Opn}\ \dfrac{P_1 \xrightarrow{\overline{x}z} Q}{(\nu z)P_1 \xrightarrow{\overline{x}(z)} Q}$$

*Par*1 : the first transition is:

$$\textbf{Par1}\ \dfrac{P_1 \overset{\gamma_1}{\longmapsto} L_1^{'} \qquad bn(\gamma_1) \cap fn(P_2) = \emptyset}{P_1|P_2 \overset{\gamma_1}{\longmapsto} L_1^{'}|P_2}$$

the transitions

$$P \overset{\gamma_1}{\longmapsto} L_1 \overset{\gamma_2}{\longmapsto} \cdots \overset{\gamma_k}{\longmapsto} L_k$$

are such that the last rule used in their derivation is *Par*1. Whether the derivation of the last transition can end with *Par*1 or *Cong*3:

*Par*1 We derive the following chain of transition:

$$P_1 \overset{\gamma_1}{\longmapsto} L_1^{'} \overset{\gamma_2}{\longmapsto} L_2^{'} \cdots L_{k-1}^{'} \overset{\gamma_k}{\longmapsto} L_k^{'} \overset{\gamma_{k+1}}{\longmapsto} Q_1$$

for rule *Low* and inductive hypothesis

$$P_1 \xrightarrow{\sigma} Q_1$$

A proof of the conclusion is

$$\textbf{Par}\ \frac{P_1 \xrightarrow{\sigma} Q_1 \qquad bn(\sigma) \cap fn(P_2) = \emptyset}{P_1|P_2 \xrightarrow{\sigma} Q_1|P_2}$$

$Cong3$ the last part of the derivation of the last transition is

$$\textbf{Cong3}\ \frac{L_k^{'}|P_2 \xmapsto{\gamma_{k+1}} Q^{'} \qquad Q^{'} \equiv Q}{L_k^{'}|P_2 \xmapsto{\gamma_{k+1}} Q}$$

because of the transitivity of $\equiv$ we can assume that there is a derivation of $L_k^{'}|P_2 \xmapsto{\gamma_{k+1}} Q^{'}$ which does not use $Cong3$ as its last instance. So this last instance can only be $Par1$ so $Q^{'} = Q^{''}|P_2$. We derive the following chain of transition:

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

for rule Low and inductive hypothesis

$$P_1 \xrightarrow{\sigma} Q^{''}$$

A proof of the conclusion is

$$\textbf{Cong3}\ \frac{\textbf{Par}\ \dfrac{P_1 \xrightarrow{\sigma} Q^{''} \qquad bn(\sigma) \cap fn(P_2) = \emptyset}{P_1|P_2 \xrightarrow{\sigma} Q^{''}|P_2} \qquad Q^{''}|P_2 \equiv Q}{P_1|P_2 \xrightarrow{\sigma} Q}$$

$Cong2$: the last rule of the derivation of the first transition is:

$$\textbf{Cong2}\ \frac{P^{'} \xmapsto{\gamma_1} L_1 \qquad P^{'} \equiv P}{P \xmapsto{\gamma_1} L_1}$$

We derive the following chain of transition:

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

for rule Low and inductive hypothesis

$$P^{'} \xrightarrow{\sigma} Q$$

A proof of the conclusion is

$$\textbf{Cong2}\ \frac{P^{'} \xrightarrow{\sigma} Q \qquad P^{'} \equiv P}{P \xrightarrow{\sigma} Q}$$

$Com3$: the last part of the derivation of the first transition, having in mind lemma 4.2.2, is:

$$\textbf{Com3}\ \frac{P_1 \xmapsto{xy} L_1^{'} \qquad P_2 \xmapsto{\overline{x}y} P_2^{'}}{P_1|P_2 \xmapsto{\epsilon} L_1^{'}|P_2^{'}}$$

the derivations of the transitions $L_1 \xmapsto{\gamma_2} L_2 \cdots L_{k-1} \xmapsto{\gamma_k} L_k$ can end only with an instance of $Par1$ so

$$\textbf{Par1}\ \frac{L_i^{'} \xmapsto{\gamma_{i+1}} L_{i+1}^{'} \qquad bn(\gamma_{i+1}) \cap fn(P_2^{'}) = \emptyset}{L_i^{'}|P_2^{'} \xmapsto{\gamma_{i+1}} L_{i+1}^{'}|P_2^{'}}$$

The derivation of the last transition $L_k \xmapsto{\gamma_{k+1}} Q$ can end with $Par1$ or with $Cong3$:

$Par1$: We derive the following chain of transition:

$$P_1 \xmapsto{xy} L_1^{'} \xmapsto{\gamma_2} L_2^{'} \cdots L_{k-1}^{'} \xmapsto{\gamma_k} L_k^{'} \xmapsto{\gamma_{k+1}} Q_1$$

for rule *Low* and inductive hypothesis

$$P_1 \xrightarrow{\sigma} Q_1$$

A proof of the conclusion is

$$\textbf{EComSeq}\ \frac{P_1 \xrightarrow{xy\cdot\gamma_2\cdot\ldots\cdot\gamma_{k+1}} Q_1 \qquad P_2 \xrightarrow{\overline{x}y} P_2^{'}}{P_1|P_2 \xrightarrow{\gamma_2\cdot\ldots\cdot\gamma_{k+1}} Q_1|P_2^{'}}$$

and $\gamma_2 \cdot \ldots \cdot \gamma_{k+1} = \sigma$ because the first action γ_1 is ϵ.

$Cong3$: e' facile da scrivere a questo punto

$Par2$: the last part of the derivation of the first transition is:

$$\textbf{Par2}\ \frac{P_1 \xmapsto{\gamma_1} L_1^{'} \qquad bn(\gamma_1) \cap fn(P_2) = \emptyset}{P_1|P_2 \xmapsto{\gamma_1} L_1^{'}| * P_2}$$

$\square$

da sistemare la parte sulle regole simmetriche di com? par? e sum?

Chapter 5

Multi π calculus with strong input and output

5.1 Syntax

As we did whit multi π calculus, we suppose that we have a countable set of names $\mathbb{N}$, ranged over by lower case letters $a, b, \cdots, z$. 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 $P, Q, \cdots$. A multi π process, in addiction to the same actions of a π process, can perform also a strong prefix:

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

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

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

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:

$$\begin{array}{rclrcl} B(\underline{x(y)}.Q, I) & = & \{y, \overline{y}\} \cup B(Q, I) & F(\underline{x(y)}.Q, I) & = & \{x, \overline{x}\} \cup (F(Q, I) - \{y, \overline{y}\}) \\ B(\underline{\overline{x}y}.Q, I) & = & B(Q, I) & F(\underline{\overline{x}y}.Q, I) & = & \{x, \overline{x}, y, \overline{y}\} \cup F(Q, I) \end{array}$$

5.2 Operational semantic

5.2.1 Early operational semantic with structural congruence

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 $\overline{x}(y)$
  - unbound output $\overline{x}y$
  - bound input $x(z)$

  We use $\alpha, \alpha_1, \alpha_2, \cdots$ to range over the set of actions, we use $\sigma, \sigma_1, \sigma_2, \cdots$ to range over the set $\mathbb{A}_m^+ \cup \{\tau\}$.
- the transition relations is $\rightarrow \subseteq \mathbb{P}_m \times (\mathbb{A}_m^+ \cup \{\tau\}) \times \mathbb{P}_m$

$$\textbf{Pref}\ \frac{\alpha\ not\ a\ strong\ prefix}{\alpha.P \xrightarrow{\alpha} P} \qquad \textbf{Par}\ \frac{P \xrightarrow{\sigma} P' \quad bn(\sigma) \cap fn(Q) = \emptyset}{P|Q \xrightarrow{\sigma} P'|Q}$$

$$\textbf{SOut}\ \frac{P \xrightarrow{\sigma} P' \quad \sigma \neq \tau}{\underline{\overline{x}y}.P \xrightarrow{\overline{x}y \cdot \sigma} P'} \qquad \textbf{LCom}\ \frac{P \xrightarrow{\sigma_1} P' \qquad Q \xrightarrow{\sigma_2} Q' \qquad Sync(\sigma_1, \sigma_2, \sigma_3, \delta_1, \delta_2)}{P|Q \xrightarrow{\sigma_3} P'\delta_1|Q'\delta_2}$$

$$\textbf{Sum}\ \frac{P \xrightarrow{\sigma} P'}{P + Q \xrightarrow{\sigma} P'} \qquad \textbf{Str}\ \frac{P \equiv P' \qquad P' \xrightarrow{\alpha} Q' \qquad Q \equiv Q'}{P \xrightarrow{\alpha} Q}$$

$$\textbf{Res}\ \frac{P \xrightarrow{\sigma} P' \quad z \notin n(\alpha)}{(\nu z)P \xrightarrow{\sigma} (\nu z)P'} \qquad \textbf{SInp}\ \frac{P \xrightarrow{\sigma} P' \quad \sigma \neq \tau}{\underline{x(y)}.P \xrightarrow{x(y) \cdot \sigma} P'}$$

Table 5.1: Multi π late 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 5.2.1. The *late transition relation with structural congruence* is the smallest relation induced by the rules in table 5.1:

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 5.2

Transactional synchronization This is an example of two processes that synchronize over a sequence of actions of length two:

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

We start first noticing that

$$\text{S4R}\ \frac{\text{S1R}\ \frac{}{Sync(\overline{a}y, a(z)\{x/w\}, \tau, \{\}, \{y/z\})}}{Sync(\overline{a}x \cdot \overline{a}y, a(w) \cdot a(z), \tau, \{\}, \{x/w\}\{y/z\})}$$

and that

$$\text{SOut}\ \frac{\text{Pref}\ \frac{}{\overline{a}y.P \xrightarrow{\overline{a}y} P}}{\underline{\overline{a}x}.\overline{a}y.P \xrightarrow{\overline{a}x \cdot \overline{a}y} P} \qquad \text{SInp}\ \frac{\text{Pref}\ \frac{}{a(z).Q \xrightarrow{a(z)} Q}}{\underline{a(w)}.a(z).Q \xrightarrow{a(w) \cdot a(z)} Q}$$

and in the end we just need to apply the rule **LCom**

$$\frac{\frac{\frac{\frac{\frac{\frac{\frac{\frac{}{\text{p}} \quad \text{o}}{\text{n}}}{\text{m}}}{\text{l}}}{\text{i}} \qquad \frac{\frac{}{\text{h}}}{\text{g}}}{\text{f}} \qquad \frac{\frac{}{\text{e}}}{\text{d}}}{\text{c}}}{\text{b}}}{\text{a}}$$

□

Multi-party synchronization In this example we have three processes that want to synchronize:

S1L $\dfrac{}{Sync(x(y), \overline{x}z, \tau, \{z/y\}, \{\})}$	S1R $\dfrac{}{Sync(\overline{x}z, x(y), \tau, \{\}, \{z/y\})}$
S2L $\dfrac{}{Sync(x(y), \overline{x}z \cdot \sigma, \sigma, \{z/y\}, \{\})}$	S2R $\dfrac{}{Sync(\overline{x}z \cdot \sigma, x(y), \sigma, \{\}, \{z/y\})}$
S3L $\dfrac{}{Sync(x(y) \cdot \sigma, \overline{x}z, \sigma\{z/y\}, \{z/y\}, \{\})}$	S3R $\dfrac{}{Sync(\overline{x}z, x(y) \cdot \sigma, \sigma\{z/y\}, \{\}, \{z/y\})}$
S4L $\dfrac{Sync(\sigma_1, \sigma_2\{z/y\}, \sigma_3, \delta_1, \delta_2)}{Sync(x(y) \cdot \sigma_1, \overline{x}z \cdot \sigma_2, \sigma_3, \{z/y\}\delta_1, \delta_2)}$	S4R $\dfrac{Sync(\sigma_1, \sigma_2\{z/y\}, \sigma_3, \delta_1, \delta_2)}{Sync(\overline{x}z \cdot \sigma_1, x(y) \cdot \sigma_2, \sigma_3, \delta_1, \{z/y\}\delta_2)}$
I1L $\dfrac{Sync(\sigma_1, \sigma_2, \tau, \delta_1, \delta_2)}{Sync(\alpha \cdot \sigma_1, \sigma_2, \alpha, \delta_1, \delta_2)}$	I1R $\dfrac{Sync(\sigma_1, \sigma_2, \tau, \delta_1, \delta_2)}{Sync(\sigma_1, \alpha \cdot \sigma_2, \alpha, \delta_1, \delta_2)}$
I2L $\dfrac{Sync(\sigma_1, \sigma_2, \sigma_3, \delta_1, \delta_2)}{Sync(\alpha \cdot \sigma_1, \sigma_2, \alpha \cdot \sigma_3, \delta_1, \delta_2)}$	I2R $\dfrac{Sync(\sigma_1, \sigma_2, \sigma_3, \delta_1, \delta_2)}{Sync(\sigma_1, \alpha \cdot \sigma_2, \alpha \cdot \sigma_3, \delta_1, \delta_2)}$
I3L $\dfrac{}{Sync(\alpha, \sigma, \alpha \cdot \sigma, \delta_1, \delta_2)}$	I3R $\dfrac{}{Sync(\sigma, \alpha, \alpha \cdot \sigma, \delta_1, \delta_2)}$
I4L $\dfrac{}{Sync(\epsilon, \sigma, \sigma, \delta_1, \delta_2)}$	I4R $\dfrac{}{Sync(\sigma, \epsilon, \sigma, \delta_1, \delta_2)}$

Table 5.2: Synchronization relation

$$\textbf{LCom}\ \dfrac{\underline{\overline{a}f}.\overline{b}g.P|a(w).Q \xrightarrow{\overline{b}g} P|Q\{f/w\} \qquad \textbf{Pref}\ \dfrac{}{b(y).R \xrightarrow{b(y)} R} \qquad \textbf{S1R}\ \dfrac{}{Sync(\overline{b}g, b(y), \tau, \emptyset, \{g/y\})}}{(\underline{\overline{a}f}.\overline{b}g.P|a(w).Q)|b(y).R \xrightarrow{\tau} (P|Q\{f/w\})|R\{g/y\}}$$

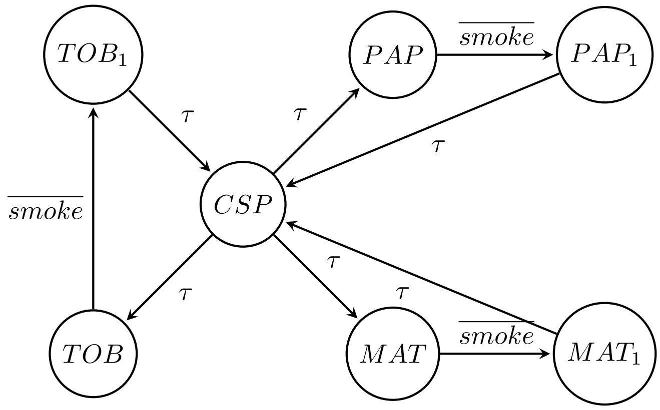
$$\textbf{LCom}\ \dfrac{\underline{\overline{a}f}.\overline{b}g.P \xrightarrow{\overline{a}f \cdot \overline{b}g} P \qquad \textbf{Pref}\ \dfrac{}{a(w).Q \xrightarrow{a(w)} Q} \qquad \textbf{S2R}\ \dfrac{}{Sync(\overline{a}f \cdot \overline{b}g, a(w), \overline{b}g, \emptyset, \{f/w\})}}{\underline{\overline{a}f}.\overline{b}g.P|a(w).Q \xrightarrow{\overline{b}g} P|Q\{f/w\}}$$

$$\textbf{SOut}\ \dfrac{\textbf{Out}\ \dfrac{}{\overline{b}g.P \xrightarrow{\overline{b}g} P}}{\underline{\overline{a}f}.\overline{b}g.P \xrightarrow{\overline{a}f \cdot \overline{b}g} P}$$

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:

$$\begin{aligned}
&Agent \stackrel{def}{=} \underline{\overline{tob}}.\overline{mat}.end().Agent + \underline{\overline{mat}}.\overline{pap}.end().Agent + \underline{\overline{pap}}.\overline{tob}.end().Agent \\
&S_{pap} \stackrel{def}{=} \underline{tob()}.mat().\overline{smoke}.\overline{end}.S_{pap} \\
&S_{tab} \stackrel{def}{=} \underline{mat()}.pap().\overline{smoke}.\overline{end}.S_{tab} \\
&S_{mat} \stackrel{def}{=} \underline{pap()}.tob().\overline{smoke}.\overline{end}.S_{mat} \\
&CSP \stackrel{def}{=} (\nu tob, pap, mat, end)(Agent|S_{tob}|S_{mat}|S_{pap})
\end{aligned}$$

The semantic of CSP is the following graph:



where

$$
\begin{aligned}
PAP &\stackrel{def}{=} (\nu tob, pap, mat, end)(end().Agent|S_{tob}|S_{mat}|\overline{smoke}.\overline{end}.S_{pap}) \\
TOB &\stackrel{def}{=} (\nu tob, pap, mat, end)(end().Agent|\overline{smoke}.\overline{end}.S_{tob}|S_{mat}|S_{pap}) \\
MAT &\stackrel{def}{=} (\nu tob, pap, mat, end)(end().Agent|S_{tob}|\overline{smoke}.\overline{end}.S_{mat}|S_{pap}) \\
PAP_1 &\stackrel{def}{=} (\nu tob, pap, mat, end)(end().Agent|S_{tob}|S_{mat}|\overline{end}.S_{pap}) \\
TOB_1 &\stackrel{def}{=} (\nu tob, pap, mat, end)(end().Agent|\overline{end}.S_{tob}|S_{mat}|S_{pap}) \\
MAT_1 &\stackrel{def}{=} (\nu tob, pap, mat, end)(end().Agent|S_{tob}|\overline{end}.S_{mat}|S_{pap})
\end{aligned}
$$

Bibliography

[1] Roberto Gorrieri, Cristian Versari, *Multi π: a calculus for mobile multi-party and transactional communication.*

[2] E. W. Dijkstra, *Hierarchical ordering of sequential processes*, Acta Informatica 1(2):115-138,

[3] controllare le etichette di tutte le semantiche: in part 1971.

[4] Robin Milner, Joachim Parrow, David Walker, *A calculus of mobile processes, part II*, 1990.

[5] Roberto Gorrieri, *A fully-abstract semantics for atomicity*, Dipartimento di scienze dell'informazione, Universitá di Bologna.

[6] Joachin Parrow, *An introduction to the π calculus*, Department Teleinformatics, Rotal Institute of Technology, Stockholm.

[7] Davide Sangiorgi, David Walker, *The π-calculus*, Cambridge University Press.

[8] Davide Sangiorgi, *A theory of bisimulation for the π-calculus*, Acta informatica, 33(1):69-97, 1996.

[9] Milner, Robin, *Communicating and mobile systems: the π-calculus*, Cambridge University Press.

[10] MohammedReza Mousavi, Michel A Reniers, *Congruence for structural congruences*, Department of Computer Science, Eindhoven University of Technology.