A compositional semantics for the reversible π-calculus

Ioana Domnina Cristescu* Jean Krivine Daniele Varacca
Preuves, Programmes et Systèmes,
UMR 7126, CNRS and University Paris Diderot

***Abstract*—In the present paper, we introduce a fully compositional semantics for the reversible π-calculus. It is the first account of a compositional definition of a reversible calculus, that has both concurrency primitives and name mobility.**

The notion of reversibility is strictly linked to the notion of causality. We discuss the notion of causality induced by our calculus, and we compare with the existing notions in the literature, in particular for what concerns the syntactic feature of scope extrusion, typical of the π-calculus.

I. Introduction

A. Reversibility matters

Being able to reverse a computation is often an important feature of computing systems although not always studied as a topic of its own. In sequential systems, step by step rewinding of a computation is a common way of debugging programs. Also (reversible) programs running on logically reversible gates are known to have good properties with respect to energy consumption [1]. In the concurrent world, reversibility is a key aspect in every system that needs to solve distributed consensus [2] in order to escape local states where the consensus cannot be found. However in the concurrent case, rewinding a computation requires to define first what is a legitimate backward move from a given state, in a context where the computation is no longer functional.

A formal model for concurrent systems needs to address two challenges at the same time: (i) how to compute without forgetting and (ii) what is an optimal notion of legitimate backward moves. Roughly speaking, the first point –that needs to be answered in the sequential world as well– is about syntax: processes need to carry a memory that keep track of everything that has been done (and of the choices that have not been made). The second point is tied to the choice of the computation's semantics. In a concurrent setting we do not want to undo the actions precisely in the opposite order than the one in which they were executed, as this order is immaterial. The concurrency relation between actions has to be taken into account. Semantics that represent explicitly the concurrency of actions usually come equipped with a notion of *causality*.

We argue that the most liberal notion of reversibility is the one that just respects causality: an action can be undone precisely after all the actions that causally depend on it have also been undone.

* supported by ANR REVER

B. Our contributions

We are not the first to observe that causality and reversibility are tightly connected notions [3], [4]. Also, there are already several accounts of reversible languages for concurrency [5], [6], [7], and even of the (higher-order) π-calculus [8]. In spite of that, we think this paper makes important contributions.

First of all, we believe the existing approaches are not fully satisfactory. Distributed computations done in CCS are rather limited in scope because of the absence of name mobility. As soon as name creation (and exchange) is enabled, computing without forgetting becomes difficult because of the variable substitutions and also because the scope of a name, that may increase in forward computation, should decrease accordingly during backtracking. Also, although the reversible Hoπ that has been proposed [8] is a clear gain in expressivity over CCS, it is only given in terms of reduction semantics and therefore not compositional.

We believe that the present study addresses the challenges that were deliberately left aside in the previous works, namely a compositional definition of a reversible calculus, that has both concurrency primitives and name mobility. As we will see, achieving compositionality is far from trivial, in the same way as the standard labelled transition semantics of the π-calculus is not a trivial extension of its reduction semantics.

But our contributions are also in the realm of the causal semantics of the π-calculus. We take the stance that the *concrete* events of a computation are the *reductions*, ie. the steps that a closed system does. Labelled transitions are then considered as *abstract* or incomplete events, that await a suitable context to become concrete. In other words, they only exist for the sake of compositionality.

As a consequence, the concrete causality relation between reductions is the one that is induced by the prefix operator (the "dot") and propagated through communications, also called *structural* dependence. Which notion of causality should then be considered on labelled transitions? For a simple calculus like CCS the answer is trivial because the causality between labelled transitions is also purely structural, but it is no longer true in the π-calculus because of the dependency induced by the scope extrusion.

To be as liberal as possible for backtracking, the causality between labelled transitions that should be respected needs

to be the *smallest* relation that is consistent with the structural causality between reductions. More precisely, there should be a causal relation between labelled transitions of a process only if every possible pair of reductions obtained by "completion" of those transitions (by parallel composition) are also causally related. This would guarantee that if a backward labelled transition is not derivable in our semantics, it is because any corresponding reduction would violate the structural causality. There are several works that add different notions of causality to the labelled transition system of the π-calculus [9], [10]. Although the causal semantics that is induced by our semantics is related to them, ours is the only one, to the best of our knowledge, that satisfies the above requirement, which is formalized by Theorem 5.5 of Section IV.

C. Other notable features

- In the purely forward direction, our semantics is just a decoration over the classical π-calculus: by forgetting additional annotations, we retrieve the (late) labelled transition semantics of the standard π-calculus. This can be considered as a sanity check.
- Our semantics is not only *compositional* but also *structural*. That is, the semantics of a process is obtained by structural rules from the semantics of its direct subprocesses. Compositionality requires in particular that rules for scope extrusion are needed. Making these rules reversible is one of the main technical challenges of the present work.
- The notion of causality that is induced by our semantics is *stable*: every event carries with itself its unambiguous causal history. This in contrast with the causal semantics of the π-calculus proposed in Ref. [11]. A full comparison of the present work with the event structure semantics is our current interest.

D. Outline

This paper is organized as follows: In Section II we introduce the syntax and the labelled transition semantics for the reversible π calculus and we show its main properties in Section III. In Section V we discuss the notion of causality induced by our semantics. In Section VI we conclude with a summary of our work and its consequences.

II. The reversible π-calculus

In this section we present the compositional semantics of the reversible π-calculus (Rπ). In order to define the reversible operational semantics (Section II-B), we need first to introduce our meta variables and go through a few definitions (Section II-A).

A. Statics

1) Terms: We use a, b, c to range over channel names and P, Q to range over π calculus processes, defined as follows:

$$P, Q ::= 0 \mid \pi.P \mid (P \mid Q) \mid \nu a(P)$$

where $\pi ::= b(c) \mid \overline{b}\langle a\rangle \mid \tau$ denotes traditional π prefixes. We introduce neither choice nor replication. This restriction of expressivity is only in order to simplify the presentation, and these operators would pose no technical issues in the following developments.

As in RCCS [5], Rπ processes are built upon simple π processes to which we add a memory that will keep track of past actions. Every entry in a memory is called a *(memory) event* and can be used to trigger backward moves. From now on the term *process* will refer to Rπ processes.

We use $\mathcal{I}$ for the set of event identifiers, with a distinguished symbol $* \in \mathcal{I}$ that will denote partial synchronization. Let i, j, k range over elements of $\mathcal{I}$ and Δ, Γ range over subsets of $\mathcal{I}$. Rπ terms are built according to the following grammar:

$$\begin{array}{l}\text{(Event labels) } \alpha ::= \overline{b}\langle a\rangle \mid b[\star/c] \mid b[a/c]\\ \text{(Memory events) } e ::= \langle i, k, \alpha\rangle\\ \text{(Memory stacks) } m ::= \varepsilon \mid \langle\uparrow\rangle.m \mid e.m\\ \text{(R}\pi\text{ processes) } R, S ::= 0 \mid m \rhd P \mid (R \parallel S) \mid \nu a_\Gamma(R)\end{array}$$

In the style of RCCS, Rπ memories are structured as stacks, the top element being of the left and the empty stack being denoted by ε. There are two types of information that can be pushed on a memory: either a fork symbol $\langle\uparrow\rangle$, which allows memory stacks to divide whenever processes are forking, and events which are triplets of the form $\langle i, k, \alpha\rangle$. For any event $e = \langle i, k, \alpha\rangle$, we say that i is the *identifier* of e, k is the identifier of its *contextual cause* and α its *label*. The label of an event used to record the prefix that was consumed during a transition, but also acts as an explicit substitution that allows one not to lose information about variable scope. We will come back to this important point in Section II-A3. The notations $id(e), c(e)$ and $\lambda(e)$ give access to the identifier, the contextual cause and the label of e respectively.

The restriction $\nu a_\Gamma(R)$ and the parallel composition of processes $R \parallel S$ reflect the corresponding operators of π-processes thanks to the following structural rules:

$$m \rhd (P_1 \mid P_2) \quad \equiv_m \quad \langle\uparrow\rangle.m \rhd P_1 \parallel \langle\uparrow\rangle.m \rhd P_2 \quad (1)$$

$$m \rhd \nu a(P) \quad \equiv_m \quad \nu a_\emptyset(m \rhd P) \text{ with } a \notin m \quad (2)$$

which distribute a memory whenever two π processes are forking (1), and push classical π calculus restrictions at the level of processes (2). Note that an Rπ restriction is indexed by a set $\Gamma \subset \mathcal{I}$ (initially empty) and behaves as a classical restriction only when $\Gamma = \emptyset$. It will be used to keep track of past variable scope whenever $\Gamma \neq \emptyset$ (see Section II-A2).

It is noteworthy that not all syntactically correct processes are semantically meaningful. Indeed processes contain a computation history composed of past interactions, stored in the memories, and past variable scope, recorded by the νa_Γ constructs. History consistency cannot be easily enforced

statically[1]. For the present work it will suffice to consider only the set of terms, called *reachable*, that contains the obviously sound process $\varepsilon \rhd P$ and closed under the operational semantics of Rπ.

2) Names, scope and substitutions: In a process R, a channel a can be *bound* ($a \in \mathrm{bn}(R)$), *free* ($a \in \mathrm{fn}(R)$) or *liberated* ($a \in \mathrm{lib}(R)$). While free and bound names are as usual, one may think of liberated names as channels that used to be under the scope of a restriction that is no longer there because of an extrusion. They are the names that fall under the scope of the construct $\nu a_{\Gamma\neq\emptyset}(R)$, which then behaves as the "ghost" of a restriction in R with the set Γ containing the identifiers of all the events that have extruded the name a out of R.

Free and liberated names are defined inductively on the structure of processes ($+$ and $-$ denote classical operations on sets, $f(a)$ denotes either $\mathrm{fn}(a)$ or $\mathrm{lib}(a)$ whenever the distinction is irrelevant):

$$
\begin{array}{l}
f(\nu a_\emptyset R) = f(R) - a \\
(\Gamma \neq \emptyset) \quad f(\nu a_\Gamma R) = f(R) + a \\
f(R \parallel S) = f(R) \parallel f(S) \\
\mathrm{fn}(m \rhd P) = names(m) + \mathrm{fn}(P) \quad \mathrm{lib}(m \rhd P) = \emptyset \\
\mathrm{fn}(b(a).P) = b + (\mathrm{fn}(P) - \{a\}) \\
\mathrm{fn}(\overline{b}\langle a\rangle.P) = \mathrm{fn}(P) + a + b
\end{array}
$$

with $names(m)$ being all the names occurring in the memory m. It is obvious from the above definition that all liberated names are free. As usual, names which are not free in R are called bound.

The operational semantics of Rπ is built on top of the so called "late" semantics of the π-calculus, where substitutions on variables occur upon synchronization. Since substitutions are forgetful operations that cannot be always reversed correctly, we replace them with *explicit* substitutions that are logged in the event labels (see Section II-B2). We will also see that a process communicating on a liberated channel, has to make an assumption on the identity of the event that made the channel public (via an extrusion), called its contextual cause. Since the initial assumption can be made more precise while more structure of the process is revealed by the LTS, the contextual cause may also be updated in a "late" fashion. We thus need to define the following special substitutions on processes:

Definition 2.1: The *synchronization update*, denoted by $R_{[a/c]@i}$, replaces the partial substitution $[\star/c]$ with the complete substitution $[a/c]$ at the event identified by i, it is defined as:

$$
\begin{array}{l}
(R \parallel S)_{[a/c]@i} = R_{[a/c]@i} \parallel S_{[a/c]@i} \\
(\nu a_\Gamma R)_{[a/c]@i} = \nu a_\Gamma (R_{[a/c]@i}) \\
(\langle i, \_, b[\star/c]\rangle.m \rhd P)_{[a/c]@i} = \langle i, \_, b[a/c]\rangle.m \rhd P \\
(m \rhd P)_{[a/c]@i} = m \rhd P \quad otherwise
\end{array}
$$

[1]We believe it should be possible to enforce consistent history through a careful typing system, but we leave this point for future investigations.

The *contextual cause update*, denoted by $R_{[k/k']@i}$ proceeds similarly but substitutes the old cause k' for a new one:

$$
\begin{array}{l}
(R \parallel S)_{[k/k']@i} = R_{[k/k']@i} \parallel S_{[k/k']@i} \\
(\nu a_\Gamma R)_{[k/k']@i} = \nu a_\Gamma (R_{[k/k']@i}) \\
(\langle i, k', \_\rangle.m \rhd P)_{[k/k']@i} = \langle i, k, \_\rangle.m \rhd P \\
(m \rhd P)_{[k/k']@i} = m \rhd P \quad otherwise
\end{array}
$$

3) Memories and events: We will use the following intuitive notations: we write $m \in R$ if there exists a context $C[\bullet]$ such that $R = C[m \rhd P]$. Similarly we write $e \in R$ when there is $m \in R$ such that $m = m_1.e.m_0$ for some (possibly empty) m_1 and m_0. Finally for all $i \in \mathcal{I}$ we write $i \in R$ if there exists $e \in R$ such that $id(e) = i$ or $c(e) = i$.

There are 3 relations between events that we need to consider.

Definition 2.2 (Relations on events): Let R be a process, we define the following relations on events of R.

- *Structural causal relation*: $e' \sqsubset_R e$ if there exists $m \in R$ such that $m = m_2.e.m_1.e'.m_0$ for some (possibly empty) m_2, m_1, m_0.
- *Contextual causal relation*: $e' \prec_R e$ if $c(e) = id(e')$.
- *Instantiation relation*: $e' \rightsquigarrow_R e$ if $e' \sqsubset_R e$ and $\lambda(e') = b[a/c]$, for some name a, b, c and c is in subject position in $\lambda(e)$. Furthermore for all memory m and all $e \in m$, such that $e = \langle i, k, b[a/c]\rangle$, we write $inst_m(c) = i$ for the identifier of the event e in m that instantiates c. Note that there is at most one such event in m. If no such event exists in m we write $inst_m(c) = *$.

Example 2.1: In the process

$$\nu a_\emptyset(\nu a_{\{i_1\}}(\langle i_1, *, \overline{b}\langle a\rangle\rangle.m_1 \rhd P_1 \| \langle i_2, i_1, a\rangle.m_2 \rhd P_2)$$

$$\| \langle i_2, *, \overline{c}\rangle.\langle i_1, *, b[a/c]\rangle.m_3 \rhd P_3)$$

we have:

$$
\begin{array}{ll}
\langle i_1, *, b[a/c]\rangle \sqsubset \langle i_2, *, \overline{c}\rangle & \langle i_1, *, \overline{b}\langle a\rangle\rangle \prec \langle i_2, i_1, a\rangle \\
\langle i_1, *, b[a/c]\rangle \rightsquigarrow \langle i_2, *, \overline{c}\rangle & inst_{\langle i_2, *, \overline{c}\rangle.\langle i_1, *, b[a/c]\rangle.m_3}(c) = i_1
\end{array}
$$

For any events $e \in R$ and $e' \in R$ such that $id(e) = i$ and $id(e') = j$, we use the overloaded notations $i \sqsubset_R j$, $i \prec_R j$ or $i \rightsquigarrow_R j$, if e and e' are in the corresponding relation. Note that there are at most two events e and e' such that $id(e) = id(e')$, in which case (e, e') forms a *synchronization pair*.

B. Dynamics

1) Transitions and transition labels: The label ζ of a transition $t : R \xrightarrow{\zeta} S$ is a quadruple of the form $(i, j, k) : \gamma$ where $i \in \mathcal{I} - \{*\}$ is the *identifier* of t, $j \in \mathcal{I}$ is the instantiator of i and $k \in \mathcal{I}$ is the contextual cause of i. The labels γ are built on the following grammar:

$$
\begin{array}{lcl}
\gamma & ::= & \alpha \mid \alpha^- \\
\alpha & ::= & b(c) \mid \overline{b}\langle a\rangle \mid \overline{b}(\nu a_\Gamma)
\end{array}
$$

where $\overline{b}(\nu a_\Gamma)$ corresponds to the bound output of the π calculus, whenever $\Gamma = \emptyset$, and otherwise corresponds to a free output, decorated with a set of event identifiers.

For all label γ of the form α or α^-, we write $subj(\gamma) = b$ if $\alpha \in \{b(c), \overline{b}\langle a\rangle, \overline{b}(\nu a_\Gamma)\}$ for some a. We also write $\mathrm{bn}(\gamma) = \{a\}$ whenever $\alpha = \overline{b}\langle \nu a_{\Gamma\neq\emptyset}\rangle$ for some b. A transition is *positive* whenever its label is of the form α, and *negative* if the label is of the form α^-. It is *derivable* if it can be obtained form the LTS presented in the next section.

As we already hinted at, Rπ substitutions are not executed directly but simply logged in event labels. As a consequence, processes need to search in their memories for the public name of a channel in order to check that a synchronization is possible. Such operation is performed only on demand, when a process is trying to reduce its prefix (see IN+ and OUT+ axioms in Section II-B2).

Definition 2.3 (Public label): For all process of the form $m \rhd \pi.P$ let $m[\pi]$ be the *public label* of π. It is defined by lexicographical induction on the pair (π,m):

$$\varepsilon[a] = a \quad m[b(c)] = m[b](c) \quad m[\overline{b}\langle a\rangle] = \overline{m[b]}\langle m[a]\rangle$$
$$(\langle i,k,b[c/a]\rangle.m)[a] = c \quad (\langle i,k,b[\star/a]\rangle.m)[a] = a$$
$$(\langle\uparrow\rangle.m)[a] = m[a] \quad (e.m)[a] = m[a] \quad otherwise$$

2) The labelled transition system (LTS): The labelled transition system of Rπ can be divided into positive and negative rules. The negative ones are derived from the positive ones by inversion (see Definition 2.4). The positive rules are:

IN+
$$\frac{i \notin m \quad j = inst_m(b)}{m \rhd b(c).P \xrightarrow{(i,j,*):m[b(c)]} \langle i,*,b[\star/c]\rangle.m \rhd P}$$

OUT+
$$\frac{i \notin m \quad j = inst_m(b)}{m \rhd \overline{b}\langle a\rangle.P \xrightarrow{(i,j,*):m[\overline{b}\langle a\rangle]} \langle i,*,\overline{b}\langle a\rangle\rangle.m \rhd P}$$

OPEN+
$$\frac{R \xrightarrow{(i,j,k):\alpha} R' \qquad \alpha = \overline{b}\langle a\rangle \vee \alpha = \overline{b}\langle \nu a_{\Gamma'}\rangle}{\nu a_\Gamma R \xrightarrow{(i,j,k):\overline{b}\langle \nu a_\Gamma\rangle} \nu a_{\Gamma+i} R'}$$

CAUSE REF+
$$\frac{R \xrightarrow{(i,j,k):\alpha} R' \ a \in subj(\alpha)}{\nu a_\Gamma R \xrightarrow{(i,j,k'):\alpha} \nu a_\Gamma R'_{[k'/k]@i}} \begin{array}{l} k = k' \ or \\ \exists k' \in \Gamma \ k \rightsquigarrow_R k' \end{array}$$

COM+
$$\frac{R \xrightarrow{(i,j,k):\overline{b}\langle a\rangle} R' \qquad S \xrightarrow{(i,j',k'):b(c)} S'}{R \parallel S \xrightarrow{(i,*,*):\tau} R' \parallel S'_{[a/c]@i}} \begin{array}{l} k =_* j' \\ k' =_* j \end{array}$$

CLOSE+
$$\frac{R \xrightarrow{(i,j,k):\overline{b}\langle \nu a_\Gamma\rangle} R' \qquad S \xrightarrow{(i,j',k'):b(c)} S'}{R \parallel S \xrightarrow{(i,*,*):\tau} \nu a_\Gamma(R' \parallel S'_{[a/c]@i})} \begin{array}{l} k =_* j' \\ k' =_* j \end{array}$$

$$with\ a \notin \mathrm{fn}(S)\ whenever\ \Gamma = \emptyset$$

PAR+
$$\frac{R \xrightarrow{(i,j,k):\alpha} R'}{R \parallel S \xrightarrow{(i,j,k):\alpha} R' \parallel S} \mathrm{bn}(\alpha) \cap \mathrm{fn}(S) = \emptyset, i \notin S$$

MEM+
$$\frac{R \equiv_m S \xrightarrow{\zeta} S' \equiv_m R'}{R \xrightarrow{\zeta} R'}$$

NEW+
$$\frac{R \xrightarrow{\zeta} R'}{\nu a_\Gamma R \xrightarrow{\zeta} \nu a_\Gamma R'} a \notin \zeta$$

with for all $i, j \in I$, $i =_* j$ if $* \in \{i, j\}$ or $i = j$.

Note that the complete positive LTS contains also the symmetrical rules for the COM+, CLOSE+ and PAR+ rules with respect to the $\parallel$ operator. For lack of space, we do not write them explicitly.

The backward rules are derived according to the following definition:

Definition 2.4 (Inverting operation): Let $\alpha^{-1} = \alpha^-$ and $(\alpha^-)^{-1} = \alpha$. Let *opp* be the operation defined in a functorial manner on labeled transition systems as:

$$opp(R \xrightarrow{(i,j,k):\gamma} S) = S \xrightarrow{(i,j,k):\gamma^{-1}} R$$

and

$$opp\left(\frac{R \xrightarrow{\zeta} S}{R' \xrightarrow{\zeta'} S'}\right) = \frac{opp(R \xrightarrow{\zeta} S)}{opp(R' \xrightarrow{\zeta'} S')}$$

$$opp\left(\frac{R_1 \xrightarrow{\zeta_1} S_1 \qquad R_2 \xrightarrow{\zeta_2} S_2}{T \xrightarrow{\zeta'} T'}\right) =$$

$$\frac{opp(R_1 \xrightarrow{\zeta_1} S_1) \qquad opp(R_2 \xrightarrow{\zeta_2} S_2)}{opp(T \xrightarrow{\zeta'} T')}$$

Side conditions are invariant. For all processs R, let $\mathcal{L}^+(R) = (R, \rightarrow)$ be the *positive* LTS of R and $\mathcal{L}^-(R) = (R, opp(\rightarrow))$ its *negative* version. The *reversible operational semantics* of R is defined as $\mathcal{L}(R) = \mathcal{L}^+(R) \cup \mathcal{L}^-(R)$.

3) Discussion:

Axioms: IN+ and OUT+ add an event e into the memory and apply the necessary substitutions on the transition label. The event identifier is locally fresh, as ensured by the side condition $i \notin m$.

Name extrusion: In Rπ, the role of the Γ-restriction $\nu a_\Gamma(R)$ is to act as a boundary that delimitates the past and present scope of a in R. Intuitively any partial synchronization (either input or output) on channel a emanating from R needs to pick inside Γ an event identifier which will act as a proof that some process in the context knows a. As a consequence, if $\Gamma = \emptyset$ no partial synchronization on a may cross this boundary and $\nu a_\emptyset$ behaves as a classical π-calculus restriction. The role of the OPEN+ rule is to update

Γ each time a process in R is sending a to the context[2] (see also Example 2.2).

Importantly, because of possible successive extrusions, Γ-restrictions may be nested inside each others. Each time a partial synchronization on a liberated name crosses such boundary, the LTS updates the contextual cause (ie. the proof that a complete synch may eventually occur) that was chosen so far. The role of the CAUSE REF+ rule is to make sure a partial synchronization on a chooses a correct contextual cause. Critically for the unicity of derivations (see Proposition 3.2), the way a cause is updated is not arbitrary, as indicated by the side condition of the CAUSE REF+ rule. In a nutshell, when passing a Γ-restriction, a contextual cause k may either be preserved if $\Gamma \in k$ or replaced by any $k' \in \Gamma$ such that $k \rightsquigarrow_R k'$. We will see that there always exists at least a $k' \in \Gamma$ such that $k \rightsquigarrow_R k'$ (see Propositions 3.4 and 3.6) so the CAUSE REF+ rule is never blocking if $\Gamma \neq \emptyset$.

Synchronizations: Two partial synchronizations may compose only if they agree on the public channel name in subject position (in the rule COM+ and CLOSE+ rules this is channel b). Such public name is deduced in the LTS at the level of the axiom applications. The side conditions of both synch rules proceeds with the following intuition: if the left premise of transition i learned the name b thanks to an earlier communication j, then $j \neq *$ in the transition label. There are then two cases for the right premise of transition i: either $k' = *$, in which case no assumption was made on the contextual cause of this transition and the synchronization may occur (since $j =_* *$), or $k' \neq *$. In the latter case, the leftmost sub-derivation had to cross a Γ-restriction and one must make sure that the chosen contextual cause k' coincides with the instantiator of the left derivation, ie. $k' = j$. The argument is symmetric if one starts with the instantiator of the leftmost derivation.

Example 2.2: Consider the process (empty memory stacks and empty π-processes are omitted):

$$R = \nu a_\emptyset \left((\overline{b}\langle a \rangle \| \overline{c}\langle a \rangle) \| a \right)$$

The following trace is derivable (we use integers for identifiers and $(i, j, k) : \alpha$ is written $i : \alpha$ whenever $j = k = *$):

$$\begin{array}{rll} R & \xrightarrow{1:\overline{b}\langle \nu a_\emptyset \rangle} & \nu a_1 \left((\langle 1, *, \overline{b}\langle a \rangle \rangle \| \overline{c}\langle a \rangle) \| a \right) \\ & \xrightarrow{2:\overline{c}\langle \nu a_1 \rangle} & \nu a_{\{1,2\}} \left((\langle 1, *, \overline{b}\langle a \rangle \rangle \| \langle 2, *, \overline{c}\langle a \rangle \rangle) \| a \right) = R' \end{array}$$

There are now two possibilities to reduce the rightmost prefix a of R': the first one assuming that event 1 is the reason why a is "known" in the context, and the other one making the complementary assumption, namely that event 2 is the culprit. This yields the following two derivable transitions from R':

$$\begin{array}{l} R' \xrightarrow{(3,*,1):a} \nu a_{\{1,2\}} \left((\langle 1, *, \overline{b}\langle a \rangle \rangle \| \langle 2, *, \overline{c}\langle a \rangle \rangle) \| \langle 3, 1, a \rangle \right) = T_1 \\ R' \xrightarrow{(3,*,2):a} \nu a_{\{1,2\}} \left((\langle 1, *, \overline{b}\langle a \rangle \rangle \| \langle 2, *, \overline{c}\langle a \rangle \rangle) \| \langle 3, 2, a \rangle \right) = T_2 \end{array}$$

[2]Conversely, OPEN- will decrease the number of identifiers in Γ in order to take into account the fact that there is one less extruder for a.

Notice here that T_1 (resp. T_2) may rollback event 2 (resp. event 1) while event 1 (resp. event 2) is backward blocked: indeed it is impossible to derive $T_1 \xrightarrow{1:\overline{b}\langle a \rangle^-}$ since the PAR- rule would require $1 \notin \langle 2, *, \overline{c}\langle a \rangle \rangle) \| \langle 3, 1, a \rangle$. In fact, we will see Section III that backtracking respects both contextual and structural causes.

In order to illustrate how synchronization is compositionally defined, let us consider the above derivations in a context where R is in parallel with $S = b(d).\overline{d}$. From S one may derive the following transition, that complements event 1:

$$S \xrightarrow{1:b(d)} \langle 1, *, b[\star/d] \rangle \rhd \overline{d}$$

Using the CLOSE+ rule, one may now compose both transitions identified by 1 (since $* =_* *$) and one gets:

$$\begin{array}{l} (R \| S) \xrightarrow{1:\tau} \nu a_\emptyset \left(\nu a_1 \left(\langle 1, *, \overline{b}\langle a \rangle \rangle \| \overline{c}\langle a \rangle \| a \right) \| \langle 1, *, b[a/d] \rangle \rhd \overline{d} \right) \\ \xrightarrow{2:\overline{c}\langle \nu a_1 \rangle} \\ \nu a_2 \left(\nu a_{\{1,2\}} \left(\langle 1, *, \overline{b}\langle a \rangle \rangle \| \langle 2, *, \overline{c}\langle a \rangle \rangle \| a \right) \| \langle 1, *, b[a/d] \rangle \rhd \overline{d} \right) \end{array}$$

using the PAR+ rule for the second transition. Now recall that there are two possible derivations from S in order to reduce the a prefix at the center of the above term. However only the first one can be composed with a transition on the $\overline{d}$ prefix on the right, since d is instantiated to a at event 1. Thus the only possibility[3] is to use the first derivation (with target T_1) in the COM+ rule composed with the derivation:

$$\langle 1, *, b[a/d] \rangle \rhd \overline{d} \xrightarrow{(3,1,*):\overline{a}} \langle 3, *, \overline{d} \rangle . \langle 1, *, b[a/d] \rangle$$

the side condition of the COM+ rule being satisfied.

Other rules: The rule PAR+ ensure freshness for the bound names and for the identifier i. In the PAR- rule, the side condition $i \notin S$ prevents a part of a synchronization to backtrack by itself. Rule MEM+- rewrites the process in a form in which it can trigger a transition. Importantly only the MEM- rule allows one to pop the $\langle \uparrow \rangle$ symbol out of a memory. This ensures that a child process cannot backtrack below its spawning point, without reuniting first with its sibling. Lastly, in rule NEW+ if $\Gamma = \emptyset$ the process cannot do a transition that has the bound name a in its subject. If $\Gamma \neq \emptyset$ then the side conditions forces the usage of rules OPEN+-, CAUSE REF+-.

Not all side conditions are necessary for the backward transitions, as most of them are in fact invariant of the history consistency of processes. For simplifying the presentation however we keep them in both directions.

III. PROPERTIES

After presenting some interesting properties of the LTS, that may shed light on some subtle point of its behavior, we show in Section III-B that the forward interpretation of an Rπ process is strongly bisimilar to its projection in the π-calculus. We then proceed, Section IV, with the proof that the backtracking of Rπ is sound with respect to a notion of trace equivalence. Before discussing and proving that this notion of equivalence is optimal in the Section V,

[3]The second derivation from R' is still applicable but can only be used for a synchronization occurring later in the context of the process.

A. Basic properties

First of all we observe that every transition can be undone.

Proposition 3.1 (Loop): For R reachable and for every forward transition $t : R \xrightarrow{\gamma} R'$ there exists a backward transition $t' : R' \xrightarrow{\gamma^-} R$, and conversely.

This is a trivial consequence of the symmetries of the rules.

An interesting property of proof systems, is whether each transition has a unique derivation. Given the complexity of our rules, in particular the choice of the contextual cause (rule CAUSE REF), it is not trivial that our system enjoys such property. Not only it does, but it does in a stronger sense for backward transitions.

Proposition 3.2: Two derivation trees have the same conclusion:

$$\frac{\pi_1}{R \xrightarrow{\zeta} S} \qquad \frac{\pi_2}{R \xrightarrow{\zeta} S}$$

iff $\pi_1 = \pi_2$.

Proposition 3.3: Suppose we have two negative transitions $R \xrightarrow{\zeta} S_1$ and $R \xrightarrow{\zeta} S_2$. Then $S_1 = S_2$.

The forward transitions do not have this property due to the nondeterminism in the choice of the synchronization partners. In the backward case, however, this form of nondeterminism disappears.

The following propositions emphasize some important properties of well-formed terms, concerning Γ-restrictions within processes. First we notice that in any $T = C[\nu a_\Gamma R]$, event identifiers in Γ correspond exactly to extruders of a that occur in the memory of R.

Proposition 3.4: For all $T = C[\nu a_\Gamma R]$ reachable, for some context $C[\bullet]$, $i \in \Gamma$ iff $m_1.\langle i, \_, \overline{f}\langle c\rangle\rangle.m_2 \in R$ such that $m_2[c] = a$, and $\langle i, \_, d[a/e]\rangle \notin R$.

Then we show that, in a reachable process, all νa_Γ's on the same name a are nested.

Proposition 3.5: Let $\Gamma, \Delta \neq \emptyset$. In $T = C[\nu a_\Gamma(R) \parallel S]$ reachable, $\nu a_\Delta \notin S$.

A liberated name a occurs in a process or in its memory if the process was within the scope of the original $\nu a_\emptyset$ or if a was received through a synchronization. This is formally stated in the following Proposition.

Proposition 3.6: In $T = C_1[C_2[\nu a_\Gamma R] \parallel S]$ reachable and $\Gamma \neq \emptyset$, if $a \in S$ then $\langle i, \_, d[a/c]\rangle \in S$, for some $i \in \Gamma$.

B. Correspondence with π-calculus

In Section II we have defined the reversible semantics of an Rπ process R as the LTS engendered by the union of $\mathcal{L}^+(R)$, the positive LTS of R, and $\mathcal{L}^-(R)$ its negative version. In order to claim that Rπ is a *reversible* π-calculus we need to prove that $\mathcal{L}^+(\varepsilon \rhd P)$, the positive interpretation of a π process P in $R\pi$, is bisimilar to the LTS of P.

We need to define a function that translates a Rπ process into a π calculus one, by:

- erasing the memories and νa_Γ, with $\Gamma \neq \emptyset$, operators from a process;
- applying the substitutions stored in the memories on the process;

Definition 3.7: Let ϕ be a function that translates $R\pi$ processes into π, by applying all the substitutions and erasing all $\nu_{\Gamma\neq\emptyset}$ from the process.

$$\begin{aligned}
&\phi(\varepsilon \rhd P) = \varepsilon \cdot P\\
&\phi(R \parallel S) = \phi(R) \mid \phi(S)\\
&\phi(\nu a_\emptyset R) = \nu a \phi(R)\\
&\phi(\nu a_{\Gamma\neq\emptyset} R) = \phi(R)\\
&\phi(\langle i, k, b[a/c]\rangle.m \rhd P) = \phi(m \rhd P\{a/c\})\\
&\phi(e.m \rhd P) = \phi(m \rhd P) otherwise
\end{aligned}$$

Proposition 3.8: (Strong bisimulation between forward Rπ and its π-image)

1) If $R \xrightarrow{\gamma} S$ then $\phi(R) \xrightarrow{\gamma} \phi(S)$.
2) If $T \xrightarrow{\gamma} U$ and $\forall R$ such that $\phi(R) = T$, $\exists S$ with $R \xrightarrow{\gamma} S$ and $\phi(S) = U$.

Propositions 3.8 can be extended to traces.

The operational correspondence is as follows:

Proposition 3.9: If $P \longrightarrow^\star Q$ and $\phi(R) = P$ then there exists S such that $R \longrightarrow^\star S$ and $\phi(S) = Q$.

Proof: Using the second part of Proposition **??** we have that for each transition of a π process there exists one for its Rπ correspondent. The result follows from induction on the length of the trace. ∎

Proposition 3.10: If $\varepsilon \rhd P \longrightarrow^\star R$ then $P \longrightarrow^\star \phi(R)$.

In practice, to better carry out the proofs, the translation is split into two parts: first removing the tagged restrictions and the memories, obtaining a π-calculs with explicit substitution. Then a second translation applies the substitutions. More details can be found in the appendix.

IV. Correctness of backtracking

In the introduction of this paper, we argued that we wanted our notion of reversibility to be as liberal as possible. As was already noted in RCCS [5] and subsequent work on reversible process algebra [7], [8], backtracking a trace should be allowed along any path that respects *causality*, or, said otherwise, backtracking can be done along any path that is *causally equivalent* to the path that was executed.

This property was formulated first in the work of the reversible CCS as a combination of a loop lemma, stating that any forward trace can be undone step by step, and a fundamental property that ensures that any two coinitial and cofinal trace are necessarily causally equivalent (see Fig. 1).

We already know that the loop property holds trivially for Rπ (Proposition 3.1). It remains to check that Rπ traces do exhibit the fundamental property, which depends on the equivalence on traces that is induced by the semantics of the language (denoted by $\sim$ in Fig. 1). For instance, the less

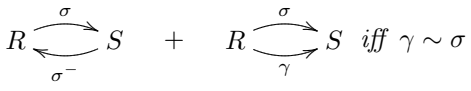


Figure 1. The conjunction of a loop property (left) and the fundamental property (right) insures that after the forward trace σ, one may rollback to R along a causally equivalent past γ^-.

liberal backtracking policy is obtained when the fundamental property holds only for equal traces.

This section follows closely the argument made in RCCS, in the context of Rπ. We will see, in Lemma 4.3, that Rπ transitions contain enough information to characterize syntactically the concurrency and causality relations[4]. This will let us characterize a notion of equivalence up-to permutation on traces in Definition 4.4 and proof the fundamental property for Rπ in Theorem 4.5. Later, in Section V, we will also aruge that our notion of equivalence is, in a sense, optimal for reversing the π-calculus.

As usual, the causal equivalence class of a path is constructed by permuting the transitions that are concurrent. We proceed by defining the concurrency relation between transitions as the complement of causality.

Definition 4.1 (Traces): Two transitions, t and t' are *composable* (denoted by $t;t'$,) if the target process of t is the source process for t'. A *trace*, denoted by $R \xrightarrow{w} S$ is a composition of transitions, with $w = (\zeta_i)^\star$. The empty trace is denoted by ϵ. Two traces are *coinitial* if they start from the same process and *cofinal* if they end in the same process.

Definition 4.2 (Causality and concurrency): Let $t_1 : R \xrightarrow{(i_1,j_1,k_1):\gamma_1} S$ and $t_2 : S \xrightarrow{(i_2,j_2,k_2):\gamma_2} T$ be two transitions, where $t_1 \neq opp(t_2)$. We say that t_1 is:

- a *structural cause* of t_2, written $t_1 \sqsubset t_2$, if $i_1 \sqsubset_T i_2$ or $i_2 \sqsubset_R i_1$
- a *contextual cause* of t_2, written $t_1 \prec t_2$, if $i_1 \prec_T i_2$ or $i_2 \prec_R i_1$.

We simply say that t_1 *causes* t_2, written $t_1 < t_2$, if either $t_1 \sqsubset t_2$ or $t_1 \prec t_2$. Otherwise we say that they are *concurrent*.

Example 4.1: Consider the following trace (with the conventions of Example 2.2):

$$\nu a_\emptyset(\overline{b}\langle a\rangle.a) \xrightarrow{1:\overline{b}\langle \nu a_\emptyset\rangle} \xrightarrow{(2,*,1):a} \nu a_1(\langle 2,1,a\rangle.\langle 1,*,\overline{b}\langle a\rangle\rangle)$$

where the first transition is both a structural and a contextual cause of the second; and consider the trace:

$$\nu a_\emptyset(\overline{b}\langle a\rangle \| \overline{c}\langle a\rangle) \xrightarrow{1:\overline{b}\langle \nu a_\emptyset\rangle} \xrightarrow{2:\overline{c}\langle \nu a_1\rangle} \nu a_{\{1,2\}}(\langle 1,*,\overline{b}\langle a\rangle\rangle \| \langle 2,*,\overline{c}\langle a\rangle\rangle)$$

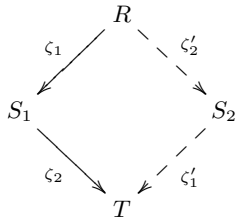
where the two transitions are this time concurrent.

We need now showing that the above syntactic definition of concurrency indeed coincides with commutability of transitions. We shall see that a particularity of Rπ is that commutation of concurrent transitions may not always be label preserving. However, commutation will be label preserving *up-to* label equivalence $=_\lambda$, defined as the least equivalence relation on labels satisfying:

$$(i,j,k) : \overline{b}(\nu a_\Gamma) =_\lambda (i,j,k) : \overline{b}(\nu a_\Delta)$$

for some $\Gamma, \Delta \subset \mathcal{I}$.

Lemma 4.3 (Square): Consider two consecutive transitions $t_1 : R \xrightarrow{\zeta_1} S_1$ and $t_2 : S_1 \xrightarrow{\zeta_2} T$. If t_1 and t_2 are concurrent, there exist $t_2' : R \xrightarrow{\zeta_2'} S_2$ and $t_1' : S_2 \xrightarrow{\zeta_1'} T$, with $\zeta_i =_\lambda \zeta_i'$, such that the following diagram commutes:



Following the standard notation, we say that t_2 is the *residual* of t_2' after t_1 and write $t_2 = t_2'/t_1$.

Example 4.2: Back to Example 4.1, swapping the two concurrent transitions one obtains:

$$\nu a_\emptyset(\overline{b}\langle a\rangle \| \overline{c}\langle a\rangle) \xrightarrow{2:\overline{c}\langle \nu a_\emptyset\rangle} \xrightarrow{1:\overline{b}\langle \nu a_1\rangle} \nu a_{\{1,2\}}(\langle 1,*,\overline{b}\langle a\rangle\rangle \| \langle 2,*,\overline{c}\langle a\rangle\rangle)$$

Definition 4.4 (Equivalence up-to permutation): Let $\sim$ be the least equivalence relation on traces satisfying:

$$t_1;(t_2/t_1) \sim t_2;(t_1/t_2) \qquad t;opp(t) \sim \epsilon \qquad opp(t);t \sim \epsilon$$

We can now state the fundamental property which proves that backtracking respects the causality induce by Rπ.

Theorem 4.5 (Fundamental property): Two traces are coinitial and cofinal if and only if they are equivalent.

The reader aware of the work on non interleaving semantics for the π-calculus may have noticed that our semantics allows more transitions to commute than the standard ones [9], [10]. This is enabled by the fact that we let commutation preserve label up-to $=_\lambda$, which amounts to saying that the *same* event may have two (slightly) different labels. We will come back formally to this important point in the Section V. In the meantime, we conclude this Section by a sanity check ensuring that the positive LTS of Rπ coincides with the late semantics of the π-calculus.

V. Causality

In the following we consider only the forward computations of our reversible semantics, and use them to define a causal semantics for the π calculus.

As we have remarked in the introduction, we believed that in a closed system (ie where only reductions are observed) the only notion of causality is the structural one, the one

[4]In other terms, Rπ can be viewed as a Levy labeling [**?**] over the π-calculus, as RCCS is a Levy labeling over CCS.

induced by the precedence operator. In fact, for reductions, contextual causality is hidden behind structural causality:

Proposition 5.1: Let $t_1 : R \xrightarrow{(i_1,*,*):\tau} S$ and $t_2 : S \xrightarrow{(i_2,*,*):\tau} T$. If t_1 and t_2 are causal then $t_1 \sqsubset_T t_2$.

We want to justify contextual causality between labelled events as an "anticipation" of the structural causality between the reductions these events will generate. Or, dually, that if two labelled events are concurrent, then it is possible from them to generate two concurrent reductions. In order to formalize this intuition, given a process and one computation trace, we need a notion of *reduction context*, that provides a synchronizing partner for every non-τ transition in the trace (see Proposition 5.4).

Then the main result of this Section (Theorem 5.5) is that two non-τ transitions are concurrent if and only if there exists a reduction context that preserves concurrency.

We introduce the projections that are then used to retrieve from a synchronization its two composing transitions.

Proposition 5.2: If $t : R \xrightarrow{(i,*,*):\tau} S$ then there exists at most one context $C[\bullet]$ such that $t : C[R_1 \| R_2] \xrightarrow{(i,*,*):\tau} C'[S_1 \| S_2]$ with $R_q \neq S_q, q \in \{1,2\}$.

Definition 5.3: The projections to the left and to the right of a transition t are defined as follows:

- if $t : C_1[R_1 \| R_2] \xrightarrow{(i,*,*):\tau} C_2[S_1 \| S_2]$ with $R_q \neq S_q, q \in \{1,2\}$ then

  $$\pi_l(t) : R_1 \xrightarrow{(i,j,k):\alpha} S_1 \qquad \pi_r(t) : R_2 \xrightarrow{(i,j',k'):\overline{\alpha}} S_2,$$

  where $\langle i,k,\alpha\rangle \in S_1$ and $j \rightsquigarrow_{R_1} i$ (similar for j',k').
- otherwise, $\pi_l(t) = \pi_r(t) = t$.

Proposition 5.4 (Reduction contexts): Given a trace

$$R_0 \xrightarrow{(i_1,j_1,k_1):\alpha_1} R_1 ... \xrightarrow{(i_n,j_n,k_n):\alpha_n} R_n$$

$\exists C[\bullet]$ such that $C[R_0] = R'_0 \xrightarrow{(i_1,*,*):\tau} R'_1 ... \xrightarrow{(i_n,*,*):\tau} R'_n$ for some $R'_1, .., R'_n$ and, for $q \in \{0, .., n\}$:

- if $\alpha_q \neq \tau$ then $\exists x, \pi_x(R'_q \xrightarrow{(i_q,*,*):\tau} R'_{q+1}) = R_q \xrightarrow{(i_q,j_q,k_q):\alpha_q} R_{q+1}$
- if $\alpha_q = \tau$ then $\exists C'[\bullet]$ such that $R'_q = C'[R_q], R'_{q+1} = C'[R_{q+1}]$.

Example 5.1: Let us consider the process $R = \nu a_\emptyset(\varepsilon \rhd \overline{b}\langle a\rangle \| \varepsilon \rhd a)$ with the trace

$$R \xrightarrow{(i_1,*,*):\overline{b}\langle \nu a_\emptyset\rangle} \xrightarrow{(i_2,*,i_1):a} S$$

A reduction context for R is $C[\bullet] = [\bullet] \| \varepsilon \rhd b(u).\overline{u}$. The trace becomes

$$C[R] \xrightarrow{(i_1,*,*):\tau} \xrightarrow{(i_2,*,*):\tau} C'[S]$$

with $i_1 \prec_S i_2$ and $i_1 <_{C'[S]} i_2$.

In the Theorem below we build a reduction context that preserves concurrency. The reverse says that all reduction contexts preserve causality.

Theorem 5.5: Let t_1 and t_2 be two transitions such that

$$t_1 : R \xrightarrow{(i_1,j_1,k_1):\alpha_1} S \qquad t_2 : S \xrightarrow{(i_2,j_2,k_2):\alpha_2} T.$$

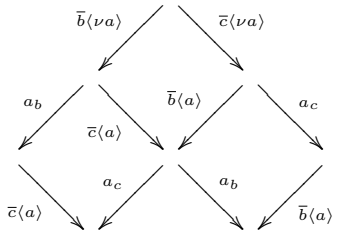
t_1 and t_2 are concurrent $\iff$ $\exists$ a reduction context $C[\bullet]$ such that

$$t'_1 : C[R] \xrightarrow{(i_1,*,*):\tau} S' \qquad t'_2 : S' \xrightarrow{(i_2,*,*):\tau} T'$$

with t'_1 and t'_2 concurrent.

Example 5.2: Let us consider the π calculus process $P = \nu a(\overline{b}\langle a\rangle \mid \overline{c}\langle a\rangle \mid a)$ with the trace $P \xrightarrow{\overline{b}\langle \nu a\rangle} \xrightarrow{\overline{c}\langle \nu a\rangle} \xrightarrow{a} Q$. A reduction context for P is $C[\bullet] = [\bullet] | b(u).\overline{u} | c(v).\overline{v}$ and we have $C[P] \xrightarrow{\tau_b} \xrightarrow{\tau_c} \xrightarrow{\tau_a} Q'$. Remark that the transitions τ_b and τ_c are concurrent and that we can interchange them in the trace.

The last synchronization on channel a corresponds to two different events: one engendered by the substitution on u and another by the substitution on v. In Rπ, the corresponding events choose as cause the transition on b and on c respectively. We can represent all the commutative transitions for P as follows:



Depending on the choice of cause we can reach two distinct processes, that allow then different backward paths.

[To Be Continued]

VI. Conclusion

The semantics of π calculus we defined guarantees that events have each a unique causal history. We believe that we can then find a correspondance between Rπ and events structure and thus introduce reversibility in event structures.

Another interesting continuation of our work consists in presenting the causality relation of π calculus using the abstract interpretation tools. The reduction context and closed trace of section V can be seen as concretizations for the open process and trace, respectively. The concretization is then a under-approximation of the concurrency relation between transitions.

Lastly, we are also interested in defining a meaningfull equivalence relation for reversible processes. The weak bisimulation, usually employed in π calculus, is no longer useful in a reversible context. As noted by [8], a reversible process is weakly bisimilar to some of its derivatives. However, we believe that our compositional semantics can be used as a starting point in defining an interesting bisimulation between reversible processes.

ACKNOWLEDGMENTS

We thank Ilaria Castellani.

REFERENCES

[1] R. Landauer, "Irreversibility and heat generation in the computing process," *IBM Journal of Research and Development*, vol. 5, pp. 183–191, 1961.

[2] L. Bougé, "On the existence of symmetric algorithms to find leaders in networks of communicating sequential processes," *Acta Inf.*, vol. 25, no. 2, pp. 179–201, 1988.

[3] R. D. Nicola, U. Montanari, and F. Vaandrager, "Back and forth bisimulations," in *Proceedings of CONCUR'90*, ser. LNCS, vol. 458, 1990, pp. 152–165.

[4] V. Danos and J. Krivine, "Transactions in RCCS," in *In Proc. of CONCUR, LNCS 3653*. Springer, 2005, pp. 398–412.

[5] ——, "Reversible communicating systems," in *Proceedings of 15th CONCUR*, ser. Lecture Notes in Computer Science, vol. 3170. Springer, 2004, pp. 292–307.

[6] I. Phillips and I. Ulidowski, "Reversing algebraic process calculi," in *Proceedings of FOSSAC'06*, ser. LNCS, vol. 3921, 2006, pp. 246–260.

[7] ——, "Reversibility and models for concurrency," *Electr. Notes Theor. Comput. Sci.*, vol. 192, no. 1, pp. 93–108, 2007, proceedings of SOS 2007.

[8] I. Lanese, C. A. Mezzina, and J.-B. Stefani, "Reversing higher-order pi," in *Proceedings of 21st CONCUR*, ser. Lecture Notes in Computer Science, vol. 6269. Springer, 2010, pp. 478–493.

[9] M. Boreale and D. Sangiorgi, "A fully abstract semantics for causality in the π-calculus," *Acta Inf.*, vol. 35, no. 5, pp. 353–400, 1998.

[10] P. Degano and C. Priami, "Non-interleaving semantics for mobile processes." *Theor. Comp. Sci.*, vol. 216, no. 1-2, pp. 237–270, 1999.

[11] S. Crafa, D. Varacca, and N. Yoshida, "Event structure semantics of the parallel extrusion in the pi -calculus," in *Proceedings of 23rd CONCUR*, ser. Lecture Notes in Computer Science. Springer, 2012, accepted.

[12] D. Hirschkoff, "Handling substitutions explicitely in the pi-calculus."

[13] P. Gardner and L. Wischik, "Explicit fusions," in *Proceedings of the 25th International Symposium on Mathematical Foundations of Computer Science*, ser. MFCS '00. London, UK: Springer-Verlag, 2000, pp. 373–382.

[14] G. L. Ferrari, U. Montanari, and P. Quaglia, "A pi-calculus with explicit substitutions: the late semantics," in *Proceedings of the 19th International Symposium on Mathematical Foundations of Computer Science 1994*, ser. MFCS '94. London, UK: Springer-Verlag, 1994, pp. 342–351.

VII. ANNEXES

A. Graphical representation of the information flow

In the following we show a graphical representation of a Rπ process in which we use annotated boxes to represent the scope of a restriction. For simplicity, we consider that the process has one liberated name. Extrusions are represented as arrows, which are decorated with the identifier of the transition. The transitions that are using the name in subject are represented as fragments on which we write their contextual cause in parathensis. A first example is given in Figure 2.

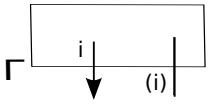


Figure 2. A simple example

In the following we ignore the empty memories and processes. We consider the following process, as an example $R = \nu a_\emptyset(\overline{b}\langle a\rangle \mid \overline{c}\langle a\rangle \| a) \| b(e).(\overline{d}\langle e\rangle \| \overline{e})$ and we draw its transitions.

Let $R \xrightarrow{1:\tau_b} R' = \nu a_\emptyset(\nu a_1(S) \| \langle 1, *, b[a/e]\rangle \rhd (\overline{d}\langle e\rangle \| \overline{e}))$, with $S = (\langle 1, *, \overline{b}\langle a\rangle\rangle \rhd 0 \| \overline{c}\langle a\rangle \| a)$, represented by the Figure 3. We have two boxes, corresponding to the two

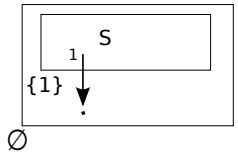


Figure 3. The synchronization on b

restrictions on a. The synchronization on b is represented by the dot.

The process then does two extrusion of a: on channel c and on on channel d (due to the substitution $[a/e]$)

$$R' \xrightarrow{2:\tau_c} \xrightarrow{3:\tau_d} R'' = \nu a_{2,3}(\nu a_{1,2}(a) \| \langle 1, *, b[a/e]\rangle \rhd \overline{e})$$

shown in Figure 4, where event 3 has as instantiator event 1, represented by the dotted line.

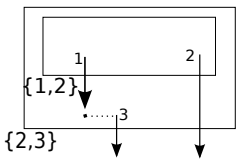


Figure 4. The extrusions on c and d

Let us now do the synchronization on channel a: $R'' \xrightarrow{4:\tau_a} T$. We have the Figure 5. We can see that if a fragment exits the box it then has to choose a cause, that is a path back

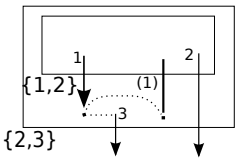


Figure 5. The synchronization on a

into the box. Paths are then build out of synchronizations and instantiations of names, the two representing the information flow.

Consider now that we do the transition on channel e, which after the substitution becomes a. We have that $R'' \xrightarrow{(4,1,3):a} \nu a_{2,3}(\nu a_{1,2}(a))$, shown in Figure 6. We can

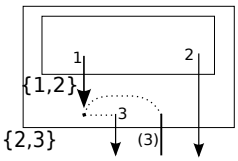


Figure 6. The transition on e

see that the instantiator of event 4 is the event 1. In order for the transition to exit the scope of the restriction $\nu a_{2,3}$ it has to choose a cause. In other words, event 4 is a synchronization occuring somewhere outside of the box where the synchronizing partner of e is a name instantiated by event 3.

We conclude with the transition: $R'' \xrightarrow{(4,*,3):a} \nu a_{2,3}(\nu a_{1,2}(0) \| \langle 1, *, b[a/e] \rangle \rhd \overline{e})$ where channel a is used in a synchronization outside the box. Initially the transitions picks as cause event 1, in order to exit the smaller box. It is then allowed to change its cause from 1 to 3, since event 3 has 1 as its instantiator. The transition is represented in Figure 7.

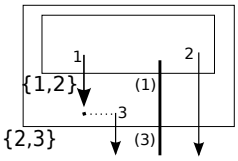


Figure 7. The transition on a

B. Proofs of section III

Proposition 7.1: Let $R \xrightarrow{(i,j,k):\alpha} S$. $\forall m \in R$

- either $m \in S$
- or $\exists e, e.m \in S$ with $id(e) = i$.

Proposition 7.2: Let $R \xrightarrow{(i,j,k):\alpha^-} S$. $\forall m \in S$

- either $m \in R$
- or $\exists e, e.m \in S$ with $id(e) = i$.

Proposition 7.3: In $R \xrightarrow{(i,j,k):\overline{b}\langle \nu a_\Gamma \rangle} S$, $\exists C$ such that $R = C[\nu a_\Gamma R']$. Similarly, in $R \xrightarrow{(i,j,k):\overline{b}\langle \nu a_\Gamma \rangle^-} S$, $\exists C$ such that $S = C[\nu a_\Gamma R']$.

Proposition 7.4: If $R = C[\nu a_{\Gamma'} R']$, then a transition that has a as the object of an output is necessarily of the form $R \xrightarrow{(i,j,k):\overline{b}\langle \nu a_\Gamma \rangle} S$.

Proposition 7.5: Let $R \xrightarrow{(i,j,k):\gamma} S$ and $n_s(\alpha) = a$. $R = C[\nu a_\Gamma R'] \iff k \neq *$.

Lemma 3.4 For all $T = C[\nu a_\Gamma R]$ reachable, for some context $C[]$, $i \in \Gamma$ iff $m_1.\langle i, \_, \overline{f}\langle c \rangle \rangle.m_2 \in R$ such that $m_2(c) = a$, and $\langle i, \_, d[a/e] \rangle \notin R$.

Proof: Suppose that $i \in \Gamma$. We reason by induction on the trace $(\varepsilon \rhd P) \longrightarrow^\star T$.

- the base case: in $\varepsilon \rhd P$ the property is true.
- the inductive case: consider the trace

$$\varepsilon \rhd P \longrightarrow ... \longrightarrow T' \longrightarrow T$$

where the property is true for T'. We have to show that it is preserved for $\forall a, \Gamma, R$ such that $T = C[\nu a_\Gamma R]$. We reason by induction on the derivation tree of $T' \longrightarrow T$:

– rule PAR+:

$$\frac{R_0 \xrightarrow{(i,j,k):\alpha} R_1}{R_0 \parallel S_0 \xrightarrow{(i,j,k):\alpha} R_1 \parallel S_0}$$

The property is true for $R_0 \parallel S_0$ hence it is also true for R_0, S_0 and by induction for R_1. Therefore it is true for $R_1 \parallel S_0$.

– rule OPEN+:

$$\frac{R_0 \xrightarrow{(i,j,k):\alpha} R_1 \qquad \alpha = \overline{b}\langle a \rangle \vee \alpha = \overline{b}\langle \nu a_{\Gamma'} \rangle}{\nu a_\Gamma R_0 \xrightarrow{(i,j,k):\overline{b}\langle \nu a_\Gamma \rangle} \nu a_{\Gamma + i} R_1}$$

The property is true in $\nu a_\Gamma R_0$. Then it is true also in R_0: if $R_0 = C[\nu a_{\Gamma'} R_0']$ then $\nu a_\Gamma R_0 = C'[\nu a_{\Gamma'} R_0']$.
By induction, we have that R_1 respects the property. It remains to show that the property is preserved for $\nu a_{\Gamma+i} R_1$ (with $C = []$). For all $i' \in \Gamma$ the property is satisfied by induction. Due to transition on α and by proposition 7.1 we have that $\langle i, \_, \overline{f}\langle c \rangle \rangle.m \in R_1$, $m(f) = b, m(c) = a$ which satisfies the property for i.

– rule CLOSE+:

$$\frac{R_0 \xrightarrow{(i,j,k):\overline{b}\langle \nu a_\Gamma \rangle} R_1 \qquad S_0 \xrightarrow{(i,j',k'):b(u)} S_1}{R_0 \parallel S_0 \xrightarrow{(i,\star,\star):\tau} \nu a_\Gamma (R_1 \parallel S_{1[a/u]@i})}$$

The property is true for $R_0 \parallel S_0$, hence for R_0 and S_0 as well. By induction it is true for R_1 and S_1. Moreover, because of the transition on $\overline{b}\langle \nu a_\Gamma \rangle$

and from proposition 7.3, we have that $\exists C'$ such that $R_0 = C'[\nu a_\Gamma R_0']$. Hence $\forall i \in \Gamma$ there is a corresponding extruder in the memory of R_0. The transition is forward and therefore none of these extruders are removed from the memory.
We can the conclude that in $\nu a_\Gamma(R_1 \parallel S_1)$ the property is true.

– rule OPEN-:

$$\frac{R_0 \xrightarrow{(i,j,k):\alpha^-} R_1 \qquad \alpha = \overline{b}\langle a\rangle \vee \alpha = \overline{b}\langle \nu a_{\Gamma'}\rangle}{\nu a_{\Gamma+i} R_0 \xrightarrow{(i,j,k):\overline{b}\langle \nu a_\Gamma\rangle^-} \nu a_\Gamma R_1}$$

By induction, the property is true for $\nu a_{\Gamma+i}R_0$. The transition removes the memory entry corresponding to i (according to proposition 7.2), hence the property is true for $\nu a_\Gamma R_1$.

Similar for the rest of the rules.

We still have to show that, in $T = C[\nu a_\Gamma R]$, if $m_1.\langle i, \_, \overline{f}\langle c\rangle\rangle.m_2 \in R$ such that $m_2(c) = a$, and $\langle i, \_, d[a/e]\rangle \notin R$ then $i \in \Gamma$. As above, we reason by induction on the trace $(\varepsilon \rhd P) \longrightarrow^\star T$. For $\varepsilon \rhd P$, there is nothing to prove since the memory is empty. For the inductive case, we have the trace: $\varepsilon \rhd P \longrightarrow ... \longrightarrow T' \longrightarrow T$ and we consider the transition $T' \longrightarrow T = C[\nu a_\Gamma R]$, on which we reason by induction.

- rule OPEN+:

$$\frac{R_0 \xrightarrow{(i,j,k):\alpha} R_1 \qquad \alpha = \overline{b}\langle a\rangle \vee \alpha = \overline{b}\langle \nu a_{\Gamma'}\rangle}{\nu a_\Gamma R_0 \xrightarrow{(i,j,k):\overline{b}\langle \nu a_\Gamma\rangle} \nu a_{\Gamma+i} R_1}$$

which adds $\langle i, \_, \overline{f}\langle c\rangle\rangle$(from proposition 7.1) into the memory and we have that $i \in \Gamma + i$.

- rule OPEN-:

$$\frac{R_0 \xrightarrow{(i,j,k):\alpha^-} R_1 \qquad \alpha = \overline{b}\langle a\rangle \vee \alpha = \overline{b}\langle \nu a_{\Gamma'}\rangle}{\nu a_{\Gamma+i} R_0 \xrightarrow{(i,j,k):\overline{b}\langle \nu a_\Gamma\rangle^-} \nu a_\Gamma R_1}$$

We have to ensure that for all extruders in the memory of R_1, there is a corresponding identifier in Γ. It follows from proposition 7.2 that the backward transition on α removes the memory entry corresponding to i.

- rule CLOSE+ is not possible because of the constraint $\langle i, \_, d[a/e]\rangle \notin R$.

For the rest of the rules, the result follows from induction. ∎

Lemma 3.5 Let $\Gamma, \Gamma' \neq \emptyset$. In $T = C[\nu a_\Gamma R \mid S]$ reachable, $\nu a_{\Gamma'} \notin S$.

Proof: We reason by induction on the trace that leads to T: $\varepsilon \rhd P \longrightarrow ... \longrightarrow T' \longrightarrow T$. The base case is trivial since $\forall \nu a_\Gamma \in P$, $\Gamma = \emptyset$. For the inductive case, we have that, in $T' \longrightarrow T$, the property is true for T'. We have to show that it is preserved in T. We reason by induction on the derivation tree of the transition:

- rule PAR+:

$$\frac{R_0 \xrightarrow{\gamma} R_1}{R_0 \parallel S_0 \xrightarrow{\gamma} R_1 \parallel S_0}$$

where the property is satisfied in $R_0 \parallel S_0$. We distinguish the following cases:

– $\nu a_\Gamma \notin R_0, S_0$ we have nothing to prove;
– $\nu a_\Gamma \in R_0$, then by induction $\nu a_{\Gamma'} \notin S_0$. The property is preserved in R_1, hence it is true also in $R_1 \parallel S_0$.
– $\nu a_\Gamma \in S_0$ and $\nu a_{\Gamma'} \notin R_0$. Remember that $\Gamma, \Gamma' \neq \emptyset$. For the property to be satisfied in $R_1 \parallel S_0$, we have to show that $\nu a_{\Gamma'} \notin R_1$.
Suppose that $\nu a_{\Gamma'} \in R_1$. Then necessarily $\nu a_\emptyset \in R_0$ and the transition is on $\gamma = \overline{b}\langle \nu a_\emptyset\rangle$. But the side condition of the rule PAR+: $\mathrm{bn}(\gamma) \cap \mathrm{fn}(S_0) = \emptyset$ is then not satisfied, and we reach a contradiction.

- rule COM+:

$$\frac{R_0 \xrightarrow{(i,j,k):\overline{b}\langle d\rangle} R_1 \qquad S_0 \xrightarrow{(i,j',k'):b(c)} S_1}{R_0 \parallel S_0 \xrightarrow{(i,\star,\star):\tau} R_1 \parallel S_{1[d/c]@i}}$$

As above, we distinguish on the process which contains νa_Γ:

– $\nu a_\Gamma \notin R_0, S_0$ we have nothing to prove;
– $\nu a_\Gamma \in R_0$, hence $\nu a_{\Gamma'} \notin S_0$, with $\Gamma, \Gamma' \neq \emptyset$. If $d = a$ then the transition on R_0 is not correct (from proposition 7.3), hence the rule cannot be applied. Therefore $d \neq a$. But then $\nu a_{\Gamma'} \notin S_1$ either since the transition does not have a as object. Therefore the process $T = R_1 \parallel S_1$ satisfies the property.
– $\nu a_\Gamma \in S_0$, similar to above.

Straightforward for the rest of the rules. ∎

Lemma 3.6 In $T = C_1[C_2[\nu a_\Gamma R] \parallel S]$ reachable and $\Gamma \neq \emptyset$, if $a \in S$ then $\langle i, \_, d[a/c]\rangle \in S$, $i \in \Gamma$.

Proof: We reason by induction on the trace $(\varepsilon \rhd P) \longrightarrow^\star T$. For $\varepsilon \rhd P$, the property follows from the fact that $\forall \nu a_\Gamma \in P$, $\Gamma = \emptyset$. For the inductive case, we have the trace: $\varepsilon \rhd P \longrightarrow ... \longrightarrow T' \longrightarrow T$ and we consider the transition $T' \longrightarrow T = C_1[C_2[\nu a_\Gamma R] \parallel S]$, on which we reason by induction. The only interesting case corresponds to rule CLOSE+:

$$\frac{R_0 \xrightarrow{(i,j,k):\overline{b}\langle \nu a_\Gamma\rangle} R_1 \qquad S_0 \xrightarrow{(i,j',k'):b(c)} S_1 \quad a \notin S_0}{R_0 \parallel S_0 \xrightarrow{(i,\star,\star):\tau} \nu a_\Gamma(R_1 \parallel S_{1[a/c]@i})}$$

The transition adds the entry $\langle i, \_, d[a/c]\rangle$ to the memory of S_1 with $m(d) = b$. Using proposition 7.3 we have that $\nu a_{\Gamma'} \in R_1$. Hence the property holds for $\nu a_\Gamma(R_1 \parallel S_1)$.

For the rest of the cases the result follows from induction. ∎

Lemma 3.2 Two derivation trees have the same conclusion:

$$\frac{\pi_1}{R \xrightarrow{\zeta} S} \qquad \frac{\pi_2}{R \xrightarrow{\zeta} S}$$

iff $\pi_1 = \pi_2$.

Proof: If $\pi_1 = \pi_2$ then it is trivial to show that both derivations reach the same conclusion. For the other direction, we proceed by induction on the derivation tree of a transition $R \xrightarrow{\zeta} S$. We show that for each rule, there is only one premise possible.

- rule OPEN+:

$$\frac{\pi_1}{\nu a_\Gamma R \xrightarrow{(i,j,k):\overline{b}\langle \nu a_\Gamma\rangle} S}$$

For the transition on $\overline{b}(\nu a_\Gamma)$ from $\nu a_\Gamma R$ only rule OPEN+ applies and we have that $S = \nu a_{\Gamma+i} S'$. Then the rules become:

$$\frac{R \xrightarrow{(i,j,k):\alpha_1} S' \qquad \alpha_1 = \overline{b}\langle a\rangle \vee \alpha_1 = \overline{b}\langle \nu a_{\Gamma'}\rangle}{\nu a_\Gamma R \xrightarrow{(i,j,k):\overline{b}\langle \nu a_\Gamma\rangle} \nu a_{\Gamma+i} S'}$$

It remains to show that $\alpha_1 = \alpha_2$. Let us suppose that $\alpha_1 = \overline{b}\langle a\rangle$. Then from proposition 7.4 $\nexists C[]$ such that $R = C[\nu a_{\Gamma'} R']$. By proposition 7.3 it follows that $\alpha_2 = \overline{b}\langle a\rangle$. Similarly, if $\alpha_1 = \overline{b}\langle \nu a_{\Gamma'}\rangle$. Then $\alpha_1 = \alpha_2$ and the two premises coincide.

- rule CAUSE REF+:

$$\frac{\pi_1}{\nu a_\Gamma R \xrightarrow{(i,j,k):\alpha} S} \qquad a \in subj(\alpha)$$

Notice that indeed for such a transition only rule CAUSE REF+ applies. Then $S = \nu a_\Gamma S'_{[k/k_1]@i} = \nu a_\Gamma S'_{[k/k_2]@i}$ with

$$\frac{R \xrightarrow{(i,j,k_1):\alpha} S'}{\nu a_\Gamma R \xrightarrow{(i,j,k):\alpha} \nu a_\Gamma S'_{[k/k_i]@i}}$$

$$a \in subj(\alpha), k \in \Gamma, k_i = k \vee k_i \rightsquigarrow_R k$$

with $i \in \{1,2\}$. We have to show that $k_1 = k_2$.
We have that $k \in \Gamma$ and, by Lemma 3.4, that $\exists m_1.\langle k, \_, \overline{f}\langle c\rangle\rangle.m_2 \in R$ with $m_2(c) = a$. We distinguish the cases:

- if $\exists \nu a_{\Gamma'} \in R$ we have, by Lemma 3.5, that $k \in \Gamma'$. Hence $k_1 = k_2 = k$.
- otherwise we have the cases:
  - $c = a$ then the constraints $k_1 \rightsquigarrow_R k$ and $k_2 \rightsquigarrow_R k$ are met only if $k_1 = k_2 = \star$ which follows from Proposition 7.5.
  - $c \neq a$ then $\exists \langle k', \_, d[a/e]\rangle \in m_2$. Then $k' \rightsquigarrow_R k$. Such a k' is unique since there is only one substitution per name. Hence $k_1 = k_2 = k'$.

- rules COM+, CLOSE+ are similar:

$$\frac{\pi_1}{R \parallel S \xrightarrow{(i,\star,\star):\tau} T}$$

If $\exists \nu a_{\Gamma'} \in R$, by proposition 7.4 the output has as object νa_Γ and only rule CLOSE+ applies. If $\nexists \nu a_{\Gamma'} \in R$, then it follows that only rule COM+ applies. Let us study the latter. We have that $T = R' \parallel S'_{[a/c]@i}$ in

$$\frac{R \xrightarrow{(i,j,k_i):\overline{b}\langle a\rangle} R' \qquad S \xrightarrow{(i,j',k'_i):b(c)} S'}{R \parallel S \xrightarrow{(i,\star,\star):\tau} R' \parallel S'_{[a/c]@i}}$$

$$k_i =_* j', k'_i =_* j$$

It remains to show that $k_1 = k_2$. We have the cases:

- If b is a free name then, by proposition 7.5, $k_1 = k'_1 = \star$ and $k_2 = k'_2 = \star$.
- If b is a liberated name then, from Lemma 3.5 we deduce that either $k_1 = \star$ or $k'_1 = \star$ but not both. Hence one of the two conditions, $k_1 =_* j'$ and $k'_1 =_* j$, is trivially true (and similar for k_2). Let us suppose that $k'_1 = \star$. Then $k'_2 = \star$ as well and $k_1, k_2 \neq \star$. In order to satisfy the side condition, then necessarily $k_1 = k_2 = j'$.

Similar for the rest of the rules.

■

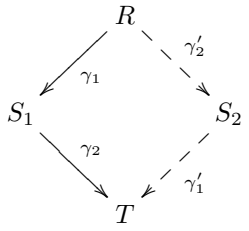
Lemma 3.1 For R reachable and for every forward transition $t : R \xrightarrow{\gamma} R'$ there exists a backward transition $t' : R' \xrightarrow{\gamma^-} R$, and conversely.

Proof: We reason by induction on derivation tree of the transition $t : R \xrightarrow{\gamma} R'$ in one direction, and on $t : R' \xrightarrow{\gamma} R$ for the converse. Trivial since the rules are symmetric for the forward and backward direction. ■

C. Proofs of section V

Proposition 7.6: In $t_1 : R \xrightarrow{(i_1,j_1,k_1):\gamma_1} S$ and $t_2 : S \xrightarrow{(i_2,j_2,k_2):\gamma_2} T$, if $\{i_1, i_2\} \cap \{k_1, k_2\} \neq \emptyset$ then $i_1 \prec_T i_2 \vee i_2 \prec_R i_1$.

Lemma 4.3 Let $t_1 : R \xrightarrow{(i_1,j_1,k_1):\gamma_1} S_1$ and $t_2 : S_1 \xrightarrow{(i_2,j_2,k_2):\gamma_2} T$ be two transitions that are not causal. Then there exists $t'_2 : R \xrightarrow{(i_2,j_2,k_2):\gamma'_2} S_2$ and $t'_1 : S_2 \xrightarrow{(i_1,j_1,k_1):\gamma'_1} T$, where $\gamma_i =_\lambda \gamma'_i$:



Proof: The grammar of a Rπ process is $T := \nu\tilde{a}_\Gamma(S \parallel T)$. Then we can rewrite R as

$$R = \nu\tilde{a_0}_{\Gamma_0}(T_0 \parallel \nu\tilde{a_1}_{\Gamma_1}(T_1 \parallel ...\nu\tilde{a_k}_{\Gamma_k}(R_1 \parallel R_2)))$$

such that t_1 involves at least a thread in R_1 and t_2 at least one in R_2. We reason by cases on whether t_1 and t_2 are synchronizations.

1) Suppose that neither t_1 nor t_2 are synchronizations. Since the transitions are not structural causal they involve separate threads. We have that

$$S_1 = \nu\tilde{a'}_{0\Gamma'_0}(T_0 \parallel \nu\tilde{a'}_{1\Gamma'_1}(T_1 \parallel ...\nu\tilde{a'}_{k\Gamma'_k}(R'_1 \parallel R_2)))$$

$$S_2 = \nu\tilde{a''}_{0\Gamma''_0}(T_0 \parallel \nu\tilde{a''}_{1\Gamma''_1}(T_1 \parallel ...\nu\tilde{a''}_{k\Gamma''_k}(R_1 \parallel R'_2))).$$

Therefore, whenever one of the two processes triggers a transition, the only modifications can appear in the process itself or in the $\nu\tilde{a}_{i\Gamma_i}$. We proceed by induction on the structure of the process. In the inductive case we have to show that if $R \xrightarrow{\zeta_1} S_1 \xrightarrow{\zeta_2} T$ and $R \xrightarrow{\zeta'_2} S_1 \xrightarrow{\zeta'_1} T$ then the following holds

- $$\nu\tilde{a}_\Gamma(R) \xrightarrow{\zeta_1} \nu\tilde{b}_\Delta(S_1) \xrightarrow{\zeta_2} \nu\tilde{a'}_{\Gamma'}(T) \text{ and}$$
  $$\nu\tilde{a}_\Gamma(R) \xrightarrow{\zeta'_2} \nu\tilde{b'}_{\Delta'}(S_2) \xrightarrow{\zeta'_1} \nu\tilde{a'}_{\Gamma'}(T)$$
- $$T_i \| R \xrightarrow{\zeta_1} T_i \| S_1 \xrightarrow{\zeta_2} T_i \| T \text{ and}$$
  $$T_i \| R \xrightarrow{\zeta'_2} T_i \| S_2 \xrightarrow{\zeta'_1} T_i \| T$$

Both subcases above are straightforward. It remains to show the base case of the induction, that is if

$$\nu\tilde{a}_\Gamma(R_1 \parallel R_2) \xrightarrow{\zeta_1} \nu\tilde{b}_\Delta(R'_1 \parallel R_2) \xrightarrow{\zeta_2} \nu\tilde{a'}_{\Gamma'}(R'_1 \parallel R'_2)$$

then

$$\nu\tilde{a}_\Gamma(R_1 \parallel R_2) \xrightarrow{\zeta'_1} \nu\tilde{b'}_{\Delta'}(R_1 \parallel R'_2) \xrightarrow{\zeta'_2} \nu\tilde{a'}_{\Gamma'}(R'_1 \parallel R'_2).$$

We only consider transitions that modify $\tilde{a}$ or Γ that is those derived using rules OPEN+-. We proceed by cases on the rules that we can apply for t_1 and t_2 when any of them uses rule OPEN+-.

- we apply rule OPEN+ on both t_1 and t_2:

  $$\nu a_\Gamma(R_1 \parallel R_2) \xrightarrow{(i_1,j_1,k_1):\overline{b}\langle\nu a_\Gamma\rangle} \nu a_{\Gamma+i_1}(R'_1 \parallel R_2)$$
  $$\xrightarrow{(i_2,j_2,k_2):\overline{c}\langle\nu a_{\Gamma+i_1}\rangle} \nu a_{\Gamma+i_2+i_1}(R'_1 \parallel R'_2).$$

  When we swap the transitions we obtain (using again rule OPEN+):

  $$\nu a_\Gamma(R_1 \parallel R_2) \xrightarrow{(i_2,j_2,k_2):\overline{c}\langle\nu a_\Gamma\rangle} \nu a_{\Gamma+i_2}(R_1 \parallel R'_2)$$
  $$\xrightarrow{(i_1,j_1,k_1):\overline{b}\langle\nu a_{\Gamma+i_2}\rangle} \nu a_{\Gamma+i_2+i_1}(R'_1 \parallel R'_2).$$

  Note that the labels of t_1 and t'_1 (or t_2 and t'_2) are not equal but are equivalent.

- we apply rule OPEN+ on t_1 and OPEN- on t_2:

  $$\nu a_{\Gamma+i_2}(R_1 \parallel R_2) \xrightarrow{(i_1,j_1,k_1):\overline{b}\langle\nu a_{\Gamma+i_2}\rangle}$$
  $$\nu a_{\Gamma+i_2+i_1}(R'_1 \parallel R_2) \xrightarrow{(i_2,j_2,k_2):\overline{c}\langle\nu a_{\Gamma+i_1}\rangle^-}$$
  $$\nu a_{\Gamma+i_1}(R'_1 \mid R'_2)$$

  and we have, using rules OPEN- and OPEN+,

  $$\nu a_{\Gamma+i_2}(R_1 \parallel R_2) \xrightarrow{(i_2,j_2,k_2):\overline{c}\langle\nu a_\Gamma\rangle^-}$$
  $$\nu a_\Gamma(R_1 \parallel R'_2) \xrightarrow{(i_1,j_1,k_1):\overline{b}\langle\nu a_\Gamma\rangle} \nu a_{\Gamma+i_1}(R'_1 \parallel R'_2).$$

- we apply rule OPEN+ on t_1 and rule CAUSE REF+ on t_2:

  $$\nu a_\Gamma(R_1 \parallel R_2) \xrightarrow{(i_1,j_1,k_1):\overline{b}\langle\nu a_\Gamma\rangle} \nu a_{\Gamma+i_1}(R'_1 \parallel R_2)$$
  $$\xrightarrow{(i_2,j_2,k_2):\alpha} \nu a_{\Gamma+i_1}(R'_1 \parallel R'_{2[k_2/k]@i_2})$$

  with $R_2 \xrightarrow{(i_2,j_2,k):\alpha} R'_2$ and $a \in subj(\alpha), k_2 \in \Gamma + i_1$ such that $k = k_2 \vee k \rightsquigarrow_{R_2} k_2$.
  If $k_2 = i_1$ then the transitions are in a causal relation. If $k = i_1$ and $k_2 \neq i_1$ then $k \rightsquigarrow_{R_2} k_2$ would imply that i_1 is a synchronization, which is not the case. Therefore $k, k_2 \neq i_1$ and the two transitions can safely interchange:

  $$\nu a_\Gamma(R_1 \parallel R_2) \xrightarrow{(i_2,j_2,k_2):\alpha} \nu a_\Gamma(R_1 \parallel R'_{2[k_2/k]@i_2})$$
  $$\xrightarrow{(i_1,j_1,k_1):\overline{b}\langle\nu a_\Gamma\rangle} \nu a_{\Gamma+i_1}(R'_1 \parallel R'_{2[k_2/k]@i_2})$$

- we apply rule NEW+- on t_1 and OPEN+ on t_2:

  $$\nu a_\Gamma(R_1 \parallel R_2) \xrightarrow{\zeta} \nu a_\Gamma(R'_1 \parallel R_2)$$
  $$\xrightarrow{(i,j,k):\overline{b}\langle\nu a_\Gamma\rangle} \nu a_{\Gamma+i}(R'_1 \parallel R'_2)$$

  and we can derive:

  $$\nu a_\Gamma(R_1 \parallel R_2) \xrightarrow{(i,j,k):\overline{b}\langle\nu a_\Gamma\rangle} \nu a_{\Gamma+i}(R_1 \parallel R'_2)$$
  $$\xrightarrow{\zeta} \nu a_{\Gamma+i}(R'_1 \parallel R'_2)$$

  by applying rules OPEN+ and NEW+-.

2) Suppose that t_2 is a synchronization, but not t_1, with $R = C[\nu\tilde{a}_\Gamma(R_1 \parallel R_2)]$. We can make the following remark:
Note 7.7: Consider the synchronization $S \xrightarrow{\zeta} S''$. Then $t : C[\nu\tilde{a}_\Gamma(S \parallel S')] \xrightarrow{\zeta} C[\nu\tilde{a}_\Gamma(S'' \parallel S')]$ (t modifies only S and nothing else). For the transition $t : C[\nu\tilde{a}_\Gamma S] \xrightarrow{\gamma} C[\nu\tilde{a'}_\Gamma S']$, the synchronizations can add or remove an element of $\tilde{a}$ but not of Γ, and does not change the context $C[\bullet]$.

   We proceed with a case analysis depending on which threads are involved in t_2. Since t_1 is not a synchronization it only involves threads in R_1.

Suppose that t_2 involves a thread from $C[\bullet]$, denoted with T. Then $R = C_1[\nu\tilde{a}_\Gamma(T \| C_2[\nu b_\Delta(R_1 \parallel R_2)])]$. Since only t_1 modifies C_1, from note 7.7, we can reason on $R = \nu\tilde{a}_\Gamma(T \| C_2[\nu b_\Delta(R_1 \parallel R_2)])$. We have the following hypothesis:

$$R \xrightarrow{\zeta_1} \nu\tilde{a'}_{\Gamma'}(T \| C'_2[\nu b'_{\Delta'}(R'_1 \parallel R_2)]) \xrightarrow{\zeta_2} \nu\tilde{a''}_{\Gamma''}(T' \| C''_2[\nu b''_{\Delta''}(R'_1 \parallel R'_2)]).$$

We denote with t' the transition $T \| C_2[\nu b_\Delta(R_1 \parallel R_2)] \xrightarrow{\zeta_1} T \| C'_2[\nu b'_{\Delta'}(R'_1 \parallel R_2)]$. From case 1) we can derive that t' interchanges with $\pi_l(t_2)$ and $\pi_r(t_2)$. Following a case analysis on the rules that are applied on t_2 we obtain the trace:

$$R \xrightarrow{\zeta_2} \nu\tilde{a}^\star_{\Gamma^\star}(T' \| C^\star_2[\nu b^\star_{\Delta^\star}(R_1 \parallel R'_2)]) \xrightarrow{\zeta'_1} \nu\tilde{a''}_{\Gamma''}(T' \| C''_2[\nu b''_{\Delta''}(R'_1 \parallel R'_2)])$$

in which the transitions have interchange their position.

Let us now suppose that t_2 does not involve the context. We proceed by an induction on the structure of the process. From the note 7.7 it follows that the inductive case is trivial (since only t_1 modifies the context). For the base case we have $R = \nu\tilde{a}_\Gamma(R_1 \parallel R_2)$. Since t_2 does not involve the context we have the following subcases:

- t_2 only involves threads in R_2. Then we have:

  $$\nu\tilde{a}_\Gamma(R_1 \parallel R_2) \xrightarrow{\zeta_1} \nu\tilde{a'}_{\Gamma'}(R'_1 \parallel R_2) \xrightarrow{\zeta_2} \nu\tilde{a'}_{\Gamma'}(R'_1 \parallel R'_2)$$

  and it is easy to show that:

  $$\nu\tilde{a}_\Gamma(R_1 \parallel R_2) \xrightarrow{\zeta_2} \nu\tilde{a}_\Gamma(R_1 \parallel R'_2) \xrightarrow{\zeta_1} \nu\tilde{a'}_{\Gamma'}(R'_1 \parallel R'_2).$$

- t_2 involves a thread in R_1 and one in R_2. We reason by cases on t_2:
  - if rule COM+- is applied then we have

    $$\nu\tilde{a}_\Gamma(R_1 \parallel R_2) \xrightarrow{\zeta_1} \nu\tilde{a'}_{\Gamma'}(R'_1 \parallel R_2) \xrightarrow{\zeta_2} \nu\tilde{a'}_{\Gamma'}(R''_1 \parallel R'_2)$$

    One has to show that transition $R_1 \xrightarrow{\zeta_1} R'_1$ interchanges with $\pi_l(t_2)$ and $\pi_r(t_2)$, which follows from case 1).
  - if rule CLOSE+ is applied then the hypothesis becomes:

    $$\nu\tilde{a}_\Gamma(R_1 \parallel R_2) \xrightarrow{\zeta_1} \nu\tilde{a'}_{\Gamma'}(R'_1 \parallel R_2) \xrightarrow{\zeta_2} \nu\tilde{a'}_{\Gamma'}\nu b_{\Gamma''}(R''_1 \parallel R'_2).$$

    If $b \notin \tilde{a}$ then similar to above. Otherwise $b = a \in \tilde{a}$ and we can rewrite the transition in the following simpler form:

    $$\nu a_\Gamma(R_1 \parallel R_2) \xrightarrow{\zeta_1} \nu a_{\Gamma'}(R'_1 \parallel R_2) \xrightarrow{\zeta_2} \nu a_{\Gamma'}\nu a_{\Gamma''}(R''_1 \parallel R'_2).$$

    We have to reason by cases on the rules that applies for t_1:
    - rule OPEN-

      $$\nu a_{\Gamma+i_1}(R_1 \parallel R_2) \xrightarrow{(i_1,j_1,k_1):\bar{b}\langle\nu a_\Gamma\rangle^-} \nu a_\Gamma(R'_1 \parallel R_2) \xrightarrow{(i_2,*,*):\tau} \nu a_\Gamma(\nu a_{\Gamma''}(R''_1 \parallel R'_2))$$

      Suppose that $R'_1 \xrightarrow{(i_2,j_2,k_2):\bar{c}\langle\nu a_{\Gamma''}\rangle} R''_1$ (similar if the output is on R_2). From proposition 7.3 $R'_1 = C[\nu a_{\Gamma''} S']$ and $R''_1 = C[\nu a_{\Gamma''+i_2} S'']$. From Lemma 3.5, $R_1 = C[\nu a_{\Gamma''+i_1} S]$. Then the transitions become:

      $$\nu a_{\Gamma+i_1}(C[\nu a_{\Gamma''+i_1} S] \parallel R_2) \xrightarrow{(i_1,j_1,k_1):\bar{b}\langle\nu a_\Gamma\rangle^-} \nu a_\Gamma(C[\nu a_{\Gamma''} S'] \parallel R_2) \xrightarrow{(i_2,*,*):\tau} \nu a_\Gamma(\nu a_{\Gamma''}(C[\nu a_{\Gamma''+i_2} S''] \parallel R'_2))$$

      We know from case 1) that

      $$R_1 \xrightarrow{(i_1,j_1,k_1):\bar{b}\langle\nu a_\Gamma\rangle} R'_1 \xrightarrow{(i_2,j_2,k_2):\bar{c}\langle\nu a_{\Gamma''}\rangle} R''_1$$

      can interchange (similar for R_2). Therefore we can derive:

      $$\nu a_{\Gamma+i_1}(C[\nu a_{\Gamma''+i_1} S] \parallel R_2) \xrightarrow{(i_2,*,*):\tau} \nu a_{\Gamma+i_1}(\nu a_{\Gamma''+i_1}(C[\nu a_{\Gamma''+i_1+i_2} S] \parallel R'_2)) \xrightarrow{(i_1,j_1,k_1):\bar{b}\langle\nu a_\Gamma\rangle^-} \nu a_\Gamma(\nu a_{\Gamma''}(C[\nu a_{\Gamma''+i_2} S''] \parallel R'_2)).$$

  - if rule CLOSE- is applied then we have the hypothesis:

    $$\nu\tilde{a}_\Gamma\nu b_{\Gamma''}(R_1 \parallel R_2) \xrightarrow{\zeta_1} \nu\tilde{a'}_{\Gamma'}\nu b_{\Gamma''}(R'_1 \parallel R_2) \xrightarrow{\zeta_2} \nu\tilde{a'}_{\Gamma'}(R''_1 \parallel R'_2)$$

    If $b \notin \tilde{a}$ then it is trivial. Otherwise $b = a \in \tilde{a}$ and we can reason on the following simpler form of the transition:

    $$\nu a_\Gamma(R_1 \parallel R_2) \xrightarrow{\zeta_1} \nu a_{\Gamma'}(R'_1 \parallel R_2) \xrightarrow{\zeta_2} R''_1 \parallel R'_2.$$

    We reason by cases on the rules applied on t_1:
    - rule OPEN+

      $$\nu a_\Gamma(R_1 \parallel R_2) \xrightarrow{(i_1,j_1,k_1):\bar{b}\langle\nu a_\Gamma\rangle} \nu a_{\Gamma+i_1}(R'_1 \parallel R_2) \xrightarrow{(i_2,\star,\star):\tau^-} R''_1 \parallel R'_2$$

From the premise of rule CLOSE- we have that $R_1' \xrightarrow{(i_2,j_2,k_2):\overline{c}\langle \nu a_{\Gamma+i1}\rangle^-} R_1''$ (similar if the output is on R_2). Then, from proposition 7.3 $R_1'' = C'[\nu a_{\Gamma+i_1} S'']$ and $R_1' = C'[\nu a_{\Gamma+i_1+i_2} S']$. From Lemma 3.4, $R_1 = C[\nu a_{\Gamma+i_2} S]$. Then the transitions become:

$$\nu a_\Gamma(C[\nu a_{\Gamma+i_2} S] \parallel R_2) \xrightarrow{(i_1,j_1,k_1):\overline{b}\langle \nu a_\Gamma\rangle}$$
$$\nu a_{\Gamma+i_1}(C'[\nu a_{\Gamma+i_1+i_2} S'] \parallel R_2) \xrightarrow{(i_2,\star,\star):\tau^-}$$
$$C'[\nu a_{\Gamma+i_1} S''] \parallel R_2'.$$

We can swap the two transitions and obtain:

$$\nu a_\Gamma(C[\nu a_{\Gamma+i_2} S] \parallel R_2) \xrightarrow{(i_2,\star,\star):\tau^-}$$
$$C[\nu a_\Gamma S'] \parallel R_2' \xrightarrow{(i_1,j_1,k_1):\overline{b}\langle \nu a_\Gamma\rangle}$$
$$C'[\nu a_{\Gamma+i_1} S''] \parallel R_2'.$$

3) If t_1 is a synchronization, but not t_2 then similar to above.
4) Suppose that both t_1 and t_2 are synchronizations. Consider that the first thread in the structure of R that is involved in one of the transitions is R_1 in t_1 (similar if in t_2). Then we write $R = C[\nu \tilde{a}_\Gamma(R_1 \parallel R_2)]$. As in the cases above, we can reason only on $R = \nu \tilde{a}_\Gamma(R_1 \parallel R_2)$. We have the following cases:
   - If t_1 involves only threads in R_1 and t_2 only threads in R_2 from note 7.7 it is enough to show that we can swap the two transitions for $R_1 \mid R_2$. We have then the hypothesis:

     $$R_1 \parallel R_2 \xrightarrow{\zeta_1} R_1' \parallel R_2 \xrightarrow{\zeta_2} R_1' \parallel R_2'$$

     and we have to show that

     $$R_1 \parallel R_2 \xrightarrow{\zeta_2} R_1 \parallel R_2' \xrightarrow{\zeta_1} R_1' \parallel R_2'$$

     which follows from a simple case analysis on t_1 and t_2.
   - If t_1 involves only threads in R_1 but t_2 involves a thread from R_1 as well then, from note 7.7, we have the following hypothesis:

     $$\nu \tilde{a}_\Gamma(R_1 \parallel R_2) \xrightarrow{\zeta_1}$$
     $$\nu \tilde{a}_\Gamma(R_1' \parallel R_2) \xrightarrow{\zeta_2} \nu \tilde{a'}_\Gamma(R_1'' \parallel R_2').$$

     From the cases above we have that the transition $R_1 \xrightarrow{\zeta_1} R_1'$ can interchange positions with $\pi_l(t_2)$ and $\pi_r(t_2)$. From a case analysis on the rules that apply for t_2 we can derive the transitions:

     $$\nu \tilde{a}_\Gamma(R_1 \parallel R_2) \xrightarrow{\zeta_2}$$
     $$\nu \tilde{a'}_\Gamma(S \parallel R_2') \xrightarrow{\zeta_1} \nu \tilde{a'}_\Gamma(R_1'' \parallel R_2').$$

   - If t_1 involves a thread in both R_1 and R_2 and t_2 only in R_2 then similar to above.
   - If t_1 and t_2 involve threads in both R_1 and R_2, then

     $$\nu \tilde{a}_\Gamma(R_1 \parallel R_2) \xrightarrow{\zeta_1}$$
     $$\nu \tilde{a'}_\Gamma(R_1' \parallel R_2') \xrightarrow{\zeta_2} \nu \tilde{a''}_\Gamma(R_1'' \parallel R_2'').$$

     From case 1) we have that $\pi_l(t_1)$ and $\pi_l(t_2)$ interchange positions and similarly for $\pi_r(t_1)$ and $\pi_r(t_2)$. We proceed with a case analysis on the transitions that apply on t_1 and t_2:
     – rule CLOSE+ on t_1 and CLOSE- on t_2:

     $$\nu a_\Gamma(R_1 \| R_2) \xrightarrow{\zeta_1}$$
     $$\nu a_\Gamma(\nu a'_{\Gamma'}(R_1' \| R_2')) \xrightarrow{\zeta_2} \nu a'_{\Gamma'}(R_1'' \| R_2'')$$

     We swap the transitions to obtain:

     $$\nu a_\Gamma(R_1 \| R_2) \xrightarrow{\zeta_2} S_1 \| S_2' \xrightarrow{\zeta_1} \nu a'_{\Gamma'}(R_1'' \| R_2'').$$

■

Events do not contain enough information to fully characterize a transition. We say that two transitions t and t' are *memory equivalent*, denoted by $t =_e t'$ if they correspond to the same event.

Example 7.1: Consider $R = \varepsilon \rhd a.P_1 \| \varepsilon \rhd a.P_2$ and the transitions

$$t : R \xrightarrow{(i,*,*):a} \langle i, *, a\rangle.\varepsilon \rhd P_1 \| \varepsilon \rhd a.P_2$$

$$t' : R \xrightarrow{(i,*,*):a} \varepsilon \rhd a.P_1 \| \langle i, *, a\rangle.\varepsilon \rhd P_2$$

$t \neq t'$ but $t =_e t'$ since both transitions add the event $\langle i, *, a\rangle$ into the memory.

Within a trace, the LTS ensures the uniqueness of the events identifiers.

Proposition 7.8: In a trace, if two transitions t and t' are memory equivalent $t = opp(t')$.

Proof: From $id(t) = id(t')$ we have that necessarily the transitions share the same threads. Hence

$$t : R \xrightarrow{(i,j,k):\alpha} S \qquad t' : S \xrightarrow{(i,j,k):\alpha^-} R$$

Then $opp(t') = R \xrightarrow{(i,j,k):\alpha} S$. From Lemma 3.2 we have that $t = opp(t')$. ■

In the example 7.1 the transitions are not equal because they do not appear on the same trace, but also because they do not share a common thread.

Corollary 7.9: If $t =_e t'$ are coinitial and share a common thread then $t = t'$.

Lemma ?? Let s be a trace. Then there exists two traces r, only backward, and r', only forward, such that $s \sim r; r'$.

Proof: Similar to the proof of [**?**]. We proceed by induction on the length of s and the distance to the earliest disagreement pair in s, that we denote with $t'; t$:

$$t' : R \xrightarrow{(i,j,k):\alpha} S \qquad t : S \xrightarrow{(i',j',k'):\alpha'^-} T$$

We have the following cases:

- t, t' are concurrent. We can apply Lemma 4.3 and swap them. As a result we decrease the distance to the earliest disagreement pair.
- t, t' are causal. We distinguish on the two possible types of causality:
  - Structural causality. It implies that the transitions share a common thread. Since the memory of a thread behaves as a stack, and that the two transitions are consecutive and of opposite direction we have that $t' =_e t$. From Proposition 7.8 $t = opp(t')$. Since $t; opp(t) \sim \epsilon$ we can replace the pair $t'; t$ in s and obtain a trace causally equivalent to s but with a smaller length, on which we can then apply the induction hypothesis.
  - Contextual causality. We have two possibilities:
    - $i = k'$. This case is not allowed by the LTS: we cannot backtrack on an event that was chosen as a contextual cause by a previous transition.
    - $i' = k$. Not allowed by the LTS: the transition $opp(t)$ was triggered before t' and we cannot choose as cause a transition that has not yet fired.

■

Lemma 7.10: Let s_1, s_2 be two coinitial and cofinal traces, where s_2 is only forward. Then there exists s_1' shorter or equal to s_1 such that $s_1 \sim s_1'$.

Proof: We proceed by induction on the length of s_1. If s_1 is forward then $s_1' = s_1$. Otherwise, using Lemma **??**, we can assume that s_1 is parabolic: $s_1 = u; t; t'; v$, where $u; t$ is backward, $t'; v$ forward and $t; t'$ is the only pair of consecutive transitions that are in opposite direction. Since s_2 is only forward and s_1, s_2 are coinitial and cofinal, whatever the transition t takes out from the memory has to be put back by some other transition in $t'; v$. We denote with t'' the earliest such transition, that is $t =_e t''$, on the same threads and of different signs. Then, from Proposition 7.8, $t = opp(t'')$. We can rewrite s_1 as $u; t; t'; v'; t''; v''$.

Let us now show that t is concurrent with all transitions up to t''. Suppose $\exists t_\star$ between t and t'' such that t and $t_\star$ are causal. We have the following cases:

- t and $t_\star$ are structural causal. Contradiction with the hypothesis that t'' is the earliest transition sharing a thread with t.
- t and $t_\star$ are contextual causal. We have that:
  - either $c(t) = id(t_\star)$. Not possible since $t_\star$ succeeds t;
  - or $c(t_\star) = id(t)$. Not possible either since t is backward;

Therefore all transition between t and t'' are concurrent to t. It follows, from Lemma 4.3, that we can swap t with each transition up to t'' and obtain $s_1 \sim u; t'; v'; t; t''; v''$. By the definition of $\sim$, $s_1 \sim u; t'; v'; v''$, hence the length decreases and we can apply the induction hypothesis. ■

Lemma 4.5 Two traces s_1, s_2 are coinitial and cofinal if and only if $s_1 \sim s_2$.

Proof: If $s_1 \sim s_2$ then we can derive that s_1, s_2 are coinitial and cofinal from the definition of $\sim$.

Suppose then that s_1, s_2 are coinitial and cofinal. Due to Lemma **??** we can suppose that they are parabolic. We reason by induction on the lengths of s_1, s_2 and on the depth of the earliest disagreement between them, denoted by the pair t_1, t_2. We have the following cases, depending on whether t_1 and t_2 are forward or not:

1) if t_1 is forward and t_2 backward, then

$$s_1 = u_1; t_1; v_1 \qquad s_2 = u_2; t_2; v_2$$

where $u_1 \sim u_2$, u_1 backwards and v_1 forward (since s_1 is parabolic). Then the traces $t_1; v_1$, $t_2; v_2$ are coinitial and cofinal, with $t_1; v_1$ only forward. We apply Lemma 7.10 and derive that there exists $s_2' \sim t_2; v_2$ shorter or equal to $t_2; v_2$. If it is equal then t_2 is forward which contradicts the hypothesis. We then proceed by induction with $u_2; s_2'$ shorter.

2) if both transitions are forward, then

$$s_1 = u_1; t_1; v_1 \qquad s_2 = u_2; t_2; v_2$$

where $u_1 \sim u_2$, and $t_1; v_1$, $t_2; v_2$ are coinitial, cofinal and both forward.

- if t_1, t_2 are concurrent, then whatever t_1 puts in the memory must be copied in v_2. Let us denote with t_1' the earliest such transition (ie $t_1' \in v_2$, $t_1' =_e t_1$ and on the same threads) and rewrite $t_2; v_2$ as $t_2; u_2; t_1'; u_2'$. We show that t_1' is concurrent with all transitions in u_2:
  - t_1' is the earliest transition on the same thread as t_1, thus it is not structural causal with any transition in $t_2; u_2$;
  - $t_1' =_e t_1$ hence $c(t_1') = c(t_1)$. Then t_1 cannot have as contextual cause a transition in $t_2; u_2$.

  Using the lemma 4.3, we have:

$$t_2; v_2 = t_2; u_2; t_1'; u_2' \sim t_1'; t_2; u_2; u_2'.$$

  From Corollary 7.9 $t_1' = t_1$. We obtain a later earliest disagreement pair, without changing the lengths of s_1 and s_2, hence we can rely on the induction hypothesis.
- if t_1, t_2 are causal, then let us show that this case is not possible. We have the following types of causality:
  - structural causal: suppose that t_1 and t_2 are not memory equivalent. Then the traces $t_1; v_1$ and $t_2; v_2$ add into the memory events in a different order, hence the traces are not cofinal, which is a contradiction.

We have then that $t_1 =_e t_2$, and by Corollary 7.9 $t_1 = t_2$, which contradicts the hypothesis. Therefore t_1 and t_2 are not structural causal.

- contextual causal: suppose that t_2 is a cause for t_1 (similar if t_1 is a cause for t_2). In order for t_1 to choose as cause t_2 then $\exists t_2' \in u_1$ and hence an earlier disagreement pair, which again contradicts the hypothesis.

3) if they are both backward, we proceed similar to above.

■

Proposition 5.2. If $t : R \xrightarrow{(i,*,*):\tau} S$ then there exists at most one context $C[\bullet]$ such that $t : C[R_1 \| R_2] \xrightarrow{(i,*,*):\tau} C'[S_1 \| S_2]$ with $R_q \neq S_q, q \in \{1,2\}$.

Proof: It follows from the structure of a Rπ process $R := \nu \tilde{a}_\Gamma (S \parallel T)$. ■

Lemma 5.4. In a trace

$$R_0 \xrightarrow{(i_1,j_1,k_1):\alpha_1} R_1 ... \xrightarrow{(i_n,j_n,k_n):\alpha_n} R_n$$

$\exists C[\bullet]$ such that $C[R_0] = R_0' \xrightarrow{(i_1,*,*):\tau} R_1' ... \xrightarrow{(i_n,*,*):\tau} R_n'$ for some $R_1', .., R_n'$ and, for $q \in \{0, .., n\}$:

- if $\alpha_q \neq \tau$ then $\exists x, \pi_x(R_q' \xrightarrow{(i_q,*,*):\tau} R_{q+1}') = R_q \xrightarrow{(i_q,j_q,k_q):\alpha_q} R_{q+1}$
- if $\alpha_q = \tau$ then $\exists C'[\bullet]$ such that $R_q' = C'[R_q], R_{q+1}' = C'[R_{q+1}]$.

Proof: We construct the context by induction on the length of the trace. We start with the last transition in the trace for which we provide the minimum reduction context. We then add a prefix to the context for each transition in the trace. In order to keep track of the context built so far we prove the following stronger induction hypothesis:

$$\text{in } R_0 \xrightarrow{(i_1,j_1,k_1):\alpha_1} R_1 ... \xrightarrow{(i_n,j_n,k_n):\alpha_n} R_n$$

$$\exists S_0, R_0 \| S_0 = R_0' \xrightarrow{(i_1,*,*):\tau} R_1' ... \xrightarrow{(i_n,*,*):\tau} R_n'$$

where $C[\bullet] = [\bullet] \| S_0$ respects the necessary properties.

- Base case. Let $R_{n-1} \xrightarrow{(i_n,j_n,k_n):\alpha_n} R_n$.
  - If $\alpha_1 = \tau$ then $S_{n-1} = 0$ trivially verifies the necessary properties.
  - If $\alpha_1 \neq \tau$ then $S_{n-1} = \varepsilon \triangleright \overline{\alpha}_n.0$. We have the following transition: $R_{n-1} \| S_{n-1} \xrightarrow{(i_n,*,*):\tau} C'[R_n \| S_n]$, where the projections are easily verified.
- Inductive case. Let $R_{p-1} \xrightarrow{(i_p,j_p,k_p):\alpha_p} R_p$. By induction we have that $\exists S_p$ such that

  $$R_p \| S_p \xrightarrow{(i_{p+1},*,*):\tau} R_{p+1}' ... \xrightarrow{(i_p,*,*):\tau} R_n'.$$

  We have to show that there exists S_{p-1} such that $R_{p-1} \| S_{p-1} \xrightarrow{(i_p,*,*):\tau} R_p'' ... \xrightarrow{(i_n,*,*):\tau} R_n''$ with $\exists x, \pi_x(R_q'' \xrightarrow{(i_q,*,*):\tau} R_{q+1}'') = R_q \xrightarrow{(i_q,j_q,k_q):\alpha_q} R_{q+1}$ for all $\alpha_q \neq \tau$ and $\exists C''[\bullet]$ such that $R_q'' = C''[R_q], R_{q+1}'' = C''[R_{q+1}]$ for $\alpha_q = \tau$ where $q \in \{0, .., n\}$.

  We proceed by cases on α_p:
  - $\alpha_p = \tau$ then let $S_{p-1} = S_p$ and the hypothesis is verified.
  - $\alpha_p = b(a)$ (or $\overline{b}\langle a \rangle$) then $\overline{\alpha}_p = \overline{b}\langle a \rangle$ (or $b(a)$ respectively). Let $S_{p-1} = \overline{\alpha_p}.S_p$.
    We have the following trace:

    $$R_{p-1} \| S_{p-1} \xrightarrow{(i_p,*,*):\tau} R_p \| S_p \xrightarrow{(i_{p+1},*,*):\tau} R_{p+1}' ... \xrightarrow{(i_n,*,*):\tau} R_n'$$

    where the first synchronization does not apply any substitution (since $(S_p)_{\{a/a\}} = S_p$). We have that

    $$\pi_l(R_{p-1} \| S_{p-1} \xrightarrow{(i_p,*,*):\tau} R_p \| S_p) = R_{p-1} \xrightarrow{(i_p,j_p,k_p):\alpha_p} R_p$$

    and for the rest of the transitions the properties follow from induction.
  - $\alpha_p = \overline{b}(\nu a_\Gamma)$ then $\overline{\alpha}_p = b(a)$. Let $S_{p-1} = \overline{\alpha_p}.S_p$ and thus the trace becomes

    $$R_{p-1} \| S_{p-1} \xrightarrow{(i_p,*,*):\tau} \nu a_\Gamma (R_p \| S_p) \xrightarrow{(i_{p+1},*,*):\tau} R_{p+1}'' = \nu a_\Gamma (R_{p+1}') ... \xrightarrow{(i_n,*,*):\tau} R_n'' = \nu a_\Gamma (R_n')$$

    since νa_Γ cannot prevent the transitions with label τ. We have that

    $$\pi_l(R_{p-1} \| S_{p-1} \xrightarrow{(i_p,*,*):\tau} \nu a_\Gamma (R_p \| S_p)) = R_{p-1} \xrightarrow{(i_p,j_p,k_p):\alpha_p} R_p.$$

    We need to verify that the properties still hold for the rest of the transitions:
    - ∗ if $\alpha_q \neq \tau$ then

      $$\exists x, \pi_x(\nu a_\Gamma (R_q') \xrightarrow{(i_q,*,*):\tau} \nu a_\Gamma (R_{q+1}')) = \pi_x(R_q' \xrightarrow{(i_q,*,*):\tau} R_{q+1}') = R_q \xrightarrow{(i_q,j_q,k_q):\alpha_q} R_{q+1}$$

      where the last equality follows from the induction hypothesis.
    - ∗ if $\alpha_q = \tau$ then by induction we have that there exists $C'[\bullet]$ such that $R_q' = C'[R_q], R_{q+1}' = C'[R_{q+1}]$. Let $C''[\bullet] = \nu a_\Gamma (C'[\bullet])$ which verifies that $C''[R_q] = \nu a_\Gamma (C'[R_q]) = R_q''$ and similarly for R_{q+1}''.

■

Lemma 5.1 Let $t_1 : R \xrightarrow{(i_1,*,*):\tau} S$ and $t_2 : S \xrightarrow{(i_2,*,*):\tau} T$. If t_1 and t_2 are causal then $t_1 \sqsubset_T t_2$.

Proof: Since t_1 and t_2 are consecutive we have that either $t_1 \sqsubset_T t_2$ or $t_1 \prec_T t_2$. Nothing to prove for the first case. Let us then consider the case $t_1 \prec_T t_2$.

From $t_1 \prec_T t_2$ and that t_2 is a synchronization it follows that

$$\pi_l(t_2) = S_1 \xrightarrow{(i_2,j_2,k_2):\alpha_2} T_1 \qquad \pi_r(t_2) = S_2 \xrightarrow{(i_2,j_2',k_2'):\overline{\alpha_2}} T_2$$

with either $k_2 = i_1$ or $k_2' = i_1$. Let us suppose that $k_2 = i_1$ (similar for $k_2' = i_1$). We denote with a the subject of α_2.

From the definition of the projections we have that $S = C[S_1 \| S_2]$. Since $k_2 = i_1$ and from Proposition 7.5, $\nu a_\Gamma \in S_1$. By Lemma 3.5, $\nu a_{\Gamma'} \notin S_2$. But $a \in subj(\overline{\alpha_2})$, then, from Lemma 3.6 $\langle i, \_, d[a/c] \rangle \in S_2$. We have then that $i \sqsubset_{T_2} i_2$ and $j_2' = i$.

We apply a synchronization rule for transition t_2 (either COM+ or CLOSE+), with the side conditions $k_2 =_* j_2'$ and $k_2' =_* j_2$. From the first, we have that $k_2 = j_2'$, hence $i = i_1$ and $i_1 \sqsubset_T i_2$. ■

Theorem 5.5 Let t_1 and t_2 be two transitions such that

$$t_1 : R \xrightarrow{(i_1,j_1,k_1):\alpha_1} S \qquad t_2 : S \xrightarrow{(i_2,j_2,k_2):\alpha_2} T.$$

t_1 and t_2 are concurrent $\iff$ $\exists$ a reduction context $C[\bullet]$ such that

$$t_1' : C[R] \xrightarrow{(i_1,*,*):\tau} S' \qquad t_2' : S' \xrightarrow{(i_2,*,*):\tau} T'$$

with t_1' and t_2' concurrent.

Proof:

1) If t_1 and t_2 are concurrent then $\exists C[\bullet]$ such that t_1' and t_2' concurrent. We proceed by cases on α_1, α_2.
   - $\alpha_1, \alpha_2 \neq \tau$. Then let $C[\bullet] = [\bullet] \| (\varepsilon \rhd \overline{\alpha_1} \| \varepsilon \rhd \overline{\alpha_2})$, where if $\alpha_q = b(a)$ (or $\overline{b}\langle a \rangle, \overline{b}(\nu a_\Gamma)$), $\overline{\alpha_q} = \overline{b}\langle a \rangle$ (or $b(a)$ respectively). We obtain the following trace (in which we ignore the empty processes):

     $$t_1' : C[R] \xrightarrow{(i_1,*,*):\tau} C'[S \| \varepsilon \rhd \overline{\alpha_2}]$$
     $$t_2' : C'[S \| \varepsilon \rhd \overline{\alpha_2}] \xrightarrow{(i_2,*,*):\tau} C''[T]$$

     It is straightforward that t_1' and t_2' are concurrent and that t_1 and t_2 respectively, are their projections.
   - If $\alpha_1 = \tau, \alpha_2 \neq \tau$ then let $C[\bullet] = [\bullet] \| \varepsilon \rhd \overline{\alpha_2}$ where if $\alpha_2 = b(a)$ (or $\overline{b}\langle a \rangle, \overline{b}(\nu a_\Gamma)$), $\overline{\alpha_2} = \overline{b}\langle a \rangle$ (or $b(a)$ respectively). We have the trace, in which we ignored the empty processes:

     $$t_1' : R \| \varepsilon \rhd \overline{\alpha_2} \xrightarrow{(i_1,*,*):\tau} C'[S \| \varepsilon \rhd \overline{\alpha_2}]$$
     $$t_2' : C'[S \| \varepsilon \rhd \overline{\alpha_2}] \xrightarrow{(i_2,*,*):\tau} C''[T]$$

     t_1' and t_2' are concurrent and the necessary properties for the reduction context $C[\bullet]$ are verified.
   - Similar for the rest of the cases.
2) If t_1 and t_2 are causal then $\forall C[\bullet]$, t_1' and t_2' are causal. We have that $e_1 = \langle i_1, k_1, \alpha_1 \rangle \in T$ and $e_2 = \langle i_2, k_2, \alpha_2 \rangle \in T$. From the properties of a reduction context we have that $e_q = \langle i_q, k_q, \alpha_q' \rangle \in T'$ where:

   $$\alpha_q' = b[c/a] \text{ if } \alpha_q = b[*/a]$$
   $$\alpha_q' = \alpha_q \text{ otherwise}$$

   for $q \in \{1, 2\}$ and $x \in \{l, r\}$. Hence if $e_1 \sqsubset_T e_2$ then $e_1 \sqsubset_{T'} e_2$ (the order in which the events are added to the memory does not change) and if $e_1 \prec_T e_2$ then $e_1 \prec_{T'} e_2$ (the events do not change their contextual cause).

■

D. Details of section III-B

In order to clarify the proofs, we introduce an intermediate calculus, similar to Rπ, but that attaches to a process a list of substitutions (instead of a memory), called an *environment*. As in Rπ, an input does not modify the process itself but adds the substitutions to the list. Only when a name is used in a transition label that the substitutions are retrieved from the list and applied. Other transitions do not modify the environment. We call the calculus π_l and say that this type of substitution is a *late* one.

1) The intermediate calculus: We denote by T, U processes of π_l and preserve the rest of the notations from the previous sections.

$$\begin{array}{ll} \xi ::= & [a/c] :: \xi \mid [*/c] :: \xi \mid \varepsilon \quad \text{(environments)} \\ T, U ::= & (\xi \cdot P \mid T) \mid \nu a T \mid \emptyset \quad (\pi_l \text{ processes}) \end{array}$$

As in Rπ, we have the following congruence rules:

$$\xi \cdot (P \mid Q) \equiv_l \xi \cdot P \mid \xi \cdot Q$$

$$\xi \cdot \nu a P \equiv_l \nu a (\xi \cdot P) \text{ if } a \notin \xi$$

There is no rule that allows a rearrangement (up to congruence) of the environment. In this manner the order in which the substitutions occurred in the process is preserved.

Similarly to Rπ, we define the functions that apply the substitution on the transition labels and update the environment after a synchronization:

Definition 7.11: Let $T_{[a/b]}$ be the synchronization update on the process T:

$$\begin{array}{ll} (T \mid U)_{[a/c]} & = T_{[a/c]} \mid U_{[a/c]} \\ (\nu a T)_{[a/c]} & = \nu a (T_{[a/c]}) \\ ([\star/b] :: \xi \cdot P)_{[a/c]} & = [a/b] :: \xi \cdot P, \text{ if } b = c \\ (\xi \cdot P)_{[a/c]} & = \xi \cdot P, \textit{otherwise} \end{array}$$

Let ξ be a function on the labels of a transition for a thread, defined inductively on the environment:

$$\varepsilon[a] = a \quad \xi[b(c)] = \xi[b](c) \quad \xi[\overline{b}\langle a \rangle] = \overline{\xi[b]}\langle \xi[a] \rangle$$
$$([a/c] :: \xi)[c] = a \quad (\_ :: \xi)[a] = \xi[a], \textit{otherwise}$$

The transition rules are similar to the ones of π calculus, except that the environment is updated.

$$\text{IN}_l \quad \xi \cdot b(c).P + Q \xrightarrow{\xi[b(c)]} [*/c] :: \xi \cdot P$$

$$\text{OUT}_l \quad \xi \cdot \bar{b}\langle a\rangle.P + Q \xrightarrow{\xi[\bar{b}\langle a\rangle]} \xi \cdot P \qquad \text{OPEN}_l \quad \frac{T \xrightarrow{\bar{b}\langle a\rangle} T'}{\nu a T \xrightarrow{\bar{b}\langle \nu a\rangle} T'}$$

$$\text{COM}_l \quad \frac{T \xrightarrow{\bar{b}\langle a\rangle} T' \qquad U \xrightarrow{b(c)} U'}{T \mid U \xrightarrow{\tau} T' \mid U'_{[a/c]}}$$

$$\text{CLOSE}_l \quad \frac{T \xrightarrow{\bar{b}\langle \nu a\rangle} T' \qquad U \xrightarrow{b(c)} U'}{T \mid U \xrightarrow{\tau} T' \mid U'_{[a/c]}} \text{ if } a \notin \text{fn}(U)/\{c\}$$

$$\text{PAR}_l \quad \frac{T \xrightarrow{\alpha} T'}{T \mid U \xrightarrow{\alpha} T' \mid U} \text{ if } \text{bn}(\alpha) \cap \text{fn}(U) = \emptyset$$

$$\text{MEM}_l \quad \frac{T \equiv_l U \xrightarrow{\alpha} U' \equiv_l T'}{T \xrightarrow{\alpha} T'} \qquad \text{NEW}_l \quad \frac{T \xrightarrow{\alpha} T'}{\nu a T \xrightarrow{\alpha} \nu a T'} \text{ if } a \notin \alpha$$

Definition 7.12: Let ϕ be a function that translates π_l processes into π, by applying all the substitutions:

$$\phi(T \mid U) = \phi(T) \mid \phi(U) \quad \phi(\nu a T) = \nu a \phi(T)$$
$$\phi([a/c] :: \xi \cdot P) = \phi(\xi \cdot P\{a/c\})$$
$$\phi([*/c] :: \xi \cdot P) = \phi(\xi \cdot P) \quad \phi(\varepsilon \cdot P) = P$$

Proposition 7.13: (Strong bisimulation between π_l and its π-image)

1) If $T \xrightarrow{\gamma} U$ and $\phi(T) = P$ then there exists Q such that $P \xrightarrow{\gamma} Q$ and $\phi(U) = Q$.
2) If $P \xrightarrow{\gamma}_\pi Q$ then for all T such that $\phi(T) = P$ there exists U with $T \xrightarrow{\gamma} U$ and $\phi(U) = Q$.

As a remark, the $\xi\pi$ calculus bears a close resemblance to a π calculus with *explicit substitution*. The latter is a calculus in which the substitution are handled explicitly, similarly to the λ calculus with explicit substitution.

Examples of such calculi are given in [12], [13] and also in [14], from which we borrowed the idea of an environment. However a difference in our approach is that we never need to apply the substitutions, as deriving the transition labels is still a meta-syntactic operation. This is a consequence of the fact that we are interested in a framework for reversibility. In this sense, the aforementioned calculi are closer to the explicit substitution of λ calculus than the late substitution we employ in π_l.

2) Correspondence between π_l and Rπ:

Definition 7.14: Let ϕ be a function that translates Rπ processes into π_l, by recording all the substitutions into the environment and erasing all $\nu_{\Gamma \neq \emptyset}$ from the process.

$$\phi(\varepsilon \rhd P) = \varepsilon \cdot P$$
$$\phi(R \parallel S) = \phi(R) \mid \phi(S)$$
$$\phi(\nu a_\emptyset R) = \nu a \phi(R)$$
$$\phi(\nu a_{\Gamma \neq \emptyset} R) = \phi(R)$$
$$\phi(\langle i, k, b[a/c]\rangle.m \rhd P) = [a/c] :: \phi(m \rhd P)$$
$$\phi(\langle i, k, b[\star/c]\rangle.m \rhd P) = [\star/c] :: \phi(m \rhd P)$$
$$\phi(e.m \rhd P) = \phi(m \rhd P) otherwise$$

Proposition 7.15: (Strong bisimulation between forward Rπ and its π_l-image)

1) If $R \xrightarrow{\gamma} S$ then $\phi(R) \xrightarrow{\gamma} \phi(S)$.
2) If $T \xrightarrow{\gamma} U$ and $\forall R$ such that $\phi(R) = T$, $\exists S$ with $R \xrightarrow{\gamma} S$ and $\phi(S) = U$.

The two Propositions 7.13, 7.15 can be extended to traces.

In order to show the correspondence between Rπ and π, we define the *forgetful map*, denoted with ϕ, as the composition of the two translations above.

Proposition 3.9 If $P \longrightarrow^\star Q$ and $\phi(R) = P$ then there exists S such that $R \longrightarrow^\star S$ and $\phi(S) = Q$.

Proof: Using the second part of the propositions 7.13 and 7.15 we have that for each transition of a π process there exists one for its Rπ correspondent. The result follows from induction on the length of the trace. ■

Proposition 3.10 If $\varepsilon \rhd P \longrightarrow^\star R$ then $P \longrightarrow^\star \phi(R)$.

3) Proofs: For the following proofs we need to present an LTS for the π calculus:

$$\text{IN}_\pi \quad b(c).P \xrightarrow{b(c)} P \qquad \text{OUT}_\pi \quad \bar{b}\langle a\rangle.P \xrightarrow{\bar{b}\langle a\rangle} P \qquad \text{OPEN}_\pi \quad \frac{T \xrightarrow{\bar{b}\langle a\rangle} T'}{\nu a T \xrightarrow{\bar{b}\langle \nu a\rangle} T'}$$

$$\text{NEW}_\pi \quad \frac{T \xrightarrow{\alpha} T'}{\nu a T \xrightarrow{\alpha} \nu a T'} \text{ if } a \notin \alpha \qquad \text{COM}_\pi \quad \frac{T \xrightarrow{\bar{b}\langle a\rangle} T' \qquad U \xrightarrow{b(c)} U'}{T \mid U \xrightarrow{\tau} T' \mid U'_{\{a/c\}}}$$

$$\text{CLOSE}_\pi \quad \frac{T \xrightarrow{\bar{b}\langle \nu a\rangle} T' \qquad U \xrightarrow{b(c)} U'}{T \mid U \xrightarrow{\tau} T' \mid U'_{\{a/c\}}} \text{ if } a \notin \text{fn}(U)/\{c\}$$

$$\text{PAR}_\pi \quad \frac{T \xrightarrow{\alpha} T'}{T \mid U \xrightarrow{\alpha} T' \mid U} \text{ if } \text{bn}(\alpha) \cap \text{fn}(U) = \emptyset$$

Proof of Lemma **??***:*

1) If $T \xrightarrow{\gamma} U$ and $\phi(T) = P$ then there exists Q such that $P \xrightarrow{\gamma} Q$ and $\phi(U) = Q$.
2) If $P \xrightarrow{\gamma}_\pi Q$ then for all T such that $\phi(T) = P$ there exists U with $T \xrightarrow{\gamma} U$ and $\phi(U) = Q$.

Proof:

1) By induction on the derivation tree of the transition $T \xrightarrow{\gamma} U$:
   - $\xi \cdot (b(c).P) \xrightarrow{\xi[b(c)]} [\star/c] :: \xi \cdot P$. Let $\phi(\xi \cdot (b(c).P)) = (b(c).P)\{\tilde{x}/\tilde{y}\} = b'(c).P'$, which

can do a transition on $b'(c)$, with $\xi(b) = b'$. We have that $b'(c).P' + Q' \xrightarrow{b'(c)} P'$ and we want to show that $\phi([\star/c] :: \xi \cdot P) = P\{\tilde{x}/\tilde{y}\}$. This follows from the definition of ϕ as $\phi(\xi \cdot P) = P\{\tilde{x}/\tilde{y}\}$ and $\phi([\star/c] :: \xi \cdot P) = \phi(\xi \cdot P) = P\{\tilde{x}/\tilde{y}\}$. Similar for the rule OUT.

- $T \xrightarrow{\gamma} T'$ with $T \equiv_l U \xrightarrow{\gamma} U' \equiv_l T'$. From $T \equiv_l U$ we deduce $\phi(T) = \phi(U)$, and similarly for $U' \equiv_l T'$. By induction, we have that $\phi(U) \xrightarrow{\gamma} \phi(U')$.
- $T \mid U \xrightarrow{\tau} T' \mid U'_{[a/c]}$. By induction, for $T \xrightarrow{\bar{b}\langle a\rangle} T'$, $U \xrightarrow{b(c)} U'_{[a/c]}$ we have that $\phi(T) = P \xrightarrow{\bar{b}\langle a\rangle} P' = \phi(T')$ and $\phi(U) = Q \xrightarrow{b(c)} Q' = \phi(U')$. Then $\phi(T \mid U) = \phi(T) \mid \phi(U) = P \mid Q$ which has the transition $P \mid Q \xrightarrow{\tau} P' \mid Q'\{a/c\}$. We have that $\phi(T' \mid U'_{[a/c]}) = P' \mid Q'\{a/c\}$, as $\phi(U'_{[a/c]}) = Q'\{a/c\}$.
- $\nu a T \xrightarrow{\gamma} \nu a U$ with $T \xrightarrow{\gamma} U$. Then by induction $\phi(T) = P \xrightarrow{\gamma} Q = \phi(U)$. We then have $\nu a P \xrightarrow{\gamma} \nu a Q$, with $\phi(\nu a U) = \nu a Q$.
  Similar for the rest of the rules.

2) By induction on the derivation tree of the transition $P \xrightarrow{\gamma}_\pi Q$:
   - $b(c).P + Q \xrightarrow{b(c)}_\pi P$. As $\phi(T) = b(c).P + Q$, we have $T = \xi \cdot b'(c).P' + Q'$ with $\xi(b') = b$ and $\phi(\xi \cdot (b'(c).P' + Q')) = (b'(c).P')\{\tilde{x}/\tilde{y}\} = b(c).P$. The transition on T is $\xi \cdot b'(c).P' \xrightarrow{b(c)} [\star/c] :: \xi \cdot P'$. We still have to show that $\phi([\star/c] :: \xi \cdot P') = P$, which results from $\phi(\xi \cdot P') = P'\{\tilde{x}/\tilde{y}\} = P$ and $\phi([\star/c] :: \xi \cdot P') = \phi(\xi \cdot P')$.
   - $P \mid Q \xrightarrow{\tau}_\pi P' \mid Q'\{a/c\}$. From $P \xrightarrow{\bar{b}\langle a\rangle}_\pi P'$, $Q \xrightarrow{b(c)}_\pi Q'$, by the induction hypothesis, it follows that $T \xrightarrow{\bar{b}\langle a\rangle} T'$, $U \xrightarrow{b(c)} U'$, with $\phi(T) = P$, $\phi(U) = Q$ and $\phi(T') = P'$, $\phi(U') = Q'$. Hence $\phi(T \mid U) = \phi(T) \mid \phi(U) = P \mid Q$, $\phi(T' \mid U'_{[a/c]}) = \phi(T') \mid \phi(U'_{[a/c]}) = P' \mid Q'\{a/c\}$. Using the rule COM_l we have $T \mid U \xrightarrow{\tau} T' \mid U'_{[a/c]}$, which concludes our proof.
     Similar for the rest of the cases.

■

Proof of Lemma 3.8:

1) If $R \xrightarrow{\gamma} S$ then $\phi(R) \xrightarrow{\gamma} \phi(S)$.
2) If $T \xrightarrow{\gamma} U$ and $\forall R$ such that $\phi(R) = T$, $\exists S$ with $R \xrightarrow{\gamma} S$ and $\phi(S) = U$.

Proof:

1) By induction on the transition $T \xrightarrow{\gamma} U$:
   - $m \rhd b(c).P \xrightarrow{(i,j,*):m[b(c)]} \langle i, *, b[\star/c]\rangle.m \rhd P$. Using the remark on ϕ, $\phi(m \rhd b(c).P) = \phi_m(m) \cdot b(c).P = \xi \cdot b(c).P$ that can do a transition $\xi \cdot b(c).P \xrightarrow{\xi[b(c)]} [\star/c] :: \xi \cdot P$. As $\phi_m(m) = \xi$ and, due to the definition of the functions m and ξ, we have $m(b) = \xi(b)$. We still have to show that $\phi(\langle i, *, b[\star/c]\rangle.m) = [\star/c] :: \xi$ which results from $\phi_m(\langle i, *, b[\star/c]\rangle.m) = [\star/c] :: \phi_m(m) = [\star/c] :: \xi$.
   - $R \parallel S \xrightarrow{(i,*,*):\tau} R' \parallel S'_{[a/c]@i}$ with $R \xrightarrow{(i,j,k):\bar{b}\langle a\rangle} R'$ and $S \xrightarrow{(i,j',k'):b(c)} S'$ (from rule COM+). From the definition of ϕ, we have $\phi(R \mid S) = \phi(R) \mid \phi(S) = T \mid U$. By induction on the hypothesis of the rule COM+ we have that $T \xrightarrow{\bar{b}\langle a\rangle} T'$ and $U \xrightarrow{b(c)} U'$ with $\phi(R') = T'$ and $\phi(S') = U'$, respectively. We apply rule COM_l and have the transition $T \mid U \xrightarrow{\tau} T' \mid U'_{[a/c]}$, where $\phi(S'_{[a/c]@i}) = U'_{[a/c]}$ follows from the definitions of the two update functions.
     Similar for the rest of the cases.

2) By induction on the transition $T \xrightarrow{\gamma} U$:
   - $\xi \cdot (b(c).P) \xrightarrow{\xi[b(c)]} [\star/c] :: \xi \cdot P$. Then $R = m \rhd b(c).P$ as $\phi(m \rhd b(c).P) = \phi_m(m) \cdot (b(c).P)$. We have that $m \rhd b(c).P \xrightarrow{(i,j,*):m[b(c)]} \langle i, *, b[\star/c]\rangle.m \rhd P$ with $\xi(b) = m(b)$. Then it remains to show that $\phi(\langle i, *, b[\star/c]\rangle.m \rhd P) = [\star/c] :: \xi \cdot P$, which follows from $\phi(\langle i, *, b[\star/c]\rangle.m \rhd P) = [\star/c] :: \phi(m \rhd P) = [\star/c] :: \phi_m(m) \cdot P = [\star/c] :: \xi \cdot P$.
   - $T \mid U \xrightarrow{\tau} T' \mid U'_{[a/c]}$ with $T \xrightarrow{\bar{b}\langle a\rangle} T'$ and $U \xrightarrow{b(c)} U'$. We have $\phi(T \mid U) = \phi(T) \mid \phi(U) = R \mid S$ and, by induction $R \xrightarrow{(i,j,k):\bar{b}\langle a\rangle} R'$ and $S \xrightarrow{(i,j',k'):b(c)} S'$ with $\phi(R') = T'$ and $\phi(S') = U'$ respectively. We use rule COM+ to obtain the transition $R \mid S \xrightarrow{(i,*,*):\tau} R' \mid S'_{[a/c]@i}$, where $\phi(S'_{[a/c]@i}) = U'_{[a/c]}$.

■

Proof of Lemma 3.10: If $\varepsilon \rhd P \longrightarrow^\star R$ then $P \longrightarrow^\star \phi(R)$.

Proof: Straightforward adaptation from the proof in [**?**]. For the trace $\varepsilon \rhd P \longrightarrow^\star R$ there is an equivalent one that is composed only of forward transitions,as we have showed in section V. Then the first part of the lemmas **??** and 3.8 apply, and we can do an induction on the length of the forward trace. ■