Comparing the expressiveness of the π-calculus and CCS

Rob van Glabbeek
Data61, CSIRO, Sydney, Australia
School of Computer Science and Engineering, University of New South Wales, Sydney, Australia
rvg@cs.stanford.edu

***Abstract*—This paper shows that the π-calculus with implicit matching is no more expressive than CCS$_\gamma$, a variant of CCS in which the result of a synchronisation of two actions is itself an action subject to relabelling or restriction, rather than the silent action τ. This is done by exhibiting a compositional translation from the π-calculus with implicit matching to CCS$_\gamma$ that is valid up to strong barbed bisimilarity.**

The full π-calculus can be similarly expressed in CCS$_\gamma$ enriched with the triggering operation of MEIJE.

I also show that these results cannot be recreated with CCS in the rôle of CCS$_\gamma$, not even up to reduction equivalence, and not even for the asynchronous π-calculus without restriction or replication.

Finally I observe that CCS cannot be encoded in the π-calculus.

I. INTRODUCTION

The π-calculus [23], [24], [22], [33] has been advertised as an "extension to the process algebra CCS" [23] adding mobility. It is widely believed that the π-calculus has features that cannot be expressed in CCS, or other *immobile* process calculi—so called in [27]—such as ACP and CSP.

> "the π-calculus has a much greater expressiveness than CCS" [Sangiorgi [32]]
>
> "Mobility – of whatever kind – is important in modern computing. It was not present in CCS or CSP, [...] but [...] the *π-calculus* [...] takes mobility of linkage as a primitive notion." [Milner [22]]

The present paper investigates this belief by formally comparing the expressive power of the π-calculus and immobile process calculi.

Following [10], [11] I define one process calculus to be at least as expressive as another up to a semantic equivalence $\sim$ iff there exists a so-called *valid translation* up to $\sim$ from the other to the one. Validity entails compositionality, and requires that each translated expression is $\sim$-equivalent to its original. This concept is parametrised by the choice of a semantic equivalence that is meaningful for both the source and the target language. Any language is as expressive as any other up to the universal relation, whereas almost no two languages are equally expressive up to the identity relation. The equivalence $\sim$ up to which a translation is valid is a measure for the quality of the translation, and thereby for the degree in which the source language can be expressed in the target.

Robert de Simone [34] showed that a wide class of process calculi, including CCS [20], CSP [6], ACP [4] and SCCS [18], are expressible up to strong bisimilarity in MEIJE [1]. In [8] I sharpened this result by eliminating the crucial rôle played by unguarded recursion in De Simone's translation, now taking aprACP$_R$ as the target language. Here aprACP$_R$ is a fragment of the language ACP of [4], enriched with relational relabelling, and using action prefixing instead of general sequential composition. It differs from CCS only in its more versatile communication format, allowing multiway synchronisation instead of merely handshaking, in the absence of a special action τ, and in the relational nature of the relabelling operator. The class of languages that can be translated to MEIJE and aprACP$_R$ are the ones whose structural operational semantics fits a format due to [34], now known as the *De Simone* format. They can be considered the "immobile process calculi" alluded to above. The π-calculus does not fit into this class—its operational semantics is not in De Simone format.

To compare the expressiveness of mobile and immobile process calculi I first of all need to select a suitable semantic equivalence that is meaningful for both kinds of languages. A canonical choice is *strong barbed bisimilarity* [26], [33]. Strong barbed bisimilarity is not a congruence for either CCS or the π-calculus, but it is used as a semantic basis for defining suitable congruences on languages [26], [33]. For CCS, the familiar notion of *strong bisimilarity* [19] arises as the congruence closure of strong barbed bisimilarity. For the π-calculus, the congruence closure of strong barbed bisimilarity yields the notion of *strong early congruence*, called *strong full bisimilarity* in [33]. In general, whatever its characterisation in a particular calculus, *strong barbed congruence* is the name of the congruence closure of strong barbed bisimilarity, and a default choice for a semantic equivalence [33].

My first research goal was to find out if there exists a translation from the π-calculus to CCS that is valid up to strong barbed bisimilarity. The answer is negative. In fact, no compositional translation of the π-calculus to CCS is possible, even when weakening the equivalence up to which it should be valid from strong barbed bisimilarity to strong reduction equivalence, and even when restricting the source language to the asynchronous π-calculus [5] without restriction and replication. This disproves a result of [3].

My next research goal was to find out if there is a translation from the π-calculus to any other immobile process calculus, and if yes, to keep the target language as close as possible

TABLE I
STRUCTURAL OPERATIONAL SEMANTICS OF CCS

$$\alpha.P \xrightarrow{\alpha} P \qquad \frac{P_j \xrightarrow{\alpha} P'_j}{\sum_{i\in I} P_i \xrightarrow{\alpha} P'_j}\ (j \in I)$$

$$\frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q} \qquad \frac{P \xrightarrow{a} P',\ Q \xrightarrow{\bar{a}} Q'}{P|Q \xrightarrow{\tau} P'|Q'} \qquad \frac{Q \xrightarrow{\alpha} Q'}{P|Q \xrightarrow{\alpha} P|Q'}$$

$$\frac{P \xrightarrow{\alpha} P'}{P\backslash L \xrightarrow{\alpha} P'\backslash L}\ (\alpha \notin L \cup \bar{L}) \qquad \frac{P \xrightarrow{\alpha} P'}{P[f] \xrightarrow{f(\alpha)} P'[f]} \qquad \frac{P \xrightarrow{\alpha} P}{A \xrightarrow{\alpha} P}\ (A \stackrel{\text{def}}{=} P)$$

to CCS. Here the answer turned out to be positive. How close the target language can be kept to CCS depends on which version of the π-calculus I take as source language. My first choice was the original π-calculus, as presented in [23], [24], as it is at least as expressive as its competitors. It turns out, however, that the matching operator $[x{=}y]P$ of [23], [24] is the source of a complication. The book [33] merely allows matching to occur as part of action prefixing, as in $[x{=}y]u(z).P$ or $[x{=}y]\bar{u}v.P$. I call this *implicit matching*. Matching was introduced in [23], [24] to facilitate complete equational axiomatisations of the π-calculus, and [33] shows that for that purpose implicit matching is sufficient.

To obtain a valid translation from the π-calculus with implicit matching (henceforth called $\pi_{\rm IM}$) to an upgraded variant of CCS, the only upgrade needed is to turn the result of a synchronisation of two actions into a visible action, subject to relabelling or restriction, rather than the silent action τ. I call this variant CCS_γ, where γ is a commutative partial binary communication function, just like in ACP [4]. CCS_γ is a fragment of aprACP_R, which also carries a parameter γ. If $\gamma(a,b) = c$, this means that an a-action of one component in a parallel composition may synchronise with a b-action of another component, into a c-action; if $\gamma(a,b)$ is undefined, the actions a and b do not synchronise. CCS can be seen as the instance of CCS_γ with $\gamma(\bar{a},a) = \tau$, and γ undefined for other pairs of actions. But as target language for my translation I will need another choice of the parameter γ.

An important feature of ACP, which greatly contributes to its expressiveness, is multiway synchronisation. This is achieved by allowing an action $\gamma(a,b)$ to synchronise with an action c into $\gamma(\gamma(a,b),c)$. This feature is not needed for the target language of my translations. So I require that $\gamma(\gamma(a,b),c)$ is always undefined.

To obtain a valid translation from the full π-calculus, with an explicit matching operator, I need to further upgrade CCS_γ with the *triggering* operator of MEIJE, which allows a relabelling of the first action of its argument only.

By a general result of [11], the validity up to strong barbed bisimilarity of my translation from $\pi_{\rm IM}$ to CCS_γ (and from π to $\text{CCS}_\gamma^{\rm trig}$) implies that it is even valid up to an equivalence on their disjoint union that on π coincides with strong barbed congruence, or strong early congruence, and on $\text{CCS}_\gamma^{\rm trig}$ is the congruence closure of strong barbed bisimilarity under translated contexts. The latter is strictly coarser than strong bisimilarity, which is the congruence closure of strong barbed bisimilarity under all $\text{CCS}_\gamma^{\rm trig}$ contexts.

Having established that $\pi_{\rm IM}$ can be expressed in CCS_γ, the possibility remains that the two languages are equally expressive. This, however, is not the case. There does not exists a valid translation (up to any reasonable equivalence) from CCS—thus neither from CCS_γ—to the π-calculus, even when disallowing the infinite sum of CCS, as well as unguarded recursion. This is a trivial consequence of the power of the CCS renaming operator, which cannot be mimicked in the π-calculus. Using a simple renaming operator that is as finite as the successor function on the natural numbers, CCS, even without infinite sum and unguarded recursion, allows the specification of a process with infinitely many weak barbs, whereas this is fundamentally impossible in the π-calculus.

II. CCS

CCS [19] is parametrised with a sets $\mathcal{K}$ of *agent identifiers* and $\mathscr{A}$ of *visible actions*. The set $\overline{\mathscr{A}}$ of *co-actions* is $\overline{\mathscr{A}} := \{\bar{a} \mid a{\in}\mathscr{A}\}$, and $\mathscr{L} := \mathscr{A} \cup \overline{\mathscr{A}}$ is the set of *labels*. The function $\bar{\cdot}$ is extended to $\mathscr{L}$ by declaring $\bar{\bar{a}} = a$. Finally, $Act := \mathscr{L} \uplus \{\tau\}$ is the set of *actions*. Below, a, b, c, ... range over $\mathscr{L}$ and α, β over Act. A *relabelling* is a function $f\colon \mathscr{L} \to \mathscr{L}$ satisfying $f(\bar{a}) = \overline{f(a)}$; it extends to Act by $f(\tau) := \tau$. The class T_{CCS} of CCS *terms*, *expressions*, *processes* or *agents* is the smallest class[1] including:

$\alpha.P$	for $\alpha \in Act$ and $P \in \text{T}_{\text{CCS}}$	*prefixing*
$\sum_{i\in I} P_i$	for I an index set and $P_i \in \text{T}_{\text{CCS}}$	*choice*
$P\|Q$	for $P, Q \in \text{T}_{\text{CCS}}$	*parallel comp.*
$P\backslash L$	for $L \subseteq \mathscr{L}$ and $P \in \text{T}_{\text{CCS}}$	*restriction*
$P[f]$	for f a relabelling and $P \in \text{T}_{\text{CCS}}$	*relabelling*
A	for $A \in \mathcal{K}$	*recursion*.

One writes $P_1 + P_2$ for $\sum_{i\in I} P_i$ when $I = \{1,2\}$, and $\mathbf{0}$ when $I = \emptyset$. Each agent identifier $A \in \mathcal{K}$ comes with a unique

[1] CCS [19], [20] allows arbitrary index sets I in summations $\sum_{i\in I} P_i$. As a consequence, T_{CCS} is a proper class rather than a set. Although this is unproblematic, many computer scientists prefer the class of terms to be a set. This can be achieved by choosing a cardinal κ and requiring the index sets I to satisfy $|I| < \kappa$. To enable my translation from the π-calculus to $\text{CCS}_\gamma^{\rm trig}$, κ should exceed the size of the set of names used in the π-calculus.

defining equation of the form $A \stackrel{\text{def}}{=} P$, with $P \in \mathrm{T}_{\mathrm{CCS}}$. The semantics of CCS is given by the labelled transition relation $\rightarrow \subseteq \mathrm{T}_{\mathrm{CCS}} \times Act \times \mathrm{T}_{\mathrm{CCS}}$. The transitions $P \xrightarrow{\alpha} Q$ with $P, Q \in \mathrm{T}_{\mathrm{CCS}}$ and $\alpha \in Act$ are derived from the rules of Table I.

Arguably, the most authentic version of CCS [20] features a recursion construct instead of agent identifiers. Since there exists a straightforward valid transition from the version of CCS presented here to the one from [20], the latter is at least as expressive. Therefore, when showing that a variant of CCS is at least as expressive as the π-calculus, I obtain a stronger result by using agent identifiers.

III. CCS$_\gamma$

CCS$_\gamma$ has four parameters: the same set $\mathcal{K}$ of *agent identifiers* as for CCS, an alphabet $\mathscr{A}$ of *visible actions*, with a subset $\mathscr{S} \subseteq \mathscr{A}$ of synchronisations[2], and a partial *communication function* $\gamma : (\mathscr{A} \backslash \mathscr{S})^2 \rightharpoonup \mathscr{S} \cup \{\tau\}$, which is commutative, i.e. $\gamma(a,b) = \gamma(b,a)$ and each side of this equation is defined just when the other side is. Compared to CCS there are no co-actions, so $Act := \mathscr{A} \uplus \{\tau\}$.

The syntax of CCS$_\gamma$ is the same as that of CCS, except that parallel composition is denoted $\|$ rather than $|$, following ACP [4], [2]. This indicates a semantic difference: the rule for communication in the middle of Table I is for CCS$_\gamma$ replaced by

$$\frac{P \xrightarrow{a} P',\ Q \xrightarrow{b} Q'}{P \| Q \xrightarrow{c} P' \| Q'} \quad (\gamma(a,b) = c).$$

Moreover, relabelling operators $f : \mathscr{A} \rightarrow Act$ are allowed to rename visible actions into τ, but not vice versa.[3] They are required to satisfy $c \in \mathscr{S} \Rightarrow f(c) \in \mathscr{S} \cup \{\tau\}$. These are the only differences between CCS and CCS$_\gamma$.

IV. Strong barbed bisimilarity

The semantics of the π-calculus and CCS can be expressed by associating a labelled or a barbed transition system with these languages, with processes as states. Semantic equivalences are defined on the states of labelled or barbed transition systems, and thereby on π- and CCS processes.

Definition 1: A *labelled transition system* (LTS) is pair $(S, \rightarrow)$ with S a class (of *states*) and $\rightarrow \subseteq S \times A \times S$ a *transition relation*, for some suitable set of *actions* A.

I write $P \xrightarrow{\alpha} Q$ for $(P, \alpha, Q) \in \rightarrow$, $P \xrightarrow{\alpha}$ for $\exists Q.\ P \xrightarrow{\alpha} Q$, and $P \not\xrightarrow{\alpha}$ for its negation. The structural operational semantics of CCS presented before creates an LTS with as states all CCS processes and the transition relation derived from the operational rules, with $A := Act$.

Definition 2: A *strong bisimulation* is a symmetric relation $\mathscr{R}$ on the states of an LTS such that

- if $P \mathscr{R} Q$ and $P \xrightarrow{\alpha} P'$ then $\exists Q'.\ Q \xrightarrow{\alpha} Q' \wedge P' \mathscr{R} Q'$.

Processes P and Q are *strongly bisimilar*—notation $P \leftrightarroweq Q$—if $P \mathscr{R} Q$ for some strong bisimulation $\mathscr{R}$.

[2]These have been added solely to prevent multiway synchronisation.

[3]Renaming into τ could already be done in CCS by means of parallel composition. Hence this feature in itself does not add extra expressiveness.

TABLE II
The actions

α	Kind	$O(\alpha)$	$\mathrm{fn}(\alpha)$	$\mathrm{bn}(\alpha)$
$M\tau$	Silent	–	$\emptyset$	$\emptyset$
$M\bar{x}y$	Free output	$\bar{x}$	$\mathrm{n}(M) \cup \{x, y\}$	$\emptyset$
$M\bar{x}(y)$	Bound output	$\bar{x}$	$\mathrm{n}(M) \cup \{x\}$	$\{y\}$
Mxy	Free input	x	$\mathrm{n}(M) \cup \{x, y\}$	$\emptyset$
$Mx(y)$	Bound input	x	$\mathrm{n}(M) \cup \{x\}$	$\{y\}$

As is well-known, $\leftrightarroweq$ is an equivalence relation, and a strong bisimulation itself. Through the operational semantics of CCS$_\gamma$, strong bisimilarity is defined on CCS$_\gamma$ processes.

Definition 3: A *barbed transition system* (BTS) is a triple $(S, \mapsto, \downarrow)$ with S a class (of *states*), $\mapsto \subseteq S \times S$ a *reduction relation*, and $\downarrow \subseteq S \times B$ an *observability predicate* for some suitable set of *barbs* B.

One writes $P{\downarrow_b}$ for $P \in S$ and $b \in B$ when $(P, b) \in \downarrow$. A BTS can be extracted from an LTS with $\tau \in A$, by means of a partial observation function $O : A \rightharpoonup B$. The states remain the same, the reductions are taken to be the transitions labelled τ (dropping the label in the BTS), and $P{\downarrow_b}$ holds exactly when there is a transition $P \xrightarrow{\alpha} Q$ with $O(\alpha) = b$.

In this paper I consider labelled transition systems whose actions $\alpha \in A$ are of the forms presented in Table II. Here x and y are *names*, drawn from the disjoint union of two sets $\mathcal{Z}$ and $\mathcal{R}$ of *public* and *private* names, and M is a (possibly empty) *matching sequence*, a sequence of *matches* $[x{=}y]$ with $x, y \in \mathcal{Z} \uplus \mathcal{R}$ and $x \neq y$. The set of names occurring in M is denoted $\mathrm{n}(M)$. In Table II, also the *free names* $\mathrm{fn}(\alpha)$ and *bound names* $\mathrm{bn}(\alpha)$ of an action α are defined. The set of *names* of α is $\mathrm{n}(\alpha) := \mathrm{fn}(\alpha) \cup \mathrm{bn}(\alpha)$. Consequently, also the actions Act of my instantiation of CCS$_\gamma$ need to have the forms of Table II. For the translation into barbed transition systems I take $B := \mathcal{Z} \cup \overline{\mathcal{Z}}$, where $\overline{\mathcal{Z}} := \{\bar{a} \mid a \in \mathcal{Z}\}$, and $O(\alpha)$ as indicated in Table II, provided $M = \varepsilon$ and $O(\alpha) \in B$.

Definition 4: A *strong barbed bisimulation* is a symmetric relation $\mathscr{R}$ on the states of a BTS such that

- if $P \mathscr{R} Q$ and $P \longmapsto P'$ then $\exists Q'.\ Q \longmapsto Q' \wedge P' \mathscr{R} Q'$
- and if $P \mathscr{R} Q$ and $P{\downarrow_b}$ then also $Q{\downarrow_b}$.

Processes P and Q are *strongly barbed bisimilar*—notation $P \barbeq Q$—if $P \mathscr{R} Q$ for some strong barbed bisimulation $\mathscr{R}$.

Again, $\barbeq$ is an equivalence relation, and a strong barbed bisimulation itself. Through the above definition, strong barbed bisimilarity is defined on all LTSs occurring in this paper, as well as on my instantiation of CCS$_\gamma$. It can also be used to compare processes from different LTSs, namely by taking their disjoint union.

V. The π-calculus

The π-calculus [23], [24] is parametrised with an infinite set $\mathcal{N}$ of *names* and, for each $n \in \mathbb{N}$, a set of $\mathcal{K}_n$ of

agent identifiers of arity n. The set T_π of π-calculus *terms*, *expressions*, *processes* or *agents* is the smallest set including:

$\mathbf{0}$		*inaction*
$\tau.P$	for $P \in \mathrm{T}_\pi$	*silent prefix*
$\bar{x}y.P$	for $x, y \in \mathcal{N}$ and $P \in \mathrm{T}_\pi$	*output prefix*
$x(y).P$	for $x, y \in \mathcal{N}$ and $P \in \mathrm{T}_\pi$	*input prefix*
$(\nu y)P$	for $y \in \mathcal{N}$ and $P \in \mathrm{T}_\pi$	*restriction*
$[x{=}y]P$	for $x, y \in \mathcal{N}$ and $P \in \mathrm{T}_\pi$	*match*
$P\|Q$	for $P, Q \in \mathrm{T}_\pi$	*parallel comp.*
$P + Q$	for $P, Q \in \mathrm{T}_\pi$	*choice*
$A(y_1, ..., y_n)$	for $A \in \mathcal{K}_n$ and $y_i \in \mathcal{N}$	*defined agent*

The order of precedence among the operators is the order of the listing above. A process $\alpha.\mathbf{0}$ with $\alpha = \tau$ or $\bar{x}y$ or $x(y)$ is often written α.

$\mathrm{n}(P)$ denotes the set of all names occurring in a process P. An occurrence of a name y in a term is *bound* if it occurs in a subterm of the form $x(y).P$ or $(\nu y)P$; otherwise it is *free*. The set of names occurring free (resp. bound) in a process P is denoted $\mathrm{fn}(P)$ (resp. $\mathrm{bn}(P)$).

Each agent identifier $A \in \mathcal{K}_n$ is assumed to come with a unique *defining equation* of the form

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

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

The π-calculus with implicit matching (π_{IM}) drops the matching operator, instead allowing prefixes of the form $M\bar{x}y.P$, $Mx(y).P$ and $M\tau.P$, with M a matching sequence.

A *substitution* is a partial function $\sigma : \mathcal{N} \rightharpoonup \mathcal{N}$ such that $\mathcal{N} \backslash (dom(\sigma) \cup range(\sigma))$ is infinite. For $\vec{x} = (x_1, \ldots, x_n)$, $\vec{y} = (y_1, \ldots, y_n) \in \mathcal{N}^n$, $\{\vec{y}/\vec{x}\}$ denotes the substitution given by $\sigma(x_i) = y_i$ for $1 \leq i \leq n$. One writes $\{y/x\}$ when $n{=}1$.

For $x \in \mathcal{N}$, $x[\sigma]$ denotes $\sigma(x)$ if $x \in dom(\sigma)$ and x otherwise; $M[\sigma]$ is the result of changing each occurrence of a name x in M into $x[\sigma]$, while dropping resulting matches $[y{=}y]$.

For a substitution σ, the process $P\sigma$ is obtained from $P \in \mathrm{T}_\pi$ by simultaneous substitution, for all $x \in dom(\sigma)$, of $x[\sigma]$ for all free occurrences of x in P, with change of bound names to avoid name capture. A formal inductive definition is:

$$\begin{aligned}
\mathbf{0}\sigma &= \mathbf{0} \\
(M\tau.P)\sigma &= M[\sigma]\tau.(P\sigma) \\
(M\bar{x}y.P)\sigma &= M[\sigma]\overline{x[\sigma]}y[\sigma].(P\sigma) \\
(Mx(y).P)\sigma &= M[\sigma]x[\sigma](z).(P\{z/y\}\sigma) \\
((\nu y)P)\sigma &= (\nu z)(P\{z/y\}\sigma) \\
([x{=}y]P)\sigma &= [x[\sigma]{=}y[\sigma]](P\sigma) \\
(P|Q)\sigma &= (P\sigma)|(Q\sigma) \\
(P + Q)\sigma &= (P\sigma) + (Q\sigma) \\
A(\vec{y})\sigma &= A(\vec{y}[\sigma])
\end{aligned}$$

where z is chosen outside $\mathrm{fn}((\nu y)P) \cup dom(\sigma) \cup range(\sigma)$; in case $y \notin dom(\sigma) \cup range(\sigma)$ one always picks $z := y$.

A *congruence* is an equivalence relation $\sim$ on T_π such that $P \sim Q$ implies $\tau.P \sim \tau.Q$, $\bar{x}y.P \sim \bar{x}y.Q$, $x(y).P \sim x(y).Q$, $(\nu y)P \sim (\nu y)Q$, $[x{=}y]P \sim [x{=}y]Q$, $P|U \sim Q|U$, $U|P \sim U|Q$, $P+U \sim Q+U$ and $U+P \sim U+Q$. Let $\equiv$ be the smallest congruence on T_π allowing renaming of bound names, i.e., that satisfies $x(y).P \equiv x(z).(P\{z/y\})$ and $(\nu y)P \equiv (\nu z)(P\{z/y\})$ for any $z \notin \mathrm{fn}((\nu y)P)$. If $P \equiv Q$, then Q is obtained from P by means of α-*conversion*. Due to the choice of z above, substitution is precisely defined only up to α-conversion.

Note that $P \equiv Q$ implies that $\mathrm{fn}(P) = \mathrm{fn}(Q)$, and also that $P\sigma \equiv Q\sigma$ for any substitution σ.

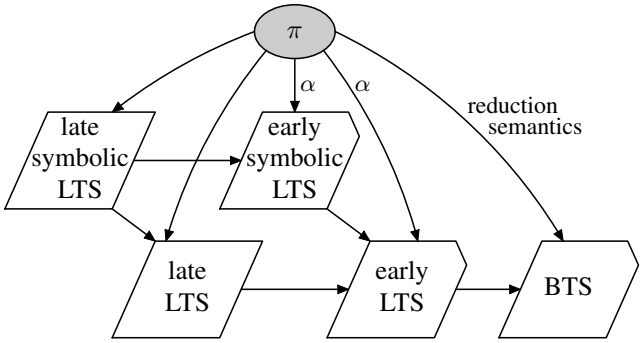


Fig. 1. Semantics of the π-calculus

VI. The semantics of the π-calculus

Whereas CCS has only one operational semantics, the π-calculus is equipped with at least five, as indicated in Figure 1. The *late* operational semantics stems from [24], the origin of the π-calculus. It is given by the action rules of Table III. These rules generate a labelled transition system in which the states are the π-calculus processes and the transitions are labelled with the actions τ, $\bar{x}y$, $\bar{x}(y)$ and $x(y)$ of Table II (always with M the empty string). Here I take $\mathcal{Z} := \mathcal{N}$ and $\mathcal{R} := \emptyset$.

For π_{IM}, rule MATCH is omitted. A process $[x{=}y]\alpha.P$ has no outgoing transitions, similar to $\mathbf{0}$.

In [24] the *late* and *early* bisimulation semantics of the π-calculus were proposed.

Definition 5: A *late bisimulation* is a symmetric relation $\mathscr{R}$ on π-processes such that, whenever $P \mathscr{R} Q$, α is either τ or $\bar{x}y$ and $z \notin \mathrm{n}(P) \cup \mathrm{n}(Q)$,

1) if $P \xrightarrow{\alpha} P'$ then $\exists Q'$ with $Q \xrightarrow{\alpha} Q'$ and $P' \mathscr{R} Q'$,
2) if $P \xrightarrow{x(z)} P'$ then $\exists Q' \forall y$. $Q \xrightarrow{x(z)} Q' \wedge P'\{y/z\} \mathscr{R} Q'\{y/z\}$,
3) if $P \xrightarrow{\bar{x}(z)} P'$ then $\exists Q'$ with $Q \xrightarrow{\bar{x}(z)} Q'$ and $P' \mathscr{R} Q'$.

Processes P and Q are *late bisimilar*—notation $P \dot{\sim}_L Q$—if $P \mathscr{R} Q$ for some late bisimulation $\mathscr{R}$. They are *late congruent*—notation $P \sim_L Q$—if $P\{\vec{y}/\vec{x}\} \dot{\sim}_L Q\{\vec{y}/\vec{x}\}$ for any substitution $\{\vec{y}/\vec{x}\}$.

Early bisimilarity ($\dot{\sim}_E$) and congruence ($\sim_E$) are defined likewise, but with $\forall y \exists Q'$ instead of $\exists Q' \forall y$. In [24], [33] it is shown that $\dot{\sim}_L$ and $\dot{\sim}_E$ are congruences for all operators of the π-calculus, except for the input prefix. $\sim_E$ and $\sim_L$ are congruence relations for the entire language; in fact they are the congruence closures of $\dot{\sim}_L$ and $\dot{\sim}_E$, respectively. By definition, $\dot{\sim}_L \subseteq \dot{\sim}_E$, and thus $\sim_L \subseteq \sim_E$.

Lemma 1 ([24]): Let $P \equiv Q$ and $\mathrm{bn}(\alpha) \cap \mathrm{n}(Q) = \emptyset$. If $P \xrightarrow{\alpha} P'$ then $Q \xrightarrow{\alpha} Q'$ for some Q' with $P' \equiv Q'$.

This implies that $\equiv$ is a late bisimulation, so that $\equiv \subset \sim_L$.

TABLE III
LATE STRUCTURAL OPERATIONAL SEMANTICS OF THE π-CALCULUS

TAU: $\tau.P \xrightarrow{\tau} P$

OUTPUT: $\bar{x}y.P \xrightarrow{\bar{x}y} P$

INPUT: $x(y).P \xrightarrow{x(z)} P\{z/y\} \quad (z \notin \mathrm{fn}((\nu y)P))$

SUM: $\dfrac{P \xrightarrow{\alpha} P'}{P+Q \xrightarrow{\alpha} P'}$

MATCH: $\dfrac{P \xrightarrow{\alpha} P'}{[x{=}x]P \xrightarrow{\alpha} P'}$

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

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

COM: $\dfrac{P \xrightarrow{\bar{x}y} P',\ Q \xrightarrow{x(z)} Q'}{P|Q \xrightarrow{\tau} P'|Q'\{y/z\}}$

CLOSE: $\dfrac{P \xrightarrow{\bar{x}(z)} P',\ Q \xrightarrow{x(z)} Q'}{P|Q \xrightarrow{\tau} (\nu z)(P'|Q')}$

RES: $\dfrac{P \xrightarrow{\alpha} P'}{(\nu y)P \xrightarrow{\alpha} (\nu y)P'} \quad (y \notin \mathrm{n}(\alpha))$

ALPHA-OPEN: $\dfrac{P \xrightarrow{\bar{x}y} P'}{(\nu y)P \xrightarrow{\bar{x}(z)} P'\{z/y\}} \quad \begin{pmatrix} y \neq x \\ z \notin \mathrm{fn}((\nu y)P') \end{pmatrix}$

The rules **SUM**, **PAR**, **COM** and **CLOSE** additionally have symmetric forms, with the rôles of P and Q exchanged.

TABLE IV
EARLY STRUCTURAL OPERATIONAL SEMANTICS OF THE π-CALCULUS

TAU: $\tau.P \xrightarrow{\tau} P$

OUTPUT: $\bar{x}y.P \xrightarrow{\bar{x}y} P$

EARLY-INPUT: $x(y).P \xrightarrow{xz} P\{z/y\}$

SUM: $\dfrac{P \xrightarrow{\alpha} P'}{P+Q \xrightarrow{\alpha} P'}$

MATCH: $\dfrac{P \xrightarrow{\alpha} P'}{[x{=}x]P \xrightarrow{\alpha} P'}$

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

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

EARLY-COM: $\dfrac{P \xrightarrow{\bar{x}y} P',\ Q \xrightarrow{xy} Q'}{P|Q \xrightarrow{\tau} P'|Q'}$

EARLY-CLOSE: $\dfrac{P \xrightarrow{\bar{x}(z)} P',\ Q \xrightarrow{xz} Q'}{P|Q \xrightarrow{\tau} (\nu z)(P'|Q')} \quad (z \notin \mathrm{fn}(Q))$

RES: $\dfrac{P \xrightarrow{\alpha} P'}{(\nu y)P \xrightarrow{\alpha} (\nu y)P'} \quad (y \notin \mathrm{n}(\alpha))$

OPEN: $\dfrac{P \xrightarrow{\bar{x}y} P'}{(\nu y)P \xrightarrow{\bar{x}(y)} P'} \quad (y \neq x)$

ALPHA: $\dfrac{P \equiv Q, \quad Q \xrightarrow{\alpha} Q'}{P \xrightarrow{\alpha} Q'}$

In [25] the *early* operational semantics of the π-calculus is proposed, presented in Table IV; it uses free input actions xy instead of bound inputs $x(y)$. This is also the semantics of [33]. The semantics in [25], [33] requires us to identify processes modulo α-conversion before applying the operational rules. This is equivalent to adding rule **ALPHA** of Table IV.

A variant of the late operational semantics incorporating rule **ALPHA** is also possible. In this setting rule **ALPHA-OPEN** can be simplified to **OPEN**, and likewise **INPUT** to $x(y).P \xrightarrow{x(y)} P$. By Lemma 1, the late operational semantics with **ALPHA** gives rise to the same notions of early and late bisimilarity as the late operational semantics without **ALPHA**; the addition of this rule is entirely optional. Interestingly, the rule **ALPHA** is not optional in the early operational semantics, not even when reinstating **ALPHA-OPEN**.

Example 1: Consider the process $P := \bar{x}y|(\nu y)(x(z))$. One has $(\nu y)(x(z)) \xrightarrow{x(z)}_L (\nu y)\mathbf{0}$ and thus $P \xrightarrow{\tau}_L \mathbf{0}|(\nu y)\mathbf{0}$ by **COM**. However, $(\nu y)(x(z)) \xrightarrow{xy}_E (\nu y)\mathbf{0}$ is forbidden by the side condition of **RES**, so in the early semantics without **ALPHA** process P cannot make a τ-step. Rule **ALPHA** comes to the rescue here, as it allows $P \equiv \bar{x}y|(\nu w)(x(z)) \xrightarrow{\tau}_E \mathbf{0}|(\nu w)\mathbf{0}$.

By the following lemma, the early transition relation $\longrightarrow_E$ is completely determined by the late transition relation $\longrightarrow_{\alpha L}$ with **ALPHA**:

Lemma 2 ([25]): Let $P \in \mathrm{T}_\pi$ and β be τ, $\bar{x}y$ or $\bar{x}(y)$.

- $P \xrightarrow{\beta}_E Q$ iff $P \xrightarrow{\beta}_{\alpha L} Q$.
- $P \xrightarrow{xy}_E Q$ iff $P \xrightarrow{x(z)}_{\alpha L} R$ for some R, z with $Q \equiv R\{y/z\}$.

The early transition relations allow a more concise definition of early bisimilarity:

Proposition 1 ([25]): An *early bisimulation* is a symmetric relation $\mathscr{R}$ on T_π such that, whenever $P \mathscr{R} Q$ and α is an action with $\mathrm{bn}(\alpha) \cap (\mathrm{n}(P) \cup \mathrm{n}(Q)) = \emptyset$,

- if $P \xrightarrow{\alpha}_E P'$ then $\exists Q'$ with $Q \xrightarrow{\alpha}_E Q'$ and $P' \mathscr{R} Q'$.

Processes P and Q are early bisimilar iff $P \mathscr{R} Q$ for some early bisimulation $\mathscr{R}$.

TABLE V
LATE SYMBOLIC STRUCTURAL OPERATIONAL SEMANTICS OF THE π-CALCULUS

TAU: $M\tau.P \xrightarrow{M\tau} P$

OUTPUT: $M\bar{x}y.P \xrightarrow{M\bar{x}y} P$

INPUT: $Mx(y).P \xrightarrow{Mx(z)} P\{z/y\} \quad (z \notin \mathrm{fn}((\nu y)P))$

SUM: $$\frac{P \xrightarrow{\alpha} P'}{P+Q \xrightarrow{\alpha} P'}$$

SYMB-MATCH: $$\frac{P \xrightarrow{\alpha} P'}{[x{=}y]P \xrightarrow{[x=y]\alpha} P'}$$

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

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

SYMB-COM: $$\frac{P \xrightarrow{M\bar{x}y} P', \ Q \xrightarrow{Nv(z)} Q'}{P|Q \xrightarrow{[x=v]MN\tau} P'|Q'\{y/z\}}$$

SYMB-CLOSE: $$\frac{P \xrightarrow{M\bar{x}(z)} P', \ Q \xrightarrow{Nv(z)} Q'}{P|Q \xrightarrow{[x=v]MN\tau} (\nu z)(P'|Q')}$$

RES: $$\frac{P \xrightarrow{\alpha} P'}{(\nu y)P \xrightarrow{\alpha} (\nu y)P'} \quad (y \notin \mathrm{n}(\alpha))$$

SYMB-ALPHA-OPEN: $$\frac{P \xrightarrow{M\bar{x}y} P'}{(\nu y)P \xrightarrow{M\bar{x}(z)} P'\{z/y\}} \quad \begin{pmatrix} y \neq x \\ z \notin \mathrm{fn}((\nu y)P') \\ y \notin \mathrm{n}(M) \end{pmatrix}$$

For the π-calculus, the blue Ms are omitted; for π_{IM} the purple rules.

TABLE VI
EARLY SYMBOLIC STRUCTURAL OPERATIONAL SEMANTICS OF THE π-CALCULUS

TAU: $M\tau.P \xrightarrow{M\tau} P$

OUTPUT: $M\bar{x}y.P \xrightarrow{M\bar{x}y} P$

EARLY-INPUT: $Mx(y).P \xrightarrow{Mxz} P\{z/y\}$

SUM: $$\frac{P \xrightarrow{\alpha} P'}{P+Q \xrightarrow{\alpha} P'}$$

SYMB-MATCH: $$\frac{P \xrightarrow{\alpha} P'}{[x{=}y]P \xrightarrow{[x=y]\alpha} P'}$$

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

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

E-S-COM: $$\frac{P \xrightarrow{M\bar{x}y} P', \ Q \xrightarrow{Nvy} Q'}{P|Q \xrightarrow{[x=v]MN\tau} P'|Q'}$$

E-S-CLOSE: $$\frac{P \xrightarrow{M\bar{x}(z)} P', \ Q \xrightarrow{Nvz} Q'}{P|Q \xrightarrow{[x=v]MN\tau} (\nu z)(P'|Q')} \quad (z \notin \mathrm{fn}(Q))$$

RES: $$\frac{P \xrightarrow{\alpha} P'}{(\nu y)P \xrightarrow{\alpha} (\nu y)P'} \quad (y \notin \mathrm{n}(\alpha))$$

SYMB-OPEN: $$\frac{P \xrightarrow{M\bar{x}y} P'}{(\nu y)P \xrightarrow{M\bar{x}(y)} P'} \quad \begin{pmatrix} y \neq x \\ y \notin \mathrm{n}(M) \end{pmatrix}$$

ALPHA: $$\frac{P \equiv Q, \quad Q \xrightarrow{\alpha} Q'}{P \xrightarrow{\alpha} Q'}$$

Through the general method of Section IV, taking $\mathcal{Z} := \mathcal{N}$ and $\mathcal{R} := \emptyset$, a barbed transition system can be extracted from the late or early labelled transition system of the π-calculus; by Lemmas 1 and 2 the same BTS is obtained either way. This defines strong barbed bisimilarity $\dot{\sim}$ on T_π. The congruence closure of $\dot{\sim}$ is early congruence [33]. In [21] a *reduction semantics* of the π-calculus is given, that yields a BTS right away. Up to strong barbed bisimilarity, this BTS is the same as the one extracted from the late or early LTS.

In [32] yet another operational semantics of the π-calculus was introduced, in a style called *symbolic* by Hennessy & Lin [16], who had proposed it for a version of value-passing CCS. It is presented in Table V. The transitions are labelled with actions α of the form $M\beta$, where M is a matching sequence and β an action as in the late operational semantics. When $x \neq y$ the matching sequence M prepended with $[x{=}y]$ is denoted $[x{=}y]M$; however, $[x{=}x]M$ simply denotes M.

In the operational semantics of CCS, τ-actions can be thought of as reactions that actually take place, whereas a transition labelled a merely represents the potential of a reaction with the environment, one that can take place only if the environment offers a complementary transition $\bar{a}$. In case the environment never does an $\bar{a}$, this potential will not be realised. A reduction semantics (as in [22]) yields a BTS that only represents directly the realised actions—the τ-transitions or *reductions*—and reasons about the potential reactions by defining the semantics of a system in terms of reductions that can happen when placing the system in various contexts. An LTS, on the other hand, directly represents transitions that could happen under some conditions only, annotated with the conditions that enable them. For CCS, this annotation is the label a, saying that the transition is conditional on an $\bar{a}$-signal from the environment. As a result of this, semantic equivalences defined on labelled transitions systems tend to be congruences for most operators right away, and do not need much closure under contexts.

Seen from this perspective, the operational semantics of the π-calculus of Table III or IV is a compromise between a pure reduction semantics and a pure labelled transition system semantics. Input and output actions are explicitly included to

signal potential reactions that are realised in the presence of a suitable communication partner, but actions whose occurrence is conditional on two different names x and y denoting the same channel are entirely omitted, even though any π-process can be placed in a context in which x and y will be identified. As a consequence of this, the early and late bisimilarities need to be closed under all possible substitutions or identifications of names before they turn into early and late congruences. The operational semantics of Table V adds the conditional transitions that where missing in Table III, and hence can be seen as a true labelled transition system semantics.

In this paper I need the early symbolic operational semantics of the π-calculus, presented in Table VI. Although new, it is the logical combination of the early and the (late) symbolic semantics. Its transitions that are labelled with actions having an empty matching sequence are exactly the transitions of the early semantics, so the BTS extracted from this semantics is the same.

For π_{IM}, rule **SYMB-MATCH** is omitted, but **TAU**, **OUTPUT** and **INPUT** carry the matching sequence M (indicated in blue).

VII. Valid translations

A *signature* Σ is a set of *operator symbols* g, each of which is equipped with an *arity* $n \in \mathbb{N}$. The set T_Σ of *closed terms* over Σ is the smallest set such that, for all $g \in \Sigma$,

$$P_1, \ldots, P_n \in \mathrm{T}_\Sigma \quad \Rightarrow \quad g(P_1, \ldots, P_n) \in \mathrm{T}_\Sigma \,.$$

Call a language *simple* if its expressions are the closed terms T_Σ over some signature Σ. The π-calculus is simple in this sense; its signature consists of the binary operators $+$ and $|$, the unary operators τ, $\bar{x}y.$, $x(y).$, (νy) and $[x{=}y]$ for $x, y \in \mathcal{N}$, and the nullary operators (or *constants*) $\mathbf{0}$ and $A(y_1, \ldots, y_n)$ for $A \in \mathcal{K}_n$ and $y_i \in \mathcal{N}$. CCS is not quite simple, since it features the infinite choice operator.

Let $\mathcal{L}$ be a language. An n-ary $\mathcal{L}$-context C is an $\mathcal{L}$-expression that may contain special *variables* $X_1, \ldots, X_n$—its *holes*. For C an n-ary context, $C[P_1, \ldots, P_n]$ is the result of substituting P_i for X_i, for each $i = 1, \ldots, n$.

Definition 6: Let $\mathcal{L}'$ and $\mathcal{L}$ languages, generating sets of closed terms $\mathrm{T}_{\mathcal{L}'}$ and $\mathrm{T}_{\mathcal{L}}$. Let $\mathcal{L}'$ be simple, with signature Σ. A *translation* from $\mathcal{L}'$ to $\mathcal{L}$ (or an *encoding* from $\mathcal{L}'$ into $\mathcal{L}$) is a function $\mathscr{T} : \mathrm{T}_{\mathcal{L}'} \to \mathrm{T}_{\mathcal{L}}$. It is *compositional* if for each n-ary operator $g \in \Sigma$ there exists an n-ary $\mathcal{L}$-context C_g such that $\mathscr{T}(g(P_1, \ldots, P_n)) = C_g[\mathscr{T}(P_1), \ldots, \mathscr{T}(P_n)]$.

Let $\sim$ be an equivalence relation on $\mathrm{T}_{\mathcal{L}'} \cup \mathrm{T}_{\mathcal{L}}$. A translation $\mathscr{T}$ from $\mathcal{L}'$ to $\mathcal{L}$ is *valid up to* $\sim$ if it is compositional and $\mathscr{T}(P) \sim P$ for each $P \in \mathrm{T}_{\mathcal{L}'}$.

The above definition stems in essence from [10], [11], but could be simplified here since [10], [11] also covered the case that $\mathcal{L}'$ is not simple. Moreover, here I restrict attention to what are called *closed term languages* in [11].

VIII. The unencodability of π into CCS

In this section I show that there exists no translation of the π-calculus to CCS that is valid up to $\mathrel{\mathring{\sim}}$. I even show this for the fragment $\pi_A^{\P}$ of the (asynchronous) π-calculus without choice, recursion, matching and restriction (thus only featuring inaction, action prefixing and parallel composition).

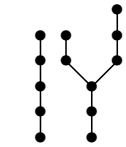
Definition 7: Strong reduction bisimilarity, $\leftrightarrows_r$, is defined just as strong barbed equivalence in Definition 4, but without the requirement on barbs.

I show that there is no translation of $\pi_A^{\P}$ to CCS that is valid up to $\leftrightarrows_r$. As $\leftrightarrows_r$ is coarser than $\mathrel{\mathring{\sim}}$, this implies my claim above. It may be useful to read this section in parallel with the first half of Section XIV.

Definition 8: Let $\leftharpoondown$ be the smallest preorder on CCS contexts such that $\sum_{i\in I} E_i \leftharpoondown E_j$ for all $j \in I$, $E|F \leftharpoondown E$, $E|F \leftharpoondown F$, $E\backslash L \leftharpoondown E$, $E[f] \leftharpoondown E$ and $A \leftharpoondown P$ for all $A \in \mathcal{K}$ with $A \stackrel{\text{def}}{=} P$. A variable X occurs *unguarded* in a context E if $E \leftharpoondown X$.

If the hole X_1 occurs unguarded in the unary context $E[\ ]$ and $U \xrightarrow{\tau}$ (resp. $U \xrightarrow{\tau}\xrightarrow{\tau}$) then $E[U] \xrightarrow{\tau}$ (resp. $E[U] \xrightarrow{\tau}\xrightarrow{\tau}$).

Lemma 3: Let $E[\ ]$ be a unary and $C[\ ,\ ]$ a binary CCS context, and $P, Q, P', Q', U \in \mathrm{T}_{\mathrm{CCS}}$. If $E[C[P,Q]] \xrightarrow{\tau}$ and $U \xrightarrow{\tau}$ but neither $E[C[P',Q]] \xrightarrow{\tau}$ nor $E[C[P,Q']] \xrightarrow{\tau}$ nor $E[U] \xrightarrow{\tau}\xrightarrow{\tau}$, then $C[P,Q] \xrightarrow{\tau}$.


CCS proof trees

Proof. Since the only rule in the operational semantics of CCS with multiple premises has a conclusion labelled τ, it can occur at most once in the derivation of a CCS transition. Thus, such a derivation is a tree with at most two branches, as illustrated at the right. Now consider the derivation of $E[C[P,Q]] \xrightarrow{\tau}$. If none of its branches prods into the subprocess P, the transition would be independent on what is substituted here, thus yielding $E[C[P',Q]] \xrightarrow{\tau}$. Thus, by symmetry, both P and Q are visited by branches of this proof. It suffices to show that these branches come together within the context C, as this implies $C[P,Q] \xrightarrow{\tau}$. So suppose, towards a contradiction, that the two branches come together in E. Then E must have the form $E_1[E_2[\ ]|E_3[\ ]]$, where the hole X_1 occurs unguarded in E_2, E_3 as well as E_1. But in that case $E[U] \xrightarrow{\tau}\xrightarrow{\tau}$, contradicting the assumptions. ∎

Lemma 4: If $D[\ ,\ ,\ ]$ is a ternary CCS context, $P_1, P_2, P_3 \in \mathrm{T}_{\mathrm{CCS}}$, and $D[P_1,P_2,P_3] \xrightarrow{\tau}$, then there exists an $i \in \{1,2,3\}$ and a CCS context $E[\ ]$ such that $D'[P] \xrightarrow{\tau} E[P]$ for any $P \in \mathrm{T}_{\mathrm{CCS}}$. Here D' is the unary context obtained from $D[\ ,\ ,\ ]$ by substituting P_j for the hole X_j, for all $j \in \{1,2,3\}$, $j \neq i$.

Proof. Since the derivation of $D[P_1,P_2,P_3] \xrightarrow{\tau}$ has at most two branches, one of the P_i is not involved in this proof at all. Thus, the derivation remains valid if any other process P is substituted in the place of that P_i; the target of the transition remains the same, except for P taking the place of P_i in it. ∎

Theorem 1: There is no translation from $\pi_A^{\P}$ to CCS that is valid up to $\leftrightarrows_r$.

Proof. Suppose, towards a contradiction, that $\mathscr{T}$ is a translation from $\pi_A^{\P}$ to CCS that is valid up to $\leftrightarrows_r$. By definition, this means that $\mathscr{T}$ is compositional and that $\mathscr{T}(P) \leftrightarrows_r P$ for any $\pi_A^{\P}$-process P.

As $\mathscr{T}$ is compositional, there exists a ternary CCS context $D[\ ,\ ,\ ]$ such that, for any $\pi_A^{\P}$-processes R, S, T,

$$\mathscr{T}\big(\bar{x}v \mid x(y).(R|S|T)\big) = D[\mathscr{T}(R), \mathscr{T}(S), \mathscr{T}(T)].$$

Since $\bar{x}v|x(y).(\mathbf{0}|\mathbf{0}|\mathbf{0}) \xrightarrow{\tau}$ as well as $\mathscr{T}\big(\bar{x}v|x(y).(\mathbf{0}|\mathbf{0}|\mathbf{0})\big) \leftrightarrows_r \bar{x}v|x(y).(\mathbf{0}|\mathbf{0}|\mathbf{0})$, it follows that $\mathscr{T}\big(\bar{x}v|x(y).(\mathbf{0}|\mathbf{0}|\mathbf{0})\big) \xrightarrow{\tau}$, i.e., $D[\mathscr{T}(\mathbf{0}), \mathscr{T}(\mathbf{0}), \mathscr{T}(\mathbf{0})] \xrightarrow{\tau}$. Hence Lemma 4 can be applied. For simplicity I assume that $i=1$; the other two cases proceed in the same way. So there is a CCS context $E[\ ]$ such that $D[P, \mathscr{T}(\mathbf{0}), \mathscr{T}(\mathbf{0})] \xrightarrow{\tau} E[P]$ for all CCS terms P. In particular, $\mathscr{T}\big((\bar{x}v|x(y).(R|\mathbf{0}|\mathbf{0})\big) = D[\mathscr{T}(R), \mathscr{T}(\mathbf{0}), \mathscr{T}(\mathbf{0})] \xrightarrow{\tau} E[\mathscr{T}(R)]$ for all $\pi_A^{\P}$-processes R. (1)

I examine the translations of the π-calculus expressions $\bar{x}v|x(y).(R|\mathbf{0}|\mathbf{0})$, for $R \in \{\bar{y}z|v(w), 0|v(w), \bar{y}z|0, \tau\}$.

Since $\bar{x}v|x(y).(\bar{y}z|v(w)|\mathbf{0}|\mathbf{0}) \xrightarrow{\tau}\xrightarrow{\tau}$ and $\mathscr{T}$ respects $\leftrightarrows_r$,

$$\mathscr{T}\big(\bar{x}v|x(y).(\bar{y}z|v(w)|\mathbf{0}|\mathbf{0})\big) \xrightarrow{\tau}\xrightarrow{\tau}$$

In the same way, neither $\mathscr{T}\big(\bar{x}v|x(y).(0|v(w)|\mathbf{0}|\mathbf{0})\big) \xrightarrow{\tau}\xrightarrow{\tau}$ nor $\mathscr{T}\big(\bar{x}v|x(y).(\bar{y}z|0|\mathbf{0}|\mathbf{0})\big) \xrightarrow{\tau}\xrightarrow{\tau}$. (2)

Furthermore, since $\mathscr{T}$ respects $\leftrightarrows_r$ and there is no $S \in \mathrm{T}_\pi$ such that $\bar{x}v|x(y).(\bar{y}z|v(w)|\mathbf{0}|\mathbf{0}) \xrightarrow{\tau} S \not\xrightarrow{\tau}$, there is no $S \in \mathrm{T}_{\mathrm{CCS}}$ with $\mathscr{T}\big(\bar{x}v|x(y).(\bar{y}z|v(w)|\mathbf{0}|\mathbf{0})\big) \xrightarrow{\tau} S \not\xrightarrow{\tau}$. (3)

By (1) and (3), $E[\mathscr{T}(\bar{y}z|v(w))] \xrightarrow{\tau}$.

By (1) and (2), $E[\mathscr{T}(0|v(w))] \not\xrightarrow{\tau}$ and $E[\mathscr{T}(\bar{y}z|0)] \not\xrightarrow{\tau}$.

Since $\mathscr{T}$ is compositional, there is a binary CCS context $C_|[\ ,\ ]$ such that $\mathscr{T}(P|Q) = C_|[\mathscr{T}(P), \mathscr{T}(Q)]$ for any $P, Q \in \mathrm{T}_\pi$. It follows that

$$E[C_|[\mathscr{T}(\bar{y}z), \mathscr{T}(v(w))]] \xrightarrow{\tau}$$
$$E[C_|[\mathscr{T}(\mathbf{0}), \mathscr{T}(v(w))]] \not\xrightarrow{\tau}$$
$$E[C_|[\mathscr{T}(\bar{y}z), \mathscr{T}(\mathbf{0})]] \not\xrightarrow{\tau}.$$

Moreover since $\tau \xrightarrow{\tau}$, also $U := \mathscr{T}(\tau) \xrightarrow{\tau}$, but, since it is not the case that $\bar{x}v|x(y).(\tau|\mathbf{0}|\mathbf{0}) \xrightarrow{\tau}\xrightarrow{\tau}\xrightarrow{\tau}$, neither holds $\mathscr{T}\big(\bar{x}v|x(y).(\tau|\mathbf{0}|\mathbf{0})\big) \xrightarrow{\tau}\xrightarrow{\tau}\xrightarrow{\tau}$, and neither $E[U] \xrightarrow{\tau}\xrightarrow{\tau}$. So by Lemma 3, $\mathscr{T}(\bar{y}z|v(w)) = C_|[\mathscr{T}(\bar{y}z), \mathscr{T}(v(w))] \xrightarrow{\tau}$, yet $\bar{y}z|v(w) \not\xrightarrow{\tau}$. This contradicts the validity of $\mathscr{T}$ up to $\leftrightarrows_r$. ∎

IX. A valid translation of π_{IM} into CCS_γ

Given a set $\mathcal{N}$ of names, I now define the parameters $\mathcal{K}$, $\mathscr{A}$ and γ of the language CCS_γ that will be the target of my encoding. First of all, $\mathcal{K}$ will be the disjoint union of all the sets $\mathcal{K}_n$ for $n \in \mathbb{N}$, of n-ary agent identifiers from the chosen instance of the π-calculus.

Take $p \notin \mathcal{N}$. Let $\mathcal{R}_0 := \{\varsigma p \mid \varsigma \in \{e, \ell, r\}^*\}$. The set $\mathcal{R}$ of *private names* is $\{u^\upsilon \mid u \in \mathcal{R}_0 \wedge \upsilon \in \{'\}^*\}$. Let $\mathcal{S} = \{s_1, s_2, \ldots\}$ be an infinite set of *spare names*, disjoint from $\mathcal{N}$ and $\mathcal{R}$. Let $\mathcal{Z} := \mathcal{N} \uplus \mathcal{S}$ and $\mathcal{H} := \mathcal{Z} \uplus \mathcal{R}$.[4]

[4]The names in $\mathcal{S}$ and in $\mathcal{R}\backslash\mathcal{R}_0$ exist solely to make the substitutions $\{\vec{y}/\vec{x}\}^{\mathcal{S}}$, η and p_y surjective. Here σ is surjective iff $dom(\sigma) \subseteq range(\sigma)$.

I take Act to be the set of all expressions α from Table II, as defined in Section IV (in terms of $\mathcal{Z}$ and $\mathcal{R}$), so $\mathscr{A} := Act\backslash\{\tau\}$. The communication function γ is given by $\gamma(M\bar{x}y, Nvy) = [x{=}v]MN\tau$, just as for rule **E-S-COM** in Table VI.

For $\vec{x} = (x_1, \ldots, x_n) \in \mathcal{N}^n$ and $\vec{y} = (y_1, \ldots, y_n) \in \mathcal{H}^n$, with the x_i distinct, let $\{\vec{y}/\vec{x}\}^{\mathcal{S}} : \mathcal{S} \cup \{x_1, \ldots, x_n\} \rightharpoonup \mathcal{H}$ be the substitution σ with $\sigma(x_i) = y_i$ and $\sigma(s_i) = x_i$ for $i = 1, ..., n$, and $\sigma(s_i) = s_{i-n}$ for $i > n$. These functions extend homomorphically to $\mathscr{A}$ and thereby constitute CCS_γ relabellings. Abbreviate $[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]$ by $[\vec{y}/\vec{x}]$ and $[\{z/y\}^{\mathcal{S}}]$ by $[z/y]$.

For $\eta \in \{\ell, r, e\}$ and $y \in \mathcal{Z}$, let the surjective substitutions $\eta : \mathcal{R} \rightharpoonup \mathcal{R}$ and $p_y : \{y\} \cup \mathcal{R} \to \{y\} \cup \mathcal{R}$ be given by:

$$\begin{array}{llll} \eta(\varsigma p) := {}^{\eta\varsigma}p & & p_y(y) := p & p_y(p') := y \\ \eta({}^{\varsigma}p^{\upsilon\prime}) := {}^{\varsigma}p^{\upsilon} & \text{if } \varsigma \neq \eta\zeta & p_y(u) := e(u) & \text{if } u \neq y, p' \end{array}$$

These $\sigma : \mathcal{H} \rightharpoonup \mathcal{H}$ are injective, i.e., $x[\sigma] \neq y[\sigma]$ when $x \neq y$. Also they yield CCS_γ relabellings. The following compositional encoding, which will be illustrated with examples in Section XII, defines my translation from π_{IM} to CCS_γ.

$$\begin{array}{lll} \mathscr{T}(\mathbf{0}) & := & \mathbf{0} \\ \mathscr{T}(M\tau.P) & := & M\tau.\mathscr{T}(P) \\ \mathscr{T}(M\bar{x}y.P) & := & M\bar{x}y.\mathscr{T}(P) \\ \mathscr{T}(Mx(y).P) & := & \sum_{z\in\mathcal{H}} Mxz.\big(\mathscr{T}(P)[z/y]\big) \\ \mathscr{T}((\nu y)P) & := & \mathscr{T}(P)[p_y] \\ \mathscr{T}(P \mid Q) & := & \mathscr{T}(P)[\ell] \parallel \mathscr{T}(Q)[r] \\ \mathscr{T}(P + Q) & := & \mathscr{T}(P) + \mathscr{T}(Q) \\ \mathscr{T}(A(\vec{y})) & := & A[\vec{y}/\vec{x}] \qquad \text{when } A(\vec{x}) \stackrel{\text{def}}{=} P \end{array}$$

where the CCS_γ agent identifier A has the defining equation $A = \mathscr{T}(P)$ when $A(\vec{x}) \stackrel{\text{def}}{=} P$ was the defining equation of the agent identifier A from the π-calculus.

To explain what this encoding does, inaction, silent prefix, output prefix and choice are translated homomorphically. The input prefix is translated into an infinite sum over all possible input values z that could be received, of the received message Mxz followed by the continuation process $\mathscr{T}(P)[z/y]$. Here $[z/y]$ is a CCS relabelling operator that simulates substitution of z for y in $\mathscr{T}(P)$. This implements the rule **EARLY-INPUT** from Table VI. Agent identifiers are also translated homomorphically, except that their arguments $\vec{y}$ are replaced by relabelling operators.

Restriction is translated by simply dropping the restriction operator, but renaming the restricted name y into a private name p that generates no barbs. The operator $[p_y]$ injectively renames all private names ςp that occur in the scope of (νy) by tagging all of them with a tag e. This ensures that the new private name p is fresh, so that no name clashes can occur that in π_{IM} would have been prevented by the restriction operator.

Parallel composition is almost translated homomorphically. However, each private name on the right is tagged with an r, and on the left with an ℓ. This guarantees that private names introduced at different sides of a parallel composition cannot interact. Interaction is only possible when the name is passed on in the appropriate way.

The main result of this paper states the validity of the above translation, and thus that CCS_γ is at least as expressive as π_{IM}:

Theorem 2: For $P \in \mathrm{T}_\pi$ one has $\mathscr{T}(P) \mathrel{\rotatebox[origin=c]{180}{$\rightsquigarrow$}} P$.

See the appendix for a proof.[5] Theorem 2 says that each π-calculus process is strongly barbed bisimilar to its translation as a CCS_γ process. The labelled transition systems of the π-calculus and CCS_γ are both of the type presented in Section IV, i.e. with transition labels taken from Table II. There also the associated barbs are defined. By Theorem 2 each π transition $P \xrightarrow{\tau} P'$ can be matched by a CCS_γ transition $\mathscr{T}(P) \xrightarrow{\tau} Q$ with $\mathscr{T}(P') \mathrel{\rotatebox[origin=c]{180}{$\rightsquigarrow$}} Q$. Likewise, each CCS_γ transition $\mathscr{T}(P) \xrightarrow{\tau} Q$ can be matched by a π transition $P \xrightarrow{\tau} P'$ with $\mathscr{T}(P') \mathrel{\rotatebox[origin=c]{180}{$\rightsquigarrow$}} Q$. Moreover, if P has a barb x (or $\bar{x}$) then so does $\mathscr{T}(P)$, and vice versa. Here a π or CCS_γ process P has a barb $a \in \mathcal{Z} \cup \overline{\mathcal{Z}}$ iff $P \xrightarrow{ay} P'$ or $P \xrightarrow{a(y)} P'$ for some name $y \in \mathcal{H}$ and process P'. Transitions $P \xrightarrow{M\bar{x}y} P'$, $P \xrightarrow{M\bar{x}(y)} P'$, $P \xrightarrow{Mxy} P'$ or $P \xrightarrow{Mx(y)} P'$ with $M \neq \varepsilon$ or $x \in \mathcal{R}$ generate no barbs.

X. The ideas behind this encoding

The above encoding combines seven ideas, each of which appears to be necessary to achieve the desired result. Accordingly, the translation could be described as the composition of seven encodings, leading from π_{IM} to CCS_γ via six intermediate languages. Here a language comprises syntax as well as semantics. Each of the intermediate languages has a labelled transition system semantics where the labels are as described in Section IV. Accordingly, at each step it is well-defined whether strong barbed bisimilarity is preserved, and one can show it is. These proofs go by induction on the derivation of transitions, where the transitions with visible labels are necessary steps even when one would only be interested in the transitions with τ-labels. There are various orders in which the seven steps can be taken. The seven steps are:

1) Moving from the late operational semantics (Table III) to the early one (Table IV). This translation is syntactically the identity function, but still its validity requires proof, as the generated LTS changes. The proof amounts to showing that the same barbed transition system is obtained before and after the translation—see Section VI.
2) Moving from a regular operational semantics (Table IV) to a symbolic one (Table VI). This step commutes with the previous one.
3) Renaming the bound names of a process in such a way that the result is clash-free [3], meaning that all bound names are different and no name occurs both free and bound. The trick is to do this in a compositional way. The relabelling operators $[\ell]$, $[r]$ and $[p_y]$ in the final encoding stem from this step.
4) Eliminating the need for rule **ALPHA** in the operational semantics. This works only for clash-free processes, as generated by the previous step.
5) Dropping the restriction operators, while preserving strong barbed bisimilarity. This eliminates the orange parts of Table VI. For this purpose clash-freedom and the elimination of **ALPHA** are necessary.
6) Changing all occurrences of substitutions into applications of CCS relabelling operators.
7) The previous six steps generate a language with a semantics in the De Simone format. So from here on a translation to MEIJE or aprACP_R is known to be possible. The last step, to CCS_γ, involves changing the remaining form of name-binding into an infinite sum.

[5] Appendices 1 and 2 present two different proofs of Theorem 2—the first shorter, but the second conceptually simpler, avoiding counterintuitive detours.

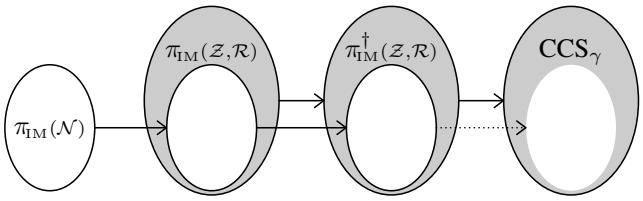


Fig. 2. Translation from the π-calculus with implicit matching to CCS_γ
The intermediate languages $\pi_{\mathrm{IM}(\mathcal{Z},\mathcal{R})}$ and $\pi^{\dagger}_{\mathrm{IM}(\mathcal{Z},\mathcal{R})}$ are not yet defined.

As indicated in Figure 2, my translation maps the π-calculus with implicit matching to a subset of CCS_γ. On that subset, π-calculus behaviour can be replayed faithfully, at least up to strong early congruence, the congruence closure of strong barbed bisimilarity (cf. [11]). However, the interaction between a translated π-calculus process and a CCS_γ process outside the image of the translation may be disturbing, and devoid of good properties. Also, in case intermediate languages are encountered on the way from π_{IM} to CCS_γ, which is just one of the ways to prove my result, no guarantees are given on the sanity of those languages outside the image of the source language, i.e. on their behaviour outside the realm of clash-free processes after Step 3 has been made.

XI. Triggering

To include the general matching operator in the source language I need to extend the target language with the *triggering* operator $s \Rightarrow P$ of MEIJE [1], [34]:

$$\frac{P \xrightarrow{\alpha} P'}{s \Rightarrow P \xrightarrow{s\alpha} P'}$$

MEIJE features *signals* and *actions*; each signal s can be "applied" to an action α, and doing so yields an action $s\alpha$. In this paper the actions are as in Table II, and a signal is an expression $[x{=}y]$ with $x, y \in \mathcal{N}$; application of a signal to an action was defined in Section VI.

Triggering cannot be expressed in CCS_γ, as rooted weak bisimilarity [2], the weak congruence of [19], [20], is a congruence for CCS_γ but not for triggering. However, rooted branching bisimilarity [12] is a congruence for triggering [9].

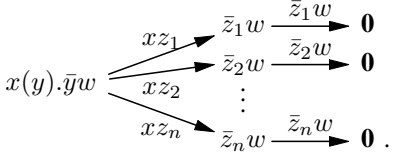
My translation from π_{IM} to CCS_γ can be extended into one from the full π-calculus to $\mathrm{CCS}^{\mathrm{trig}}_\gamma$ by adding the clause

$$\mathscr{T}([x{=}y]P) \;:=\; [x{=}y] \Rightarrow \mathscr{T}(P).$$

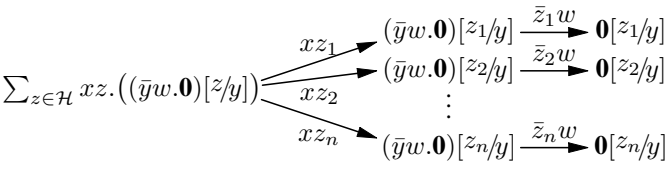
Theorem 2 applies to this extended translation as well.

XII. Examples

Example 2: The outgoing transitions of $x(y).\bar{y}w$ are

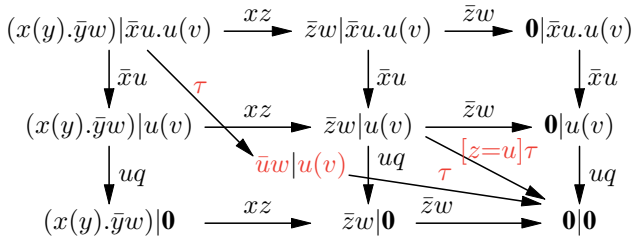


The same applies to its translation $\sum_{z\in\mathcal{H}} xz.\big((\bar{y}w.\mathbf{0})[z/y]\big)$.



Here the z_i range over all names in $\mathcal{N}$. Below I flatten such a picture by drawing the arrows only for one name z, which however still ranges over $\mathcal{N}$.

Example 3: The transitions of $P = x(y).\bar{y}w \mid \bar{x}u.u(v)$ are



Here $\bar{u}w|u(v)$ is the special case of $\bar{z}w|u(v)$ obtained by taking $z := u$. It thus also has outgoing transitions labelled $\bar{u}w$ and uq, for $q \in \mathcal{N}$.

Up to strong bisimilarity, the same transition system is obtained by the translation $\mathscr{T}(P)$ of P in CCS$_\gamma$.

$$\mathscr{T}(P) = \left(\sum_{z\in\mathcal{H}} xz.((\bar{y}w.\mathbf{0})[z/y])\right)[\ell] \,\Big\|\, \left(\bar{x}u.\sum_{z\in\mathcal{H}} uz(\mathbf{0}[z/v])\right)[r]$$

Since there are no restriction operators in this example, the relabelling operators $[\ell]$ and $[r]$ are of no consequence. Here $\mathscr{T}(P) \xrightarrow{\tau} (\bar{y}w.\mathbf{0})[u/y][\ell] \,\|\, \sum_{z\in\mathcal{H}} uz(\mathbf{0}[z/v])[r] \xrightarrow{\tau} \mathbf{0}[u/y][\ell] \,\|\, \mathbf{0}[w/v][r]$.

Example 4: Let $Q = (\nu x)\big(x(y).\bar{y}w \mid (\nu u)\big(\bar{x}u.u(v)\big)\big)$. It has no other transitions than

$$Q \xrightarrow{\tau} (\nu x)(\nu u)\big(\bar{u}w|u(v)\big) \xrightarrow{\tau} (\nu x)(\nu u)(\mathbf{0}|\mathbf{0}).$$

Its translation $\mathscr{T}(Q)$ into CCS$_\gamma$ is

$$\left(\left(\sum_{z\in\mathcal{H}} xz.((\bar{y}w.\mathbf{0})[z/y])\right)[\ell] \,\Big\|\, \left(\bar{x}u.\sum_{z\in\mathcal{H}} uz(\mathbf{0}[z/v])\right)[p_u][r]\right)[p_x]$$

Up to strong bisimilarity, its transition system is the same as that of P or $\mathscr{T}(P)$ from Example 3, except that in transition labels the name u is renamed into the private name ${}^{er}p$, and x is renamed into the private name p. One has $\mathscr{T}(Q) \,\dot{\sim}\, Q$, since private names generate no barbs.

Example 5: The process $(\nu x)(x(y)) \mid (\nu x)(\bar{x}u)$ has no outgoing transitions. Accordingly, its translation

$$\left(\sum_{z\in\mathcal{H}} xz.(\mathbf{0}\{z/y\})\right)[p_x][\ell] \,\Big\|\, (\bar{x}u)[p_x][r]$$

only has outgoing transitions labelled ${}^{\ell}pz$ for $z \in \mathcal{H}$ and $\overline{{}^{r}p}u$. Since the names ${}^{\ell}p$ and ${}^{r}p$ are private, these transitions generate no barbs. In this example, the relabelling operators $[\ell]$ and $[r]$ are essential. Without them, the mentioned transitions would have complementary names, and communicate into a τ-transition.

Example 6: Let $P = (\nu y)\big(\bar{x}y.\bar{y}w\big) \mid x(u).u(v)$. Then

$$P \xrightarrow{\tau} (\nu y)\big(\bar{y}w \mid y(v)\big) \xrightarrow{\tau} (\nu y)(\mathbf{0}|\mathbf{0}).$$

Now $\mathscr{T}\big((\nu y)\big(\bar{x}y.\bar{y}w\big)\big) = (\bar{x}y.\bar{y}w.\mathbf{0})[p_y]$ and

$$\mathscr{T}(x(u).u(v)) = \sum_{z\in\mathcal{H}} xz.\left(\left(\sum_{z\in\mathcal{H}} uz.(\mathbf{0}[z/v])\right)[z/u]\right).$$

Hence $\mathscr{T}\big((\nu y)\big(\bar{x}y.\bar{y}w\big)\big)[\ell] \xrightarrow{\bar{x}\,{}^{\ell}p} (\bar{y}w.\mathbf{0})[p_y][\ell]$. Since the substitution r used in the relabelling operator $[r]$ is surjective, there is a name s that is mapped to ${}^{\ell}p$, namely ${}^{\ell}p'$. Considering that $\mathscr{T}(x(u).u(v)) \xrightarrow{xs} \mathscr{T}(u(v))[s/u]$,

$$\mathscr{T}(P) \xrightarrow{\tau} (\bar{y}w.\mathbf{0})[p_y][\ell] \,\Big\|\, \left(\sum_{z\in\mathcal{H}} (uz.\mathbf{0})[z/v]\right)[s/u][r]$$

These parallel components can perform actions $\overline{{}^{\ell}p}w$ and ${}^{\ell}pw$, synchronising into a τ-transition, and thereby mimicking the behaviour of P.

Example 7: Let $P = (\nu y)\big(\bar{x}y.(\nu y)(\bar{y}w)\big) \mid x(u).u(v)$. Then $P \xrightarrow{\tau} (\nu y)\big((\nu y)(\bar{y}w) \mid y(v)\big) \not\xrightarrow{\tau}$. One obtains

$$\mathscr{T}(P) \xrightarrow{\tau} (\bar{y}w.\mathbf{0})[p_y][p_y][\ell] \,\Big\|\, \left(\sum_{z\in\mathcal{H}} uz.(\mathbf{0}[z/v])\right)[s/u][r]$$

for a name s that under $[r]$ maps to ${}^{\ell}p$. Now the left component can do an action $\overline{{}^{\ell e}p}w$, whereas the left component can merely match with $\overline{{}^{\ell}p}w$. No synchronisation is possible. This shows why it is necessary that the relabelling $[p_y]$ not only renames y into p, but also p into ${}^{e}p$.

Example 8: Let $P = x(y).x(w).\bar{w}u$. Then

$$P|\bar{x}v.\bar{x}y.y(v) \xrightarrow{\tau} x(w).\bar{w}u|\bar{x}y.y(v) \xrightarrow{\tau} \bar{y}u|y(v) \xrightarrow{\tau} \mathbf{0}|\mathbf{0}.$$

Therefore, $\mathscr{T}(P|\bar{x}v.\bar{x}y.y(v))$ must also be able to start with three consecutive τ-transitions. Note that

$$\mathscr{T}(P|\bar{x}v.\bar{x}y.y(v)) = \mathscr{T}(P)[\ell] \,\Big\|\, \left(\bar{x}v.\bar{x}y.\sum_{z\in\mathcal{H}} yz(\mathbf{0}[z/y])\right)[r]$$

with

$$\mathscr{T}(P) = \sum_{z\in\mathcal{H}} xz.\left(\left(\sum_{z\in\mathcal{H}} xz.((\bar{w}u.\mathbf{0})[z/w])\right)[z/y]\right).$$

The only way to obtain $\mathscr{T}(P|\bar{x}v.\bar{x}y.y(v)) \xrightarrow{\tau}\xrightarrow{\tau}\xrightarrow{\tau}$ is when $\mathscr{T}(P) \xrightarrow{xv} Q \xrightarrow{xy}\xrightarrow{\bar{y}u}$. The CCS$_\gamma$ process Q must be

$$\left(\sum_{z\in\mathcal{H}} xz.((\bar{w}u.\mathbf{0})[z/w])\right)[v/y].$$

Given the semantics of the CCS relabelling operator, one must have $\sum_{z\in\mathcal{H}} xz.((\bar{w}u.\mathbf{0})[z/w]) \xrightarrow{\alpha}$, such that applying the relabelling $[v/y]$ to α yields xy. When simply taking $[\{v/y\}]$ for $[v/y]$, that is, the relabelling that changes all occurrences of the name y in a transition label into v, this is not possible. This shows that a simplification of my translation without use of the spare names $\mathcal{S}$ would not be valid.

Crucial for this example is that I only use surjective substitutions. $[v/y]$ is an abbreviation of $[\{v/y\}^{\mathcal{S}}]$. Here $\{v/y\}^{\mathcal{S}}$ is a surjective substitution that not only renames y into v, but also sends a spare name s to y. This allows me to take $\alpha := xs$. Consequently, in deriving the transition $\sum_{z\in\mathcal{H}} xz.((\bar{w}u.\mathbf{0})[z/w]) \xrightarrow{\alpha}$, I choose z to be s, so that

$$\sum_{z\in\mathcal{H}} xz.((\bar{w}u.\mathbf{0})[z/w]) \xrightarrow{xs} (\bar{w}u.\mathbf{0})[s/w] \xrightarrow{\bar{s}u} \mathbf{0}[s/w].$$

Putting this in the scope of the relabelling $[v/y]$ yields

$$Q \xrightarrow{xy} (\bar{w}u.\mathbf{0})[s/w][v/y] \xrightarrow{\bar{y}u} \mathbf{0}[s/w][v/y]$$

as desired, and the example works out.[6]

This example shows that spare names play a crucial role in intermediate states of CCS$_\gamma$-translations. In general this leads to stacked relabellings from true names into spare ones and back. Making sure that in the end one always ends up with the right names calls for particularly careful proofs that do not cut corners in the bookkeeping of names.

A last example showing a crucial feature of my translation is discussed in Section XIV.

XIII. The unencodability of CCS into π

Let $f : \mathscr{A} \to \mathscr{A}$ be a CCS relabelling function satisfying $f(x_i y) = x_{i+1}y$. Here $(x_i)_{i=0}^{\infty}$ is an infinite sequence of names, and $\mathscr{A}$ is as in Section IV. The CCS process A defined by

$$A := x_0 y.\mathbf{0} + \tau.(A[f])$$

satisfies $\exists P.\ A \xrightarrow{\tau}{}^{*}\ P \wedge P{\downarrow_{x_i}}$ for all $i \geq 0$, i.e., it has infinitely many *weak barbs*. It is easy to check that all weak barbs of a π-calculus process Q must be free names of Q, of which there are only finitely many. Consequently, there is no π-calculus process Q with $A \mathrel{\dot{\leftrightarrow}} Q$, and hence no translation of CCS in the π-calculus that is valid up to $\mathrel{\dot{\leftrightarrow}}$.[7]

[6]This use of spare names solves the problem raised in [3, Footnote 5].

[7]In [28] it was already mentioned, by reference to Pugliese [personal communication, 1997] that CCS relabelling operators cannot be encoded in the π-calculus.

XIV. Related work

My translation from π_{IM} to CCS$_\gamma$ is inspired by an earlier translation $\mathcal{E}$ from a version of the π-calculus to CCS, proposed by Banach & van Breugel [3]. The paper [3] takes $\mathscr{A} := \{\langle x, y\rangle \mid x, y \in \mathcal{N}\}$ for the visible CCS actions; action $\langle x, y\rangle$ corresponds with my xy, and its complement $\overline{\langle x, y\rangle}$ with my $\bar{x}y$. On the fragment of π featuring inaction, prefixing, choice and parallel composition, the encoding of [3] is given by

$$\begin{array}{lcl}
\mathcal{E}(\mathbf{0}) & := & \mathbf{0}\\
\mathcal{E}(\tau.P) & := & \tau.\mathcal{E}(P)\\
\mathcal{E}(\bar{x}y.P) & := & \overline{\langle x, y\rangle}.\mathcal{E}(P)\\
\mathcal{E}(x(y).P) & := & \sum_{z\in\mathcal{N}}\langle x, z\rangle.\big(\mathcal{E}(P)[z/y]\big)\\
\mathcal{E}(P \mid Q) & := & \mathcal{E}(P) \mid \mathcal{E}(Q)\\
\mathcal{E}(P + Q) & := & \mathcal{E}(P) + \mathcal{E}(Q).
\end{array}$$

The main result of [3] (Theorem 5.3), stating the correctness of this encoding, says that $P \leftrightarrows_r Q$ iff $\mathcal{E}(P) \leftrightarrows_r \mathcal{E}(Q)$, for all π-processes P and Q. Here $\leftrightarrows_r$ is strong reduction bisimilarity—see Definition 7. In fact, replacing the call to Lemma 3.5 in the proof of this theorem by a call to Lemma 3.4, they could equally well have claimed the stronger result that $P \leftrightarrows_r \mathcal{E}(P)$ for all π-processes P, i.e., that $\mathcal{E}$ is valid up to $\leftrightarrows_r$.

This result contradicts my Theorem 1 and thus must be flawed. Where it fails can be detected by pushing the counterexample process $P := \bar{x}v \mid x(y).R$ with $R := \bar{y}u|v(w)$, used in the proof of Theorem 1, through the encoding of [3]. I claim that while $P \xrightarrow{\tau} \bar{v}u|v(w) \xrightarrow{\tau}$, its translation $\mathcal{E}(P)$ cannot do two τ-steps. Hence $P \not\leftrightarrows_r \mathcal{E}(P)$. Using a trivial process Q such that $P \leftrightarrows_r Q \leftrightarrows_r \mathcal{E}(Q)$, this also constitutes a counterexample to [3, Theorem 5.3].

Note that $\mathcal{E}(R) = \overline{\langle y, u\rangle}.\mathbf{0} \mid \sum_{z\in N}\langle v, z\rangle.(\mathbf{0}[z/w])$. This process can perform the actions $\overline{\langle y, u\rangle}$ as well as $\langle v, u\rangle$, but no action τ, since $y \neq v$. Now

$$\mathcal{E}(P) = \overline{\langle x, v\rangle}.\mathbf{0} \mid \sum_{z\in N}\langle x, z\rangle.(\mathcal{E}(R)[z/y]).$$

Its only τ-transition goes to $\mathbf{0} \mid \mathcal{E}(R)[v/y]$. This process can perform the actions $\overline{\langle v, u\rangle}$ as well as $\langle v, u\rangle$, but still no action τ, since $[v/y]$ is a CCS relabelling operator rather than a substitution, and it is applied only after any synchronisations between $\overline{\langle y, u\rangle}.\mathbf{0}$ and $\sum_{z\in N}\langle v, z\rangle.(\mathbf{0}[z/w])$ are derived.

My own encoding $\mathscr{T}$ translates the processes P and R essentially in the same way, but now there is a transition $\mathscr{T}(R) \xrightarrow{[y=v]\tau} (\mathbf{0}\|\mathbf{0}[u/w])$. The renaming $[v/y]$ turns this synchronisation into a τ:

$$\mathscr{T}(P) \xrightarrow{\tau} \mathscr{T}(R)[v/y] \xrightarrow{\tau} (\mathbf{0}\|\mathbf{0}[u/w])[v/y].$$

The crucial innovation of my approach over [3] in this regard is the switch from the early to the early symbolic semantics of the π-calculus, combined with a switch from CCS as target language to CCS$_\gamma$.

In [31], Roscoe argues that CSP is at least as expressive as the π-calculus. As evidence he present a translation from the latter to the former. Roscoe does not provide a criterion for the validity of such a translation, nor a result implying that a suitable criterion has been met. The following observations show

that his transition is not compositional, and that it is debatable whether it preserves a reasonable semantic equivalence.

(1) Roscoe translates $\tau.P$ as $tau \rightarrow \text{CSP}[P]$, where $\rightarrow$ is CSP action prefixing and $\text{CSP}[P]$ is the translation of the π-expression P. Here tau is a visible CSP action, that is renamed into τ only later in the translation, when combining prefixes into summations. Thus, on the level of prefixes, the translation does not preserve (strong) barbed bisimilarity or any other suitable semantic equivalence. This problem disappears when we stop seeing prefixing and choice as separate operators in the π-calculus, instead using a guarded choice $\sum_{i\in I} \alpha_i.P_i$.

(2) Roscoe translates $x(y).P$ into $x?z \rightarrow \text{CSP}[P\{z/y\}]$. This is not compositional, since the translation of $x(y).P$ does not merely call the translation of P as a building block, but the result of applying a substitution to P. Substitution is not a CSP operator; it is applied to the π-expression P before translating it. While this mode of translation has some elegance, it is not compositional, and it remains questionable whether a suitable weaker correctness criterion can be formulated that takes the place of compositionality here.

(3) To deal with restriction, [31] works with translations $\text{CSP}[P]_{\kappa,\sigma}$, where two parameters κ and σ are passed along that keep track of sets of fresh names to translate restricted names into. The set of fresh names σ is partitioned in the translation of $P|Q$ (page 388), such that both sides get disjoint sets of fresh names to work with. Although the idea is rather similar to the one used here, the passing of the parameters makes the translation non-compositional. In a compositional translation $\text{CSP}[P|Q]$ the arguments P and Q may appear in the translated CSP process only in the shape $\text{CSP}[P]$ and $\text{CSP}[Q]$, not $\text{CSP}[P]_{\kappa,\sigma'}$ for new values of σ'.

As pointed out in [14], [29], even the most bizarre translations can be found valid if one only imposes requirements based on semantic equivalence, and not compositionality. Roscoe's translation is actually rather elegant. However, we do not have a decent criterion to say to what extent it is a valid translation. The expressiveness community strongly values compositionality as a criterion, and this attribute is the novelty brought in by my translation.

XV. Conclusion

This paper exhibited a compositional translation from the π-calculus to CCS_γ extended with triggering that is valid up to strong barbed bisimilarity, thereby showing that the latter language is at least as expressive as the former. Triggering is not needed when restricting to the π-calculus with implicit matching (as used for instance in [33]). Conversely, I observed that CCS (and thus certainly CCS_γ) cannot be encoded in the π-calculus. I also showed that the upgrade of CCS to CCS_γ is necessary to capture the expressiveness of the π-calculus.

A consequence of this work is that any system specification or verification that is carried out in the setting of the π-calculus can be replayed in CCS_γ. The main idea here is to replace the names that are kept private in the π-calculus by means of the restriction operator, by names that are kept private by means of a careful bookkeeping ensuring that the same private name is never used twice. Of course this in no way suggests that it would be preferable to replay π-calculus specifications or verifications in CCS_γ.

My translation encodes the restriction operator (νy) from the π-calculus by renaming y into a "private name". Crucial for this approach is that private names generate no barbs, in contrast with standard approaches where all names generate barbs. This use of private names is part of the definition of strong barbed bisimilarity $\overset{\bullet}{\sim}$ on my chosen instance of CCS_γ, and justified since that definition is custom made in the present paper. The use of private names can be avoided by placing an outermost CCS restriction operator around any translated π-process. This, however, would violate the compositionality of my translation.

The use of infinite summation in my encoding might be considered a serious drawback. However, when sticking to a countable set of π-calculus names, only countable summation is needed, which, as shown in [8], can be eliminated in favour of unguarded recursion with infinitely many recursion equations. As the original presentation of the π-calculus already allows unguarded recursion with infinitely many recursion equations [24] the latter can not reasonably be forbidden in the target language of the translation. Still, it is an interesting question whether infinite sums or infinite sets of recursion equations can be avoided in the target language if we rule them out in the source language. My conjecture is that this is possible, but at the expense of further upgrading CCS_γ, say to aprACP^τ_R. This would however require work that goes well beyond what is presented here.

An alternative approach is to use a version of CCS featuring a *choice quantifier* [17] instead of infinitary summation, a construct that looks remarkably like an infinite sum, but is as finite as any quantifier from predicate logic. A choice quantifier binds a data variable z (here ranging over names) to a single process expression featuring z. The present application would need a function from names to CCS relabelling operators. When using this approach, the size of translated expressions becomes linear in the size of the originals.

It could be argued that choice quantification is a step towards mobility. On the other hand, if mobility is associated more with scope extrusion than with name binding itself, one could classify CCS_γ with choice quantification as an immobile process algebra. A form of choice quantification is standard in mCRL2 [15], which is often regarded "immobile".

My translation from π to CCS_γ has a lot in common with the attempted translation of π to CCS in [3]. That one is based on the early operational semantics of CCS, rather than the early symbolic one used here. As a consequence, substitutions there cannot be eliminated in favour of relabelling operators.

A crucial step in my translation yields an intermediate language with an operational semantics in De Simone format. In [7] another representation of the π-calculus is given through an operational semantics in the De Simone format. It uses

a different way of dealing with substitutions. This type of semantics could be an alternative stepping stone in an encoding from the π-calculus into CCS_γ.

In [28] Palamidessi showed that there exists no uniform encoding of the π-calculus into a variant of CCS. Here *uniform* means that $\mathscr{T}(P|Q) = \mathscr{T}(P)|\mathscr{T}(Q)$. This does not contradict my result in any way, as my encoding is not uniform. Palamidessi [28] finds uniformity a reasonable criterion for encodings, because it guarantees that the translation maintains the degree of distribution of the system. In [30], however, it is argued that it is possible to maintain the degree of distribution of a system upon translation without requiring uniformity. In fact, the translation offered here is a good example of one that is not uniform, yet maintains the degree of distribution.

Gorla [13] proposes five criteria for valid encodings, and shows that there exists no valid encoding of the π-calculus (even its asynchronous fragment) into CCS. Gorla's proof heavily relies on the criterion of *name invariance* imposed on valid encodings. It requires for $P \in \mathrm{T}_\pi$ and an injective substitution σ that $\mathscr{T}(P\sigma) = \mathscr{T}(P)\sigma'$ for some substitution σ' that is obtained from σ through a *renaming policy*. Furthermore, the renaming policy is such that if $dom(\sigma)$ is finite, then also $dom(\sigma')$ is finite. This latter requirement is not met by the encoding presented here, for a single name $x \in \mathcal{N}$ corresponds with an infinite set of actions xy, the "names" of CCS, and a substitution that merely renames x into z must rename each action xy into zy at the CCS end, thus violating the finiteness of $dom(\sigma')$.

My encoding also violates Gorla's compositionality requirement, on grounds that $\mathscr{T}(P)$ appears multiple times (actually, infinitely many) in the translation of $Mx(y).P$. It is however compositional by the definition in [10] and elsewhere. My encoding satisfies all other criteria of [13] (operational correspondence, divergence reflection and success sensitiveness).

References

[1] D. Austry & G. Boudol (1984): *Algèbre de processus et synchronisations*. TCS 30(1), pp. 91–131, doi:10.1016/0304-3975(84)90067-7.

[2] J.C.M. Baeten & W.P. Weijland (1990): *Process Algebra*. Cambridge Tracts in Theoretical Computer Science 18, Cambridge University Press, doi:10.1017/CBO9780511624193.

[3] R. Banach & F. van Breugel (1998): *Mobility and Modularity: expressing π-calculus in CCS*. Preprint. Available at http://www.cs.man.ac.uk/~banach/some.pubs/Pi.CCS.ext.abs.pdf.

[4] J.A. Bergstra & J.W. Klop (1986): *Algebra of communicating processes*. In: *Mathematics and Computer Science*, CWI Monograph 1, North-Holland, pp. 89–138.

[5] G. Boudol (1992): *Asynchrony and the π-calculus (Note)*. Tech. Rep. 1702, INRIA.

[6] S.D. Brookes, C.A.R. Hoare & A.W. Roscoe (1984): *A theory of communicating sequential processes*. *Journal of the ACM* 31(3), pp. 560–599, doi:10.1145/828.833.

[7] G.L. Ferrari, U. Montanari & P. Quaglia (1996): *A Pi-Calculus with Explicit Substitutions*. Theoretical Computer Science 168(1), pp. 53–103, doi:10.1016/S0304-3975(96)00063-1.

[8] R.J. van Glabbeek (1994): *On the expressiveness of ACP (extended abstract)*. In: Proc. ACP'94, Workshops in Computing, Springer, pp. 188–217, doi:10.1007/978-1-4471-2120-6_8.

[9] R.J. van Glabbeek (2011): *On Cool Congruence Formats for Weak Bisimulations*. Theoretical Computer Science 412(28), pp. 3283–3302, doi:10.1016/j.tcs.2011.02.036.

[10] R.J. van Glabbeek (2012): *Musings on Encodings and Expressiveness*. In: Proc. EXPRESS/SOS'12, EPTCS 89, Open Publishing Association, pp. 81–98, doi:10.4204/EPTCS.89.7.

[11] R.J. van Glabbeek (2018): *A Theory of Encodings and Expressiveness*. In: Proc. FoSSaCS'18, LNCS 10803, Springer, pp. 183–202, doi:10.1007/978-3-319-89366-2_10.

[12] R.J. van Glabbeek & W.P. Weijland (1996): *Branching Time and Abstraction in Bisimulation Semantics*. *Journal of the ACM* 43(3), pp. 555–600, doi:10.1145/233551.233556.

[13] D. Gorla (2010): *Towards a unified approach to encodability and separation results for process calculi*. *Information and Computation* 208(9), pp. 1031–1053, doi:10.1016/j.ic.2010.05.002.

[14] D. Gorla & U. Nestmann (2016): *Full abstraction for expressiveness: history, myths and facts*. *Mathematical Structures in Computer Science* 26(4), pp. 639–654, doi:10.1017/S0960129514000279.

[15] J.F. Groote & M.R. Mousavi (2014): *Modeling and Analysis of Communicating Systems*. MIT Press.

[16] M. Hennessy & H. Lin (1995): *Symbolic Bisimulations*. Theoretical Comp. Sc. 138(2), pp. 353–389, doi:10.1016/0304-3975(94)00172-F.

[17] B. Luttik (2003): *On the expressiveness of choice quantification*. *Ann. Pure Appl. Logic* 121, pp. 39–87, doi:10.1016/S0168-0072(02)00082-9.

[18] R. Milner (1983): *Calculi for synchrony and asynchrony*. Theoretical Comp. Sc. 25, pp. 267–310, doi:10.1016/0304-3975(83)90114-7.

[19] R. Milner (1989): *Communication and Concurrency*. Prentice Hall, Englewood Cliffs.

[20] R. Milner (1990): *Operational and algebraic semantics of concurrent processes*. In: *Handbook of Theoretical Computer Science*, chapter 19, Elsevier Science Publishers B.V. (North-Holland), pp. 1201–1242.

[21] R. Milner (1992): *Functions as Processes*. *Mathematical Structures in Computer Science* 2(2), pp. 119–141, doi:10.1017/S0960129500001407.

[22] R. Milner (1999): *Communicating and Mobile Systems: the π-Calculus*. Cambridge University Press.

[23] R. Milner, J. Parrow & D. Walker (1992): *A Calculus of Mobile Processes, I*. I&C 100, pp. 1–40, doi:10.1016/0890-5401(92)90008-4.

[24] R. Milner, J. Parrow & D. Walker (1992): *A Calculus of Mobile Processes, II*. I&C 100, pp. 41–77, doi:10.1016/0890-5401(92)90009-5.

[25] R. Milner, J. Parrow & D. Walker (1993): *Modal Logics for Mobile Processes*. TCS 114, pp. 149–171, doi:10.1016/0304-3975(93)90156-N.

[26] R. Milner & D. Sangiorgi (1992): *Barbed Bisimulation*. In: Proc. ICALP'92, LNCS 623, Springer, pp. 685–695, doi:10.1007/3-540-55719-9_114.

[27] U. Nestmann (2006): *Welcome to the Jungle: A Subjective Guide to Mobile Process Calculi*. In: Proc. CONCUR'06, LNCS 4137, Springer, pp. 52–63, doi:10.1007/11817949_4.

[28] C. Palamidessi (2003): *Comparing The Expressive Power Of The Synchronous And Asynchronous Pi-Calculi*. *Mathematical Structures in Comp. Science* 13(5), pp. 685–719, doi:10.1017/S0960129503004043.

[29] J. Parrow (2016): *General conditions for full abstraction*. *Math. Struct. in Comp. Sc.* 26(4), pp. 655–657, doi:10.1017/S0960129514000280.

[30] K. Peters, U. Nestmann & U. Goltz (2013): *On Distributability in Process Calculi*. In: *Proc. ESOP'13*, LNCS 7792, Springer, pp. 310–329, doi:10.1007/978-3-642-37036-6_18.

[31] A.W. Roscoe (2010): *CSP is Expressive Enough for π*. In: *Reflections on the Work of C.A.R. Hoare*, Springer, pp. 371–404, doi:10.1007/978-1-84882-912-1_16.

[32] D. Sangiorgi (1996): *A Theory of Bisimulation for the pi-Calculus*. *Acta Informatica* 33(1), pp. 69–97, doi:10.1007/s002360050036.

[33] D. Sangiorgi & D. Walker (2001): *The π-calculus: A Theory of Mobile Processes*. Cambridge University Press.

[34] R. de Simone (1985): *Higher-level synchronising devices in* MEIJE-*SCCS*. TCS 37, pp. 245–267, doi:10.1016/0304-3975(85)90093-3.

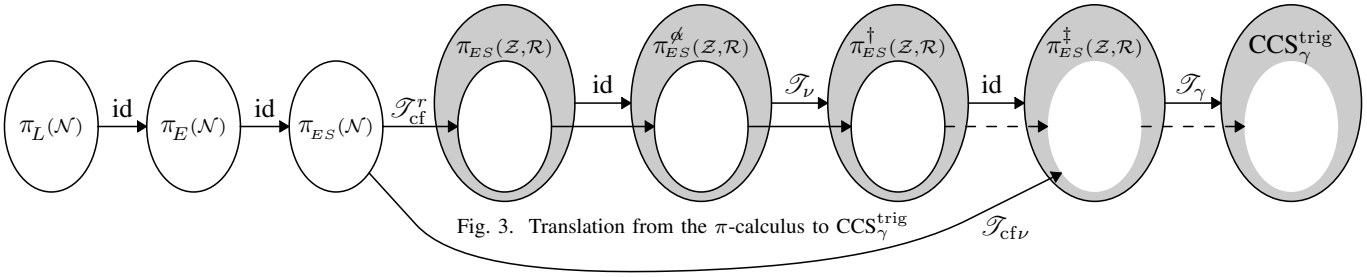


Fig. 3. Translation from the π-calculus to $\mathrm{CCS}^{\rm trig}_\gamma$

APPENDIX 1

As indicated in Figure 3, my translation from π to $\mathrm{CCS}^{\rm trig}_\gamma$ proceeds in seven steps. Section IX presents a translation in one step: essentially the composition of these constituent translations. Its decomposition in Figure 3 describes both how I found it, and how I prove its validity.

Each of the eight languages in Figure 3 comprises syntax, determining what are the valid expressions or processes, a structural operational semantics generating an LTS, and a BTS extracted from the LTS in the way described in Section IV. The subscript L, E or ES in Figure 3 tells whether I mean the π-calculus equipped with the late, the early, or the early symbolic semantics. The argument $\mathcal{N}$ denotes the set of names employed by this version of the π-calculus. In step 3 the set of names is extended from $\mathcal{N}$ to $\mathcal{Z} \uplus \mathcal{R}$, where $\mathcal{N} \subset \mathcal{Z}$. I write $\mathcal{Z},\mathcal{R}$ instead of $\mathcal{Z} \uplus \mathcal{R}$ to indicate that only the names in $\mathcal{Z}$—the *public* ones—generate barbs, and to impose a mild restriction on which names to allow within defining equations of agent identifiers. The superscript $^{\not\alpha}$ indicates that rule **ALPHA** is deleted from the operational semantics, and † that moreover the restriction operator is dropped. The superscript $\ddagger$ indicates a variant of the calculus to which relabelling operators have been added, and where the substitutions in rules **EARLY-INPUT** and **IDE** are replaced by relabelling operators. The interior white ellipses denote the classes of *clash-free* processes, defined in Sections A and E.

My translation starts from the π-calculus $\pi_L(\mathcal{N})$ with the late operational semantics, as defined in [24]. The first step is the identity mapping to $\pi_E(\mathcal{N})$, the calculus with the same syntax but the early operational semantics. The validity of this translation step is the statement that each $\pi_L(\mathcal{N})$-expression P is strongly barbed bisimilar with the same expression P, but now seen as a state in the LTS generated by the early operational semantics. As remarked in Section VI, this is an immediate consequence of Lemmas 1 and 2.

The second translation step likewise goes to the π-calculus with the early symbolic semantics. Its validity has been concluded at the end of Section VI.

Skipping step 3 for the moment, Section B describes the fourth translation step by studying the identity translation from $\pi_{ES}(\mathcal{Z},\mathcal{R})$ to $\pi^{\not\alpha}_{ES}(\mathcal{Z},\mathcal{R})$. As shown by Example 1, rule **ALPHA** is not redundant in the early (symbolic) semantics, and thus this step in not valid in general. As a consequence, $\pi^{\not\alpha}_{ES}(\mathcal{Z},\mathcal{R})$ is a weird calculus, that no doubt is unsuitable for many practical purposes. Nevertheless, Section B shows that this translation step is valid on the subclass of clash-free processes, defined in Section A. That is, each clash-free process P in $\pi_{ES}(\mathcal{Z},\mathcal{R})$ is strongly barbed bisimilar to the same process P seen as a state in $\pi^{\not\alpha}_{ES}(\mathcal{Z},\mathcal{R})$.

Step 5, recorded in Section C, eliminates the restriction operator from the language by translating (sub)expressions $(\nu z)P$ into P. This step does not preserve $\clubsuit$ for the language as a whole, but, since there are no $\mathcal{R}$-barbs, it does so on the sublanguage that arises as the image of the previous translation steps, namely on the clash-free processes in $\pi^{\not\alpha}_{ES}(\mathcal{Z},\mathcal{R})$.

Section D shows that the substitutions that occur in the operational semantics of Table VI may be replaced by relabelling operators, while preserving strong barbed bisimilarity on clash-free processes. This proves the validity of Step 6, the identity translation from $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ to $\pi^{\ddagger}_{ES}(\mathcal{Z},\mathcal{R})$ After this step, the resulting language is in De Simone format. Moreover, as shown in Section G, it is easily translated into $\mathrm{CCS}^{\rm trig}_\gamma$.

To compose the above fourth step with the first two steps of my translation I need an intermediate step that maps each process P in $\pi_{ES}(\mathcal{N})$ to a clash-free process. A first proposal $\mathscr{T}^r_{\rm cf}$ for such a translation appears in Section E. It replicates agent identifiers and their defining equations—this surely preserves $\clubsuit$—and renames bound names by giving all binders (νy) a fresh name—this preserves $\equiv$ and thus certainly $\clubsuit$. Due to the introduction of fresh names, the target of the translation is $\pi_{ES}(\mathcal{Z} \uplus \mathcal{R})$ rather than $\pi_{ES}(\mathcal{N})$.

Even though $\mathscr{T}^r_{\rm cf}$ preserves $\clubsuit$, I have to reject it as a valid translation. The main objection is that it employs operations on processes—*ruthless substitutions*—that are not syntactic operators of the target language. Thereby it fails the criterion of compositionality, even if the ruthless substitutions are applied in a compositional manner. In addition, it creates an infinite amount of replicated agent identifiers, whereas I strive not to increase the number of agent identifiers found in the source language.

To overcome these problems, Section F studies the composed translation $\mathscr{T}_\nu \circ \mathscr{T}^r_{\rm cf}$ from $\pi_{ES}(\mathcal{N})$ to $\pi^{\ddagger}_{ES}(\mathcal{Z},\mathcal{R})$, and shows that in this translation the applications of ruthless substitutions can be replaced by applications of relabelling operators. In fact, the latter can be seen as syntactic counterparts of the ruthless substitutions. This replacement preserves the validity of the translation. This change also makes the replication of agent identifiers unnecessary, and thus solves both objections against $\mathscr{T}^r_{\rm cf}$.

Section H shows that the composed translation so obtained is equivalent to the one presented in Section IX.

A different proof, which first moves from substitutions to relabellings and only then eliminates restriction, is presented in Appendix 2.

A. Clash-free processes

In this section I consider the π-calculus $\pi(\mathcal{Z}\uplus\mathcal{R})$ where the set of available names is the disjoint union of sets $\mathcal{Z}$ and $\mathcal{R}$ of *public* and *private* names. For the purposes of this section it doesn't matter whether the calculus is equipped with the late, the early, the late symbolic or the early symbolic operational semantics.

A process P in $\pi(\mathcal{Z}\uplus\mathcal{R})$ is *well-typed* w.r.t. the partition $(\mathcal{Z},\mathcal{R})$ if for each binder $x(z)$ occurring in P one has $z\in\mathcal{Z}$, and for each binder (νy) occurring in P one has $z\in\mathcal{R}$. Let $\pi(\mathcal{Z},\mathcal{R})$ be the variant of $\pi(\mathcal{Z}\uplus\mathcal{R})$ in which all defining equations $A(\vec{x})\stackrel{\rm def}{=}P$ satisfy the restriction that P is well-typed and $x_i\in\mathcal{Z}$ for $i=1,\ldots,n$. In addition, when extracting a BTS from the LTS generated by this version of the π-calculus, only names in $\mathcal{Z}$ generate barbs.

In the setting of $\pi(\mathcal{Z},\mathcal{R})$ I sharpen the definition of $P\sigma$ from Section V, the application of a substitution σ to a process P. Namely, when choosing a name $z\notin{\rm fn}((\nu y)P)\cup dom(\sigma)\cup range(\sigma)$ to replace a bound name y, one always picks $z\in\mathcal{Z}$ if $y\in\mathcal{Z}$ and $z\in\mathcal{R}$ if $y\in\mathcal{R}$.

In [3] a process P is called *clash-free* if all occurrences of binders $x(y)$ and (νy) use a different name y and no name occurs both free and bound. Below I employ a more liberal version of this notion by requiring this only for binders (νy), and by allowing the arguments of a $+$-operator to share bound names. At the same time I sharpen the concept by including as binders of P not only those occurring in P, but also those occurring in the body Q of a defining equation $A(\vec{x})\stackrel{\rm def}{=}Q$ for which A, directly or indirectly, occurs in P. Furthermore, I require clash-free processes to be well-typed.

Definition 9: Let $h(P)$, the *hereditary subprocesses* of $P\in{\rm T}_\pi$, be the smallest set of processes containing P such that

- if $Q\in h(P)$ and R is a subterm of Q then $R\in h(P)$, and
- if $A(\vec{y})\in h(P)$ and $A(\vec{x})\stackrel{\rm def}{=}Q$ then $Q\in h(P)$.

Let ${\rm RN}(P)$, the *restriction-bound names* of P, be the set of all names y such that a process $(\nu y)Q$ occurs in $h(P)$.

Observation 1: If P is a well-typed $\pi(\mathcal{Z},\mathcal{R})$ process then ${\rm RN}(P)\subseteq\mathcal{R}$.

Definition 10: A $\pi(\mathcal{Z},\mathcal{R})$ process P is *clash-free*[8] if

1) P is well-typed,
2) for each $(\nu y)Q\in h(P)$ one has $y\notin{\rm RN}(Q)$,
3) for each $Q|R\in h(P)$ one has ${\rm RN}(Q)\cap{\rm RN}(R)=\emptyset$, and
4) ${\rm fn}(P)\cap{\rm RN}(P)=\emptyset$.

A substitution σ is *clash-free* on a well-typed $\pi(\mathcal{Z},\mathcal{R})$ process P if ${\rm RN}(P)\cap(dom(\sigma)\cup range(\sigma))=\emptyset$. In that case, $P\sigma$, defined in Section V, does not involve renaming of bound names from $\mathcal{R}$.

[8]The concept defined here ought to be called "restriction-clash-free", but is abbreviated "clash-free" in Sections A–C. In Section E a more general notion of "full" clash-freedom will be defined.

Observation 2: If P is clash-free and σ is clash-free on P, then $P\sigma$ is clash-free and ${\rm RN}(P\sigma)={\rm RN}(P)$.

Lemma 5: If $A(\vec{y})$ is clash-free and $A(\vec{x})\stackrel{\rm def}{=}P$ then the substitution $\{\vec{y}/\vec{x}\}$ from rule IDE is clash-free on P.

Proof. Since $x_i\in\mathcal{Z}$ for $i=1,\ldots,n$, one has $x_i\notin\mathcal{R}\supseteq{\rm RN}(P)$. Since $y_i\in{\rm fn}(A(\vec{y}))$ one has $y_i\notin{\rm RN}(A(\vec{y}))={\rm RN}(P)$. ■

Lemma 6: If $Mx(y).P$ is clash-free and $z\notin{\rm RN}(Mx(y).P)$ then substitution $\{z/y\}$ from EARLY-INPUT is clash-free on P.

Proof. One has $y\in\mathcal{Z}$, so $y\notin{\rm RN}(P)\subseteq\mathcal{R}$, and $z\notin{\rm RN}(P)$ because ${\rm RN}(P)={\rm RN}(Mx(y).P)$. ■

Definition 11: Let $\leftharpoonup\subseteq{\rm T}_\pi\times{\rm T}_\pi$ be the smallest preorder such that $P+Q\leftharpoonup P$, $P+Q\leftharpoonup Q$, $P|Q\leftharpoonup P$, $P|Q\leftharpoonup Q$, $[x{=}y]P\leftharpoonup P$, $(\nu y)P\leftharpoonup P$ and $A(\vec{y})\leftharpoonup P\{\vec{y}/\vec{x}\}$ when $A(\vec{x})\stackrel{\rm def}{=}P$.

The relation $\leftharpoonup$, like in Definition 8, connects the left-hand sides of conclusions in operational rules with the left-hand sides of the corresponding premises.

Lemma 7: If $P\leftharpoonup Q$ and P is clash-free, then so is Q.

Proof. Only the last two cases are nontrivial. For $(\nu y)P\leftharpoonup P$ use: ${\rm fn}(P)\subseteq{\rm fn}((\nu y)P)\cup\{y\}$ and ${\rm RN}(P)\subseteq{\rm RN}((\nu y)P)\setminus\{y\}$.

Suppose $A(\vec{y})$ is clash-free and $A(\vec{x})\stackrel{\rm def}{=}P$. By definition P is clash-free. Apply Lemma 5 and Observation 2. ■

B. The elimination of ALPHA

As a consequence of Lemma 1, up to late congruence, rule ALPHA is redundant in the late operational semantics of the π-calculus, and hence not included in Table III. As shown by Example 1, ALPHA is not redundant in the early operational semantics, or in the early symbolic one. However, this section establishes a few basic properties of the early (symbolic) operational semantics, leading to the conclusion that restricted to the class of clash-free $\pi(\mathcal{Z},\mathcal{R})$ processes, ALPHA is redundant up to $\frown$. These results apply to the early as well as the early symbolic semantics.

Let $P\stackrel{\alpha}{\longrightarrow}_\bullet Q$ denote that the transition $P\stackrel{\alpha}{\longrightarrow}Q$ is derivable from the rules of Table VI without using rule ALPHA.

Lemma 8: Let $P\stackrel{\alpha}{\longrightarrow}Q$. If $\alpha=M\bar{x}y$ or $M\tau$ then ${\rm n}(\alpha)\subseteq{\rm fn}(P)$. If $\alpha=Mxy$ or $\alpha=M\bar{x}(y)$ then ${\rm n}(M)\cup\{x\}\subseteq{\rm fn}(P)$.

Proof. Trivial inductions on the inference of $P\stackrel{\alpha}{\longrightarrow}Q$. ■

Lemma 9: If $P\stackrel{\alpha}{\longrightarrow}Q$ then ${\rm fn}(Q)\subseteq{\rm fn}(P)\cup{\rm n}(\alpha)$.

Proof. A trivial induction on the inference of $P\stackrel{\alpha}{\longrightarrow}Q$. ■

Lemma 10: If $P\stackrel{M\bar{x}(y)}{\longrightarrow}_\bullet Q$ then $y\in{\rm RN}(P)$.

Proof. A trivial induction on the inference of $P\stackrel{M\bar{x}(y)}{\longrightarrow}Q$. ■

Lemma 11: If R is clash-free, α is not of the form Mxz with $z\in{\rm RN}(R)$, and $R\stackrel{\alpha}{\longrightarrow}_\bullet R'$, then R' is clash-free and ${\rm RN}(R')\subseteq{\rm RN}(R)$. Moreover, ${\rm bn}(\alpha)\cap{\rm RN}(R')=\emptyset$.

Proof. With induction on the derivation of $R\stackrel{\alpha}{\longrightarrow}_\bullet R'$.

- The cases that $R \xrightarrow{\alpha}_\bullet R'$ is derived by rule **TAU** or **EARLY-OUTPUT** are trivial, using that R' is a subexpression of R.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **EARLY-INPUT**. Then $R = Mx(y).P$, $\alpha = Mxz$ with $z \notin \text{RN}(R)$ and $R' = P\{z/y\}$. Since $\text{fn}(P) \subseteq \text{fn}(R) \cup \{y\}$, $\text{RN}(P) = \text{RN}(R)$ and $\mathcal{Z} \ni y \notin \text{RN}(P)$, the process P is clash-free. By Lemma 6, the substitution $\{z/y\}$ is clash-free on P. Thus $R' = P\{z/y\}$ is clash-free by Observation 2, and $\text{RN}(R') = \text{RN}(R)$.
- The cases that $R \xrightarrow{\alpha}_\bullet R'$ is derived by rule **SUM**, **IDE** or **SYMB-MATCH** are trivial.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **PAR**. Then $R = P|Q$, $P \xrightarrow{\alpha}_\bullet P'$, $R' = P'|Q$, and if $\alpha = Mxz$ then $z \notin \text{RN}(P)$. By Lemma 7, P and Q are clash free. So by induction P' is clash-free and $\text{RN}(P') \subseteq \text{RN}(P)$. Moreover, $\text{bn}(\alpha) \cap \text{RN}(P') = \emptyset$. If $z \in \text{bn}(\alpha)$ then $z \in \text{RN}(P)$ by Lemma 10 and thus $z \notin \text{RN}(Q)$ by the clash-freedom of R. Hence $z \notin \text{RN}(R')$. Moreover $\text{RN}(R') \subseteq \text{RN}(R)$. I still need to check that R' is clash-free.
  Since $\text{RN}(P) \cap \text{RN}(Q) = \emptyset$, also $\text{RN}(P') \cap \text{RN}(Q) = \emptyset$. It remains to establish that $\text{fn}(P'|Q) \cap \text{RN}(P'|Q) = \emptyset$. So suppose $y \in \text{fn}(P'|Q)$. In case $y \in \text{fn}(P|Q)$ then $y \notin \text{RN}(P|Q) \supseteq \text{RN}(P'|Q)$ by the clash-freedom of R. Hence, in view of Lemmas 8 and 9, the only remaining cases are that α has the shape Mxy or $M\bar{x}(y)$, and $y \in \text{fn}(P')$. In the first case the assumption made in Lemma 11 yields $y \notin \text{RN}(R) \supseteq \text{RN}(P'|Q)$. In the latter case $y \in \text{RN}(P)$ by Lemma 10, so $y \notin \text{RN}(Q)$ by the clash-freedom of R, and $y \notin \text{RN}(P')$ as P' is clash-free.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **E-S-COM**. Then $R = P|Q$, $\alpha = [x{=}v]MN\tau$, $P \xrightarrow{M\bar{x}y}_\bullet P'$, $Q \xrightarrow{Nvy}_\bullet Q'$ and $R' = P'|Q'$. By Lemma 8, $y \in \text{fn}(P)$, so $y \notin \text{RN}(Q)$ by the clash-freedom of R. By Lemma 7, P and Q are clash free. Hence, by induction, P' and Q' are clash-free, $\text{RN}(P') \subseteq \text{RN}(P)$ and $\text{RN}(Q') \subseteq \text{RN}(Q)$.
  Since $\text{RN}(P) \cap \text{RN}(Q) = \emptyset$, also $\text{RN}(P') \cap \text{RN}(Q') = \emptyset$. Since $y \in \text{fn}(P)$, $\text{fn}(P') \cup \text{fn}(Q') \subseteq \text{fn}(P) \cup \text{fn}(Q)$ by Lemmas 8 and 9. Thus $\text{fn}(R') \cap \text{RN}(R') = \emptyset$ by the clash-freedom of R, so R' is clash-free.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **E-S-CLOSE**. Then $R = P|Q$, $\alpha = [x{=}v]MN\tau$, $P \xrightarrow{M\bar{x}(z)}_\bullet P'$, $Q \xrightarrow{Nvz}_\bullet Q'$, $R' = (\nu z)(P'|Q')$ and $z \notin \text{fn}(Q)$. By Lemma 10, $z \in \text{RN}(P)$, so $z \notin \text{RN}(Q)$ by the clash-freedom of R. By Lemma 7, P and Q are clash free. Hence, by induction, P' and Q' are clash-free, $\text{RN}(P') \subseteq \text{RN}(P)$ and $\text{RN}(Q') \subseteq \text{RN}(Q)$. Moreover, $z \notin \text{RN}(P')$. As $z \in \text{RN}(P) \subseteq \text{RN}(R)$, it follows that $\text{RN}(R') \subseteq \text{RN}(R)$.
  Since $\text{RN}(P) \cap \text{RN}(Q) = \emptyset$, also $\text{RN}(P') \cap \text{RN}(Q') = \emptyset$. Since $\text{fn}(R') = (\text{fn}(P') \cup \text{fn}(Q')) \setminus \{z\} \subseteq \text{fn}(P) \cup \text{fn}(Q) = \text{fn}(R)$ by Lemmas 8 and 9, $\text{fn}(R') \cap \text{RN}(R') = \emptyset$ by the clash-freedom of R. Finally, as $z \notin \text{RN}(Q) \supseteq \text{RN}(Q')$ and $z \notin \text{RN}(P')$, $z \notin \text{RN}(P'|Q')$. Hence R' is clash-free.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **RES**. Then $R = (\nu y)P$, $R' = (\nu y)P'$, and $P \xrightarrow{\alpha}_\bullet P'$. Moreover, $y \notin \text{n}(\alpha)$. By Lemma 7, P is clash-free. If α is of the form Mxz then $z \neq y$ and thus $z \notin \text{RN}(P)$. Hence, by induction, P' is clash-free, $\text{RN}(P') \subseteq \text{RN}(P)$ and $\text{bn}(\alpha) \cap \text{RN}(P') = \emptyset$. It follows that $\text{RN}(R') \subseteq \text{RN}(R)$ and $\text{bn}(\alpha) \cap \text{RN}(R') = \emptyset$. Since R is clash-free, $y \notin \text{RN}(P) \supseteq \text{RN}(P')$. It remains to establish that $\text{fn}(R') \cap \text{RN}(R') = \emptyset$. So suppose $w \in \text{fn}(R') \subseteq \text{fn}(P')$. Then $w \neq y$. Moreover, $w \notin \text{RN}(P')$ by the clash-freedom of P'. Hence $w \notin \text{RN}(R')$.
- Finally, suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **SYMB-OPEN**. Then $R = (\nu y)P$, $\alpha = M\bar{x}(y)$ and $P \xrightarrow{M\bar{x}y}_\bullet R'$. Moreover, $y \neq x$ and $y \notin \text{n}(M)$. By Lemma 7, P is clash-free. Hence, by induction, R' is clash-free and $\text{RN}(R') \subseteq \text{RN}(P) \subseteq \text{RN}(R)$. Since R is clash-free, $y \notin \text{RN}(P) \supseteq \text{RN}(R')$. ■

Lemma 12: If $P \xrightarrow{Mxy}_\bullet Q$ with $y \notin \text{fn}(P)$ and $z \notin \text{RN}(P)$ then $P \xrightarrow{Mxz}_\bullet P'$ for some P' with $P' \equiv Q\{z/y\}$.

Proof. A trivial induction on the inference of $P \xrightarrow{Mxy} Q$. ■

Let χ be an inference of $P \xrightarrow{\alpha} Q$. Then $\text{bn}_\chi(P)$ is the union of all $\text{bn}(P')$ for $P' \xrightarrow{\beta} Q'$ a transition that appears in χ.

Lemma 13: Let χ be an inference of $P \xrightarrow{Mxy} Q$ with $y \notin \text{fn}(P)$ and $z \notin \text{bn}_\chi(P)$, then $P \xrightarrow{Mxz} P'$ for some $P' \equiv Q\{z/y\}$. Moreover, $P \xrightarrow{Mxz} P'$ has an inference no deeper than χ.

Proof. A trivial induction on χ. ■

Given a substitution $\sigma : \mathcal{Z} \uplus \mathcal{R} \rightharpoonup \mathcal{Z} \uplus \mathcal{R}$, let $\sigma[y \mapsto z]$ denote the substitution with $dom(\sigma[y \mapsto z]) = dom(\sigma) \cup \{y\}$, defined by $\sigma[y \mapsto z](y) := z$ and $\sigma[y \mapsto z](x) := \sigma(x)$ when $x \neq y$.

Observation 3: If $w \notin \text{fn}((\nu y)P) \cup dom(\sigma) \cup range(\sigma)$ then $P\sigma[y \mapsto z] \equiv P\{w/y\}\sigma\{z/w\}$.

Recall that $((\nu y)P)\sigma := (\nu w)(P\{w/y\}\sigma)$ for some $w \notin \text{fn}((\nu y)P) \cup dom(\sigma) \cup range(\sigma)$. Thus if $U \equiv ((\nu y)P)\sigma$, then, w.l.o.g. α-converting the topmost name w first, $U = (\nu z)Q$ with $Q \equiv P\{w/y\}\sigma\{z/w\}$. By Observation 3 $Q \equiv P\sigma[y \mapsto z]$. Here $z \notin \text{fn}((\nu w)(P\{w/y\}\sigma)) = \text{fn}(((\nu y)P)\sigma)$. Likewise, if $U \equiv (Mx(y).P)\sigma$ then $U = M[\sigma]x[\sigma](z).Q$ with $Q \equiv P\sigma[y \mapsto z]$.

Corollary 1: If $z \notin \text{fn}(((\nu y)P)\sigma)$ then $(\nu z)(P\sigma[y \mapsto z]) \equiv ((\nu y)P)\sigma$.

In the next lemma I am mostly interested in the case $dom(\sigma) = \emptyset$, but the general case is needed to deal inductively with the cases **RES** and **SYMB-OPEN**. The condition $\text{fn}(\alpha[\sigma]) \cap \text{RN}(U) = \emptyset$ is needed for case **RES** only.

Lemma 14: Suppose $R \xrightarrow{\alpha} R'$, U is clash-free, σ is a finite substitution with $\text{fn}(\alpha[\sigma]) \cap \text{RN}(U) = \emptyset$ and $U \equiv R\sigma$.

(a) If $\text{bn}(\alpha) = \emptyset$ then $\exists U'.\ U \xrightarrow{\alpha[\sigma]}_\bullet U' \equiv R'\sigma$.
(b) If $\alpha = M\bar{x}(y)$ then $\exists z, U'.\ U \xrightarrow{M[\sigma]\bar{x}[\sigma](z)}_\bullet U' \equiv R'\sigma[y \mapsto z]$.

Proof. By induction on the depth of the inference of $R \xrightarrow{\alpha} R'$.
Ad (a):

- The cases that $R \xrightarrow{\alpha} R'$ is derived by rule **TAU** or **OUTPUT** are trivial.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by **EARLY-INPUT**. Then $R = Mx(y).P$, $\alpha = Mxz$ and $R' = P\{z/y\}$. Since $U \equiv R\sigma$, $U = M[\sigma]x[\sigma](w).Q$ with $w \notin \text{fn}(((\nu y)P)\sigma)$ and

$Q \equiv P\sigma[y \mapsto w]$. By application of rule **EARLY-INPUT**, $U \xrightarrow{\alpha[\sigma]}_\bullet Q\{z[\sigma]/w\}$. Moreover,

$$Q\{z[\sigma]/w\} \equiv P\sigma[y \mapsto w]\{z[\sigma]/w\} \equiv P\{z/y\}\sigma = R'\sigma.$$

- The cases that $R \xrightarrow{\alpha} R'$ is derived by rule **SUM**, **SYMB-MATCH**, **PAR** or **ALPHA** are trivial.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by **IDE**. Then $R = A(\vec{y})$, $A(\vec{x}) \stackrel{\text{def}}{=} P$ and $P\{\vec{y}/\vec{x}\} \xrightarrow{\alpha} R'$. Furthermore, $U = R\sigma$. By Lemma 7 $P\{\vec{y}[\sigma]/\vec{x}\}\big(\equiv P\{\vec{y}/\vec{x}\}\sigma\big)$ is clash-free. By Lemma 5 and Observation 2, $\text{RN}(P\{\vec{y}[\sigma]/\vec{x}\}) = \text{RN}(P) = \text{RN}(A(\vec{y})) = \text{RN}(U)$, so $\text{fn}(\alpha[\sigma]) \cap \text{RN}(P\{\vec{y}[\sigma]/\vec{x}\}) = \emptyset$. Thus, by induction, $P\{\vec{y}[\sigma]/\vec{x}\} \xrightarrow{\alpha}_\bullet U' \equiv R'\sigma$. By rule **IDE** $U \xrightarrow{\alpha}_\bullet U'$.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by **E-S-COM**. Then $R = P|Q$, $\alpha = [x{=}v]MN\tau$, $P \xrightarrow{M\bar{x}y} P'$, $Q \xrightarrow{Nvy} Q'$ and $R' = P'|Q'$. Moreover, $U = V|W$ with $V \equiv P\sigma$ and $W \equiv Q\sigma$. Since $y \in \text{fn}(P)$ by Lemma 8, $y[\sigma] \in \text{fn}(V) \subseteq \text{fn}(U)$. As U is clash-free, $y[\sigma] \notin \text{RN}(U) \supseteq \text{RN}(V) \cup \text{RN}(W)$. Thus $\text{fn}((M\bar{x}y)[\sigma]) \cap \text{RN}(V) = \text{fn}((Nvy)[\sigma]) \cap \text{RN}(W) = \emptyset$. So, by induction, $\exists V'.\ V \xrightarrow{M\bar{x}y}_\bullet V' \equiv P'\sigma$ as well as $\exists W'.\ W \xrightarrow{Nvy}_\bullet W' \equiv Q'\sigma$. Therefore, applying **E-S-COM**, $U \xrightarrow{[x=v]MN\tau} V'|W' \equiv P'\sigma|Q'\sigma = R'\sigma$.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by **RES**. Then $R = (\nu y)P$, $R' = (\nu y)P'$, and $P \xrightarrow{\alpha} P'$ is obtained by a shallower inference. Moreover, $y \notin \text{n}(\alpha)$. Since $U \equiv R\sigma$, $U = (\nu z)Q$ with $z \notin \text{fn}((\nu y)P)\sigma)$ and $Q \equiv P\sigma[y \mapsto z]$. So $\text{fn}(\alpha[\sigma]) \cap (\text{RN}(Q) \cup \{z\}) = \emptyset$ and $\alpha[\sigma[y \mapsto z]] = \alpha[\sigma]$. By Lemma 7 process Q is clash-free. Thus, by induction, $\exists Q'.Q \xrightarrow{\alpha[\sigma]}_\bullet Q' \equiv P'\sigma[y \mapsto z]$. Hence, as $z \notin \text{n}(\alpha[\sigma])$, $U \xrightarrow{\alpha}_\bullet (\nu z)Q'$ by **RES**. Since $z \notin \text{fn}(((\nu y)P)\sigma)$ and $z \notin \text{n}(\alpha[\sigma])$, $z \notin \text{fn}(((\nu y)P')\sigma)$ by Lemma 9. [Namely, if $w \in \text{fn}(((\nu y)P')\sigma)$ then $w = v[\sigma]$ with $v \in \text{fn}(P') \setminus \{y\}$. Thus $v \in \big(\text{fn}(P) \cup \text{n}(\alpha)\big) \setminus \{y\} \subseteq \text{fn}((\nu y)P) \cup \text{n}(\alpha)$, and $w \in \text{fn}(((\nu y)P)\sigma) \cup \text{n}(\alpha[\sigma])$.] Hence, by Corollary 1, $(\nu z)Q' \equiv (\nu z)(P'\sigma[y \mapsto z]) \equiv ((\nu y)P')\sigma = R'\sigma$.
- $R \xrightarrow{\alpha} R'$ cannot be derived by **SYMB-OPEN**, as $\text{bn}(\alpha) = \emptyset$.
- Finally, suppose $R \xrightarrow{\alpha} R'$ is derived by **E-S-CLOSE**. Then $R = P|Q$, $\alpha = [x{=}v]MN\tau$, $P \xrightarrow{M\bar{x}(y)} P'$, $Q \xrightarrow{Nvy} Q'$, $y \notin \text{fn}(Q)$ and $R' = (\nu y)(P'|Q')$. Moreover, $U = V|W$ with $V \equiv P\sigma$ and $W \equiv Q\sigma$. As $\text{fn}(M\bar{x}(y)[\sigma]) \subseteq \text{fn}(\alpha[\sigma])$, $\text{fn}(M\bar{x}(y)[\sigma]) \cap \text{RN}(V) = \emptyset$. Consequently, by induction,

  $$\exists z, V'.\ V \xrightarrow{M[\sigma]\bar{x}[\sigma](z)}_\bullet V' \equiv P'\sigma[y \mapsto z].$$

  Let χ be the derivation of $Q \xrightarrow{Nvy} Q'$. Pick $w \notin \text{bn}_\chi(Q) \cup \text{RN}(W) \cup \text{fn}(W) \cup \text{fn}((\nu y)Q') \cup dom(\sigma) \cup range(\sigma)$. Applying Lemma 13, $Q \xrightarrow{Nvw} Q''$ for some $Q'' \equiv Q'\{w/y\}$. Moreover, the inference of this transition is not as deep as the one of $R \xrightarrow{\alpha} R'$. As $\text{fn}((Nvw)[\sigma]) \cap \text{RN}(W) = \emptyset$,

  $$\exists W'.\ W \xrightarrow{N[\sigma]v[\sigma]w}_\bullet W' \equiv Q''\sigma \equiv Q'\{w/y\}\sigma$$

  by induction. Since $z \in \text{RN}(V)$ by Lemma 10, and U is clash-free, $z \notin \text{RN}(W)$. Applying Lemma 12,

  $$\exists W''.\ W \xrightarrow{N[\sigma]v[\sigma]z}_\bullet W'' \equiv W'\{z/w\} \equiv Q'\{w/y\}\sigma\{z/w\}$$

  By Observation 3, $Q'\{w/y\}\sigma\{z/w\} \equiv Q'\sigma[y \mapsto z]$. Thus, applying **E-S-CLOSE**, $U \xrightarrow{([x=v]MN\tau)[\sigma]} (\nu z)(V'|W'')$. Since $z \in \text{RN}(V) \subseteq \text{RN}(U)$ and U is clash-free, $z \notin \text{fn}(U) = \text{fn}(R\sigma) \supseteq \text{fn}(R'\sigma)$, the last step by Lemmas 8 and 9. Therefore, by Corollary 1, $(\nu z)(V'|W'') \equiv (\nu z)((P'|Q')\sigma[y \mapsto z]) \equiv ((\nu y)(P'|Q'))\sigma = R'\sigma$.

Ad (b):

- Suppose $R \xrightarrow{M\bar{x}(y)} R'$ is derived by **SYMB-OPEN**. Then $R = (\nu y)P$ and $P \xrightarrow{M\bar{x}y} R'$ is obtained by a shallower inference. Moreover, $y \neq x$ and $y \notin \text{n}(M)$. Since $U \equiv R\sigma$, $U = (\nu z)Q$ with $z \notin \text{fn}((\nu y)P)\sigma)$ and $Q \equiv P\sigma[y \mapsto z]$. By Lemma 7, Q is clash-free. Since U is clash-free, $z \notin \text{RN}(Q)$. So $\text{fn}((M\bar{x}y)[\sigma[y \mapsto z]]) \cap \text{RN}(Q) = \emptyset$. Consequently, by induction,

  $$\exists U'.\ Q \xrightarrow{M[\sigma]\bar{x}[\sigma]z}_\bullet U' \equiv R'\sigma[y \mapsto z].$$

  By Lemma 8, $\text{n}(M\bar{x}y) \subseteq \text{fn}(P)$ and hence $\text{n}(M) \cup \{x\} \subseteq \text{fn}((\nu y)P)$, so $\text{n}(M[\sigma]) \cup \{x[\sigma]\} \subseteq \text{fn}(((\nu y)P)\sigma) \not\ni z$. Therefore, by **SYMB-OPEN**, $U \xrightarrow{M[\sigma]\bar{x}[\sigma](z)}_\bullet U'$.
- The cases that $R \xrightarrow{M\bar{x}(y)} R'$ is derived by rule **SUM**, **SYMB-MATCH** or **ALPHA** are again trivial.
- Derivations of $R \xrightarrow{M\bar{x}(y)} R'$ by **TAU**, **OUTPUT**, **EARLY-INPUT**, **E-S-COM** or **E-S-CLOSE** cannot occur.
- The case that $R \xrightarrow{M\bar{x}(y)} R'$ is derived by rule **IDE** proceeds just as for statement (a) above.
- Suppose $R \xrightarrow{M\bar{x}(y)} R'$ is derived by **PAR**. Then $R = P|Q$, $P \xrightarrow{M\bar{x}(y)} P'$ is obtained by a shallower inference, and $R' = P'|Q$. Moreover, $y \notin \text{fn}(Q)$, and $U = V|W$ with $V \equiv P\sigma$ and $W \equiv Q\sigma$. Using that V is clash-free and $\text{fn}((M\bar{x}(y))[\sigma]) \cap \text{RN}(V) = \emptyset$, by induction

  $$\exists z, V'.\ V \xrightarrow{M[\sigma]x[\sigma](z)}_\bullet V' \equiv P'\sigma[y \mapsto z].$$

  By Lemma 10 $z \in \text{RN}(V)$, so $z \notin \text{fn}(W)$ by the clash-freedom of U. By **PAR** $U \xrightarrow{M[\sigma]x[\sigma](z)}_\bullet V'|W$. Finally, $V'|W \equiv P'\sigma[y \mapsto z] \mid Q\sigma[y \mapsto z] \equiv R'\sigma[y \mapsto z]$.
- Suppose $R \xrightarrow{M\bar{x}(v)} R'$ is derived by **RES**. Then $R = (\nu y)P$, $P \xrightarrow{M\bar{x}(v)} P'$, $R' = (\nu y)P'$ and $y \notin \text{n}(M\bar{x}(v))$. Since $U \equiv R\sigma$, $U = (\nu z)Q$ with $z \notin \text{fn}((\nu y)P)\sigma)$ and $Q \equiv P\sigma[y \mapsto z]$. So $\text{fn}(M\bar{x}(v)[\sigma]) \cap (\text{RN}(Q) \cup \{z\}) = \emptyset$, $M[\sigma[y \mapsto z]] = M[\sigma]$ and $\bar{x}[\sigma[y \mapsto z]] = \bar{x}[\sigma]$. Moreover, Q is clash-free by Lemma 7. By induction,

  $$\exists w, Q'.\ Q \xrightarrow{M[\sigma]\bar{x}[\sigma](w)}_\bullet Q' \equiv P'\sigma[y \mapsto z][v \mapsto w].$$

  By Lemma 10, $w \in \text{RN}(Q)$, so $w \neq z$ by the clash-freedom of U. By Lemma 8, $\text{n}(M) \cup \{x\} \subseteq \text{fn}(P)$, so $\text{n}(M[\sigma]) \cup \{x[\sigma]\} \subseteq \text{fn}(((\nu y)P)\sigma) \not\ni z$. Hence, by **RES**,

  $$U \xrightarrow{M[\sigma]\bar{x}[\sigma](w)}_\bullet (\nu z)Q'.$$

  Using Lemmas 8 and 9, $\text{fn}(P') \subseteq \text{fn}(P) \cup \{v\}$ and thus $\text{fn}(((\nu y)P')\sigma[v \mapsto w]) \subseteq \text{fn}(((\nu y)P)\sigma[v \mapsto w]) \cup \{w\}$.

As $z \notin \mathrm{fn}((\nu y)P)\sigma) \wedge z \neq w$, $z \notin \mathrm{fn}(((\nu y)P')\sigma[v \mapsto w])$. It follows that

$$\begin{array}{rcll}(\nu z)Q' & \equiv & (\nu z)\big(P'\sigma[y \mapsto z][v \mapsto w]\big) & \\ & = & (\nu z)\big(P'\sigma[v \mapsto w][y \mapsto z]\big) & y \neq v \\ & \equiv & \big((\nu y)P'\big)\sigma[v \mapsto w] & \text{Corollary 1} \\ & = & R'\sigma[v \mapsto w]. & \blacksquare\end{array}$$

Now let $\mathscr{T}_\alpha$ be the identity translation from $\pi_{ES}(\mathcal{Z},\mathcal{R})$ to $\pi^{\not\alpha}_{ES}(\mathcal{Z},\mathcal{R})$, where $\pi^{\not\alpha}_{ES}(\mathcal{Z},\mathcal{R})$ is the variant of the $\pi_{ES}(\mathcal{Z},\mathcal{R})$ without rule **ALPHA**.

Theorem 3: If P is clash-free then $\mathscr{T}_\alpha(P) \nsim P$.

Proof. I have to provide a strong barbed bisimulation on the disjoint union of the LTSs of $\pi_{ES}(\mathcal{Z},\mathcal{R})$ and $\pi^{\not\alpha}_{ES}(\mathcal{Z},\mathcal{R})$. Since they have the same states, I make them disjoint by tagging each process in $\pi^{\not\alpha}_{ES}(\mathcal{Z},\mathcal{R})$ with a superscript $^{\not\alpha}$. Let

$$\mathscr{R} := \{(R, U^{\not\alpha}), (U^{\not\alpha}, R) \mid U \text{ is clash-free and } R \equiv U\}.$$

Clearly, $\mathscr{R}$ relates P and $\mathscr{T}_\alpha(P)$ for clash-free P, so it suffices to show that $\mathscr{R}$ is a strong barbed bisimulation. So suppose U is clash-free and $R \equiv U$, so that $U^{\not\alpha}\ \mathscr{R}\ R$ and $R\ \mathscr{R}\ U^{\not\alpha}$.

Let $U \xrightarrow{\tau}_\bullet Q$. Then $R \xrightarrow{\tau} Q$ by rule **ALPHA**. Moreover, Q is clash-free by Lemma 11, so $Q^{\not\alpha}\ \mathscr{R}\ Q$.

Let $R \xrightarrow{\tau} R'$. Then $\exists P.\ U \xrightarrow{\tau}_\bullet P \equiv R'$ by Lemma 14(i). Moreover, P is clash-free by Lemma 11, so $R'\ \mathscr{R}\ P^{\not\alpha}$.

Now let $U^{\not\alpha}{\downarrow_b}$ with $b \in \mathcal{Z} \cup \overline{\mathcal{Z}}$. Then $U \xrightarrow{by}_\bullet U'$ or $U \xrightarrow{b(y)}_\bullet U'$ for some y and U', using the definition of O in Section IV. So $R \xrightarrow{by} U'$ or $R \xrightarrow{b(y)} U'$ by rule **ALPHA**. Thus $R{\downarrow_b}$.

Finally, let $R{\downarrow_b}$ with $b \in \mathcal{Z} \cup \overline{\mathcal{Z}}$. Then $R \xrightarrow{by}_\bullet R'$ or $R \xrightarrow{b(y)}_\bullet R'$ for some y and R'. Hence $U \xrightarrow{by} U'$ or $U \xrightarrow{b(z)} U'$ for some z and U' by Lemma 14. Thus $U{\downarrow_b}$. $\blacksquare$

C. Eliminating restriction operators from the π-calculus

The fifth step $\mathscr{T}_\nu$ of my translation simply drops all restriction operators. It is defined compositionally by $\mathscr{T}_\nu((\nu y)P) = \mathscr{T}_\nu(P)$ and $\mathscr{T}_\nu(A) = A_\nu$, where A_ν is a fresh agent identifier with defining equation $A_\nu(\vec{x}) \stackrel{\mathrm{def}}{=} \mathscr{T}_\nu(P)$ when $A(\vec{x}) \stackrel{\mathrm{def}}{=} P$ was the defining equation of A; the translation $\mathscr{T}_\nu$ acts homomorphically on all other constructs.

The source of this translation step is the well-typed fragment of $\pi^{\not\alpha}_{ES}(\mathcal{Z},\mathcal{R})$. Its target is $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$, a calculus that differs in three ways from $\pi^{\not\alpha}_{ES}(\mathcal{Z},\mathcal{R})$. First of all it only allows well-typed-processes. Secondly, it allows defining equations

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

with $\mathrm{fn}(P) \subseteq \{x_1,\ldots,x_n\} \cup \mathcal{R}$, which is more liberal than the restriction $\mathrm{fn}(P) \subseteq \{x_1,\ldots,x_n\}$ imposed by $\pi^{\not\alpha}_{ES}(\mathcal{Z},\mathcal{R})$.

Intuitively, one may think of such an equation as

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

in the sense that besides the declared names x_i also all names in $\mathcal{R}$ are implicitly declared. A call $A(\vec{y})$ of agent A can likewise be seen as a call $A(\vec{y},\mathcal{R})$. Crucial here is that although $\vec{y}$ is being substituted for $\vec{x}$, the implicit substitution of $\mathcal{R}$ for $\mathcal{R}$ is the identity. To make that stick, the restriction to well-typed processes has been imposed. All substitutions $\{z/y\}$ and $\{\vec{y}/\vec{x}\}$ that are induced by the operational semantics, namely in rules **EARLY-INPUT** and **IDE**, have the property that $y \in \mathcal{Z}$, respectively $x_i \in \mathcal{Z}$. So there never is a need to apply a substitution renaming an element of $\mathcal{R}$.

The third and last difference between $\pi^{\not\alpha}_{ES}(\mathcal{Z},\mathcal{R})$ and $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ is that the latter does not feature restriction operators. Hence there is no need for rules **RES** and **SYMB-OPEN**. Since the resulting semantics cannot generate transitions labelled $M\bar{x}(z)$, rule **E-S-CLOSE** can be dropped as well. Moreover, $\mathrm{bn}(\alpha)=\emptyset$ for all transition labels α. So the side condition of rule **PAR** can be dropped too. I also leave out **ALPHA**.

Lemma 15: Let P and Q be $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ processes. If $P \equiv Q$ and $P \xrightarrow{\alpha} P'$ then $Q \xrightarrow{\alpha} Q'$ for some Q' with $P' \equiv Q'$. Moreover, the size of the derivation of $Q \xrightarrow{\alpha} Q'$ is the same as that of $P \xrightarrow{\alpha} P'$.

Proof. A trivial induction on the inference of $P \xrightarrow{\alpha} P'$. $\blacksquare$

This testifies that **ALPHA** is redundant. Thus, in total, all orange parts of Table VI are dropped.

I proceed to prove the validity of translation $\mathscr{T}_\nu$. For α an action in $\pi^{\not\alpha}_{ES}(\mathcal{Z},\mathcal{R})$ one defines the *debinding* of α by $\langle\alpha\rangle{:=}\alpha$ if α has the form $M\tau$, Mxz or $M\bar{x}y$, and $\langle M\bar{x}(y)\rangle := M\bar{x}y$.

Lemma 16: If R is clash-free and $R \xrightarrow{\alpha}_\bullet R'$, where α is not of the form Mxz with $z{\in}\mathrm{RN}(R)$, then $\mathscr{T}_\nu(R) \xrightarrow{\langle\alpha\rangle}{\equiv}\ \mathscr{T}_\nu(R')$.

Proof. By induction on the derivation of $R \xrightarrow{\alpha}_\bullet R'$.

- The cases that $R \xrightarrow{\alpha}_\bullet R'$ is derived by **TAU** or **OUTPUT** are trivial.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **EARLY-INPUT**. Then $R = Mx(y).P$, $\alpha = Mxz$, $R' = P\{z/y\}$ and $\mathscr{T}_\nu(R) = Mx(y).\mathscr{T}_\nu(P)$. Moreover, $\mathscr{T}_\nu(R) \xrightarrow{\langle\alpha\rangle} \mathscr{T}_\nu(P)\{z/y\}$. Thus it suffices to show that $\mathscr{T}_\nu(P)\{z/y\} \equiv \mathscr{T}_\nu(P\{z/y\})$. This holds because the substitution $\{z/y\}$ is clash-free on P, using Lemma 6.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **IDE**. Then $R = A(\vec{y})$ with $A(\vec{x}) \stackrel{\mathrm{def}}{=} P$, so $P\{\vec{y}/\vec{x}\} \xrightarrow{\alpha}_\bullet R'$. Moreover $\mathscr{T}_\nu(R) = A_\nu(\vec{y})$, with A_ν defined by $A_\nu(\vec{x}) \stackrel{\mathrm{def}}{=} \mathscr{T}_\nu(P)$. By Lemma 7 $P\{\vec{y}/\vec{x}\}$ is clash-free. Since $\mathrm{RN}(P\{\vec{y}/\vec{x}\}) = \mathrm{RN}(P) = \mathrm{RN}(R)$, α is not of the form Mxz with $z \in \mathrm{RN}(P\{\vec{y}/\vec{x}\})$. So by induction $\mathscr{T}_\nu(P\{\vec{y}/\vec{x}\}) \xrightarrow{\langle\alpha\rangle}{\equiv}\mathscr{T}_\nu(R')$. By Lemma 5, the substitution $\{\vec{y}/\vec{x}\}$ is clash-free on P. Hence $\mathscr{T}_\nu(P\{\vec{y}/\vec{x}\}) \equiv \mathscr{T}_\nu(P)\{\vec{y}/\vec{x}\}$. Lemma 15 yields $\mathscr{T}_\nu(P)\{\vec{y}/\vec{x}\} \xrightarrow{\langle\alpha\rangle}{\equiv}\ \mathscr{T}_\nu(R')$. Applying rule **IDE** yields $\mathscr{T}_\nu(R) \xrightarrow{\langle\alpha\rangle}{\equiv}\ \mathscr{T}_\nu(R')$.
- The cases that $R \xrightarrow{\alpha}_\bullet R'$ is derived by **SUM**, **SYMB-MATCH** or **PAR** are trivial.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **E-S-COM**. Then $R = P|Q$, $\alpha = [x{=}v]MN\tau$, $P \xrightarrow{M\bar{x}y}_\bullet P'$, $Q \xrightarrow{Nvy}_\bullet Q'$, $R' = P'|Q'$ and $\mathscr{T}_\nu(R) = \mathscr{T}_\nu(P)|\mathscr{T}_\nu(Q)$. By Lemma 7, P and Q are clash-free. By Lemma 8, $y \in \mathrm{fn}(P) \subseteq \mathrm{fn}(R)$, so $y \notin \mathrm{RN}(Q)$ by the clash-freedom of R. By induction, $\mathscr{T}_\nu(P) \xrightarrow{M\bar{x}y}{\equiv}\ \mathscr{T}_\nu(P')$ and $\mathscr{T}_\nu(Q) \xrightarrow{Nvy}{\equiv}\ \mathscr{T}_\nu(Q')$. Thus $\mathscr{T}_\nu(R) \xrightarrow{\langle\alpha\rangle}{\equiv}\ \mathscr{T}_\nu(R')$ by application of rule **E-S-COM**.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **E-S-CLOSE**. Then $R = P|Q$, $\alpha = [x{=}v]MN\tau$, $P \xrightarrow{M\bar{x}(z)}_\bullet P'$, $Q \xrightarrow{Nvz}_\bullet Q'$ with

$z \notin \mathrm{fn}(Q)$, $R' = P'|Q'$, and $\mathscr{T}_\nu(R) = \mathscr{T}_\nu(P)|\mathscr{T}_\nu(Q)$. By Lemma 7, P and Q are clash-free. By Lemma 10, $z \in \mathrm{RN}(P)$, so $z \notin \mathrm{RN}(Q)$ by the clash-freedom of R. Consequently, by induction, $\mathscr{T}_\nu(P) \xrightarrow{M\bar{x}z} \equiv \mathscr{T}_\nu(P')$ and $\mathscr{T}_\nu(Q) \xrightarrow{Nvz} \equiv \mathscr{T}_\nu(Q')$. Thus $\mathscr{T}_\nu(R) \xrightarrow{\langle\alpha\rangle} \equiv \mathscr{T}_\nu(R')$ by application of rule **E-S-COM**.

- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **RES**. Then $R = (\nu y)P$, $R' = (\nu y)P'$, $P \xrightarrow{\alpha}_\bullet P'$ and $y \notin \mathrm{n}(\alpha)$. By Lemma 7, P is clash-free. Moreover, α is not of the form Mxz with $z \in \mathrm{RN}(P) \subseteq \mathrm{RN}(R)$. By induction $\mathscr{T}_\nu(P) \xrightarrow{\langle\alpha\rangle} \equiv \mathscr{T}_\nu(P')$. Now $\mathscr{T}_\nu(R) = \mathscr{T}_\nu(P) \xrightarrow{\langle\alpha\rangle} \equiv \mathscr{T}_\nu(P') = \mathscr{T}_\nu(R')$.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **SYMB-OPEN**. Then $R = (\nu y)P$, $\alpha = M\bar{x}(y)$ and $P \xrightarrow{M\bar{x}y}_\bullet R'$. By Lemma 7, P is clash-free. By induction $\mathscr{T}_\nu(P) \xrightarrow{M\bar{x}y} \equiv \mathscr{T}_\nu(R')$. Now $\mathscr{T}_\nu(R) = \mathscr{T}_\nu(P) \xrightarrow{\langle\alpha\rangle} \equiv \mathscr{T}_\nu(R')$. ■

The next lemma makes use of the set $\mathrm{no}(\beta)$ of *non-output* names of an action β in $\pi^\dagger_{ES}(\mathcal{Z},\mathcal{R})$. Here $\mathrm{no}(\beta) := \mathrm{n}(\beta)$ if β has the form $M\tau$ or Mxz, whereas $\mathrm{no}(M\bar{x}y) := \mathrm{n}(M) \cup \{x\}$.

Lemma 17: If R is clash-free and $\mathscr{T}_\nu(R) \xrightarrow{\beta} U$, where $\mathrm{no}(\beta) \cap \mathrm{RN}(R) = \emptyset$, then $R \xrightarrow{\alpha}_\bullet R'$ for some α and R' with $\langle\alpha\rangle = \beta$ and $\mathscr{T}_\nu(R') \equiv U$.

Proof. By induction on the derivation of $\mathscr{T}_\nu(R) \xrightarrow{\beta} U$, and a nested structural induction on R.

- The cases $R = M\tau.P$ and $R = M\bar{x}y.P$ are trivial.
- Let $R = Mx(y).P$. Then $\mathscr{T}_\nu(R) = Mx(y).\mathscr{T}_\nu(P)$. Hence $\beta = Mxz$ and $U = \mathscr{T}_\nu(P)\{z/y\}$. Furthermore, $R \xrightarrow{Mxz}_\bullet P\{z/y\}$. Finally, $\mathscr{T}_\nu(P\{z/y\}) \equiv \mathscr{T}_\nu(P)\{z/y\}$, since $\{z/y\}$ is clash-free on P, using Lemma 6.
- Let $R = A(\vec{y})$ with $A(\vec{x}) \stackrel{\mathrm{def}}{=} P$. Then $\mathscr{T}_\nu(R) = A_\nu(\vec{y})$ with $A_\nu(\vec{x}) \stackrel{\mathrm{def}}{=} \mathscr{T}_\nu(P)$. So $\mathscr{T}_\nu(P)\{\vec{y}/\vec{x}\} \xrightarrow{\beta} U$. By Lemma 5, the substitution $\{\vec{y}/\vec{x}\}$ is clash-free on P. Hence $\mathscr{T}_\nu(P)\{\vec{y}/\vec{x}\} \equiv \mathscr{T}_\nu(P\{\vec{y}/\vec{x}\})$ and Lemma 15 yields $\mathscr{T}_\nu(P\{\vec{y}/\vec{x}\}) \xrightarrow{\beta} \equiv U$. By Lemma 7 $P\{\vec{y}/\vec{x}\}$ is clash-free. Since $\mathrm{RN}(P\{\vec{y}/\vec{x}\}) = \mathrm{RN}(P) = \mathrm{RN}(R)$ one has $\mathrm{no}(\beta) \cap \mathrm{RN}(P\{\vec{y}/\vec{x}\}) = \emptyset$. So by induction $P\{\vec{y}/\vec{x}\} \xrightarrow{\alpha}_\bullet R'$ for some α and R' with $\langle\alpha\rangle = \beta$ and $\mathscr{T}_\nu(R') \equiv U$. Now $R \xrightarrow{\alpha}_\bullet R'$ by **IDE**.
- Let $R = P|Q$. Then $\mathscr{T}_\nu(R) = \mathscr{T}_\nu(P)|\mathscr{T}_\nu(Q)$. Suppose $\mathscr{T}_\nu(R) \xrightarrow{\beta} U$ is derived by **PAR**. Then $\mathscr{T}_\nu(P) \xrightarrow{\beta} V$ and $U = V|\mathscr{T}_\nu(Q)$. By Lemma 7 P is clash-free. Since $\mathrm{RN}(P) \subseteq \mathrm{RN}(P|Q)$, $\mathrm{no}(\beta) \cap \mathrm{RN}(P) = \emptyset$. So by induction $P \xrightarrow{\alpha}_\bullet P'$ for some α and P' with $\langle\alpha\rangle = \beta$ and $\mathscr{T}_\nu(P') \equiv V$. By Lemma 10 $\mathrm{bn}(\alpha) \subseteq \mathrm{RN}(P) \subseteq \mathrm{RN}(R)$, so $\mathrm{bn}(\alpha) \cap \mathrm{fn}(Q) = \emptyset$ by the clash-freedom of R. Thus $R \xrightarrow{\alpha}_\bullet P'|Q$ by **PAR**, and $\mathscr{T}_\nu(P'|Q) \equiv V|\mathscr{T}_\nu(Q) = U$.

  Now suppose $\mathscr{T}_\nu(R) \xrightarrow{\beta} U$ is derived by **E-S-COM**. Then $\beta = [x{=}v]MN\tau$, $\mathscr{T}_\nu(P) \xrightarrow{M\bar{x}y} V$, $\mathscr{T}_\nu(Q) \xrightarrow{Nvy} W$ and $U = V|W$. By Lemma 7 P and Q are clash-free. Since $\mathrm{RN}(P) \subseteq \mathrm{RN}(P|Q)$, $\mathrm{no}(M\bar{x}y) \cap \mathrm{RN}(P) = \emptyset$. So by induction either $P \xrightarrow{M\bar{x}y}_\bullet P'$ or $P \xrightarrow{M\bar{x}(y)}_\bullet P'$ for some P' with $\mathscr{T}_\nu(P') \equiv V$.

  In the first case $y \in \mathrm{fn}(P) \subseteq \mathrm{fn}(R)$ by Lemma 8, so $y \notin \mathrm{RN}(R) \supseteq \mathrm{RN}(Q)$ by the clash-freedom of R. Hence also $\mathrm{no}(Nvy) \cap \mathrm{RN}(Q) = \emptyset$. By induction $Q \xrightarrow{Nvy}_\bullet Q'$ for some Q' with $\mathscr{T}_\nu(Q') = W$. So $R \xrightarrow{\alpha}_\bullet P'|Q'$ by **E-S-COM**, and $\mathscr{T}_\nu(P'|Q') \equiv V|W = U$.

  In the second case $y \in \mathrm{RN}(P) \subseteq \mathrm{RN}(R)$ by Lemma 10, so $y \notin \mathrm{fn}(Q) \cup \mathrm{RN}(Q)$ by the clash-freedom of R. Hence $\mathrm{no}(Nvy) \cap \mathrm{RN}(Q) = \emptyset$. By induction $Q \xrightarrow{Nvy}_\bullet Q'$ for some Q' with $\mathscr{T}_\nu(Q') = W$. So $R \xrightarrow{\alpha}_\bullet (\nu y)(P'|Q')$ by **E-S-CLOSE**, and $\mathscr{T}_\nu((\nu y)(P'|Q')) = \mathscr{T}_\nu(P'|Q') \equiv V|W = U$.
- Finally, let $R = (\nu y)P$. Then $\mathscr{T}_\nu(R) = \mathscr{T}_\nu(P)$. Moreover, $\mathrm{no}(\beta) \cap \mathrm{RN}(P) = \emptyset$. By induction, $P \xrightarrow{\alpha}_\bullet P'$ for some α and P' with $\langle\alpha\rangle = \beta$ and $\mathscr{T}_\nu(P') \equiv U$.

  In case $y \notin \mathrm{n}(\alpha)$, $R \xrightarrow{\alpha}_\bullet (\nu y)P'$ by **RES**. Moreover, $\mathscr{T}_\nu((\nu y)(P')) = \mathscr{T}_\nu(P') \equiv U$.

  In case $y \in \mathrm{n}(\alpha) = \mathrm{n}(\beta)$, using that $\mathrm{RN}(R) \ni y \notin \mathrm{no}(\beta)$, β must have the form $M\bar{x}y$ with $y \neq x$ and $y \notin \mathrm{n}(M)$. So α is either $M\bar{x}(y)$ or $M\bar{x}y$. If $\alpha = M\bar{x}(y)$ then $y \in \mathrm{RN}(P)$ by Lemma 10, contradicting the clash-freedom of R. So $\alpha = M\bar{x}y$. Now $R \xrightarrow{M\bar{x}(y)}_\bullet P'$ by **SYMB-OPEN**. ■

Lemma 18: Let P be a process in $\pi_{ES}(\mathcal{Z},\mathcal{R})$. If $P \xrightarrow{Mxy}_\bullet Q$ then $\exists R.\ P \xrightarrow{Mxz}_\bullet R$ for any $z \notin \mathrm{RN}(P)$.

Proof. A trivial induction on the inference of $P \xrightarrow{Mxy} Q$. ■

Theorem 4: If P is clash-free then $\mathscr{T}_\nu(P) \mathrel{\dot{\sim}} P$.

Proof. Let

$$\mathscr{R} := \left\{ (R,U),(U,R) \;\middle|\; \begin{array}{l} R \text{ in } \pi^{\not\alpha}_{ES}(\mathcal{Z},\mathcal{R}) \text{ is clash-free} \\ \text{and } U \equiv \mathscr{T}_\nu(R) \end{array} \right\}.$$

It suffices to show that $\mathscr{R}$ is a strong barbed bisimulation. So suppose R is clash-free and $U \equiv \mathscr{T}_\nu(R)$.

Let $R \xrightarrow{\tau}_\bullet R'$. Then $\mathscr{T}_\nu(R) \xrightarrow{\tau} \equiv \mathscr{T}_\nu(R')$ by Lemma 16, and $U \xrightarrow{\tau} U' \equiv \mathscr{T}_\nu(R')$ by Lemma 15. Moreover, R' is clash-free by Lemma 11, so $R' \mathscr{R} U'$.

Let $U \xrightarrow{\tau} U'$. Then $\mathscr{T}_\nu(R) \xrightarrow{\tau} \equiv U'$ by Lemma 15. Thus, by Lemma 17, $R \xrightarrow{\tau}_\bullet R'$ for some R' with $\mathscr{T}_\nu(R') \equiv U'$. Moreover, R' is clash-free by Lemma 11, so $U' \mathscr{R} R'$.

Now let $R{\downarrow_b}$ with $b \in \mathcal{Z} \cup \overline{\mathcal{Z}}$. Then $R \xrightarrow{by}_\bullet R'$ or $R \xrightarrow{b(y)}_\bullet R'$ for some y and R', using the definition of O in Section IV. By Lemma 18 I may assume, w.l.o.g., that if $b \in \mathcal{Z}$ then $y \notin \mathrm{RN}(R)$. So $\mathscr{T}_\nu(R) \xrightarrow{by} \equiv \mathscr{T}_\nu(R')$ by Lemma 16, and $U \xrightarrow{by} \equiv \mathscr{T}_\nu(R')$ by Lemma 15. Thus $U{\downarrow_b}$.

Finally, let $U{\downarrow_b}$ with $b = x \in \mathcal{Z}$ or $b = \bar{x}$ with $x \in \overline{\mathcal{Z}}$. Then $U \xrightarrow{by} U'$ for some y and U'. So $\mathscr{T}_\nu(R) \xrightarrow{by} U''$ for some y and U'' by Lemma 15. Since R is well-typed, $\mathrm{RN}(R) \subseteq \mathcal{R}$, so $x \notin \mathrm{RN}(P)$. By Lemma 18 I may assume, w.l.o.g., that if $b = x$ then $y \notin \mathrm{RN}(R)$. Hence $\mathrm{no}(by) \cap \mathrm{RN}(R) = \emptyset$. Thus, by Lemma 17, $R \xrightarrow{by}_\bullet R'$ or $R \xrightarrow{b(y)}_\bullet R'$ for some R'. Hence $R{\downarrow_b}$. ■

D. Replacing substitution by relabelling

Recall that $\pi^\dagger_{ES}(\mathcal{Z},\mathcal{R})$ is the version of the π-calculus without restriction, equipped with the early symbolic operational semantics (Table VI without the orange rules), using $\mathcal{Z} \uplus \mathcal{R}$ as the set of names, subject to the following typing restrictions: (i) each binder $x(z)$ occurring in a process satisfies $z \in \mathcal{Z}$, and

(ii) each defining equation $A(x_1...,x_n) \stackrel{\text{def}}{=} P$ satisfies $x_i \in \mathcal{Z}$ and $\text{fn}(P) \subseteq \{x_1,\ldots,x_n\} \cup \mathcal{R}$.

$\pi^{\ddagger-}_{ES}(\mathcal{Z},\mathcal{R})$ is the variant of $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ to which has been added a relabelling operator $[\sigma]$ for each substitution σ with $dom(\sigma)$ finite and $dom(\sigma) \cap \mathcal{R} = \emptyset$; its structural operation semantics is given by

$$\frac{P \stackrel{\alpha}{\longrightarrow} P'}{P[\sigma] \stackrel{\alpha[\sigma]}{\longrightarrow} P'[\sigma]}\,.$$

Moreover, the substitutions $\{z/y\}$ and $\{\vec{y}/\vec{x}\}$ that appear in rules **early-input** and **ide** are replaced by applications of the relabelling operators $[\{z/y\}]$ and $[\{\vec{y}/\vec{x}\}]$, respectively.

This section defines a collection of "clash-free" $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ processes, and shows that on clash-free processes the identity translation from $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ to $\pi^{\ddagger-}_{ES}(\mathcal{Z},\mathcal{R})$ preserves $\leftrightarrows$.

Lemma 19: Let R be a $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ process and σ a substitution with $dom(\sigma)$ finite and $dom(\sigma) \cap \mathcal{R} = \emptyset$. If $R \stackrel{\alpha}{\longrightarrow} R'$ then $R\sigma \stackrel{\alpha[\sigma]}{\longrightarrow} \equiv R'\sigma$.

Proof. With induction on the derivation of $R \stackrel{\alpha}{\longrightarrow} R'$.

- The cases that $R \stackrel{\alpha}{\longrightarrow} R'$ is derived by rule **tau** or **output** are trivial.
- Suppose $R \stackrel{\alpha}{\longrightarrow} R'$ is derived by rule **early-input**. Then $R = Mx(y).P$, $\alpha = Mxz$ and $R' = P\{z/y\}$. So $R\sigma = M[\sigma]x[\sigma](w).(P\{w/y\}\sigma) \wedge \alpha[\sigma] = M[\sigma]x[\sigma]z[\sigma]$ where w is chosen outside $\text{fn}((\nu y)P) \cup dom(\sigma) \cup range(\sigma)$. By **early-input** $R\sigma \stackrel{\alpha[\sigma]}{\longrightarrow} P\{w/y\}\sigma\{z[\sigma]/w\} \equiv P\{z/y\}\sigma$.
- Suppose $R = A(\vec{y})$. Let $A(\vec{x}) \stackrel{\text{def}}{=} P$. Say $\vec{x} = (x_1,\ldots,x_n)$. Then $P\{\vec{y}/\vec{x}\} \stackrel{\alpha}{\longrightarrow} R'$, so by induction $P\{\vec{y}/\vec{x}\}\sigma \stackrel{\alpha[\sigma]}{\longrightarrow} \equiv R'\sigma$. Moreover, $R\sigma = A(\vec{y}[\sigma])$ and $P\{\vec{y}[\sigma]/\vec{x}\} \equiv P\{\vec{y}/\vec{x}\}\sigma$. Here I use that $\text{fn}(P) \cap dom(\sigma) \subseteq \{x_1,\ldots,x_n\}$. So $P\{\vec{y}[\sigma]/\vec{x}\} \stackrel{\alpha[\sigma]}{\longrightarrow} \equiv R'\sigma$ by Lemma 15. Thus, by rule **ide**, $R\sigma \stackrel{\alpha[\sigma]}{\longrightarrow} \equiv R'\sigma$.
- The cases that $R \stackrel{\alpha}{\longrightarrow} R'$ is derived by rule **sum**, **symb-match**, **par** or **e-s-com** are trivial. ■

Define the *input arguments* $\iota(\alpha)$ of an action α by

$$\iota(M\bar{x}y) = \iota(M\tau) := \emptyset \quad \text{and} \quad \iota(Mxz) = \{z\}.$$

Lemma 20: Let P be a $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ process. If $P \stackrel{\alpha}{\longrightarrow} Q$ then $\text{n}(\alpha)\backslash\iota(\alpha) \subseteq \text{fn}(P) \cup \mathcal{R}$ and $\text{fn}(Q) \subseteq \text{fn}(P) \cup \iota(\alpha) \cup \mathcal{R}$.

Proof. A trivial induction on the inference of $P \stackrel{\alpha}{\longrightarrow} Q$. ■

Lemmas 8 and 9 make the same statements for the calculus $\pi_{ES}(\mathcal{Z},\mathcal{R})$, but without the additions $\cup\mathcal{R}$. These additions are needed for the case of recursion, because the bodies of defining equations may introduce names from $\mathcal{R}$.

Lemma 21: Let P be a $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ process. If $P \stackrel{Mxz}{\longrightarrow} P_z$ and $w \notin \text{fn}(P)$ then there is a process P_w such that $P \stackrel{Mxw}{\longrightarrow} P_w$ and $P_z \equiv P_w\{z/w\}$. Moreover, the size of the derivation of $P \stackrel{Mxw}{\longrightarrow} P_w$ is the same as that of $P \stackrel{Mxz}{\longrightarrow} P_z$.

Proof. A trivial induction on the inference of $P \stackrel{Mxz}{\longrightarrow} P_z$. ■

Lemma 22: Let P be a $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ process. If $P \stackrel{Mxw}{\longrightarrow} P_w$ with $w \notin \text{fn}(P)$ then for each name z there is a process P_z such that $P \stackrel{Mxz}{\longrightarrow} P_z$ and $P_z \equiv P_w\{z/w\}$. Moreover, the size of the derivation of $P \stackrel{Mxz}{\longrightarrow} P_z$ is the same as that of $P \stackrel{Mxw}{\longrightarrow} P_w$.

Proof. A trivial induction on the inference of $P \stackrel{Mxw}{\longrightarrow} P_w$. ■

Lemma 23: Let R be a $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ process and σ a substitution with $dom(\sigma)$ finite and $dom(\sigma) \cap \mathcal{R} = \emptyset$. If $R\sigma \stackrel{\beta}{\longrightarrow} U$ with $\iota(\beta) \cap dom(\sigma) \subseteq range(\sigma)$ then $R \stackrel{\alpha}{\longrightarrow} R'$ for some α and R' with $\alpha[\sigma] = \beta$ and $R'\sigma \equiv U$. Moreover, the size of the derivation of $R \stackrel{\alpha}{\longrightarrow} R'$ is the same as that of $R\sigma \stackrel{\beta}{\longrightarrow} U$.

Proof. With induction on the size of the derivation of $R\sigma \stackrel{\beta}{\longrightarrow} U$.

- The cases that $R\sigma \stackrel{\beta}{\longrightarrow} U$ is derived by rule **tau** or **output** are trivial.
- Suppose $R\sigma \stackrel{\beta}{\longrightarrow} U$ is derived by **early-input**. Then $R = Mx(y).P$ and $R\sigma = M[\sigma]x[\sigma](w).(P\{w/y\}\sigma)$ with $w \notin \text{fn}((\nu y)P) \cup dom(\sigma) \cup range(\sigma)$. So $\beta = M[\sigma]x[\sigma]v$ and $U = P\{w/y\}\sigma\{v/w\}$. Since $v \in \iota(\beta)$, there is a z with $z[\sigma] = v$. By **early-input** $R \stackrel{\alpha}{\longrightarrow} R'$ with $\alpha = Mxz$ and $R' = P\{z/y\}$. Now $\alpha[\sigma] = \beta$ and $R'\sigma \equiv U$.
- Suppose $R\sigma \stackrel{\beta}{\longrightarrow} U$ is derived by **ide**. Then $R = A(\vec{y})$ and $R\sigma = A(\vec{y}[\sigma])$. Let $A(\vec{x}) \stackrel{\text{def}}{=} P$. Say $\vec{x} = (x_1,\ldots,x_n)$. Then $P\{\vec{y}[\sigma]/\vec{x}\} \stackrel{\beta}{\longrightarrow} U$. Since $P\{\vec{y}[\sigma]/\vec{x}\} \equiv P\{\vec{y}/\vec{x}\}\sigma$, using that $\text{fn}(P) \cap dom(\sigma) \subseteq \{x_1,\ldots,x_n\}$, Lemma 15 yields $P\{\vec{y}/\vec{x}\}\sigma \stackrel{\beta}{\longrightarrow} \equiv U$. So by induction $P\{\vec{y}/\vec{x}\} \stackrel{\alpha}{\longrightarrow} R'$ for some α and R' with $\alpha[\sigma] = \beta$ and $R'\sigma \equiv U$. By rule **ide** $R \stackrel{\alpha}{\longrightarrow} R'$.
- The cases that $R\sigma \stackrel{\beta}{\longrightarrow} U$ is derived by rule **sum**, **symb-match** or **par** are trivial.
- Suppose $R\sigma \stackrel{\beta}{\longrightarrow} U$ is derived by **e-s-com**. Then $R = P|Q$, $R\sigma = P\sigma|Q\sigma$, $\beta = [x{=}v]MN\tau$, $P\sigma \stackrel{M\bar{x}y}{\longrightarrow} V$, $Q\sigma \stackrel{Nvy}{\longrightarrow} W$ and $U = V|W$. By Lemma 20, $y \in \text{fn}(P\sigma)$, so $y \in range(\sigma)$ or $y \notin dom(\sigma)$. Hence $\iota(Nvy) \cap dom(\sigma) \subseteq range(\sigma)$. By induction, there are matching sequences K, L with $K[\sigma] = M$ and $L[\sigma] = N$, names q, r, z, u with $q[\sigma] = x$, $r[\sigma] = v$, $z[\sigma] = y$ and $u[\sigma] = y$, and processes P' and Q' with $P'\sigma \equiv V$ and $Q'\sigma \equiv W$, such that $P \stackrel{K\bar{q}z}{\longrightarrow} P'$ and $Q \stackrel{Lru}{\longrightarrow} Q'$. Pick $w \notin \text{fn}(Q)$. By Lemma 21 there is a process P_w such that $Q \stackrel{Lrw}{\longrightarrow} Q_w$ and $Q' \equiv Q_w\{u/w\}$. By Lemma 22 there is a process P_z such that $Q \stackrel{Lrz}{\longrightarrow} Q_z$ and $Q_z \equiv Q_w\{z/w\}$. Note that $Q'\sigma \equiv Q_z\sigma$. By **e-s-com** $R \stackrel{[q=r]KL\tau}{\longrightarrow} P'|Q_z$. Moreover, $([q{=}r]KL\tau)[\sigma] = [x{=}v]MN\tau$ and $(P'|Q_z)\sigma \equiv V|W = U$. ■

For P a $\pi^{\ddagger-}_{ES}(\mathcal{Z},\mathcal{R})$ process, let $\widehat{P}$ be the $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ process obtained from P by recursively replacing each subterm $Q[\sigma]$ by $Q\sigma$, and each agent identifier A by $A\hat{}$. Here $A\hat{}$ is a fresh agent identifier with defining equation $A\hat{}(\vec{x}) \stackrel{\text{def}}{=} \widehat{P}$ when $A(\vec{x}) \stackrel{\text{def}}{=} P$ was the defining equation of A.

Lemma 24: If $R \stackrel{\alpha}{\longrightarrow} R'$ then $\widehat{R} \stackrel{\alpha}{\longrightarrow} \equiv \widehat{R'}$.

Proof. By induction of the inference of $R \stackrel{\alpha}{\longrightarrow} R'$.

- Suppose $R \stackrel{\alpha}{\longrightarrow} R'$ is derived by **tau**. Then $R = M\tau.P$, $\alpha = M\tau$ and $R' = P$. Moreover, $\widehat{R} = M\tau.\widehat{P}$ and $\widehat{R} \stackrel{\alpha}{\longrightarrow} \widehat{R'}$.

- The case that $R \xrightarrow{\alpha} R'$ is derived by **OUTPUT** proceeds likewise.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by **EARLY INPUT**. Then $R = Mx(y).P$, $\alpha = Mxz$ and $R' = P[z/y]$. Moreover $\widehat{R} = Mx(y).\widehat{P}$ and $\widehat{R} \xrightarrow{\alpha} \widehat{P}\{z/y\} = \widehat{P[\{z/y\}]} = \widehat{R'}$.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by **SUM**. Then $R = P + Q$ and $P \xrightarrow{\alpha} R'$. Now $\widehat{R} = \widehat{P} + \widehat{Q}$. By induction $\widehat{P} \xrightarrow{\alpha} \equiv \widehat{R'}$. Hence, by **SUM**, $\widehat{R} \xrightarrow{\alpha} \equiv \widehat{R'}$.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by **SYMB-MATCH**. Then $R = [x{=}y]P$, $P \xrightarrow{\beta} R'$ and $\alpha = [x{=}y]\beta$. Now $\widehat{R} = [x{=}y]\widehat{P}$. By induction $\widehat{P} \xrightarrow{\beta} \equiv \widehat{R'}$. Hence, by **SYMB-MATCH**, $\widehat{R} \xrightarrow{\alpha} \equiv \widehat{R'}$.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by **IDE**. Then $R = A(\vec{y})$ with $A(\vec{x}) \stackrel{\text{def}}{=} P$ and $P[\{\vec{y}/\vec{x}\}] \xrightarrow{\alpha} R'$. Now $\widehat{P[\{\vec{y}/\vec{x}\}]} = \widehat{P}\{\vec{y}/\vec{x}\}$. By induction $\widehat{P[\{\vec{y}/\vec{x}\}]} = \widehat{P}\{\vec{y}/\vec{x}\} \xrightarrow{\alpha} \equiv \widehat{R'}$. Hence, by **IDE**, $\widehat{R} = \widehat{A}(\vec{y}) \xrightarrow{\alpha} \equiv \widehat{R'}$.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by **PAR**. Then $R = P|Q$, $P \xrightarrow{\alpha} P'$ and $R' = P'|Q$. Now $\widehat{R} = \widehat{P}|\widehat{Q}$ and $\widehat{R'} = \widehat{P'}|\widehat{Q}$. By induction $\widehat{P} \xrightarrow{\alpha} \equiv \widehat{P'}$. Thus $\widehat{R} \xrightarrow{\alpha} \equiv \widehat{R'}$.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by **E-S-COM**. Then $R = P|Q$, $P \xrightarrow{M\bar{x}y} P'$, $Q \xrightarrow{Nvy} Q'$, $R' = P'|Q'$, and $\alpha = [x{=}v]MN\tau$. Now $\widehat{R} = \widehat{P}|\widehat{Q}$ and $\widehat{R'} = \widehat{P'}|\widehat{Q'}$. By induction $\widehat{P} \xrightarrow{M\bar{x}y} \equiv \widehat{P'}$ and $\widehat{Q} \xrightarrow{Nvy} \equiv \widehat{Q'}$. Thus $\widehat{R} \xrightarrow{\alpha} \equiv \widehat{R'}$.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by **RELABELLING**. Then $R{=}P[\sigma]$, $P \xrightarrow{\beta} P'$, $R'{=}P'[\sigma]$ and $\alpha{=}\beta[\sigma]$. By induction $\widehat{P} \xrightarrow{\beta} \equiv \widehat{P'}$. By Lemma 19, $\widehat{R} = \widehat{P}\sigma \xrightarrow{\alpha} \equiv \widehat{P'}\sigma = \widehat{R'}$. ■

For $A \in \mathcal{K}_n$ an agent identifier with $A(x_1, \ldots, x_n) \stackrel{\text{def}}{=} P$, let $\mathrm{dn}(A) = \{x_1, \ldots, x_n\}$ be the set of *declared names* of A.

Definition 12: Let $\mathrm{BN}(P)$, the *hereditary bound names* of a $\pi^{\ddagger-}_{ES}(\mathcal{Z}, \mathcal{R})$ process P, be the set of all names y such that $h(P)$ contains either a process $Mx(y)Q$, or a process $A(\vec{z})$ with $y \in \mathrm{dn}(A)$, or a process $Q[\sigma]$ with $y \in dom(\sigma)$.

Definition 13: The *free names* of a $\pi^{\ddagger-}_{ES}(\mathcal{Z}, \mathcal{R})$-process P are defined inductively as follows:

$$\begin{array}{rcl} \mathrm{fn}(\mathbf{0}) & = & \emptyset \\ \mathrm{fn}(M\tau.P) & = & \mathrm{n}(M) \cup \mathrm{fn}(P) \\ \mathrm{fn}(M\bar{x}y.P) & = & \mathrm{n}(M) \cup \{x, y\} \cup \mathrm{fn}(P) \\ \mathrm{fn}(Mx(y).P) & = & \mathrm{n}(M) \cup \{x\} \cup \mathrm{fn}(P) \backslash \{y\} \\ \mathrm{fn}([x{=}y]P) & = & \{x, y\} \cup \mathrm{fn}(P) \\ \mathrm{fn}(P + Q) = \mathrm{fn}(P \mid Q) & = & \mathrm{fn}(P) \cup \mathrm{fn}(Q) \\ \mathrm{fn}(A(y_1, \ldots, y_n)) & = & \{y_1, \ldots, y_n\} \\ \mathrm{fn}(P[\sigma]) & = & \{x[\sigma] \mid x \in \mathrm{fn}(P)\} \end{array}$$

In the absence of relabelling operators, this definition agrees with the one from Section V.

Definition 14: A $\pi^{\ddagger-}_{ES}(\mathcal{Z}, \mathcal{R})$ process P is *clash-free* if

1) for each $A(\vec{z}){\in}h(P)$ with $A(\vec{x}) \stackrel{\text{def}}{=} Q$ one has $x_i \notin \mathrm{BN}(Q)$,
2) for each $Q[\sigma] \in h(P)$ one has $dom(\sigma) \cap \mathrm{BN}(Q) = \emptyset$,
3) for each $Mx(y).Q \in h(P)$ one has $y \notin \mathrm{BN}(Q)$, and
4) $(\mathrm{fn}(P) \cup \mathcal{R}) \cap \mathrm{BN}(P) = \emptyset$.

This definition applies equally well to $\pi^{\dagger}_{ES}(\mathcal{Z}, \mathcal{R})$ processes; here Clause 2 is moot, as there are no relabelling operators in $\pi^{\dagger}_{ES}(\mathcal{Z}, \mathcal{R})$.

The relation $\leftharpoondown$ from Definition 11 applies to $\pi^{\ddagger-}_{ES}(\mathcal{Z}, \mathcal{R})$ as well, except that there is a clause $P[\sigma] \leftharpoondown P$, and the last clause is $A(\vec{y}) \leftharpoondown P[\{\vec{y}/\vec{x}\}]$ when $A(\vec{x}) \stackrel{\text{def}}{=} P$.

Lemma 25: If $P \leftharpoondown Q$ and P is a clash-free $\pi^{\ddagger-}_{ES}(\mathcal{Z}, \mathcal{R})$ process, then so is Q.

Proof. Only the cases of relabelling and recursion are non-trivial. For $P[\sigma] \leftharpoondown P$ use: $\mathrm{fn}(P) \subseteq \mathrm{fn}(P[\sigma]) \cup dom(\sigma)$ and $\mathrm{BN}(P) = \mathrm{BN}(P[\sigma]) \backslash dom(\sigma)$.

Suppose $A(\vec{y})$ is clash-free and $A(\vec{x}) \stackrel{\text{def}}{=} P$. By definition P satisfies Clauses 1–3. Hence $P[\{\vec{y}/\vec{x}\}]$ satisfies Clauses 1 and 3. Moreover $P[\{\vec{y}/\vec{x}\}]$ satisfies Clause 2 since $A(\vec{y})$ satisfies Clause 1. Finally, $\mathrm{BN}(P[\{\vec{y}/\vec{x}\}]) = \mathrm{BN}(P) \cup \{x_1, \ldots, x_n\} = \mathrm{BN}(A(\vec{y}))$ and $\mathrm{fn}(P[\{\vec{y}/\vec{x}\}]) \subseteq \mathrm{fn}(A(\vec{y})) \cup \mathcal{R}$, so Clause 4 holds too. ■

Note that Lemma 20 holds also for $\pi^{\ddagger-}_{ES}(\mathcal{Z}, \mathcal{R})$ processes.

Lemma 26: If R is clash-free and $\widehat{R} \xrightarrow{\alpha} U$ with $\iota(\alpha) \cap \mathrm{BN}(R) = \emptyset$ then $R \xrightarrow{\alpha} R'$ for some R' with $\widehat{R'} \equiv U$.

Proof. By induction on the size of the derivation of $\widehat{R} \xrightarrow{\alpha} U$, with a nested induction on the number of topmost renaming operators in R.

- Suppose that R is not of the form $P[\sigma]$. The cases that $\widehat{R} \xrightarrow{\alpha} U$ is derived by **TAU**, **OUTPUT**, **EARLY-INPUT**, **SUM**, **SYMB-MATCH**, **IDE** or **PAR** are trivial, similar to the cases spelled out in the proof of Lemma 24, but using Lemma 25 to establish clash-freedom when applying the induction hypothesis, and also using that $R \leftharpoondown P$ implies $\mathrm{BN}(P) \subseteq \mathrm{BN}(R)$.
  Suppose $\widehat{R} \xrightarrow{\alpha} U$ is derived by rule **E-S-COM**. Then $R = P|Q$, $\widehat{R}{=}\widehat{P}|\widehat{Q}$, $\alpha = [x{=}v]MN\tau$, $\widehat{P} \xrightarrow{M\bar{x}y} V$, $\widehat{Q} \xrightarrow{Nvy} W$ and $\widehat{R'} = V|W$. By Lemma 25 P and Q are clash-free. By induction $P \xrightarrow{M\bar{x}y} P'$ for some P' with $\widehat{P'} \equiv V$. By Lemma 20 $y \in \mathrm{fn}(P) \cup \mathcal{R} \subseteq \mathrm{fn}(R) \cup \mathcal{R}$, and hence $\iota(Nvy) \cap \mathrm{BN}(Q) = \emptyset$ by the clash-freedom of R. So by induction $Q \xrightarrow{Nvy} Q'$ for some Q' with $\widehat{Q'} \equiv W$. By **E-S-COM** $R \xrightarrow{\alpha} P'|Q' \equiv U$.
- Now suppose $R = P[\sigma]$. Then $\iota(\alpha) \cap dom(\sigma) = \emptyset$ and $\widehat{R} = \widehat{P}\sigma$. By Lemma 23 $\widehat{P} \xrightarrow{\beta} V$ for some β and V with $\beta[\sigma]{=}\alpha$ and $V\sigma{\equiv}U$. Moreover, the size of the derivation of $\widehat{P} \xrightarrow{\beta} V$ is the same as that of $\widehat{P}\sigma \xrightarrow{\alpha} U$. Suppose $\iota(\beta) = \{z\}$. Then either $z \in dom(\sigma)$, so $z \notin \mathrm{BN}(P)$ by the clash-freedom of R, or $\iota(\beta) = \iota(\alpha)$ and $z \notin \mathrm{BN}(R) \supseteq \mathrm{BN}(P)$. So by induction $P \xrightarrow{\beta} P'$ for some P' with $\widehat{P'} \equiv V$. By rule **RELABELLING** $R \xrightarrow{\alpha} P'[\sigma]$. Furthermore one has $\widehat{P'[\sigma]} = \widehat{P'}\sigma \equiv V\sigma \equiv U$. ■

Lemma 27: If R in $\pi^{\ddagger-}_{ES}(\mathcal{Z}, \mathcal{R})$ is clash-free, $\iota(\alpha) \cap \mathrm{BN}(R) = \emptyset$ and $R \xrightarrow{\alpha} R'$, then R' is clash-free and $\mathrm{BN}(R') \subseteq \mathrm{BN}(R)$.

Proof. First I show that $\mathrm{BN}(R') \subseteq \mathrm{BN}(R)$ implies that R' meets Clause 4 of Definition 14. That $\mathcal{R} \cap \mathrm{BN}(R') = \emptyset$ follows

since $\mathcal{R}\cap\text{BN}(R)=\emptyset$ by the clash-freedom of R. Now suppose $y\in\text{fn}(R')\cap\text{BN}(R')$. The case that $y\in\text{fn}(R)$ contradicts the clash-freedom of R. In view of Lemma 20, the only remaining case is that $y\in\iota(\alpha)$. However, in that case $y\notin\text{BN}(R)\supseteq\text{BN}(R')$ by the assumption of the lemma.

The rest of the proof proceeds with induction on the derivation of $R\xrightarrow{\alpha}R'$.

- The cases that $R\xrightarrow{\alpha}R'$ is derived by rule **TAU** or **EARLY-OUTPUT** are trivial, using that R' is a subexpression of R.
- Suppose $R\xrightarrow{\alpha}R'$ is derived by **EARLY-INPUT**. Then $R=Mx(y).P$, $\alpha=Mxz$ with $z\notin\text{BN}(R)$ and $R'=P[\{z/y\}]$. By definition, P satisfies Clauses 1–3 of Definition 14. Hence R' satisfies Clauses 1 and 3. Moreover R' satisfies Clause 2 since R satisfies Clause 3. Finally, $\text{BN}(R')=\text{BN}(P)\cup\{y\}=\text{BN}(R)$.
- The cases that $R\xrightarrow{\alpha}R'$ is derived by rule **SUM**, **IDE** or **SYMB-MATCH** are trivial.
- Suppose $R\xrightarrow{\alpha}R'$ is derived by **PAR**. Then $R=P|Q$, $P\xrightarrow{\alpha}P'$, $R'=P'|Q$, and if $\alpha=Mxz$ then $z\notin\text{BN}(P)$. By Lemma 25, P and Q are clash free. So by induction P' is clash-free and $\text{BN}(P')\subseteq\text{BN}(P)$. Hence $\text{BN}(R')\subseteq\text{BN}(R)$ and R' is clash-free.
- Suppose $R\xrightarrow{\alpha}R'$ is derived by **E-S-COM**. Then $R=P|Q$, $\alpha=[x{=}v]MN\tau$, $P\xrightarrow{M\tilde{x}y}P'$, $Q\xrightarrow{Nvy}Q'$ and $R'=P'|Q'$. By Lemma 20, $y\in\text{fn}(P)$, so $y\notin\text{BN}(Q)$ by the clash-freedom of R. By Lemma 25, P and Q are clash free. Hence, by induction, P' and Q' are clash-free, $\text{RN}(P')\subseteq\text{RN}(P)$ and $\text{RN}(Q')\subseteq\text{RN}(Q)$. Thus $\text{BN}(R')\subseteq\text{BN}(R)$ and R' is clash-free.
- Suppose $R\xrightarrow{\alpha}R'$ is derived by **RELABELLING**. Then $R=P[\sigma]$, $P\xrightarrow{\beta}P'$, $\beta[\sigma]=\alpha$ and $R'=P'[\sigma]$. By Lemma 25, P is clash free. Suppose $\iota(\beta)=\{z\}$. Then either $z\in dom(\sigma)$, so $z\notin\text{BN}(P)$ by the clash-freedom of R, or $\iota(\beta)=\iota(\alpha)$ and $z\notin\text{BN}(R)\supseteq\text{BN}(P)$. So by induction P' is clash-free and $\text{BN}(P')\subseteq\text{BN}(P)$. Hence $\text{BN}(R')=\text{BN}(P')\cup dom(\sigma)\subseteq\text{BN}(P)\cup dom(\sigma)=\text{BN}(R)$. It remains to show that R' is clash-free.
  Clauses 1 and 3 of Definition 14 hold trivially for R', since they hold for P'. The clash-freedom of R yields $dom(\sigma)\cap\text{BN}(P)=\emptyset$, hence $dom(\sigma)\cap\text{BN}(P')=\emptyset$ and Clause 2 holds for R' as well. ■

Let $\mathscr{T}_\rho$ be the identity translation from $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ to $\pi^{\ddagger-}_{ES}(\mathcal{Z},\mathcal{R})$.

Theorem 5: $\mathscr{T}_\rho(P)\mathrel{\dot{\approx}}P$ for any clash-free P in $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$.

Proof. Let

$$\mathscr{R}:=\left\{(R,U),(U,R)\;\middle|\;\begin{array}{l}R\text{ in }\pi^{\ddagger-}_{ES}(\mathcal{Z},\mathcal{R})\text{ is clash-free}\\ \text{and }U\equiv\widehat{R}\text{ in }\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})\end{array}\right\}.$$

Since any clash-free P in $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ satisfies $\mathscr{T}_\rho(P)=P$ and $\widehat{P}=P$, and thus $\mathscr{T}_\rho(P)\mathrel{\mathscr{R}}P$, it suffices to show that $\mathscr{R}$ is a strong barbed bisimulation.

So suppose R in $\pi^{\ddagger-}_{ES}(\mathcal{Z},\mathcal{R})$ is clash-free and $U\equiv\widehat{R}$.

Let $R\xrightarrow{\tau}R'$. Then $\widehat{R}\xrightarrow{\tau}\equiv\widehat{R'}$ by Lemma 24, and $U\xrightarrow{\tau}U'\equiv\widehat{R'}$ by Lemma 15. Moreover, R' is clash-free by Lemma 27, so $R'\mathrel{\mathscr{R}}U'$.

Let $U\xrightarrow{\tau}U'$. Then $\widehat{R}\xrightarrow{\tau}\equiv U'$ by Lemma 15. Thus, by Lemma 26, $R\xrightarrow{\tau}R'$ for some R' with $\widehat{R'}\equiv U'$. Moreover, R' is clash-free by Lemma 27, so $U'\mathrel{\mathscr{R}}R'$.

Now let $R{\downarrow_b}$ with $b\in\mathcal{Z}\cup\overline{\mathcal{Z}}$. Then $R\xrightarrow{by}R'$ for some y and R', using the definition of O in Section IV. So $\widehat{R}\xrightarrow{by}\equiv\widehat{R'}$ by Lemma 24, and $U\xrightarrow{by}\equiv\widehat{R'}$ by Lemma 15. Thus $U{\downarrow_b}$.

Finally, let $U{\downarrow_b}$ with $b=x\in\mathcal{Z}$ or $b=\bar{x}$ with $x\in\overline{\mathcal{Z}}$. Then $U\xrightarrow{by}U'$ for some y and U'. So $\widehat{R}\xrightarrow{by}U''$ for some y and U'' by Lemma 15. By Lemma 21 I may assume, w.l.o.g., that if $b=x$ then $y\notin\text{BN}(R)$. Hence $\iota(by)\cap\text{BN}(R)=\emptyset$. Thus, by Lemma 26, $R\xrightarrow{by}R'$ for some R'. Hence $R{\downarrow_b}$. ■

E. Making processes clash-free

Call a $\pi_{ES}(\mathcal{Z},\mathcal{R})$ or $\pi^{\not\alpha}_{ES}(\mathcal{Z},\mathcal{R})$ process *fully clash-free* if it is restriction-clash-free according to Definition 10 and satisfies Clauses 1, 3 and 4 of Definition 14. Note that any $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ or $\pi^{\ddagger-}_{ES}(\mathcal{Z},\mathcal{R})$ process is trivially restriction-clash-free, as these languages only contain well-typed processes P with $\text{RN}(P)=\emptyset$. If P is fully clash-free, then $\mathscr{T}_\nu(P)$ is a clash-free $\pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R})$ process according to Definition 14. Thus, to connect Steps 1 and 2 of my overall translation from $\pi_L(\mathcal{N})$ to $\text{CCS}^{\text{trig}}_{\gamma}$ to steps 4, 5 and 6, the intermediate Step 3 needs to convert each $\pi_{ES}(\mathcal{N})$ process into a fully clash-free $\pi_{ES}(\mathcal{Z},\mathcal{R})$ process. This section describes this third step $\mathscr{T}^{r}_{\text{cf}}$.

Let $\mathcal{B}=\{\varsigma s\mid\varsigma\in s^*\}$ and $\mathcal{D}=\{\varsigma d_i\mid\varsigma\in d^*\wedge i>0\}$ be the sets of *symbolic* and *declared names*, respectively. The set $\mathcal{S}$ of *spare names* and $\mathcal{R}$ of *private names* were defined in Section IX. The set $\mathcal{Z}$ of *public names* is $\mathcal{N}\uplus\mathcal{S}\uplus\mathcal{B}\uplus\mathcal{D}$, and the set $\mathcal{H}$ of all names of the target language will be $\mathcal{H}=\mathcal{Z}\uplus\mathcal{R}$. The string ς in a symbolic, declared or private name is called its *modifier*.

Definition 15: The *ruthless application* $P\{\{\sigma\}\}$ of a substitution σ to a process P is the result of simultaneously replacing each occurrence of an agent identifier A in P by ${}^{\sigma}A$ and each occurrence of a name x in P by $x[\sigma]$. Here it does not matter if the occurrence of x is free or bound. Furthermore, ${}^{\sigma}A$ is a fresh agent identifier, with defining equation ${}^{\sigma}A(\vec{x}[\sigma])\stackrel{\text{def}}{=}P\{\{\sigma\}\}$ when $A(\vec{x})\stackrel{\text{def}}{=}P$ was the defining equation of A.

A substitution $\sigma:\mathcal{H}\rightharpoonup\mathcal{H}$ is called *injective* if $x[\sigma]\neq y[\sigma]$ for all names $x\neq y$. It is *surjective* if $dom(\sigma)\subseteq range(\sigma)$, and *bijective* if it is injective as well as surjective.

Ruthless substitution may lead to name capture: a free occurrence of a name becoming bound in $P\{\{\sigma\}\}$. When σ is injective, which it will be in my applications, this is not possible.

For $y\in\mathcal{N}$ and $\vec{x}=(x_1,\ldots,x_n)\in\mathcal{N}^n$, let the substitutions $s_y:\{y\}\cup\mathcal{B}\cup\mathcal{S}\to\{y\}\cup\mathcal{B}\cup\mathcal{S}$ and $d_{\vec{x}}:\{x_1,...,x_n\}\cup\mathcal{D}\cup\mathcal{S}\rightharpoonup\{x_1,...,x_n\}\cup\mathcal{D}\cup\mathcal{S}$ be defined by

$$\begin{array}{llllll}s_y(y) & := s & d_{\vec{x}}(x_i) & := d_i & \text{for } 1\leq i\leq n\\ s_y({}^{\varsigma}s) & := {}^{s\varsigma}s & d_{\vec{x}}({}^{\varsigma}d_i) & := {}^{d\varsigma}d_i & \text{for } 1\leq i\leq n\\ s_y(s_1) & := y & d_{\vec{x}}(s_i) & := x_i & \text{for } 1\leq i\leq n\\ s_y(s_{i+1}) & := s_i & d_{\vec{x}}(s_{i+n}) & := s_i\,. & \end{array}$$

Also recall the functions ℓ, r and p_y defined in Section IX. Note that all substitutions ℓ, r, p_y, s_y and $d_{\vec{x}}$ are bijective. Define the translation $\mathscr{T}^r_{\rm cf}$ from $\pi(\mathcal{N})$ to $\pi(\mathcal{Z},\mathcal{R})$ inductively by:

$$\begin{array}{lcl}
\mathscr{T}^r_{\rm cf}(\mathbf{0}) & := & \mathbf{0}\\
\mathscr{T}^r_{\rm cf}(\tau.P) & := & \tau.\mathscr{T}^r_{\rm cf}(P)\\
\mathscr{T}^r_{\rm cf}(\bar{x}y.P) & := & \bar{x}y.\mathscr{T}^r_{\rm cf}(P)\\
\mathscr{T}^r_{\rm cf}(x(y).P) & := & x(s).\big(\mathscr{T}^r_{\rm cf}(P)\{\!\{s_y\}\!\}\big)\\
\mathscr{T}^r_{\rm cf}((\nu y)P) & := & (\nu p)\big(\mathscr{T}^r_{\rm cf}(P)\{\!\{p_y\}\!\}\big)\\
\mathscr{T}^r_{\rm cf}([x{=}y]P) & := & [x{=}y]\mathscr{T}^r_{\rm cf}(P)\\
\mathscr{T}^r_{\rm cf}(P\mid Q) & := & \mathscr{T}^r_{\rm cf}(P)\{\!\{\ell\}\!\}\mid\mathscr{T}^r_{\rm cf}(Q)\{\!\{r\}\!\}\\
\mathscr{T}^r_{\rm cf}(P+Q) & := & \mathscr{T}^r_{\rm cf}(P)+\mathscr{T}^r_{\rm cf}(Q)\\
\mathscr{T}^r_{\rm cf}(A(\vec{y})) & := & A_{\rm cf}(\vec{y})
\end{array}$$

where $A_{\rm cf}$ is a fresh agent identifier, with defining equation $A_{\rm cf}(\vec{x}[d_{\vec{x}}]) \stackrel{\rm def}{=} \mathscr{T}^r_{\rm cf}(P)\{\!\{d_{\vec{x}}\}\!\}$ when $A(\vec{x}) \stackrel{\rm def}{=} P$ was the defining equation of A (which may now be dropped).

Example 9: Let $P := (\nu y)\bar{x}y.A(y,z)$ where $A(x_1,x_2) \stackrel{\rm def}{=} (\nu z).A(z,x_1)$. Then $\mathscr{T}^r_{\rm cf}(P) = (\nu p)\bar{x}p.{}^{p_y}A_{\rm cf}(p,z)$ where

$$\begin{array}{lcl}
A_{\rm cf}(d_1,d_2) & \stackrel{\rm def}{=} & (\nu p).{}^{d_{\vec{x}}p_y}A_{\rm cf}(p,d_1),\\
{}^{p_y}A_{\rm cf}(d_1,d_2) & \stackrel{\rm def}{=} & (\nu {}^{e}p).{}^{p_y d_{\vec{x}}p_y}A_{\rm cf}({}^{e}p,d_1),\\
{}^{p_y d_{\vec{x}}p_y}A_{\rm cf}({}^{d}d_1,{}^{d}d_2) & \stackrel{\rm def}{=} & (\nu {}^{ee}p).{}^{p_y d_{\vec{x}}p_y d_{\vec{x}}p_y}A_{\rm cf}({}^{ee}p,{}^{d}d_1),\ldots
\end{array}$$

Theorem 6: Each process $\mathscr{T}^r_{\rm cf}(P)$ is fully clash-free.

Proof. In $\mathscr{T}^r_{\rm cf}((\nu y)P)$ the operation $\{\!\{p_y\}\!\}$ injectively renames all private names ${}^{\varsigma}p$ in $\mathscr{T}^r_{\rm cf}(P)$ by adding a tag e in front of their modifiers. This frees up the name p. The translation of $(\nu y)P$ takes advantage of this by changing the bound name y into p. This ensures that Clause 2 of Definition 10 is met.

Likewise, in $\mathscr{T}^r_{\rm cf}(x(y).P)$ the $\{\!\{s_y\}\!\}$ injectively renames all symbolic names ${}^{\varsigma}s$ in $\mathscr{T}^r_{\rm cf}(P)$ by adding a tag s in front of their modifiers. This frees up the name s. The translation of $Mx(y)P$ finishes by changing the bound name y into s. As a result also Clause 3 of Definition 14 is met.

In $\mathscr{T}^r_{\rm cf}(A(\vec{y}))$ the operation $\{\!\{d_{\vec{x}}\}\!\}$ injectively renames all declared names ${}^{\varsigma}d_i$ for $1 \leq i \leq n$ by adding a tag d in front of their modifiers ${}^{\varsigma}$. This frees up the names $d_1,\ldots,d_n$. The translation of defining equations takes advantage of that by renaming all declared names x_i into d_i. This way Clause 1 of Definition 14 is met.

In $\mathscr{T}^r_{\rm cf}(P|Q)$ the operations $\{\!\{\ell\}\!\}$ and $\{\!\{r\}\!\}$ injectively rename private names in $\mathscr{T}^r_{\rm cf}(P)$ and $\mathscr{T}^r_{\rm cf}(Q)$ by adding a tag ℓ or r in front of their modifiers, depending on whether they occur in the left or the right argument. This validates Clause 3 of Definition 10.

Free names of a $\pi(\mathcal{N})$ process P are not renamed in $\mathscr{T}^r_{\rm cf}(P)$. As a consequence, one has $\mathrm{fn}(\mathscr{T}^r_{\rm cf}(P)) \subseteq \mathcal{N}$. Moreover, declared names always end up in $\mathcal{D}$ and names bound by input prefixes and restriction always in $\mathcal{B}$ and $\mathcal{R}$, respectively. Therefore, translated processes $\mathscr{T}^r_{\rm cf}(P)$ are always well-typed, and Clauses 1 and 4 of Definition 10 as well as 4 of Definition 14 are met. ∎

Moreover, after translation, any defining equation $A(\vec{x}) \stackrel{\rm def}{=} P$ satisfies the condition that P is well-typed and $x_i \in \mathcal{Z}$ for $i = 1,\ldots,n$. This ensures that this third step of my translation composes fruitfully with steps four, five and six.

Since all $\mathscr{T}^r_{\rm cf}$ does is replication of defining equations and renaming of bound and declared variables, it preserves strong barbed bisimilarity, regardless whether all barbs are considered, or only barbs from $\mathcal{Z}$.

F. Replacing ruthless substitution by relabelling

Even though $\mathscr{T}^r_{\rm cf}$ preserves $\mathrel{\dot{\sim}}$, I have to reject it as a valid translation. The main objection is that it employs operations on processes—*ruthless substitutions*—that are not syntactic operators of the target language. Thereby it fails the criterion of compositionality, even if the ruthless substitutions are applied in a compositional manner. In addition, it creates an infinite amount of replicated agent identifiers, whereas I strive not to increase the number of agent identifiers found in the source language. To overcome these problems, this section shows that the ruthless substitutions can be replaced by applications of relabelling operators.

Let $\mathscr{T}^r_{{\rm cf}\nu} := \mathscr{T}_\nu \circ \mathscr{T}^r_{\rm cf}$ be the composed translation from $\pi_{ES}(\mathcal{N})$ to $\pi^\dagger_{ES}(\mathcal{Z},\mathcal{R})$. It can be inductively defined just as $\mathscr{T}^r_{\rm cf}$ in Section E, except that the clause for restriction reads

$$\mathscr{T}^r_{{\rm cf}\nu}((\nu y)P) := \mathscr{T}^r_{\rm cf}(P)\{\!\{p_y\}\!\}.^{9}$$

By Theorems 3 and 4 this translation preserves strong barbed bisimilarity: $\mathscr{T}^r_{{\rm cf}\nu}(P) \mathrel{\dot{\sim}} P$ for all $\pi_{ES}(\mathcal{N})$ processes P.

Now let $\pi^\ddagger_{ES}(\mathcal{Z},\mathcal{R})$ be the variant of $\pi^{\ddagger-}_{ES}(\mathcal{Z},\mathcal{R})$ enriched with relabelling operators $[\sigma]$ for any bijective substitution $\sigma : \mathcal{H} \rightharpoonup \mathcal{H}$ that satisfies that $y \in \mathcal{R} \Rightarrow \sigma(y) \in \mathcal{R}$. Like $\pi^{\ddagger-}_{ES}(\mathcal{Z},\mathcal{R})$, it also contains all relabelling operators $[\sigma]$ with $dom(\sigma)$ finite, satisfying $dom(\sigma) \cap \mathcal{R} = \emptyset$.

Finally, let $\mathscr{T}_{{\rm cf}\nu}$ be the translation from $\pi_{ES}(\mathcal{N})$ to $\pi^\ddagger_{ES}(\mathcal{Z},\mathcal{R})$ defined inductively exactly as $\mathscr{T}^r_{{\rm cf}\nu}$, except that each substitution $\{\!\{\sigma\}\!\}$ is replaced by the relabelling operator $[\sigma]$. Theorem 7 below will show that $\mathscr{T}^r_{{\rm cf}\nu}(P) \mathrel{\dot{\sim}} \mathscr{T}_{{\rm cf}\nu}(P)$ for any $\pi_{ES}(\mathcal{N})$ process P. This entails that also the translation $\mathscr{T}_{{\rm cf}\nu}$ preserves strong barbed bisimilarity: $\mathscr{T}_{{\rm cf}\nu}(P) \mathrel{\dot{\sim}} P$ for all $\pi_{ES}(\mathcal{N})$ processes P.

The translation $\mathscr{T}_{{\rm cf}\nu}$ replaces each defining equation $A(\vec{x}) \stackrel{\rm def}{=} P$ that was present in $\pi_{ES}(\mathcal{N})$ by a defining equation $A_{{\rm cf}\nu}(\vec{x}[d_{\vec{x}}]) \stackrel{\rm def}{=} \mathscr{T}_{{\rm cf}\nu}(P)[d_{\vec{x}}]$. The total number of agent identifiers and defining equations in the language does not change. In particular, the translation $\mathscr{T}_{{\rm cf}\nu}$ does not introduce the infinite set of fresh agent identifiers ${}^{\sigma}A$ of Definition 15, and their defining equations ${}^{\sigma}A(\vec{x}[\sigma]) \stackrel{\rm def}{=} P\{\!\{\sigma\}\!\}$. These where introduced by the translation $\mathscr{T}^r_{\rm cf}$, but can be dropped as soon as we have Theorem 7. They do not form a part of the ultimate translation from $\pi_L(\mathcal{N})$ to $\mathrm{CCS}^{\rm trig}_\gamma$; they merely played a rôle in the validity proof of that translation.

Lemma 28: Let R be a $\pi^\dagger_{ES}(\mathcal{Z},\mathcal{R})$ process and σ an injective substitution. If $R \xrightarrow{\alpha} R'$ then $R\{\!\{\sigma\}\!\} \xrightarrow{\alpha[\sigma]} \equiv R'\{\!\{\sigma\}\!\}$.

Proof. With induction on the derivation of $R \xrightarrow{\alpha} R'$.

[9] Note that $\mathscr{T}_\nu(\mathscr{T}^r_{\rm cf}(P)\{\!\{d\}\!\}) = \mathscr{T}_\nu(\mathscr{T}^r_{\rm cf}(P))\{\!\{d\}\!\}$.

- The cases that $R \xrightarrow{\alpha} R'$ is derived by rule **TAU** or **OUTPUT** are trivial.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by rule **EARLY-INPUT**. Then $R = Mx(y).P$, $\alpha = Mxz$ and $R' = P\{z/y\}$. So $R\{\{\sigma\}\} = M[\sigma]x[\sigma](y[\sigma]).(P\{\{\sigma\}\}) \wedge \alpha[\sigma] = M[\sigma]x[\sigma]z[\sigma]$. By rule **EARLY-INPUT** $R\{\{\sigma\}\} \xrightarrow{\alpha[\sigma]} P\{\{\sigma\}\}\{z[\sigma]/y[\sigma]\} \equiv P\{z/y\}\{\{\sigma\}\} = R'\{\{\sigma\}\}$. The last step uses that σ is injective.
- Suppose $R = A(\vec{y})$. Let $A(\vec{x}) \stackrel{\text{def}}{=} P$. Say $\vec{x} = (x_1, \ldots, x_n)$. Then $P\{\vec{y}/\vec{x}\} \xrightarrow{\alpha} R'$, so by induction
$$P\{\vec{y}/\vec{x}\}\{\{\sigma\}\} \xrightarrow{\alpha[\sigma]} \equiv R'\{\{\sigma\}\}.$$
Moreover, $R\{\{\sigma\}\} = {}^{\sigma}A(\vec{y}[\sigma])$ and
$$P\{\{\sigma\}\}\{\vec{y}[\sigma]/\vec{x}[\sigma]\} \equiv P\{\vec{y}/\vec{x}\}\{\{\sigma\}\}.$$
This step uses that σ is injective. So by Lemma 15 $P\{\{\sigma\}\}\{\vec{y}[\sigma]/\vec{x}[\sigma]\} \xrightarrow{\alpha[\sigma]} \equiv R'\{\{\sigma\}\}$. Thus, by rule **IDE**, $R\{\{\sigma\}\} \xrightarrow{\alpha[\sigma]} \equiv R'\{\{\sigma\}\}$.
- The cases that $R \xrightarrow{\alpha} R'$ is derived by rule **SUM**, **SYMB-MATCH**, **PAR** or **E-S-COM** are trivial. ∎

Lemma 29: Let R be a $\pi^{\dagger}_{ES}(\mathcal{Z}, \mathcal{R})$ process and σ a bijective substitution. If $R\{\{\sigma\}\} \xrightarrow{\beta} U$ then $R \xrightarrow{\alpha} R'$ for some α and R' with $\alpha[\sigma] = \beta$ and $R'\{\{\sigma\}\} \equiv U$. Moreover, the size of the derivation of $R \xrightarrow{\alpha} R'$ is the same as that of $R\{\{\sigma\}\} \xrightarrow{\beta} U$.

Proof. With induction on the size of the derivation of the transition $R\{\{\sigma\}\} \xrightarrow{\beta} U$.

- The cases that $R\{\{\sigma\}\} \xrightarrow{\beta} U$ is derived by rule **TAU** or **OUTPUT** are trivial.
- Suppose $R\{\{\sigma\}\} \xrightarrow{\beta} U$ is derived by **EARLY-INPUT**. Then $R = Mx(y).P$ and $R\sigma = M[\sigma]x[\sigma](y[\sigma]).(P\{\{\sigma\}\})$. So $\beta = M[\sigma]x[\sigma]v$ and $U = P\{\{\sigma\}\}\{v/y[\sigma]\}$. Since σ is surjective, there is a z with $z[\sigma] = v$. By **EARLY-INPUT** $R \xrightarrow{\alpha} R'$ with $\alpha = Mxz$ and $R' = P\{z/y\}$. Now $\alpha[\sigma] = \beta$ and $R'\{\{\sigma\}\} = P\{z/y\}\{\{\sigma\}\} \equiv P\{\{\sigma\}\}\{z[\sigma]/y[\sigma]\} = U$. The last step uses that σ is injective.
- Suppose $R\{\{\sigma\}\} \xrightarrow{\beta} U$ is derived by **IDE**. Then $R = A(\vec{y})$ and $R\{\{\sigma\}\} = {}^{\sigma}A(\vec{y}[\sigma])$. Let $A(\vec{x}) \stackrel{\text{def}}{=} P$. Then ${}^{\sigma}A(\vec{x}[\sigma]) \stackrel{\text{def}}{=} P\{\{\sigma\}\}$. So $P\{\{\sigma\}\}\{\vec{y}[\sigma]/\vec{x}[\sigma]\} \xrightarrow{\beta} U$. Since $P\{\{\sigma\}\}\{\vec{y}[\sigma]/\vec{x}[\sigma]\} \equiv P\{\vec{y}/\vec{x}\}\{\{\sigma\}\}$, using that σ is injective, Lemma 15 yields $P\{\vec{y}/\vec{x}\}\{\{\sigma\}\} \xrightarrow{\beta} \equiv U$. So by induction $P\{\vec{y}/\vec{x}\} \xrightarrow{\alpha} R'$ for some α and R' with $\alpha[\sigma] = \beta$ and $R'\{\{\sigma\}\} \equiv U$. By rule **IDE** $R \xrightarrow{\alpha} R'$.
- The cases that $R\{\{\sigma\}\} \xrightarrow{\beta} U$ is derived by rule **SUM**, **SYMB-MATCH** or **PAR** are trivial. The case for **E-S-COM** is also trivial, when using injectivity of σ to conclude that there is a unique z with $z[\sigma] = y$. This solves the only complication in the corresponding case of the proof of Lemma 23. ∎

For P a $\pi^{\ddagger}_{ES}(\mathcal{Z}, \mathcal{R})$ process, let $\widehat{P}$ be the $\pi^{\dagger}_{ES}(\mathcal{Z}, \mathcal{R})$ process obtained from P by recursively replacing each subterm $Q[\sigma]$ by $Q\sigma$ if $dom(\sigma)$ is finite and $dom(\sigma) \cap \mathcal{R} = \emptyset$, and by $Q\{\{\sigma\}\}$ if $dom(\sigma)$ is infinite and σ bijective, and each agent identifier A by $A^{\wedge}$. Here $A^{\wedge}$ is a fresh agent identifier with defining equation $A^{\wedge}(\vec{x}) \stackrel{\text{def}}{=} \widehat{P}$ when $A(\vec{x}) \stackrel{\text{def}}{=} P$ was the defining equation of A. This definition extends the mapping $\widehat{\cdot}$ defined in Section D from $\pi^{\ddagger-}_{ES}(\mathcal{Z}, \mathcal{R})$ to $\pi^{\ddagger}_{ES}(\mathcal{Z}, \mathcal{R})$.

Lemma 30: If $R \xrightarrow{\alpha} R'$ then $\widehat{R} \xrightarrow{\alpha} \equiv \widehat{R'}$.

Proof. The proof is the same as the proof of Lemma 24, except that the case of **RELABELLING** $R = [\sigma]$ involves a further case distinction, depending on whether $dom(\sigma)$ is finite and $dom(\sigma) \cap \mathcal{R} = \emptyset$, or $dom(\sigma)$ is infinite and σ is bijective. In the first case Lemma 19 is called, and in the second case Lemma 28. ∎

Definition 16: Let BN be the smallest function from $\pi^{\ddagger}_{ES}(\mathcal{Z}, \mathcal{R})$ processes to sets of names, such that

- $y \in \text{BN}(Mx(y).P)$,
- $x_1, \ldots, x_n \in \text{BN}(A(\vec{y})$ if $A(x_1, \ldots, x_n) \stackrel{\text{def}}{=} P$,
- $y \in \text{BN}(P[\sigma])$ if $y \in dom(\sigma)$ and $dom(\sigma)$ is finite,
- $\text{BN}(P) \subseteq \text{BN}(M\tau.P)$,
- $\text{BN}(P) \subseteq \text{BN}(M\bar{x}y.P)$ and $\text{BN}(P) \subseteq \text{BN}(Mx(y).P)$,
- $\text{BN}(P) \cup \text{BN}(Q) \subseteq \text{BN}(P + Q)$,
- $\text{BN}(P) \cup \text{BN}(Q) \subseteq \text{BN}(P|Q)$,
- $\text{BN}(P) \subseteq \text{BN}([x{=}y]P)$,
- $\text{BN}(P) \subseteq \text{BN}(A(\vec{y}))$ if $A(x_1, \ldots, x_n) \stackrel{\text{def}}{=} P$,
- $\text{BN}(P) \subseteq \text{BN}(P[\sigma])$ if $dom(\sigma)$ is finite.
- $\text{BN}(P[\sigma]) = \{\sigma(y) \mid y \in \text{BN}(P)\}$ if $dom(\sigma)$ is infinite and σ bijective.

Note that restricted to $\pi^{\ddagger-}_{ES}(\mathcal{Z}, \mathcal{R})$ processes, so that the last clause above does not apply, this definition agrees with Definition 12. Using this definition of BN, Definition 14 of clash-freedom extends to the general case with the stipulation that Clause 2 on $Q[\sigma]$ only applies when σ is a finite substitution with $dom(\sigma) \cap \mathcal{R} = \emptyset$; the notion does not restrict the use of relabelling operators σ with $dom(\sigma)$ infinite and σ bijective.

I now show that Lemma 25 (and the definition of $\leftarrow\!\!\!\cdot$) extends to $\pi^{\ddagger}_{ES}(\mathcal{Z}, \mathcal{R})$.

Lemma 31: If $P \leftarrow\!\!\!\cdot\ Q$ and P is a clash-free $\pi^{\ddagger}_{ES}(\mathcal{Z}, \mathcal{R})$ process, then so is Q.

Proof. Let $P[\sigma]$ be a clash-free $\pi^{\ddagger}_{ES}(\mathcal{Z}, \mathcal{R})$ process with $dom(\sigma)$ infinite and σ bijective. By definition of $\pi^{\ddagger}_{ES}(\mathcal{Z}, \mathcal{R})$, σ satisfies $y \in \mathcal{R} \Rightarrow \sigma(y) \in \mathcal{R}$. This implies that $\mathcal{R} \cap \text{BN}(P) = \emptyset$. The other clauses of Definition 14 are satisfied trivially, so P is clash-free.

All other cases go as in the proof of Lemma 25. ∎

Note that Lemma 20 holds also for $\pi^{\ddagger}_{ES}(\mathcal{Z}, \mathcal{R})$ processes.

Lemma 32: If R is clash-free and $\widehat{R} \xrightarrow{\alpha} U$ with $\iota(\alpha) \cap \text{BN}(R) = \emptyset$ then $R \xrightarrow{\alpha} R'$ for some R' with $\widehat{R'} \equiv U$.

Proof. By induction on the size of the derivation of $\widehat{R} \xrightarrow{\alpha} U$, with a nested induction on the number of topmost renaming operators in R.

- The cases that R is not of the form $P[\sigma]$, or that $R = P[\sigma]$ with $dom(\sigma)$ finite and $dom(\sigma) \cap \mathcal{R} = \emptyset$, go exactly as in the proof of Lemma 26.

- Now suppose $R = P[\sigma]$ with $dom(\sigma)$ infinite and σ bijective. Then $\widehat{R} = \widehat{P}\{\{\sigma\}\}$. By Lemma 29 $\widehat{P} \xrightarrow{\beta} V$ for some β and V with $\beta[\sigma] = \alpha$ and $V\{\{\sigma\}\} \equiv U$. Moreover, the size of the derivation of $\widehat{P} \xrightarrow{\beta} V$ is the same as that of $\widehat{P}\{\{\sigma\}\} \xrightarrow{\alpha} U$. By Lemma 31 P is clash-free and trivially $\iota(\beta) \cap \text{Bn}(P) = \emptyset$. So by induction $P \xrightarrow{\beta} P'$ for some P' with $\widehat{P'} \equiv V$. By rule **RELABELLING** $R \xrightarrow{\alpha} P'[\sigma]$. Furthermore one has $\widehat{P'[\sigma]} = \widehat{P'}\{\{\sigma\}\} \equiv V\{\{\sigma\}\} \equiv U$. ■

The following result is in analogy with Theorem 6.

Lemma 33: Each process $R := \mathscr{T}_{\text{cf}\nu}(P)$ is clash-free and $\text{Bn}(R) \subseteq \mathcal{B} \cup \mathcal{D}$.

Proof. In $\mathscr{T}_{\text{cf}\nu}(x(y).P) = x(s).\big(\mathscr{T}_{\text{cf}\nu}(P)[s_y]\big)$ the relabelling $[s_y]$ injectively relabels all symbolic names $^{\varsigma}s$ in $\text{Bn}(\mathscr{T}_{\text{cf}\nu}(P))$ by adding a tag s to their modifiers. This frees up the name s. The translation of $Mx(y)P$ takes advantage of this by changing the bound name y into $s \in \mathcal{B}$. This ensures that Clause 3 of Definition 14 is met.

Likewise, in $\mathscr{T}_{\text{cf}\nu}(A(\vec{y})) = A_{\text{cf}\nu}(\vec{y})$, where $A_{\text{cf}\nu}$ is an agent identifier with defining equation $A_{\text{cf}\nu}(\vec{x}[d_{\vec{x}}]) \stackrel{\text{def}}{=} \mathscr{T}_{\text{cf}\nu}(P)[d_{\vec{x}}]$ when $A(\vec{x}) \stackrel{\text{def}}{=} P$ was the defining equation of A, the $[d_{\vec{x}}]$ injectively renames all declared names $^{\varsigma}d_i$ for $1 \leq i \leq n$ in $\text{Bn}(\mathscr{T}_{\text{cf}\nu}(P))$ by adding a tag d to their modifiers $^{\varsigma}$. This frees up the names $d_1, \ldots, d_n$. The translation of defining equations takes advantage of that by renaming all declared names x_i into $d_i \in \mathcal{D}$. This way Clause 1 of Definition 14 is met.

Relabelling operators $[\sigma]$ with $dom(\sigma)$ finite do not occur in processes $\mathscr{T}_{\text{cf}\nu}(P)$. Hence Clause 2 of Definition 14 is trivially met. Furthermore, since all relabelling operators employed in the translation never take a name from $\mathcal{B}$ or $\mathcal{D}$ outside of $\mathcal{B}$ or $\mathcal{D}$, respectively, one has $\text{Bn}(R) \subseteq \mathcal{B} \cup \mathcal{D}$.

In $\mathscr{T}_{\text{cf}\nu}((\nu y)P) = \mathscr{T}_{\text{cf}\nu}(P)[p_y]$ the bound name y is turned into a free name $p \in \mathcal{R}$, and all relabelling operators keep such names within $\mathcal{R}$. Free names of a $\pi(\mathcal{N})$ process P are not renamed in $\mathscr{T}_{\text{cf}\nu}(P)$. As a consequence, one has $\text{fn}(\mathscr{T}_{\text{cf}\nu}(P)) \subseteq \mathcal{N} \cup \mathcal{R}$. Therefore, Clause 4 of Definition 14 is met as well. ■

Lemma 34: If R in $\pi^{\ddagger}_{ES}(\mathcal{Z}, \mathcal{R})$ is clash-free, $\iota(\alpha) \cap \text{Bn}(R) = \emptyset$ and $R \xrightarrow{\alpha} R'$, then R' is clash-free and $\text{Bn}(R') \subseteq \text{Bn}(R)$.

Proof. The proof is the same as the one of Lemma 27, except that there is now one extra case to consider: Suppose $R \xrightarrow{\alpha} R'$ is derived by **RELABELLING** and $R = P[\sigma]$ with $dom(\sigma)$ infinite and σ bijective. Then $P \xrightarrow{\beta} P'$, $\beta[\sigma] = \alpha$ and $R' = P'[\sigma]$. By Lemma 31, P is clash free. Since $\iota(\alpha) \cap \text{Bn}(R) = \emptyset$ and σ is bijective, $\iota(\beta) \cap \text{Bn}(P) = \emptyset$. So by induction P' is clash-free and $\text{Bn}(P') \subseteq \text{Bn}(P)$. Hence $\text{Bn}(R') \subseteq \text{Bn}(R)$ and R' is clash-free. ■

Theorem 7: $\mathscr{T}^{\text{r}}_{\text{cf}\nu}(P) \curvearrowright \mathscr{T}_{\text{cf}\nu}(P)$ for any $\pi_{ES}(\mathcal{N})$ process P.

Proof. Let

$$\mathscr{R} := \left\{ (R,U), (U,R) \;\middle|\; \begin{array}{l} R \text{ in } \pi^{\ddagger}_{ES}(\mathcal{Z},\mathcal{R}) \text{ is clash-free} \\ \text{and } U \equiv \widehat{R} \text{ in } \pi^{\dagger}_{ES}(\mathcal{Z},\mathcal{R}) \end{array} \right\}.$$

Lemma 33 shows that $\mathscr{T}_{\text{cf}\nu}(P)$ is clash-free for each $\pi_{ES}(\mathcal{N})$ process P. Since $\widehat{\mathscr{T}_{\text{cf}\nu}(P)} = \mathscr{T}^{r}_{\text{cf}\nu}(P)$ for any $\pi_{SE}(\mathcal{N})$ process P, it suffices to show that $\mathscr{R}$ is a strong barbed bisimulation. This follows exactly as in the proof of Theorem 5, but using Lemmas 30, 32 and 34 instead of 24, 26 and 27. ■

G. The last steps

The translation from $\pi^{\ddagger}_{ES}(\mathcal{Z}, \mathcal{R})$ to $\text{CCS}^{\text{trig}}_{\gamma}$, depicted as the last step in Figure 3, actually consists of 2 small steps. The first of those consists in moving the relabelling operator $[\{\vec{y}/\vec{x}\}]$ that occurs in rule **IDE**, as well as the relabelling $[d_{\vec{x}}]$ that was added to defining equations of agent identifiers, forwards. The target language is the variant $\pi^{\S}_{ES}(\mathcal{Z}, \mathcal{R})$ of $\pi^{\ddagger}_{ES}(\mathcal{Z}, \mathcal{R})$ in which agent identifiers may be called only with their own declared names as parameters, i.e. such that $\vec{y} = \vec{x}$ in rule **IDE**. As a result of this, the relabelling or substitution operator in this rule can be dropped. Let $\mathscr{T}'_{\text{cf}\nu}(P)$ be the translation from $\pi(\mathcal{N})$ to $\pi^{\S}_{ES}(\mathcal{Z}, \mathcal{R})$ defined inductively as follows:

$$\begin{array}{lcl}
\mathscr{T}'_{\text{cf}\nu}(\mathbf{0}) & := & \mathbf{0} \\
\mathscr{T}'_{\text{cf}\nu}(\tau.P) & := & \tau.\mathscr{T}'_{\text{cf}\nu}(P) \\
\mathscr{T}'_{\text{cf}\nu}(\bar{x}y.P) & := & \bar{x}y.\mathscr{T}'_{\text{cf}\nu}(P) \\
\mathscr{T}'_{\text{cf}\nu}(x(y).P) & := & x(s).\big(\mathscr{T}'_{\text{cf}\nu}(P)[s_y]\big) \\
\mathscr{T}'_{\text{cf}\nu}((\nu y)P) & := & \mathscr{T}'_{\text{cf}\nu}(P)[p_y] \\
\mathscr{T}'_{\text{cf}\nu}([x{=}y]P) & := & [x{=}y]\mathscr{T}'_{\text{cf}\nu}(P) \\
\mathscr{T}'_{\text{cf}\nu}(P \mid Q) & := & \mathscr{T}'_{\text{cf}\nu}(P)[\ell] \mid \mathscr{T}'_{\text{cf}\nu}(Q)[r] \\
\mathscr{T}'_{\text{cf}\nu}(P + Q) & := & \mathscr{T}'_{\text{cf}\nu}(P) + \mathscr{T}'_{\text{cf}\nu}(Q) \\
\mathscr{T}'_{\text{cf}\nu}(A(\vec{y})) & := & A'_{\text{cf}\nu}(\vec{x})[d_{\vec{x}}][\{\vec{y}/\vec{x}[d_{\vec{x}}]\}]
\end{array}$$

where $A'_{\text{cf}\nu}$ is a fresh agent identifier, with defining equation $A'_{\text{cf}\nu}(\vec{x}) \stackrel{\text{def}}{=} \mathscr{T}'_{\text{cf}\nu}(P)$ when $A(\vec{x}) \stackrel{\text{def}}{=} P$ was the defining equation of A. This translation differs from $\mathscr{T}_{\text{cf}\nu}$ only in the case of agent identifiers. There $\mathscr{T}_{\text{cf}\nu}(A(\vec{y})) := A_{\text{cf}\nu}(\vec{y})$ where $A_{\text{cf}\nu}$ is a fresh agent identifier, with defining equation $A_{\text{cf}\nu}(\vec{x}[d_{\vec{x}}]) \stackrel{\text{def}}{=} \mathscr{T}_{\text{cf}\nu}(P)[d_{\vec{x}}]$ when $A(\vec{x}) \stackrel{\text{def}}{=} P$ was the defining equation of A.

Theorem 8: $\mathscr{T}'_{\text{cf}\nu}(P) \leftrightarrows \mathscr{T}_{\text{cf}\nu}(P)$ for any $\pi_{ES}(\mathcal{N})$ process P.

Proof. Let $\mathscr{R}$ be the symmetric closure of the smallest relation between $\pi^{\S}_{ES}(\mathcal{Z}, \mathcal{R})$ and $\pi^{\ddagger}_{ES}(\mathcal{Z}, \mathcal{R})$ processes such that $\mathbf{0}\,\mathscr{R}\,\mathbf{0}$, $A'_{\text{cf}\nu}(\vec{x})[d_{\vec{x}}][\{\vec{y}/\vec{x}[d_{\vec{x}}]\}]\,\mathscr{R}\,A_{\text{cf}\nu}(\vec{y})$, and $P\,\mathscr{R}\,V \wedge Q\,\mathscr{R}\,W$ implies $\bar{x}y.P\,\mathscr{R}\,\bar{x}y.V$, $x(y).P\,\mathscr{R}\,x(y).V$, $\tau.P\,\mathscr{R}\,\tau.V$, $[x{=}y]P\,\mathscr{R}\,[x{=}y]V$, $P|Q\,\mathscr{R}\,V|W$, $P{+}Q\,\mathscr{R}\,V{+}W$ and $P[\sigma]\,\mathscr{R}\,V[\sigma]$. Then $\mathscr{T}'_{\text{cf}\nu}(P)\,\mathscr{R}\,\mathscr{T}_{\text{cf}\nu}(P)$ for any $\pi_{ES}(\mathcal{N})$ process P, so it suffices to show that $\mathscr{R}$ is a strong bisimulation, i.e. that

$$\text{if } R\,\mathscr{R}\,U \text{ and } R \xrightarrow{\alpha} R' \text{ then } \exists U'.\ U \xrightarrow{\alpha} U' \wedge R'\,\mathscr{R}\,U'.$$

The proof is by induction on the derivation of $R \xrightarrow{\alpha} R'$, while making a case distinction based on the construction of $R\,\mathscr{R}\,U$.

- Suppose $R = A'_{\text{cf}\nu}(\vec{x})[d_{\vec{x}}][\{\vec{y}/\vec{x}[d_{\vec{x}}]\}$ and $U = A_{\text{cf}\nu}(\vec{y})$. Then $A'_{\text{cf}\nu}(\vec{x}) \xrightarrow{\beta} Q'$ for some β and Q' such that $\beta[d_{\vec{x}}][\{\vec{y}/\vec{x}[d_{\vec{x}}]\} = \alpha$ and $Q'[d_{\vec{x}}][\{\vec{y}/\vec{x}[d_{\vec{x}}]\} = R'$. Let $A(\vec{x}) \stackrel{\text{def}}{=} P$ be the defining equation of A. Then, by **IDE**, $\mathscr{T}'_{\text{cf}\nu}(P) \xrightarrow{\beta} Q'$ and thus $\mathscr{T}'_{\text{cf}\nu}(P)[d_{\vec{x}}][\{\vec{y}/\vec{x}[d_{\vec{x}}]\} \xrightarrow{\alpha} R'$. By induction, $\mathscr{T}_{\text{cf}\nu}(P)[d_{\vec{x}}][\{\vec{y}/\vec{x}[d_{\vec{x}}]\} \xrightarrow{\alpha} U'$ for some U' with $R'\,\mathscr{R}\,U'$. Thus, again by **IDE**, $A_{\text{cf}\nu}(\vec{y}) \xrightarrow{\alpha} U'$.

The symmetric case goes likewise.

- Suppose $R \mathrel{\mathscr{R}} U$ holds because $R = P|Q$ and $U = V|W$ with $P \mathrel{\mathscr{R}} V$ and $Q \mathrel{\mathscr{R}} W$.
  - First suppose that $R \xrightarrow{\alpha} R'$ is derived by **PAR**. Then $P \xrightarrow{\alpha} P'$ has a smaller derivation and $R' = P'|Q$. By induction $V \xrightarrow{\alpha} V'$ for some V' with $P' \mathrel{\mathscr{R}} V'$. So by **PAR** $U \xrightarrow{\alpha} V'|W$ and $V'|W \mathrel{\mathscr{R}} P'|Q = R'$.
  - The case that $R \xrightarrow{\alpha} R'$ is derived by **E-S-COM** (or by the symmetric form of **PAR**) is equally trivial.
- All other cases are also trivial. ∎

The language $\pi^{\S}_{ES}(\mathcal{Z},\mathcal{R})$ can almost be recognised as an instance of CCS$_\gamma$. As parameters of CCS$_\gamma$ I take $\mathscr{A}$ to be the set of all actions $M\tau$, $M\bar{x}y$ and Mxy with names from $\mathcal{H}$. The communication function $\gamma : \mathcal{H} \rightharpoonup \mathcal{H}$ is given by $\gamma(M\bar{x}y, Nvy) = [x{=}v]MN\tau$, and its commutative variant. Now the parallel composition of $\pi^{\S}_{ES}(\mathcal{Z},\mathcal{R})$ turns out to be the same as for this instance of CCS$_\gamma$. Likewise, the silent and output prefixes are instances of CCS$_\gamma$ prefixing. Moreover, when simply writing A for $A(\vec{x})$, the agent identifiers of $\pi^{\S}_{ES}(\mathcal{Z},\mathcal{R})$ are no different from CCS$_\gamma$ agent identifiers.[10] However, the input prefix of $\pi^{\S}_{ES}(\mathcal{Z},\mathcal{R})$ does not occur in CCS$_\gamma$. Yet, one can identify $Mx(y).P$ with $\sum_{z\in\mathcal{H}} Mxz.(P[\{z/y\}])$, for both processes have the very same outgoing transitions. The $\pi^{\S}_{ES}(\mathcal{Z},\mathcal{R})$ matching operator is no different from the triggering operator of MEIJE or CCS$^{\text{trig}}_\gamma$ (see Section XI): both rename only the first actions their argument process can perform, namely by adding a single match $[x{=}y]$ in front of each of them—this match is suppressed when $x{=}y$.

This yields to the following translation from $\pi^{\S}_{ES}(\mathcal{Z},\mathcal{R})$ to CCS$^{\text{trig}}_\gamma$:

$$\begin{array}{lcl}
\mathscr{T}_\gamma(\mathbf{0}) & := & \mathbf{0}\\
\mathscr{T}_\gamma(M\tau.P) & := & M\tau.\mathscr{T}_\gamma(P)\\
\mathscr{T}_\gamma(M\bar{x}y.P) & := & M\bar{x}y.\mathscr{T}_\gamma(P)\\
\mathscr{T}_\gamma(Mx(y).P) & := & \sum_{z\in\mathcal{H}} Mxz.\big(\mathscr{T}_\gamma(P)[\{z/y\}]\big)\\
\mathscr{T}_\gamma([x{=}y]P) & := & [x{=}y]{\Rightarrow}\mathscr{T}_\gamma(P)\\
\mathscr{T}_\gamma(P \mid Q) & := & \mathscr{T}_\gamma(P)\|\mathscr{T}_\gamma(Q)\\
\mathscr{T}_\gamma(P+Q) & := & \mathscr{T}_\gamma(P)+\mathscr{T}_\gamma(Q)\\
\mathscr{T}_\gamma(A(\vec{x})) & := & A
\end{array}$$

where the CCS$_\gamma$ agent identifier A has the defining equation $A = \mathscr{T}_\gamma(P)$ when $A(\vec{x}) \stackrel{\text{def}}{=} P$ was the defining equation of the $\pi^{\S}_{ES}(\mathcal{Z},\mathcal{R})$ agent identifier A.

Here the use of the triggering operator can be avoided by restricting attention to the π-calculus with implicit matching. For that language the clause for $\mathscr{T}_\gamma([x{=}y]P)$ can be dropped, at the expense of the addition of the blue Ms above, which are absent when dealing with the full π-calculus.

Theorem 9: $\mathscr{T}_\gamma(P) \leftrightarroweq P$ for each $\pi^{\S}_{ES}(\mathcal{Z},\mathcal{R})$ process P.

Proof. Trivial. ∎

[10]Here I assume that all sets $\mathcal{K}_n$ are disjoint, i.e., the same π-calculus agent identifier does occur with multiple arities. When this assumption is not met, an arity-index at the CCS$_\gamma$ identifier is needed.

H. A small simplification

Putting all steps of my translation from $\pi_L(\mathcal{N})$ to CCS$^{\text{trig}}_\gamma$ together, I obtain

$$\begin{array}{lcl}
\mathscr{T}_1(\mathbf{0}) & := & \mathbf{0}\\
\mathscr{T}_1(M\tau.P) & := & M\tau.\mathscr{T}_1(P)\\
\mathscr{T}_1(M\bar{x}y.P) & := & M\bar{x}y.\mathscr{T}_1(P)\\
\mathscr{T}_1(Mx(y).P) & := & \sum_{z\in\mathcal{H}} Mxz.\big(\mathscr{T}_1(P)[s_y][\{z/s\}]\big)\\
\mathscr{T}_1((\nu y)P) & := & \mathscr{T}_1(P)[p_y]\\
\mathscr{T}_1([x{=}y]P) & := & [x{=}y]{\Rightarrow}\mathscr{T}_1(P)\\
\mathscr{T}_1(P \mid Q) & := & \mathscr{T}_1(P)[\ell] \parallel \mathscr{T}_1(Q)[r]\\
\mathscr{T}_1(P+Q) & := & \mathscr{T}_1(P)+\mathscr{T}_1(Q)\\
\mathscr{T}_1(A(\vec{y})) & := & A[d_{\vec{x}}][\{\vec{y}/\vec{x}[d_{\vec{x}}]\}]
\end{array}$$

where the CCS$_\gamma$ agent identifier A has the defining equation $A = \mathscr{T}_1(P)$ when $A(\vec{x}) \stackrel{\text{def}}{=} P$ was the defining equation of the $\pi_L(\mathcal{N})$ agent identifier A.

Now I discuss two simplification that can be made to this translation. The first is obtained by replacing the nested relabellings $[s_y][\{z/s\}]$ and $[d_{\vec{x}}][\{\vec{y}/\vec{x}[d_{\vec{x}}]\}]$ in the clauses for input prefix and agent identifiers by single relabellings $[\{z/s\} \circ s_y]$ and $[\{\vec{y}/\vec{x}[d_{\vec{x}}]\} \circ d_{\vec{x}}]$. This surely preserves strong bisimilarity. After this change, all CCS$^{\text{trig}}_\gamma$ relabelling operators $[\sigma]$ that are introduced by the translation have the property that $x \in \mathcal{B}\cup\mathcal{D} \Leftrightarrow x[\sigma] \in \mathcal{B}\cup\mathcal{D}$.

Now let $\mathscr{T}_{\mathcal{BD}}$ be the translation from CCS$^{\text{trig}}_\gamma$ to CCS$^{\text{trig}}_\gamma$ that replaces each sum $\sum_{z\in\mathcal{H}}$ into a sum $\sum_{z\in\mathcal{N}\cup\mathcal{R}\cup\mathcal{S}}$ and each relabelling operator $[\sigma]$ by $[\sigma \upharpoonright \mathcal{N}\cup\mathcal{R}\cup\mathcal{S}]$. It is trivial to show that each process of the form $\mathscr{T}(P)$ with $P \in \pi_L(\mathcal{N})$ is strongly barbed bisimilar with $\mathscr{T}_{\mathcal{BD}}(\mathscr{T}(P))$. The idea is that names from $\mathcal{B}\cup\mathcal{D}$ are never introduced and thus can just as well be dropped from the language.

The resulting simplifications of the renamings $[\{z/s\} \circ s_y]$ and $[\{\vec{y}/\vec{x}[d_{\vec{x}}]\} \circ d_{\vec{x}}]$ can now be written as $[\{z/y\}^{\mathcal{S}}]$ and $[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]$, or as $[z/y]$ and $[\vec{y}/\vec{x}]$ for short. Here $[\{z/y\}^{\mathcal{S}}]$ was defined in Section IX. It relabels $y \in \mathcal{N}$ into z and bijectively maps $\mathcal{S}$ to $\mathcal{S}\cup\{y\}$, leaving all other names unchanged. Likewise, $[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]$ relabels x_i into y_i for $i = 1,\ldots,n$ and bijectively maps $\mathcal{S}$ to $\mathcal{S}\cup\{x_1,\ldots,x_n\}$. This yields the translation $\mathscr{T}$ presented in Section IX.

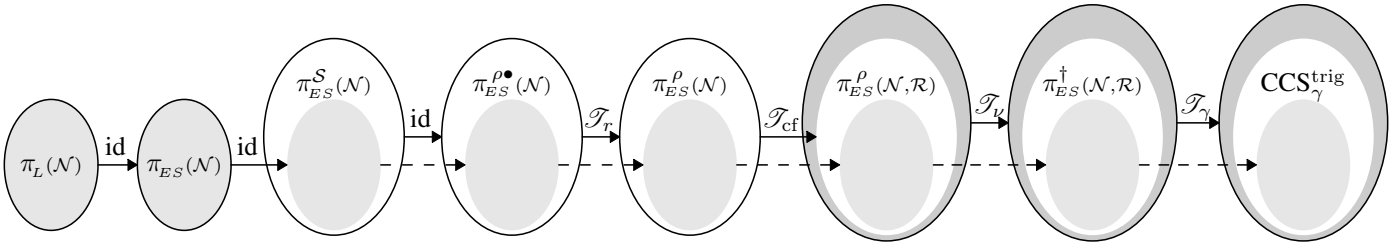


Fig. 4. Translation from the π-calculus to $\mathrm{CCS}^{\rm trig}_\gamma$

APPENDIX 2

As indicated in Figure 4, my translation from π to $\mathrm{CCS}^{\rm trig}_\gamma$ proceeds in seven steps, or actually ten when decomposing steps 1, 3 and 6 into two smaller steps each. Section IX presents the translation in one step: the composition of these constituent translations. Its decomposition in Figure 4 describes how this appendix proves its validity.

Each of the eight languages in Figure 4 comprises syntax, determining what are the valid expressions or processes, a structural operational semantics generating an LTS, and a BTS extracted from the LTS in the way described in Section IV.

My translation starts from the π-calculus $\pi_L(\mathcal{N})$ with the late operational semantics, as defined in [24]. The argument $\mathcal{N}$ denotes the set of names taken as a parameter of the π-calculus. The first step is the identity mapping to $\pi_{ES}(\mathcal{N})$, the calculus with the same syntax but the early symbolic operational semantics. The validity of this translation step is the statement that each $\pi_L(\mathcal{N})$-expression P is strongly barbed bisimilar with the same expression P, but now seen as a state in the LTS generated by the early symbolic operational semantics. This has been concluded already in Section VI. This first translation step can be decomposed into two smaller steps by taking either the π-calculus with the early semantics or the one with the late symbolic semantics as an halfway point between $\pi_L(\mathcal{N})$ and $\pi_{ES}(\mathcal{N})$.

The calculus $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$ is a variant of $\pi_{ES}(\mathcal{N})$ in which the substitutions that occur in the early symbolic operational rules **EARLY-INPUT** and **IDE** are changed into surjective substitutions. This is achieved by extending the set of names from $\mathcal{N}$ to $\mathcal{Z} := \mathcal{N} \uplus \mathcal{S}$, for a countable set of *spare names* $\mathcal{S}$. In order to preserve the integrity of the calculus, this change forces new definitions of α-conversion, the free names of a process, and the application of a substitution to a process. These concepts differ from the old ones only when spare names are involved. The $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$-processes that employ names from $\mathcal{N}$ only form a subcalculus of $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$ that behaves just like $\pi_{ES}(\mathcal{N})$. This yields the second translation step.

The calculus $\pi^{\rho\bullet}_{ES}(\mathcal{N})$ is a variant of $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$ enriched with a CCS-style relabelling operator $[\sigma]$ for each surjective substitution σ. Moreover, all substitutions that are used in the operational semantics are replaced by relabelling operators. Section D documents that the identity mapping is a valid translation from $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$ to $\pi^{\rho\bullet}_{ES}(\mathcal{N})$. This works only when using surjective substitutions, and that is the reason $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$ appears in the previous translation step.

In $\pi^{\rho\bullet}_{ES}(\mathcal{N})$ there is no room for rule **ALPHA**. Hence, before translating $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$ into $\pi^{\rho\bullet}_{ES}(\mathcal{N})$ I show that on $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$ one can equally well use a version of the early symbolic operational semantics in which all α-conversion has been moved into stronger versions of rules **RES** and **SYMB-OPEN**. This can be seen as an intermediate translation step.

The calculus $\pi^{\rho}_{ES}(\mathcal{N})$ is the variant of $\pi^{\rho\bullet}_{ES}(\mathcal{N})$ in which agent identifiers may be called only with their own declared names as parameters, i.e. such that $\vec{y} = \vec{x}$ in rule **IDE**. In Section F I establish the validity of a translation from $\pi^{\rho\bullet}_{ES}(\mathcal{N})$ to $\pi^{\rho}_{ES}(\mathcal{N})$ that turns each call $A(\vec{y})$ to an agent identifier with defining equation $A(\vec{x}) \stackrel{\rm def}{=} P$ into the expression $A(\vec{x})[\vec{y}/\vec{x}]$.

In $\pi^{\rho}_{ES}(\mathcal{N},\mathcal{R})$ the set of names employed is further extended with a collection $\mathcal{R}$ of *private names*. I write $\mathcal{N},\mathcal{R}$ instead of $\mathcal{N} \uplus \mathcal{R}$ to indicate, amongst others, that only the names in $\mathcal{Z} := \mathcal{N} \uplus \mathcal{S}$—the *public* ones—generate barbs. The interior white ellipse denotes the class of *clash-free* processes within $\pi^{\rho}_{ES}(\mathcal{N},\mathcal{R})$, defined in Section G. That section also presents the translation $\mathscr{T}_{\rm cf}$ that turns each $\pi^{\rho}_{ES}(\mathcal{N})$ process into a clash-free $\pi^{\rho}_{ES}(\mathcal{N},\mathcal{R})$ process.

Step 6, recorded in Sections H–I, eliminates the restriction operator from the language by translating (sub)expressions $(\nu z)P$ into P. This step does not preserve $\nsim$ for the language as a whole, but, since there are no $\mathcal{R}$-barbs, it does so on the sublanguage that arises as the image of the previous translation steps, namely on the clash-free processes in $\pi^{\rho}_{ES}(\mathcal{N},\mathcal{R})$. This step is delivered in two substeps, the first of which eliminates α-conversion from the operational semantics, and the second the restriction operator.

After this step, the resulting language $\pi^{\dagger}_{ES}(\mathcal{N},\mathcal{R})$ is in De Simone format. Moreover, as shown in Section J, it is easily translated into $\mathrm{CCS}^{\rm trig}_\gamma$.

A different proof of the same result, Theorem 2, saying that $\mathscr{T}$ is a valid translation from $\pi_L(\mathcal{N})$ to $\mathrm{CCS}^{\rm trig}_\gamma$, up to strong barbed bisimilarity, appears in Appendix 1. That proof avoids step 2 from the current approach, which is the most laborious one, as well as step 3a. The remaining steps are delivered in the order $1a{-}1b{-}5{-}6a{-}6b{-}3b{-}4{-}7$. However, that proof appears less direct: its third step, making processes clash-free, introduces two auxiliary sets of names $\mathcal{B}$ and $\mathcal{D}$, as well as "ruthless substitutions", with an infinity of replicated agent identifiers. The ruthless substitutions are eliminated again after the sixth step, and the names from $\mathcal{B}$ and $\mathcal{D}$ at the very end. The proof in this appendix avoids those detours.

A. Substitution

This section shows to what extend substitution preserves and reflects the behaviour of π-calculus processes. As behaviour I will use the labelled translation relation generated by the early symbolic operational semantics.

In the next section, I will need that the results of this section remain valid for an adapted version of the π-calculus, in which the meaning of the construct $x(y).P$ will be changed. Technically, this will manifest itself by

- the use of surjective substitutions $\{z/y\}^{\mathcal{S}}$ and $\{\vec{y}/\vec{x}\}^{\mathcal{S}}$ instead of $\{z/y\}$ and $\{\vec{y}/\vec{x}\}$ in rules **EARLY-INPUT** and **IDE**,
- the use of a relation $\equiv^{\mathcal{S}}$ instead of $\equiv$ in rule **ALPHA**,
- the use of a function FN instead of fn in rules **PAR** and **E-S-CLOSE**,
- a different definition of $(x(y).P)\sigma$.

The function FN will be defined inductively by

$$\begin{array}{lcl}
\text{FN}(\mathbf{0}) & := & \emptyset \\
\text{FN}(M\tau.P) & := & \text{n}(M) \cup \text{FN}(P) \\
\text{FN}(M\bar{x}y.P) & := & \text{n}(M) \cup \{x,y\} \cup \text{FN}(P) \\
\text{FN}(Mx(y).P) & := & \text{n}(M) \cup \{x\} \cup \text{FN}(P)^-_y \\
\text{FN}((\nu y)P) & := & \text{FN}(P)\backslash\{y\} \\
\text{FN}([x{=}y]P) & := & \{x,y\} \cup \text{FN}(P) \\
\text{FN}(P|Q) & := & \text{FN}(P) \cup \text{FN}(Q) \\
\text{FN}(P+Q) & := & \text{FN}(P) \cup \text{FN}(Q) \\
\text{FN}(A(\vec{y})) & := & \{y_1,\ldots,y_n\}.
\end{array}$$

where $\text{FN}(P)^-_y$ is yet to be defined. So FN differs from fn only when applied to processes $x(y).P$.

I will use the following definition of substitution:

$$\begin{array}{rcl}
\mathbf{0}\sigma & := & \mathbf{0} \\
(M\tau.P)\sigma & := & M[\sigma]\tau.(P\sigma) \\
(M\bar{x}y.P)\sigma & := & M[\sigma]\overline{x[\sigma]}y[\sigma].(P\sigma) \\
(Mx(y).P)\sigma & := & M[\sigma]x[\sigma](z).(P\sigma[y\mapsto z]) \\
((\nu y)P)\sigma & := & (\nu z)(P\{z/y\}\sigma) \\
([x{=}y]P)\sigma & := & [x[\sigma]{=}y[\sigma]](P\sigma) \\
(P|Q)\sigma & := & (P\sigma)|(Q\sigma) \\
(P+Q)\sigma & := & (P\sigma)+(Q\sigma) \\
A(\vec{y})\sigma & := & A(\vec{y}[\sigma])
\end{array}$$

where $P\sigma[y{\mapsto}z]$ is yet to be defined. Here z is chosen outside $\text{FN}((\nu y)P)\cup dom(\sigma)\cup range(\sigma)$; when $y\notin dom(\sigma)\cup range(\sigma)$ one always picks $z := y$. The original definition of substitution is retrieved by taking $P\sigma[y\mapsto z] := P\{z/y\}\sigma$.

To facilitate reuse, I record below the properties of substitutions, FN and $\equiv^{\mathcal{S}}$ that will be needed in my proofs.

$$\text{FN}(P\{\vec{y}/\vec{x}\}^{\mathcal{S}}) \subseteq \text{FN}(A(\vec{y})) \quad (4)$$

$$P \equiv^{\mathcal{S}} Q \Rightarrow \text{FN}(P) = \text{FN}(Q) \quad (5)$$

$$\text{FN}(P\{z/y\}^{\mathcal{S}}) \subseteq \text{FN}(x(y).P) \cup \{z\} \quad (6)$$

$$P\sigma[y\mapsto w]\{z[\sigma]/w\}^{\mathcal{S}} \equiv^{\mathcal{S}} P\{z/y\}^{\mathcal{S}}\sigma \quad (7)$$

$$\text{FN}(P\sigma) = \{x[\sigma] \mid x \in \text{FN}(P)\} \quad (8)$$

$$P\{\vec{y}[\sigma]/\vec{x}\}^{\mathcal{S}} \equiv^{\mathcal{S}} P\{\vec{y}/\vec{x}\}^{\mathcal{S}}\sigma \quad (9)$$

$$((\nu y)P) \equiv^{\mathcal{S}} (\nu w)(P\{w/y\}) \quad (10)$$

$$(P\sigma_1)\sigma_2 \equiv^{\mathcal{S}} P(\sigma_2\circ\sigma_1)^{11} \quad (11)$$

$$(\forall x \in \text{FN}(P).\, x[\sigma] = x[\sigma']) \Rightarrow P\sigma \equiv^{\mathcal{S}} P\sigma' \quad (12)$$

$$P \equiv^{\mathcal{S}} Q \Rightarrow P\sigma \equiv^{\mathcal{S}} Q\sigma \quad (13)$$

$$P\{w/y\}^{\mathcal{S}}\{z/w\} \equiv^{\mathcal{S}} P\{z/y\}^{\mathcal{S}} \quad (14)$$

$$P\epsilon \equiv^{\mathcal{S}} P \quad (15)$$

$$U \equiv^{\mathcal{S}} (x(y).P)\sigma \Rightarrow \begin{array}{l} U = x[\sigma](w).V \text{ with} \\ V \equiv^{\mathcal{S}} P\sigma[y\mapsto w] \end{array} \quad (16)$$

$$U \equiv^{\mathcal{S}} (\nu y)P \Rightarrow \begin{array}{l} U = (\nu w)V \text{ with} \\ V \equiv^{\mathcal{S}} P\{y/w\} \end{array} \quad (17)$$

Here $w \in \mathcal{N}\setminus\text{FN}((\nu y)P)$ in (10), $y\in\mathcal{N}$ and $w\in\text{FN}(x(y)P)$ in (14), $w \in \mathcal{N}\setminus(\text{FN}((\nu y)P)\cup dom(\sigma)\cup range(\sigma))$ in (7), $A(\vec{x}) \stackrel{\text{def}}{=} P$ in (4) and (9), and ϵ is the empty substitution. As is easy to check, these properties hold for the original π-calculus, that is, when using $\{z/y\}$ and $\{\vec{y}/\vec{x}\}$ for $\{z/y\}^{\mathcal{S}}$ and $\{\vec{y}/\vec{x}\}^{\mathcal{S}}$, $\equiv$ for $\equiv^{\mathcal{S}}$, fn for FN (i.e., $\text{FN}(P)^-_y := \text{FN}(P)\backslash\{y\}$), and $P\sigma[y\mapsto z] := P\{z/y\}\sigma$.

First I state two standard properties of names in transitions.

Define the *input arguments* $\iota(\alpha)$ of an action α by

$$\iota(M\bar{x}(y)) = \iota(M\bar{x}y) = \iota(M\tau) := \emptyset \quad \text{and} \quad \iota(Mxz) = \{z\}.$$

Lemma 35: If $P \xrightarrow{\alpha} Q$ then $\text{n}(\alpha)\backslash(\iota(\alpha)\cup\text{bn}(\alpha)) \subseteq \text{FN}(P)$.

Proof. A trivial induction on the inference of $P \xrightarrow{\alpha} Q$, using (4) and (5) when $P \xrightarrow{\alpha} Q$ is derived by **IDE** or **ALPHA**. ■

Lemma 36: If $P \xrightarrow{\alpha} Q$ then $\text{FN}(Q) \subseteq \text{FN}(P)\cup\iota(\alpha)\cup\text{bn}(\alpha)$.

Proof. A trivial induction on the inference of $P \xrightarrow{\alpha} Q$, using (6) when $P \xrightarrow{\alpha} Q$ is derived by **EARLY INPUT**, Lemma 35 when $P \xrightarrow{\alpha} Q$ is derived by **E-S-COM**, and (4) and (5) when $P \xrightarrow{\alpha} Q$ is derived by **IDE** or **ALPHA**. ■

Definition 17: The *depth* of an interference of a transition (from the rules in the operational semantics) is the length of the longest path in the proof tree of that inference. The *α-depth* is defined likewise, but not counting applications of rule **ALPHA**.

Following [24], in the following lemmas the phrase

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

means that if $P \xrightarrow{\alpha} P'$ may be inferred then so, by an inference of the same α-depth, may be $Q \xrightarrow{\alpha} Q'$.

For $\sigma : \mathcal{N} \rightharpoonup \mathcal{N}$ a substitution and α an action $M\tau$, Mxy, $M\bar{x}y$ or $M\bar{x}(z)$, write $\alpha[\sigma]$ for $M[\sigma]\tau$, $M[\sigma]x[\sigma]y[\sigma]$, $M[\sigma]\bar{x}[\sigma]y[\sigma]$ and $M[\sigma]\bar{x}[\sigma](z[\sigma])$, respectively.

Lemma 41 below shows how substitutions preserve behaviour. On the way to obtain it, I start with the special case of input and (bound) output transitions.

Lemma 37: If $R \xrightarrow{\alpha} R'$, with α not of the form $M\tau$, and σ is a substitution with $\text{bn}(\alpha[\sigma]) \cap \text{FN}(R\sigma) = \emptyset$, then equally $R\sigma \xrightarrow{\alpha[\sigma]} \equiv^{\mathcal{S}} R'\sigma$.

Proof. With induction on the inference of $R \xrightarrow{\alpha} R'$.

- Suppose $R \xrightarrow{\alpha} R'$ is derived by rule **EARLY-INPUT**. Then $R = Mx(y).P$, $\alpha = Mxz$ and $R' = P\{z/y\}^{\mathcal{S}}$. So $R\sigma = M[\sigma]x[\sigma](w).(P\sigma[y\mapsto w])\wedge\alpha[\sigma]{=}M[\sigma]x[\sigma]z[\sigma]$ where

[11] For substitutions σ_1 and σ_2, their composition $\sigma_2\circ\sigma_1$ is defined by $dom(\sigma_2\circ\sigma_1) = dom(\sigma_1)\cup dom(\sigma_2)$ and $x[\sigma_2\circ\sigma_1] = x[\sigma_1][\sigma_2]$.

w is chosen outside $\text{FN}((\nu y)P) \cup dom(\sigma) \cup range(\sigma)$. By rule **EARLY-INPUT** and (7)

$$R\sigma \xrightarrow{\alpha[\sigma]} P\sigma[y \mapsto w]\{z[\sigma]/w\}^{\mathcal{S}} \equiv^{\mathcal{S}} P\{z/y\}^{\mathcal{S}}\sigma = R'\sigma.$$

- The case that $R \xrightarrow{\alpha} R'$ is derived by **OUTPUT** is trivial.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by rule **IDE**. Then $R = A(\vec{y})$. Let $A(\vec{x}) \stackrel{\text{def}}{=} P$. By rule **IDE**, $P\{\vec{y}/\vec{x}\}^{\mathcal{S}} \xrightarrow{\alpha} R'$. Since $\text{FN}(P\{\vec{y}/\vec{x}\}^{\mathcal{S}}\sigma) \subseteq \text{FN}(R\sigma)$ by (4) and (8), $\text{bn}(\alpha[\sigma]) \cap \text{FN}(P\{\vec{y}/\vec{x}\}^{\mathcal{S}}\sigma) = \emptyset$. Hence $P\{\vec{y}/\vec{x}\}^{\mathcal{S}}\sigma \xrightarrow{\alpha[\sigma]} \equiv^{\mathcal{S}} R'\sigma$, by induction. By Property (9), $P\{\vec{y}[\sigma]/\vec{x}\}^{\mathcal{S}} \equiv^{\mathcal{S}} P\{\vec{y}/\vec{x}\}^{\mathcal{S}}\sigma$. Therefore $P\{\vec{y}[\sigma]/\vec{x}\}^{\mathcal{S}} \xrightarrow{\alpha[\sigma]} \equiv^{\mathcal{S}} R'\sigma$, by rule **ALPHA**. Thus $R\sigma = A(\vec{y}[\sigma]) \xrightarrow{\alpha[\sigma]} \equiv^{\mathcal{S}} R'\sigma$ by **IDE**.
- The cases that $R \xrightarrow{\alpha} R'$ is derived by rule **SUM**, **SYMB-MATCH** or **ALPHA** (using (5)) are trivial.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by **PAR**. Then $R = P|Q$, $P \xrightarrow{\alpha} P'$ and $R' = P'|Q$. Since $\text{bn}(\alpha[\sigma]) \cap \text{FN}(R\sigma) = \emptyset$, $\text{bn}(\alpha[\sigma]) \cap \text{FN}(P\sigma) = \emptyset$. So by induction $P\sigma \xrightarrow{\alpha[\sigma]} \equiv^{\mathcal{S}} P'\sigma$. Moreover $R\sigma = P\sigma|Q\sigma$ and $\text{bn}(\alpha[\sigma]) \cap \text{FN}(Q\sigma) = \emptyset$. So $R\sigma \xrightarrow{\alpha[\sigma]} \equiv^{\mathcal{S}} P'\sigma|Q\sigma = R'\sigma$ by rule **PAR**.
- Let $R \xrightarrow{\alpha} R'$ be derived by rule **RES**. Then $R = (\nu y)P$, $P \xrightarrow{\alpha} P'$, $y \notin \text{n}(\alpha)$ and $R' = (\nu y)P'$. Pick a w outside

  $\text{FN}((\nu y)P') \cup \text{FN}((\nu y)P) \cup dom(\sigma) \cup range(\sigma) \cup \text{n}(\alpha[\sigma])$.

  Then $R\sigma \equiv^{\mathcal{S}} ((\nu w)(P\{w/y\}))\sigma = (\nu w)(P\{w/y\}\sigma)$ by (10) and (13). Since $w \notin \text{bn}(\alpha[\sigma])$ and $\text{bn}(\alpha[\sigma]) \cap \text{FN}(R\sigma) = \emptyset$, by (5) also $\text{bn}(\alpha[\sigma]) \cap \text{FN}(P\{y/w\}\sigma) = \emptyset$. So by induction, using (11) and the substitution $\sigma \circ \{w/y\}$,

  $$P\{y/w\}\sigma \equiv^{\mathcal{S}} \xrightarrow{\alpha[\sigma]} \equiv^{\mathcal{S}} P'\{y/w\}\sigma.$$

  Thus $R\sigma \xrightarrow{\alpha[\sigma]} \equiv^{\mathcal{S}} (\nu w)(P'\{y/w\}\sigma)$ by **RES** and **ALPHA**. Furthermore, $(\nu w)(P'\{y/w\}\sigma) \equiv^{\mathcal{S}} R'\sigma$ by (10) and (13).
- Let $R \xrightarrow{M\bar{x}(y)} R'$ be derived by rule **SYMB-OPEN**. Then $R = (\nu y)P$, $P \xrightarrow{M\bar{x}y} R'$, $y \neq x$ and $y \notin \text{n}(M)$. Now $R\sigma = (\nu z)(P\{z/y\}\sigma)$ for some $z \notin \text{FN}((\nu y)P) \cup dom(\sigma) \cup range(\sigma)$. Since $w := y[\sigma] \notin \text{FN}(R\sigma)$ (the side-condition of this lemma), $R\sigma \equiv^{\mathcal{S}} (\nu w)(P\{z/y\}\sigma\{w/z\})$ by (10). Note that $u[\{w/z\} \circ \sigma \circ \{z/y\}] = u[\sigma]$ for $u \in \text{FN}(P)$. Hence $P\{z/y\}\sigma\{w/z\} \equiv^{\mathcal{S}} P\sigma$ by (11) and (12). By induction,

  $$P\sigma \xrightarrow{M[\sigma]\bar{x}[\sigma]w} \equiv^{\mathcal{S}} R'\sigma.$$

  By Lemma 35, $\text{n}(M) \cup \{x, y\} \subseteq \text{FN}(P)$. Hence $\text{n}(M) \cup \{x\} \subseteq \text{FN}(R) = \text{FN}(P) \backslash \{y\}$, using that $y \notin \text{n}(M) \cup \{x\}$. Thus $\text{n}(M[\sigma]) \cup \{x[\sigma]\} \subseteq \text{FN}(R\sigma)$ by (8), and therefore $w \notin \text{n}(M[\sigma]) \cup \{x[\sigma]\}$. Thus, by **SYMB-OPEN** and **ALPHA**,

  $$R\sigma \equiv^{\mathcal{S}} (\nu w)(P\sigma) \xrightarrow{M[\sigma]\bar{x}[\sigma](w)} \equiv^{\mathcal{S}} R'\sigma. \quad \blacksquare$$

Before generalising the above result to include the case $\alpha = M\tau$, I cover three more helpful standard properties.

Lemma 38: If $R \xrightarrow{Mxz} R_z$ and $w \notin \text{FN}(R)$ then equally $R \xrightarrow{Mxw} R_w$ for some process R_w with $R_z \equiv^{\mathcal{S}} R_w\{z/w\}$.

Proof. With induction on the inference of $P \xrightarrow{Mxz} P_z$.

- Suppose $R \xrightarrow{Mxz} R_z$ is derived by rule **EARLY-INPUT**. Then $R = Mx(y).P$ and $R_z = P\{z/y\}^{\mathcal{S}}$. Moreover, $R \xrightarrow{Mxw} R_w := P\{w/y\}^{\mathcal{S}}$. By (14), $R_z \equiv^{\mathcal{S}} R_w\{z/w\}$.
- Suppose $R \xrightarrow{Mxz} R_z$ is derived by **PAR**. Then $R = P|Q$, $P \xrightarrow{Mxz} P_z$ and $R_z = P_z|Q$. So by induction there is a process P_w such that $P \xrightarrow{Mxw} P_w$ and $P_z \equiv^{\mathcal{S}} P_w\{z/w\}$. Thus $R \xrightarrow{Mxw} P_w|Q$ by **PAR**. Moreover, $(P_w|Q)\{z/w\} = P_w\{z/w\}|Q\{z/w\} \equiv^{\mathcal{S}} P_z|Q = R_z$ by (12) and (15), using that $w \notin \text{FN}(Q)$.
- Let $R \xrightarrow{Mxz} R_z$ be derived by rule **RES**. Then $R = (\nu y)P$, $P \xrightarrow{Mxz} P_z$, $y \notin \text{n}(M) \cup \{x, z\}$ and $R_z = (\nu y)P_z$. Pick a $v \notin \text{FN}(P) \cup \text{n}(M) \cup \{x, y\}$. By induction there is a process P_v such that $P \xrightarrow{Mxv} P_v$ and $P_z \equiv^{\mathcal{S}} P_v\{z/v\}$. So $R \xrightarrow{Mxv} (\nu y)P_v$ by **RES**. By (12), (15) and Lemma 37,

  $$R \equiv^{\mathcal{S}} R\{w/v\} \xrightarrow{Mxw} R_w \equiv^{\mathcal{S}} ((\nu y)P_v)\{w/v\}$$

  for some R_w. Hence $R \xrightarrow{Mxw} R_w$ by rule **ALPHA**. By Lemma 36 $\text{FN}(P_v) \backslash \{y, v\} \subseteq \text{FN}(P) \backslash \{y\} = \text{FN}(R) \not\ni w$. Pick $u \notin \text{FN}(P_v) \cup \text{FN}(P_v\{z/v\}) \cup \{v, w\}$. Then

  $$\begin{aligned} R_w\{z/w\} &\equiv^{\mathcal{S}} ((\nu y)P_v)\{w/v\}\{z/w\} \equiv^{\mathcal{S}} \\ ((\nu u)P_v\{u/v\})\{w/v\}\{z/w\} &\equiv^{\mathcal{S}} (\nu u)(P_v\{u/y\}\{w/v\}\{z/w\}) \\ \equiv^{\mathcal{S}} (\nu u)(P_v\{z/v\}\{u/y\}) &\equiv^{\mathcal{S}} (\nu y)(P_v\{z/v\}) \equiv^{\mathcal{S}} R_z \end{aligned}$$

  by (10)–(13), using that $v \neq y \neq z$ and $w \neq u \neq v$.
- The other four cases are trivial, using (4) and (5) when $P \xrightarrow{\alpha} Q$ is derived by **IDE** or **ALPHA**. $\blacksquare$

Lemma 39: If $R \xrightarrow{Mxw} R_w$ with $w \notin \text{FN}(R)$, and z a name, then equally $R \xrightarrow{Mxz} R_z$ for some R_z with $R_z \equiv^{\mathcal{S}} R_w\{z/w\}$.

Proof. With induction on the inference of $R \xrightarrow{Mxw} R_w$.

- Let $R \xrightarrow{Mxw} R_w$ be derived by rule **RES**. Then $R = (\nu y)P$, $P \xrightarrow{Mxw} P_w$, $y \notin \text{n}(M) \cup \{x, w\}$ and $R_w = (\nu y)P_w$. Pick a $v \notin \text{FN}(P) \cup \text{FN}(P_w) \cup \text{n}(M) \cup \{x, y\}$. By induction there is a process P_v with $P \xrightarrow{Mxv} P_v$ and $P_v \equiv^{\mathcal{S}} P_w\{z/w\}$. So $R \xrightarrow{Mxv} (\nu y)P_v$ by **RES**. By (12), (15) and Lemma 37,

  $$R \equiv^{\mathcal{S}} R\{z/v\} \xrightarrow{Mxz} R_z \equiv^{\mathcal{S}} ((\nu y)P_v)\{z/v\}$$

  for some R_z. Hence $R \xrightarrow{Mxz} R_z$ by rule **ALPHA**. Pick $u \notin \text{FN}(P_w) \cup \text{FN}(P_v) \cup \{z, v, w\}$. Then

  $$\begin{aligned} R_w\{z/w\} &\equiv^{\mathcal{S}} ((\nu y)P_w)\{z/w\} \equiv^{\mathcal{S}} \\ ((\nu u)P_w\{u/y\})\{z/w\} &\equiv^{\mathcal{S}} ((\nu u)P_w\{u/y\}\{z/w\}) \equiv^{\mathcal{S}} \\ (\nu u)(P_w\{z/w\}\{u/y\}\{z/v\}) &\equiv^{\mathcal{S}} (\nu u)(P_v\{u/y\}\{z/v\}) \\ \equiv^{\mathcal{S}} (\nu u)(P_v\{u/y\})\{z/v\} &\equiv^{\mathcal{S}} ((\nu y)P_v)\{z/v\} \equiv^{\mathcal{S}} R_z \end{aligned}$$

  by (10)–(13), using that $v \neq z \neq y \neq w \neq u \neq v$.
- All other cases proceed as in the proof of Lemma 38. $\blacksquare$

The above proofs caters both to the full π-calculus and to π_{IM}, namely by considering explicitly the input prefix $Mx(y)$. Henceforth I focus on the full π-calculus, and drop the M from the input prefix. It can always be retrieved via an application of **SYMB-MATCH**.

Lemma 40: If $R \xrightarrow{Mx(z)} R'$ and $w \notin \text{FN}(R)$ then equally $R \xrightarrow{Mx(w)} \equiv^{\mathcal{S}} R'\{w/z\}$.

Proof. With induction on the inference of $R \xrightarrow{Mx(z)} R'$.

- Let $R \xrightarrow{M\bar{z}(z)} R'$ be derived by rule **SYMB-OPEN**. Then $R = (\nu z)P$, $P \xrightarrow{M\bar{x}z} R'$, $z \neq x$ and $z \notin \text{n}(M)$. By

Lemma 37, $P\{w/z\} \xrightarrow{M\bar{x}w} \equiv^{\mathcal{S}} R'\{w/z\}$. By Lemma 35, $\mathrm{n}(M)\cup\{x\} \subseteq \mathrm{FN}(R)$. Hence $w \notin \mathrm{n}(M)\cup\{x\}$. By (10), $R \equiv^{\mathcal{S}} (\nu w)(P\{w/z\})$, so rules **SYMB-OPEN** and **ALPHA** yield $R \xrightarrow{M\bar{x}(w)} \equiv^{\mathcal{S}} R'\{w/z\}$.

- Let $R \xrightarrow{M\bar{x}(z)} R'$ be derived by rule **RES**. Then $R{=}(\nu y)P$, $P \xrightarrow{M\bar{x}(z)} P'$, $y \notin \mathrm{n}(M)\cup\{x,z\}$ and $R' = (\nu y)P'$. Pick a $v \notin \{y\}\cup\mathrm{FN}(P)\cup\mathrm{FN}(P')\cup\mathrm{n}(M)\cup\{x\}$. By induction $P \xrightarrow{M\bar{x}(v)} \equiv^{\mathcal{S}} P'\{v/z\}$. So $R \xrightarrow{M\bar{x}(v)} \equiv^{\mathcal{S}} (\nu y)(P'\{v/z\})$ by **RES**. By (12), (15) and Lemma 37,
$$R \equiv^{\mathcal{S}} R\{w/v\} \xrightarrow{M\bar{x}(w)} \equiv^{\mathcal{S}} ((\nu y)(P'\{v/z\}))\{w/v\}.$$
Pick a $u \notin \mathrm{FN}(P')\cup\mathrm{FN}(P'\{v/z\})\cup\{z,v,w\}$. Then
$$((\nu y)(P'\{v/z\}))\{w/v\} \equiv^{\mathcal{S}} (\nu u)(P'\{v/z\}\{u/y\}\{w/v\}) \equiv^{\mathcal{S}} (\nu u)(P'\{u/y\}\{w/z\}) \equiv^{\mathcal{S}} ((\nu y)P')\{w/z\} = R'\{w/z\},$$
by (10)–(13), using that $\mathrm{FN}(P') \not\ni v \neq y \neq z \neq u \neq v$. By rule **ALPHA**, $R \xrightarrow{M\bar{x}(w)} \equiv^{\mathcal{S}} R'\{w/z\}$.
- The other five cases are trivial. ∎

Lemma 41: If $R \xrightarrow{\alpha} R'$ and σ is a substitution with $\mathrm{bn}(\alpha[\sigma]) \cap \mathrm{FN}(R\sigma) = \emptyset$ then equally $R\sigma \xrightarrow{\alpha[\sigma]} \equiv^{\mathcal{S}} R'\sigma$.

Proof. With induction on the inference of $R \xrightarrow{\alpha} R'$.

- The case that $R \xrightarrow{\alpha} R'$ is derived by rule **TAU** is trivial.
- The cases that $R \xrightarrow{\alpha} R'$ is derived by rule **OUTPUT**, **EARLY-INPUT**, **SUM**, **SYMB-MATCH**, **IDE**, **PAR**, **RES**, **SYMB-OPEN** or **ALPHA** are already covered by Lemma 37.
- The case that $R \xrightarrow{\alpha} R'$ is derived by **E-S-COM** is trivial.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by **E-S-CLOSE**. Then $R = P|Q$, $P \xrightarrow{M\bar{x}(z)} P'$, $Q \xrightarrow{N\bar{v}z} Q'$, $z \notin \mathrm{FN}(Q)$, $\alpha = [x{=}v]MN\tau$ and $R' = (\nu z)(P'|Q')$. Pick $w \notin \mathrm{FN}(R)\cup\mathrm{FN}(R')\cup dom(\sigma)\cup range(\sigma)$. Now $P \xrightarrow{M\bar{x}(w)} \equiv^{\mathcal{S}} P'\{w/z\}$ and $Q \xrightarrow{N\bar{v}w} \equiv^{\mathcal{S}} Q'\{w/z\}$ by Lemmas 40 and 39. Thus $P\sigma \xrightarrow{M[\sigma]\bar{x}[\sigma](w)} \equiv^{\mathcal{S}} P'\{w/z\}\sigma$, $Q\sigma \xrightarrow{N[\sigma]\bar{v}[\sigma]w} \equiv^{\mathcal{S}} Q'\{w/z\}\sigma$ by Lemma 37, using that $w[\sigma] = w \notin \mathrm{FN}(P\sigma)$, which follows by (8) from $w \notin \mathrm{FN}(P)\cup range(\sigma)$. Hence
$$R\sigma \xrightarrow{[x[\sigma]=v[\sigma]]M[\sigma]N[\sigma]\tau} (\nu w)((P'|Q')\{w/z\}\sigma)$$
by rule **E-S-CLOSE**. As $w \notin \mathrm{FN}((\nu z)(P'|Q'))$, by (10),
$$R'\sigma \equiv^{\mathcal{S}} (\nu w)((P'|Q')\{w/z\}\sigma).$$ ∎

Next I would have liked to show that substitutions also reflect behaviour, in the sense that, possibly under the same side condition as in Lemma 41, $R\sigma \xrightarrow{\beta} U$ implies $R \xrightarrow{\alpha} R'$ for some α and R' with $\alpha[\sigma] = \beta$ and $R'\sigma \equiv^{\mathcal{S}} U$. Unfortunately, this property does not hold.

Example 10: $(x(y).P)\{w/z\} \xrightarrow{xz} P\{w/z\}\{z/y\}$, yet there is no α such that $\alpha[\{w/z\}] = xz$.

In order to rescue the reflection property, I restrict attention to surjective substitutions.

Definition 18: A substitution $\sigma : \mathcal{N} \rightharpoonup \mathcal{N}$ is called *surjective* if $dom(\sigma) \subseteq range(\sigma)$.

Using rule **ALPHA**, the below reflection lemma can equally well be formulated with $R\sigma \xrightarrow{\beta} U'$ instead of $R\sigma \equiv^{\mathcal{S}} U \xrightarrow{\beta} U'$. However, its inductive proof requires the form stated.

Lemma 42: If $R\sigma \equiv^{\mathcal{S}} U \xrightarrow{\beta} U'$ and σ is a surjective substitution, then equally $R \xrightarrow{\alpha} R'$ for some α and R' with $\alpha[\sigma] = \beta$ and $R'\sigma \equiv^{\mathcal{S}} U'$.

Proof. With induction on the α-depth of the inference of $U \xrightarrow{\beta} U'$, with a nested induction on the number of applications of rule **ALPHA** at the end of that inference.

- The cases that $U \xrightarrow{\beta} U'$ is derived by rule **TAU** or **OUTPUT** are trivial.
- Suppose $U \xrightarrow{\beta} U'$ is derived by **EARLY-INPUT**. Then $R = x(y).P$ and $U = x[\sigma](w).V$ with $V \equiv^{\mathcal{S}} (P\sigma[y \mapsto w])$ by (16). So $\beta = x[\sigma]v$ and $U' = V\{v/w\}^{\mathcal{S}}$. Since σ is surjective, there is a z with $z[\sigma] = v$. By **EARLY-INPUT** $R \xrightarrow{\alpha} R'$ with $\alpha{=}Mxz$ and $R'{=}P\{z/y\}^{\mathcal{S}}$. Now $\alpha[\sigma]{=}\beta$ and $R'\sigma = P\{z/y\}^{\mathcal{S}}\sigma \equiv^{\mathcal{S}} P\sigma[y \mapsto w]\{z[\sigma]/w\}^{\mathcal{S}} \equiv^{\mathcal{S}} V\{v/w\}^{\mathcal{S}} = U'$, using (7) and (12).
- Suppose $U \xrightarrow{\beta} U'$ is derived by **IDE**. Then $R = A(\vec{y})$ and $U{=}R\sigma{=}A(\vec{y}[\sigma])$. Let $A(\vec{x}) \stackrel{\text{def}}{=} P$. Then $P\{\vec{y}[\sigma]/\vec{x}\} \xrightarrow{\beta} U'$. Since $P\{\vec{y}[\sigma]/\vec{x}\} \equiv^{\mathcal{S}} P\{\vec{y}/\vec{x}\}\sigma$ by (9), $P\{\vec{y}/\vec{x}\}\sigma \xrightarrow{\beta} U'$ by **ALPHA**. Moreover, the inference of $P\{\vec{y}/\vec{x}\}\sigma \xrightarrow{\beta} U'$ has the same α-depth as that of $P\{\vec{y}[\sigma]/\vec{x}\} \xrightarrow{\beta} U'$, which is smaller than the α-depth of the inference of $U \xrightarrow{\beta} U'$. Hence, by induction, $P\{\vec{y}/\vec{x}\} \xrightarrow{\alpha} R'$ for some α and R' with $\alpha[\sigma] = \beta$ and $R'\sigma \equiv^{\mathcal{S}} U'$. By rule **IDE** $R \xrightarrow{\alpha} R'$.
- The cases that $U \xrightarrow{\beta} U'$ is derived by rule **SUM** or **SYMB-MATCH** are trivial.
- Suppose $U \xrightarrow{\beta} U'$ is derived by **PAR**. Then $R = P|Q$, $R\sigma = P\sigma|Q\sigma$, $U = V|W$, $P\sigma \equiv^{\mathcal{S}} V \xrightarrow{\beta} V'$, $Q\sigma \equiv^{\mathcal{S}} W$, $\mathrm{bn}(\beta)\cap\mathrm{FN}(W){=}\emptyset$ and $U'{=}V'|W$. By induction $P \xrightarrow{\alpha} P'$ for some α and P' with $\alpha[\sigma]{=}\beta$ and $P'\sigma \equiv^{\mathcal{S}} V'$. By (5) $\mathrm{bn}(\alpha[\sigma]) \cap \mathrm{FN}(Q\sigma) = \emptyset$, so by (8) $\mathrm{bn}(\alpha) \cap \mathrm{FN}(Q) = \emptyset$. Consequently, $R = P|Q \xrightarrow{\alpha} P'|Q$ by rule **PAR**, and $(P'|Q)\sigma \equiv^{\mathcal{S}} V'|W = U'$.
- Suppose $U \xrightarrow{\beta} U'$ is derived by **E-S-COM**. In that case $R = P|Q$, $R\sigma = P\sigma|Q\sigma$, $U = V|W$, $\beta = [x{=}v]MN\tau$, $P\sigma \equiv^{\mathcal{S}} V \xrightarrow{M\bar{x}y} V'$, $Q\sigma \equiv^{\mathcal{S}} W \xrightarrow{Nvy} W'$ and $U'{=}V'|W'$. By induction, there are matching sequences K, L with $K[\sigma] = M$ and $L[\sigma] = N$, names q, r, z, u with $q[\sigma] = x$, $r[\sigma] = v$, $z[\sigma] = y$ and $u[\sigma] = y$, and processes P' and Q' with $P'\sigma \equiv^{\mathcal{S}} V'$ and $Q'\sigma \equiv^{\mathcal{S}} W'$, such that $P \xrightarrow{K\bar{q}z} P'$ and $Q \xrightarrow{Lru} Q'$.
Pick $w \notin \mathrm{FN}(Q)$. By Lemma 38 there is a process P_w such that $Q \xrightarrow{Lrw} Q_w$ and $Q' \equiv^{\mathcal{S}} Q_w\{u/w\}$. By Lemma 39 there is a P_z such that $Q \xrightarrow{Lrz} Q_z$ and $Q_z \equiv^{\mathcal{S}} Q_w\{z/w\}$. By rule **E-S-COM** $R \xrightarrow{[q=r]KL\tau} R' := P'|Q_z$. By (11)–(13) $Q'\sigma \equiv^{\mathcal{S}} Q_z\sigma$. So $([q{=}r]KL\tau)[\sigma] = [x{=}v]MN\tau$ and $R'\sigma = (P'|Q_z)\sigma \equiv^{\mathcal{S}} V'|W' = U'$.
- Suppose $U \xrightarrow{\beta} U'$ is derived by rule **E-S-CLOSE**. Then $R = P|Q$, $R\sigma = P\sigma|Q\sigma$, $U = V|W$, $\beta = [x{=}v]MN\tau$, $P\sigma \equiv^{\mathcal{S}} V \xrightarrow{M\bar{x}(z)} V'$, $Q\sigma \equiv^{\mathcal{S}} W \xrightarrow{Nvz} W'$ and $U' = (\nu z)(V'|W')$ for some $z \notin \mathrm{FN}(W)$. Pick $w \notin \mathrm{FN}(U)\cup dom(\sigma)\cup range(\sigma)$. By (5) $w \notin \mathrm{FN}(W) = \mathrm{FN}(Q\sigma)$ so $w \notin \mathrm{FN}(Q)$ by (8). Now $V \xrightarrow{M\bar{x}(w)} \equiv^{\mathcal{S}} V'\{w/z\}$ and $W \xrightarrow{Nvw} \equiv^{\mathcal{S}} W'\{w/z\}$ by Lemmas 40 and 39. Moreover, the inferences of these transitions have a smaller α-depth

than that of $U \xrightarrow{\beta} U'$. By induction, using that $w \notin dom(\sigma) \cup range(\sigma)$, there are matching sequences K, L with $K[\sigma] = M$ and $L[\sigma] = N$, names q, r with $q[\sigma] = x$ and $r[\sigma] = v$, and processes P' and Q' with $P'\sigma \equiv^{\mathcal{S}} V'\{w/z\}$ and $Q'\sigma \equiv^{\mathcal{S}} W'\{w/z\}$, such that $P \xrightarrow{K\bar{q}(w)} P'$ and $Q \xrightarrow{Lrw} Q'$. Therefore $R \xrightarrow{[q=r]KL\tau} (\nu w)(P'|Q')$ by **E-S-COM**. Moreover, $([q{=}r]KL\tau)[\sigma] = \beta$. By Lemma 36, $\text{FN}(U') \subseteq \text{FN}(U) \not\ni w$. So by (10) $((\nu w)(P'|Q'))\sigma = (\nu w)(P'\sigma|Q'\sigma) \equiv^{\mathcal{S}} (\nu w)(V'\{w/z\}|W'\{w/z\}) \equiv^{\mathcal{S}} U'$.

- Suppose $U \xrightarrow{\beta} U'$ is derived by **RES**. Then $R = (\nu y)P$ and $R\sigma = (\nu z)(P\{z/y\}\sigma)$ for some $z \notin \text{FN}((\nu y)P) \cup dom(\sigma) \cup range(\sigma)$. By (17), $U = (\nu w)V$ with $V \equiv^{\mathcal{S}} P\{z/y\}\sigma\{w/z\}$. So $w \notin \text{n}(\beta)$, $V \xrightarrow{\beta} V'$ and $U' = (\nu w)V'$. Since σ is surjective, there is a u with $z[\sigma] = w$. Now $V \equiv^{\mathcal{S}} P\{z/y\}\sigma\{w/z\} \equiv^{\mathcal{S}} P\{u/y\}\sigma$ by (11) and (12). By rule **ALPHA** $P\{u/y\}\sigma \xrightarrow{\beta} V'$. Moreover, the inference of $P\{u/y\}\sigma \xrightarrow{\beta} V'$ has the same α-depth as that of $V \xrightarrow{\beta} V'$, which is smaller than the one of $U \xrightarrow{\beta} U'$. So by induction $P\{u/y\} \xrightarrow{\alpha} P'$ for some α and P' with $\alpha[\sigma] = \beta$ and $P'\sigma \equiv^{\mathcal{S}} V'$. Since $w \notin \text{n}(\beta)$ one has $u \notin \text{n}(\alpha)$. Hence $(\nu u)(P\{u/y\}) \xrightarrow{\alpha} (\nu u)P'$ by **RES**. Moreover, since $w \notin \text{FN}((\nu w)V) = \text{FN}(U) = \text{FN}(R\sigma)$, using (5), $u \notin \text{FN}(R)$ by (8). Hence $R \equiv^{\mathcal{S}} (\nu u)(P\{u/y\})$ by (10) and $R \xrightarrow{\alpha} (\nu u)P'$ by **ALPHA**.
  Suppose that there is an $x \in \text{FN}((\nu u)P')$ with $x[\sigma] = w$. Then, by Lemma 36, $x \in \text{n}(\alpha)$ or $x \in \text{FN}(R)$. In the first case $w \in \text{n}(\alpha[\sigma]) = \text{n}(\beta)$, which has been ruled out. In the second case, by (8), $w \in \text{FN}(R\sigma) = \text{FN}(U)$, contradicting $U = (\nu w)V$. Hence there is no such x. Now pick $v \notin \text{FN}(P') \cup dom(\sigma) \cup range(\sigma) \cup \text{FN}(P'\sigma)$ and obtain

$$\begin{array}{lll} ((\nu u)P')\sigma & \equiv^{\mathcal{S}} ((\nu v)(P'\{v/u\}))\sigma & \text{by (10), (13)} \\ & = (\nu v)(P'\{v/u\}\sigma) & \text{by definition} \\ & \equiv^{\mathcal{S}} (\nu v)(P'\sigma\{v/w\}) & \text{by (11), (12)} \\ & \equiv^{\mathcal{S}} (\nu w)(P'\sigma) & \text{by (10)} \\ & \equiv^{\mathcal{S}} (\nu w)V' = U'. & \end{array}$$

- Suppose $U \xrightarrow{\beta} U'$ is derived by **SYMB-OPEN**. Then $R = (\nu y)P$ and $R\sigma = (\nu z)(P\{z/y\}\sigma)$ for some $z \notin \text{FN}((\nu y)P) \cup dom(\sigma) \cup range(\sigma)$. By (17), $U = (\nu w)V$ with $V \equiv^{\mathcal{S}} P\{z/y\}\sigma\{w/z\}$. So $\beta = M\bar{x}(w)$, $V \xrightarrow{M\bar{x}w} U'$ and $w \notin \text{n}(M) \cup \{x\}$. Since σ is surjective, there is a u with $u[\sigma] = w$. Now $V \equiv^{\mathcal{S}} P\{z/y\}\sigma\{w/z\} \equiv^{\mathcal{S}} P\{u/y\}\sigma$ by (11) and (12). By rule **ALPHA** $P\{u/y\}\sigma \xrightarrow{M\bar{x}w} U'$. Moreover, the inference of $P\{u/y\}\sigma \xrightarrow{M\bar{x}w} U'$ has the same α-depth as that of $V \xrightarrow{M\bar{x}w} U'$, which is smaller than the one of $U \xrightarrow{\beta} U'$. By induction $P\{u/y\} \xrightarrow{K\bar{q}r} P'$ for some K, q, r and P' with $K[\sigma]{=}M$, $q[\sigma]{=}x$, $r[\sigma]{=}w$ and $P'\sigma \equiv^{\mathcal{S}} U'$.
  Since $u[\sigma] = w \notin \text{FN}((\nu w)V) = \text{FN}(U) = \text{FN}(R\sigma)$, using (5), $u \notin \text{FN}(R)$ by (8). Hence $R \equiv^{\mathcal{S}} (\nu u)(P\{u/y\})$ by (10). By Lemma 35 $r \in \text{FN}(P\{u/y\})$. So if $r \neq u$ then $r \in \text{FN}((\nu u)(P\{u/y\}) = \text{FN}(R)$, again using (5), and $w = r[\sigma] \in \text{FN}(R\sigma)$ by (8), yielding a contradiction. Thus $r = u$. Since $w = u[\sigma] \notin \text{n}(M) \cup \{x\}$ one has $y \notin \text{n}(K) \cup \{q\}$. Hence $(\nu u)(P\{u/y\}) \xrightarrow{K\bar{q}(u)} P'$ by **SYMB-OPEN** and $R \xrightarrow{K\bar{q}(u)} P'$ by **ALPHA**.
- Suppose $U \xrightarrow{\beta} U'$ is derived by **ALPHA**. Then there is a $V \equiv^{\mathcal{S}} U$ such that $V \xrightarrow{\alpha} U'$ is derived by a simpler proof. So $R\sigma \equiv^{\mathcal{S}} V$ and by induction $R \xrightarrow{\alpha} R'$ for some α and R' with $\alpha[\sigma] = \beta$ and $R'\sigma \equiv^{\mathcal{S}} U'$. ■

B. A π-calculus with surjective substitutions

When needed I will address the π-calculus of Section V as $\pi(\mathcal{N})$, to make the choice of the set $\mathcal{N}$ of names explicit. Here I introduce a variant $\pi^{\mathcal{S}}(\mathcal{N})$ of the π-calculus that additionally employs a countably infinite collection of *spare names* $\mathcal{S} = \{s_1, s_2, \ldots\}$, disjoint with $\mathcal{N}$. Its syntax is the same as the one of $\pi(\mathcal{N} \uplus \mathcal{S})$, except that in expressions $x(y).P$ one must take $y \in \mathcal{N}$. Moreover, in defining equations $A(\vec{x}) \stackrel{\text{def}}{=} P$ I require that $x_1, \ldots, x_n \in \mathcal{N}$. The definition of substitution on $\pi^{\mathcal{S}}(\mathcal{N})$ is fine-tuned by requiring that when a bound name $y \in \mathcal{N}$ is changed into a bound name z to avoid name capture, one always chooses $z \in \mathcal{N}$.

As semantics of $\pi^{\mathcal{S}}(\mathcal{N})$ I use the variant $\xrightarrow{\alpha}{}^{\mathcal{S}}_{ES}$ of the early symbolic transition relation $\xrightarrow{\alpha}_{ES}$ where the substitutions $\{z/y\}$ and $\{\vec{y}/\vec{x}\}$ introduced by the operational rules **EARLY-INPUT** and **IDE** are changed into surjective substitutions $\{z/y\}^{\mathcal{S}}$ and $\{\vec{y}/\vec{x}\}^{\mathcal{S}}$. The motivation for this is to make Lemma 42 applicable to these substitutions. In order to preserve the validity of (4)–(17), this change forces new definitions of $\equiv^{\mathcal{S}}$, FN and $\sigma[y \mapsto w]$.

Definition 19: Let $\mathcal{H}$ be the set of all names currently in use, for now that is $\mathcal{N} \cup \mathcal{S}$. For $\vec{x} = (x_1, \ldots, x_n) \in \mathcal{N}^n$ and $\vec{y} = (y_1, \ldots, y_n) \in \mathcal{H}^n$, where the names x_i are all distinct, define the surjective substitution

$$\{\vec{x}/\vec{y}\}^{\mathcal{S}} : \mathcal{S} \cup \{x_1, \ldots, x_n\} \to \mathcal{S} \cup \{x_1, \ldots, x_n, y_1, \ldots, y_n\}$$

by $\{\vec{x}/\vec{y}\}^{\mathcal{S}}(x_i) = y_i$ and $\{\vec{x}/\vec{y}\}^{\mathcal{S}}(s_i) = x_i$ for $i = 1, \ldots, n$, and $\{\vec{x}/\vec{y}\}^{\mathcal{S}}(s_i) = s_{i-n}$ for $i > n$.

This is in fact the simplest possible adaptation of the substitution $\{\vec{y}/\vec{x}\}$ that makes it surjective. It is essential for my purposes that the axioms (4)–(17) of Section A be satisfied, so that Lemmas 41 and 42 hold for $\pi^{\mathcal{S}}(\mathcal{N})$. In view of Property (11), to obtain (7) I need to ensure that

$$\{z[\sigma]/w\}^{\mathcal{S}} \circ \sigma[y \mapsto w] = \sigma \circ \{z/y\}^{\mathcal{S}}. \tag{18}$$

The definition of $\sigma[y \mapsto w]$ can be found by solving this equation. In Table VII, $\sigma : \mathcal{H} \rightharpoonup \mathcal{H}$ is an arbitrary substitution, $y, w \in \mathcal{N}$, and $w \notin dom(\sigma) \cup range(\sigma)$. Thus, $\sigma[y \mapsto w]$ is defined as the substitution that sends a name from the first column to the corresponding name from the last column. The middle columns show that now (18) is satisfied.

Definition 20: For $x \in \mathcal{H}$ and $y \in \mathcal{N}$, let

$$x^-_y := \begin{cases} y & \text{if } x = s_1 \\ s_k & \text{if } x = s_{k+1} \\ x & \text{otherwise} \end{cases} \quad \text{and} \quad x^+_y := \begin{cases} s_1 & \text{if } x = y \\ s_{k+1} & \text{if } x = s_k \\ x & \text{otherwise.} \end{cases}$$

Now $\sigma[y \mapsto w]$, with $y, w \in \mathcal{N}$, is the substitution with $dom(\sigma[y \mapsto w]) = dom(\sigma) \cup \{y, w\} \cup \mathcal{S}$ given by

$$x[\sigma[y \mapsto w]] := \begin{cases} w & \text{if } x = y \\ (x^-_y[\sigma])^+_w & \text{otherwise.} \end{cases}$$

Name n	$n[\{z/y\}^{\mathcal{S}}]$	$n[\{z/y\}^{\mathcal{S}}][\sigma] = n[\sigma[y\mapsto w]][\{z[\sigma]/w\}^{\mathcal{S}}]$		$n[\sigma[y\mapsto w]]$
y	z	$z[\sigma]$		w
w (if $\neq y$)	w	w		s_1
s_1 if $y=w$	w	w		s_1
s_1 if $y\neq w$	y	$y[\sigma]\neq w$	if $y[\sigma]\in\mathcal{N}$ if $y[\sigma]=s_k$	$y[\sigma]$ s_{k+1}
s_{i+1}	s_i	$s_i[\sigma]\neq w$	if $s_i[\sigma]\in\mathcal{N}$ if $s_i[\sigma]=s_k$	$s_i[\sigma]$ s_{k+1}
$x\in\mathcal{N}$ $x\neq y,w$	x	$x[\sigma]\neq w$	if $x[\sigma]\in\mathcal{N}$ if $x[\sigma]=s_k$	$x[\sigma]$ s_{k+1}

TABLE VII
SOLVING (18)

As far as the computational interpretation of the π-calculus concerns, the replacement of $P\{\vec{y}/\vec{x}\}$ by $P\{\vec{y}/\vec{x}\}^{\mathcal{S}}$ in rule **IDE** is of no consequence, as the names from $\mathcal{S}$ do not occur free in P anyway. However, the replacement of $P\{z/y\}$ by $P\{z/y\}^{\mathcal{S}}$ in rule **EARLY-INPUT** changes the meaning of the construct $x(y).P$. In the original π-calculus there are no free occurrences of y in $x(y).P$, and thus, after receiving a name $z\neq y$, the resulting process $P\{z/y\}$ cannot do an action $\xrightarrow{\bar{y}w}$. Here, however, upon receiving a name $z\neq y$, in the resulting process $P\{z/y\}^{\mathcal{S}}$ the spare name s_1 is elevated to y, so that $P\{z/y\}^{\mathcal{S}}\xrightarrow{\bar{y}w}$ is possible. For this reason, y can be considered a free name of $x(y).P$ when s_1 is a free name of P. More in general, a spare name s_k can be seen as a potential name $y\in\mathcal{N}$. This potential is realised when s_k occurs in the scope of k input prefixes, and the choice of y is made by the outermost of these. Hence the definition of FN, partly given in Section A, is completed by defining $\text{FN}(P)^-_y := \{x^-_y \mid x\in\text{FN}(P)\backslash\{y\}\}$.

Definition 20 matches this intuition. To apply $\sigma[y\mapsto w]$ to a name $x\neq y$, first elevate x one level, then apply σ, and finally undo the elevation.

Note that $\text{T}_{\pi(\mathcal{N})}\subseteq\text{T}_{\pi^{\mathcal{S}}(\mathcal{N})}$, that is, the ordinary π-calculus processes form a subset of the processes in the π-calculus with surjective substitutions. A process $P\in\text{T}_{\pi(\mathcal{N})}$ by definition satisfies $\text{fn}(P)\subseteq\mathcal{N}$. The following lemma implies that on $\text{T}_{\pi(\mathcal{N})}$ there is no difference between FN and fn.

Lemma 43: If $\text{fn}(P)\subseteq\mathcal{N}$ then $\text{FN}(P)=\text{fn}(P)$.

Proof. A trivial structural induction on P. ∎

I now proceed to show that the axioms (4), (6) and (8) from Section A—the ones not mentioning $\equiv^{\mathcal{S}}$—are satisfied.

Lemma 44: $\text{FN}(P\sigma)=\{x[\sigma]\mid x\in\text{FN}(P)\}$.

Proof. With induction on the size of the parse tree of P. Note that this size is preserved under substitution. All cases are trivial, except for the two detailed below.

Let $P=x(y).Q$. Then $P\sigma=x[\sigma](z).(Q\sigma[y\mapsto z])$, so $\text{FN}(P\sigma)=\{x[\sigma]\}\cup\text{FN}(Q\sigma[y\mapsto z])^-_z$. By induction,

$$\text{FN}(Q\sigma[y\mapsto z])=\{v[\sigma[y\mapsto z]]\mid v\in\text{FN}(Q)\}=\{z\mid y\in\text{FN}(Q)\}\cup\{(v^-_y[\sigma])^+_z\mid v\in\text{FN}(Q)\backslash\{y\}\},$$

so $\text{FN}(Q\sigma[y\mapsto z])^-_z=\{x^-_z\mid x\in\text{FN}(Q\sigma[y\mapsto z])\backslash\{z\}\}=\{v^-_y[\sigma]\mid v\in\text{FN}(Q)\backslash\{y\}\}$. Here I use that $u^+_z\neq z$ and $(u^+_z)^-_z=u$ for all names u.

Since $\text{FN}(P)=\{x\}\cup\{v^-_y\mid v\in\text{FN}(Q)\backslash\{y\}\}$, it follows that $\text{FN}(P\sigma)=\{x[\sigma]\mid x\in\text{FN}(P)\}$.

Now let $P=(\nu y)Q$. Then $P\sigma=(\nu z)(Q\{z/y\}\sigma)$, so $\text{FN}(P\sigma)=\text{FN}(Q\{z/y\}\sigma)\backslash\{z\}$. By induction (applied twice) $\text{FN}(Q\{z/y\}\sigma)=\{v[\{z/y\}][\sigma]\mid v\in\text{FN}(Q)\}$. Considering that $z\notin\text{FN}(Q)\cup dom(\sigma)\cup range(\sigma)$,

$$\text{FN}(Q\{z/y\}\sigma)\backslash\{z\}=\{v[\sigma]\mid v\in\text{FN}(Q)\backslash\{y\}\}.$$

Since $\text{FN}(P)=\text{FN}(Q)\backslash\{y\}$, again the claim follows. ∎

For recursion equations $A(\vec{x})\stackrel{\text{def}}{=}P$ the π-calculus requires that $\text{fn}(P)\subseteq\{x_1,\ldots,x_n\}$, and I didn't bother to change that for $\pi^{\mathcal{S}}$. Since I require that $x_1,\ldots,x_n\in\mathcal{N}$, Lemma 43 implies that also $\text{FN}(P)\subseteq\{x_1,\ldots,x_n\}$. Using this, (4) immediately follows from (8), which is Lemma 44.

Lemma 45: $\text{FN}(P\{z/y\}^{\mathcal{S}})\subseteq\text{FN}(x(y).P)\cup\{z\}$.

Proof. If $v\in\text{FN}(P\{z/y\}^{\mathcal{S}})$ then $v=x[\{z/y\}^{\mathcal{S}}]$ for some $x\in\text{FN}(P)$, using Lemma 44. If moreover $v\neq z$ then $v=x^-_y$. Hence $v\in\text{FN}(P)^-_y\subseteq\text{FN}(x(y).P)$. ∎

Now I define $\equiv^{\mathcal{S}}$ in such a way that (5) holds.

Definition 21: For $y,z\in\mathcal{N}$ with $y\neq z$, let $\{z/y\}^3$ be the substitution σ with $dom(\sigma)=\{y,z,s_1\}$, $\sigma(y)=z$, $\sigma(z)=s_1$ and $\sigma(s_1)=y$. Moreover, let $\{y/y\}^3:=\epsilon$, the substitution with $dom(\epsilon)=\emptyset$. Let $\equiv^{\mathcal{S}}$ be the smallest congruence satisfying

(i) $(\nu y)P\equiv^{\mathcal{S}}(\nu z)(P\{z/y\})$ for any $z\notin\text{FN}((\nu y)P)$,
(ii) and $x(y).P\equiv^{\mathcal{S}}x(z).(P\{z/y\}^3)$ for any $z\in\mathcal{N}$.

Note that in $x(y).P$ the spare name s_1 occurring in P will be elevated to y, but in $x(z).Q$ it will be elevated to z. Therefore, when renaming the bound name y into z, the spare name s_1 should be renamed into y right away. Moreover, there is no reason to require that z not occur free in P; its free occurrences can simply be renamed into s_1.[12]

Lemma 46: $P\equiv^{\mathcal{S}}Q\Rightarrow\text{FN}(P)=\text{FN}(Q)$.

Proof. Using transitivity of $\equiv^{\mathcal{S}}$, it suffices to prove this for the special case that P and Q differ by only one application of the generating equations from Definition 21. Below I deal with the special case that this single application does not occur within a proper subterm of P; the general case then follows by a straightforward structural induction on P. There are two possibilities to consider.

Let $P=x(y).R$ and $Q=x(z).(R\{z/y\}^3)$. In this case $\text{FN}(P)=\{x\}\cup\text{FN}(R)^-_y$ and $\text{FN}(Q)=\{x\}\cup\text{FN}(R\{z/y\}^3)^-_z$. Moreover, $\text{FN}(R)^-_y=\{v^-_y\mid v\in\text{FN}(R)\backslash\{y\}\}$ and, by Lemma 44,

$$\text{FN}(R\{z/y\}^3)^-_z=\{v[\{z/y\}^3]^-_z\mid v\in\text{FN}(R)\wedge v[\{z/y\}^3]\neq z\}.$$

[12] An alternative form of Definition 21 requires (ii) only for $z\notin\text{FN}((\nu y)P)$. This yields the same equivalence $\equiv^{\mathcal{S}}$ as Definition 21, since even when $z\in\text{FN}((\nu y)P)$ one derives $x(y).P\equiv^{\mathcal{S}}x(w).(P\{w/y\}^3)\equiv^{\mathcal{S}}x(z).(P\{z/y\}^3)$ for some $w\notin\text{FN}((\nu y)P)$, using (19), that $P\{z/y\}^3\{w/z\}^3\equiv^{\mathcal{S}}P\{w/y\}^3$.

Note that $v = y$ iff $v[\{z/y\}^3] = z$. So take $v \neq y$. A simple case distinction shows that $v[\{z/y\}^3]^-_z = v^-_y$. Consequently, $\text{FN}(R)^-_y = \text{FN}(R\{z/y\}^3)^-_z$ and thus $\text{FN}(P) = \text{FN}(Q)$.

Let $P{=}(\nu y).R$ and $Q{=}(\nu z).(R\{z/y\})$ with $z{\notin}\text{FN}((\nu y)P)$. Then $\text{FN}(P) = \text{FN}(R)\backslash\{y\}$ and $\text{FN}(Q) = \text{FN}(R\{z/y\})\backslash\{z\}$. By Lemma 44, $\text{FN}(R\{z/y\}) = \{v[\{z/y\}] \mid v \in \text{FN}(R)\}$. When $v \neq y, z$ one has $v = v[\{z/y\}] \neq z$, and when $v[\{z/y\}] \neq z$ then $v \neq y, z$. Consequently, $\text{FN}(P) = \text{FN}(Q)$. ∎

The next two lemmas pave the way for Lemma 49, which establishes the validity of Properties (11) and (12).

Lemma 47: Let $y, w, z \in \mathcal{N}$, Then, for all $x \in \mathcal{H}$,

$$x[\sigma_1[y \mapsto w]][\sigma_2[w \mapsto z]] = x[(\sigma_2 \circ \sigma_1)[y \mapsto z]].$$

Proof. By Definition 20, $x[\sigma_1[y \mapsto w]] = w$ only if $x = y$. So

$$x[\sigma_1[y \mapsto w]][\sigma_2[w \mapsto z]] = \begin{cases} z & \text{if } x = y \\ ((x^-_y[\sigma_1])^+_w)^-_w[\sigma_2]^+_z & \text{otherwise.} \end{cases}$$

Moreover,

$$x[(\sigma_2 \circ \sigma_1)[y \mapsto z]] = \begin{cases} z & \text{if } x = y \\ (x^-_y[\sigma_1][\sigma_2])^+_z & \text{otherwise.} \end{cases}$$

Since $(v^+_w)^-_w = v$ for all $v \in \mathcal{H}$, the lemma follows. ∎

Lemma 48: Let $y, w, z \in \mathcal{N}$. Then, for all $x \in \mathcal{H}$,

$$x[\sigma[y \mapsto w]][\{z/w\}^3] = x[\sigma[y \mapsto z]]$$

$$x[\{w/y\}^3][\sigma[w \mapsto z]] = x[\sigma[y \mapsto z]].$$

Proof. If $x{=}y$ then $x[\sigma[y \mapsto w]][\{z/w\}^3] = z = x[\sigma[y \mapsto z]]$. Otherwise, considering that $v^+_w[\{z/w\}^3] = v^+_z$ for any $v \in \mathcal{H}$, $x[\sigma[y \mapsto w]][\{z/w\}^3] = (x^-_y[\sigma])^+_w[\{z/w\}^3] = (x^-_y[\sigma])^+_z = x[\sigma[y \mapsto z]]$. The second statement follows likewise, using that $x[\{w/y\}^3]^-_w = x^-_y$ when $x \neq y$, ∎

Lemma 49:

1) $(P\sigma_1)\sigma_2 \equiv^{\mathcal{S}} P(\sigma_2 \circ \sigma_1)$.
2) $(\forall x \in \text{FN}(P).\, x[\sigma] = x[\sigma']) \Rightarrow P\sigma \equiv^{\mathcal{S}} P\sigma'$.

Proof. I prove both statements by simultaneous structural induction on P.

1) All cases are trivial, except for the ones where P has the form $x(y).Q$ or $(\nu y)Q$.
   Let $P = x(y).Q$. By the definition of substitution, there is a u such that the last step in the below derivation holds. Likewise, there are w and z such that the first two steps hold.

$$\begin{aligned}(P\sigma_1)\sigma_2 &= \big(x[\sigma_1](w).(Q\sigma_1[y \mapsto w])\big)\sigma_2 \\ &= x[\sigma_1][\sigma_2](z).\big((Q\sigma_1[y \mapsto w])\sigma_2[w \mapsto z]\big) \\ &\equiv^{\mathcal{S}} x[\sigma_2 \circ \sigma_1](z).\big(Q(\sigma_2[y \mapsto w] \circ \sigma_1[w \mapsto z])\big) \\ &\equiv^{\mathcal{S}} x[\sigma_2 \circ \sigma_1](z).\big(Q((\sigma_2 \circ \sigma_1)[y \mapsto z])\big) \\ &\equiv^{\mathcal{S}} x[\sigma_2 \circ \sigma_1](u).\big(Q((\sigma_2 \circ \sigma_1)[y \mapsto z]\{u/z\}^3)\big) \\ &\equiv^{\mathcal{S}} x[\sigma_2 \circ \sigma_1](u).\big(Q((\sigma_2 \circ \sigma_1)[y \mapsto u])\big) \\ &= P(\sigma_2 \circ \sigma_1).\end{aligned}$$

   Here the third step is an application of the induction hypotheses regarding statement 1) and the fourth step applies the induction hypotheses regarding statement 2), also using Lemma 47. The fifth step applies Definition 21 of $\equiv^{\mathcal{S}}$. The sixth step applies the induction hypotheses regarding both statements, also using Lemma 48.
   Let $P = (\nu y)Q$. By the definition of substitution, there is an u such that the last step in the below derivation holds. Likewise, there are w and z such that the first two steps hold. Now choose a name v outside $\text{FN}\big(Q\{w/y\}\sigma_1\{z/w\}\sigma_2\big) \cup \text{FN}\big(Q\{u/y\}(\sigma_2 \circ \sigma_1)\big) \cup \{z, u\}$.

$$\begin{aligned}(P\sigma_1)\sigma_2 &= ((\nu w)(Q\{w/y\}\sigma_1))\sigma_2 \\ &= (\nu z)(Q\{w/y\}\sigma_1\{z/w\}\sigma_2) \\ &\equiv^{\mathcal{S}} (\nu v)(Q\{w/y\}\sigma_1\{z/w\}\sigma_2\{v/z\}) \\ &\equiv^{\mathcal{S}} (\nu v)(Q\{u/y\}(\sigma_2 \circ \sigma_1)\{v/u\}) \\ &\equiv^{\mathcal{S}} (\nu u)(Q\{u/y\}(\sigma_2 \circ \sigma_1)) \\ &= P(\sigma_2 \circ \sigma_1).\end{aligned}$$

   The third and fifth step apply Definition 21 of $\equiv^{\mathcal{S}}$. The fourth applies the induction hypotheses regarding both statements, also using that, for all $x \in \text{FN}(Q)$,

$$x[\{w/y\}][\sigma_1][\{z/w\}][\sigma_2][\{v/z\}] = x[\{u/y\}][\sigma_2 \circ \sigma_1][\{v/u\}].$$

   The latter follows by a straightforward case distinction, considering that $w \notin \text{FN}(Q)\backslash\{y\} \cup dom(\sigma_1) \cup range(\sigma_1)$, that $z \notin \text{FN}(Q\{w/y\}\sigma_1)\backslash\{w\} \cup dom(\sigma_2) \cup range(\sigma_2)$ and that $u \notin \text{FN}((\nu y)Q) \cup dom(\sigma_2 \circ \sigma_1) \cup range(\sigma_2 \circ \sigma_1)$.
2) Again all cases are trivial, except for the ones that P has the form $x(y).Q$ or $(\nu y)Q$.
   Let $P = x(y).Q$ and $u[\sigma] = u[\sigma']$ for all $u \in \text{FN}(P)$. Then, for certain $z, w \in \mathcal{N}$, $P\sigma = x[\sigma](z).(Q\sigma[y \mapsto z])$ and $P\sigma' = x[\sigma'](w).(Q\sigma'[y \mapsto w])$. By Definition 21, $P\sigma' \equiv^{\mathcal{S}} x[\sigma'](z).(Q\sigma'[y \mapsto w]\{z/w\}^3)$. By the induction hypotheses and Lemma 48, $P\sigma' \equiv^{\mathcal{S}} x[\sigma'](z).(Q\sigma'[y \mapsto z])$. By Lemma 35, $x \in \text{FN}(P)$, and thus $x[\sigma] = x[\sigma']$. To round of the proof by another appeal to the induction hypotheses regarding statement 2), it suffices to show that $u[\sigma[y \mapsto z]] = u[\sigma'[y \mapsto z]]$ for all $u \in \text{FN}(Q)$. This amounts to showing that $(u^-_y[\sigma])^+_z = (u^-_y[\sigma'])^+_z$ for all $u \in \text{FN}(Q)\backslash\{y\}$. Now for each $u \in \text{FN}(Q)\backslash\{y\}$ one has $u^-_y \in \text{FN}(P)$, by the definition of $\text{FN}(Q)^-_y$. Hence $u^-_y[\sigma] = u^-_y[\sigma']$, and the proof of this case is done.
   Let $P = (\nu y)Q$ and let $u[\sigma] = u[\sigma']$ for all $u \in \text{FN}(P)$. Then $P\sigma = (\nu z)(Q\{z/y\}\sigma)$ and $P\sigma' = (\nu w)(Q\{w/y\}\sigma')$ for certain names z, w. Pick $v \notin \text{FN}(P\sigma) \cup \{z, w\} \cup dom(\sigma) \cup dom(\sigma')$. Then, by Definition 21, $P\sigma \equiv^{\mathcal{S}} (\nu v)(Q\{z/y\}\sigma\{v/z\})$ and $P\sigma' \equiv^{\mathcal{S}} (\nu v)(Q\{w/y\}\sigma'\{v/w\})$. Now $u[\{z/y\}][\sigma][\{v/z\}] = u[\{v/y\}][\sigma]$ for all $u \in \text{FN}(Q)$, using that $z{\notin}\text{FN}(P){\cup}dom(\sigma){\cup}range(\sigma)$ and $v{\notin}dom(\sigma)$. Hence, by induction, $Q\{z/y\}\sigma\{v/z\} \equiv^{\mathcal{S}} Q\{v/y\}\sigma$. Likewise, $Q\{w/y\}\sigma'\{v/w\} \equiv^{\mathcal{S}} Q\{v/y\}\sigma'$. By Lemma 44 $\text{FN}(Q\{v/y\})\backslash\{v\} \subseteq \text{FN}(P)$, so, again by induction, $Q\{v/y\}\sigma \equiv^{\mathcal{S}} Q\{v/y\}\sigma'$, and hence $P\sigma \equiv^{\mathcal{S}} P\sigma'$. ∎

Property (10) follows immediately from Definition 21, and (7) follows from (11) (i.e. Lemma 49.1) and (18). Also (9) follows from Lemma 49, since $\text{FN}(P) \subseteq \{x_1, \ldots, x_n\}$. Hence it remains to verify Properties (13)–(17).

Lemma 50: $P \equiv^{\mathcal{S}} Q \Rightarrow P\sigma \equiv^{\mathcal{S}} Q\sigma$.

Proof. Using transitivity of $\equiv^{\mathcal{S}}$, it suffices to prove this for the special case that P and Q differ by only one application of the generating equations from Definition 21. Below I deal with the special case that this single application does not occur within a proper subterm of P; the general case then follows by a structural induction on the size of the parse tree of P, recalling that this size is preserved under substitution.

There are two possibilities to consider.

Let $P = x(y).R$ and $Q = x(z).(R\{z/y\}^3)$. In this case $P\sigma{=}x[\sigma](w).(R\sigma[y{\mapsto}w])$, $Q\sigma{=}x[\sigma](u).((R\{z/y\}^3)\sigma[z{\mapsto}u])$. By Definition 21 $Q\sigma \equiv^{\mathcal{S}} x[\sigma](w).(R\{z/y\}^3\sigma[z \mapsto u]\{w/u\}^3)$, so it suffices to show that

$$R\sigma[y \mapsto w] \equiv^{\mathcal{S}} R\{z/y\}^3\sigma[z \mapsto u]\{w/u\}^3,$$

which follows from Lemmas 48 and 49.

Let $P = (\nu y)R$ and $Q = (\nu z)(R\{z/y\})$ with $z \notin \text{FN}(R)$. Then $P\sigma = (\nu w)(R\{w/y\}\sigma)$, $Q\sigma = (\nu u)(R\{z/y\}\{u/z\}\sigma)$ for names w, u with $w, u \notin \text{FN}(P) \cup dom(\sigma) \cup range(\sigma)$ (using that $\text{FN}(P) = \text{FN}(Q)$, by Lemma 46). Choose a fresh name $v \notin \text{FN}(R\{w/y\}\sigma) \cup \text{FN}(R\{z/y\}\{u/z\}\sigma) \cup \{w, u\}$. Then $P\sigma \equiv^{\mathcal{S}} (\nu v)(R\{w/y\}\sigma\{v/w\})$ and $Q\sigma \equiv^{\mathcal{S}} (\nu v)(R\{z/y\}\{u/z\}\sigma\{v/u\})$. It suffices to show that $R\{w/y\}\sigma\{v/w\} \equiv^{\mathcal{S}} R\{z/y\}\{u/z\}\sigma\{v/u\}$. This follows from Lemma 49, provided that, for all $x \in \text{FN}(R)$,

$$x[\{w/y\}][\sigma][\{v/w\}] = x[\{z/y\}][\{u/z\}][\sigma][\{v/u\}].$$

The latter is established by a simple case distinction. ∎

Lemma 51: Let $y \in \mathcal{N}$ and $w \notin \text{FN}(x(y).P)$. Then $P\{w/y\}^{\mathcal{S}}\{z/w\} \equiv^{\mathcal{S}} P\{z/y\}^{\mathcal{S}}$.

Proof. Since $w \notin \text{FN}(P)^-_y$, one has $w^+_y \notin \text{FN}(P)$. Now the statement follows from Lemma 49, considering that, for all $x \in \text{FN}(P)$, $x[\{w/y\}^{\mathcal{S}}][\{z/w\}] = x[\{z/y\}^{\mathcal{S}}]$. ∎

Property (15) follows by a straightforward structural induction on P, using Lemma 49 for the cases that $P = x(y).Q$ or $P = (\nu y)Q$.

Using Lemma 49 it is trivial to check, for $y, w, z \in \mathcal{N}$, that

$$P\{w/y\}^3\{z/w\}^3 \equiv^{\mathcal{S}} P\{z/y\}^3 \tag{19}$$

$$P\{w/y\}^3\{u/w\}^{\mathcal{S}} \equiv^{\mathcal{S}} P\{u/y\}^{\mathcal{S}}. \tag{20}$$

Lemma 52: If $U \equiv^{\mathcal{S}} x(y).P$ then $U = x(w).V$ for some $V \equiv^{\mathcal{S}} P\{w/y\}^3$. Likewise, if $U \equiv^{\mathcal{S}} (\nu y)P$ then $U = (\nu w)V$ for some $w \notin \text{FN}(U)$ and $V \equiv^{\mathcal{S}} P\{w/y\}$.

Proof. Since $U \equiv^{\mathcal{S}} x(y).P$, one has $U = U_0 \equiv^{\mathcal{S}} U_1 \equiv^{\mathcal{S}} U_2 \equiv^{\mathcal{S}} \cdots \equiv^{\mathcal{S}} U_n = x(y).P$, and each step $U_i \equiv^{\mathcal{S}} U_{i+1}$ involves exactly one application of the generating equations from Definition 21. I prove the statement by induction on n.

The case that $n = 0$ is trivial; take $w := y$.

Let $n > 0$. By induction $U_1 = x(w).V$ with $V \equiv^{\mathcal{S}} P\{w/y\}^3$.

If for $U = U_0 \equiv^{\mathcal{S}} U_1 = x(w).V$ the application of the generating equation from Definition 21 occurs entirely within V, one has $U = x(w).W$ with $W \equiv^{\mathcal{S}} V \equiv^{\mathcal{S}} P\{w/y\}^3$.

If the application is at top level, then $U = x(z).W$ and either $W = V\{z/w\}^3$ or $V = W\{w/z\}^3$. By (19), (15), in either case $W \equiv^{\mathcal{S}} V\{z/w\}^3$. Thus $W \equiv^{\mathcal{S}} P\{z/y\}^3$ by (19).

The second statement is obtained in the same way. ∎

The last property to be checked, (16), follows from Lemma 52.

Corollary 2: If $U \equiv^{\mathcal{S}} (x(y).P)\sigma$ then $U = x[\sigma](w).V$ with $V \equiv^{\mathcal{S}} P\sigma[y \mapsto w]$.

Proof. By definition $(x(y).P)\sigma = x[\sigma](z).(P\sigma[y \mapsto z])$ for some $z \in \mathcal{N}$. By Lemma 52 $U = x(w).V$ for some $V \equiv^{\mathcal{S}} P\sigma[y \mapsto z]\{w/z\}^3$. By Lemmas 48 and 49 $V \equiv^{\mathcal{S}} P\sigma[y \mapsto w]$. ∎

Since the axioms (4)–(17) of Section A hold for $\pi^{\mathcal{S}}(\mathcal{N})$, so do Lemmas 35–42 from Section A.

As mentioned earlier, the classical π-calculus $\pi(\mathcal{N})$ can be seen as a subcalculus of the π-calculus with surjective substitutions $\pi^{\mathcal{S}}(\mathcal{N})$; its processes P satisfy $\text{fn}(P) \subseteq \mathcal{N}$. Lemma 43 shows that on this subcalculus one has $\text{fn}(P) = \text{FN}(P)$, so that FN can be seen as an extension of fn from $\pi(\mathcal{N})$ to all of $\pi^{\mathcal{S}}(\mathcal{N})$.

Likewise, the application $P\sigma$ of a substitution σ to a process P is unchanged up to $\equiv$ when $\text{fn}(P) \subseteq \mathcal{N}$ and $\sigma : \mathcal{H} \rightharpoonup \mathcal{N}$. Namely, if $\text{fn}(Q) \subseteq \mathcal{N}$, $w \notin \text{FN}((\nu y)Q) \cup dom(\sigma) \cup range(\sigma)$ and $\sigma : \mathcal{H} \rightharpoonup \mathcal{N}$, then $Q\sigma[y \mapsto w] \equiv Q\{w/y\}\sigma$.

The next lemma shows that also $\equiv^{\mathcal{S}}$ and $\equiv$ coincide on the subcalculus $\pi(\mathcal{N})$ of $\pi^{\mathcal{S}}(\mathcal{N})$.

Lemma 53: Let $\text{fn}(P|Q) \subseteq \mathcal{N}$. Then $P \equiv^{\mathcal{S}} Q \Leftrightarrow P \equiv Q$.

Proof. "⇒": It suffices to consider the case that P and Q differ by only one application of the generating equations from Definition 21. Below I deal with the special case that this single application does not occur within a proper subterm of P; the general case then follows by structural induction on P.

The case that $P = (\nu y)R$ and $Q = (\nu z)(R\{z/y\})$ with $z \notin \text{FN}((\nu y)R)$ is trivial.

So let $P = x(y).R$ and $Q = x(z).(R\{z/y\}^3)$. As $\text{fn}(P) \subseteq \mathcal{N}$ and $\text{fn}(Q) \subseteq \mathcal{N}$, also $\text{fn}(R) \subseteq \mathcal{N}$ and $\text{fn}(R\{z/y\}^3) \subseteq \mathcal{N}$, so $z \notin \text{fn}((\nu y)R)$. Considering that $v[\{z/y\}^3] = v[\{z/y\}]$ for all $v \in \text{FN}(R)$, and that the classical notion of substitution may be applied here, $R\{z/y\}^3 \equiv R\{z/y\}$. Hence $P \equiv Q$.

"⇐": It suffices to consider the case that P and Q differ by only one application of the generating equations from Section V. Below I deal with the special case that this single application does not occur within a proper subterm of P; the general case then follows by structural induction on P.

The case that $P = (\nu y)R$ and $Q = (\nu z)(R\{z/y\})$ with $z \notin \text{fn}((\nu y)R)$ is trivial.

So let $P = x(y).R$ and $Q = x(z).(R\{z/y\})$ with $z \notin \text{fn}((\nu y)R)$. In that case $\text{FN}(R) = \text{fn}(R) \subseteq \mathcal{N}$. Considering that $v[\{z/y\}] = v[\{z/y\}^3]$ for all $v \in \text{FN}(R)$, Lemma 49.2 yields $R\{z/y\} \equiv^{\mathcal{S}} R\{z/y\}^3$. Hence $P \equiv^{\mathcal{S}} Q$. ∎

Now I will show that the identity $\text{id} : \text{T}_{\pi_{ES}(\mathcal{N})} \to \text{T}_{\pi^{\mathcal{S}}_{ES}(\mathcal{N})}$ is a valid translation from $\pi_{ES}(\mathcal{N})$ to $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$. My definition of $R \xrightarrow{\alpha}_{ES} R'$ implies that $R, R' \in \text{T}_{\pi_{ES}(\mathcal{N})}$ and $\text{n}(\alpha) \subseteq \mathcal{N}$.

Lemma 54: If $R \xrightarrow{\alpha}_{ES} R'$ then equally $R \xrightarrow{\alpha}^{\mathcal{S}}_{ES} U'$ for some $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$ process U' with $U' \equiv^{\mathcal{S}} R'$.

Proof. With induction on the inference of $R \xrightarrow{\alpha}_{ES} R'$.

- Suppose $R \xrightarrow{\alpha}_{ES} R'$ is derived by rule **EARLY-INPUT**. Then $R = x(y).P$, $\alpha = xz$ and $R' = P\{z/y\}$. Using **EARLY-INPUT** from $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$, $R \xrightarrow{\alpha}^{\mathcal{S}}_{ES} U' := P\{z/y\}^{\mathcal{S}}$. By Lemma 43 $\text{FN}(P) \subseteq \mathcal{N}$. So by (12) $U' \equiv^{\mathcal{S}} R'$.
- Suppose $R \xrightarrow{\alpha}_{ES} R'$ is derived by **IDE**. Then $R = A(\vec{y})$. Let $A(\vec{x}) \stackrel{\text{def}}{=} P$. Then $P\{\vec{y}/\vec{x}\} \xrightarrow{\alpha}_{ES} R'$. So by induction $P\{\vec{y}/\vec{x}\} \xrightarrow{\alpha}^{\mathcal{S}}_{ES} U'$ for some $U' \equiv^{\mathcal{S}} R'$. As $\text{fn}(P) \subseteq \mathcal{N}$, $P\{\vec{y}/\vec{x}\} \equiv^{\mathcal{S}} P\{\vec{y}/\vec{x}\}^{\mathcal{S}}$ by (12) and Lemma 43. Hence $P\{\vec{y}/\vec{x}\}^{\mathcal{S}} \xrightarrow{\alpha}^{\mathcal{S}}_{ES} U'$ by **ALPHA**. Thus $R \xrightarrow{\alpha}^{\mathcal{S}}_{ES} U'$ by rule **IDE** from $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$.
- All other cases are trivial. ∎

Lemma 55: If $R \in \text{T}_{\pi_{ES}(\mathcal{N})}$ and $R \equiv^{\mathcal{S}} U \xrightarrow{\alpha}^{\mathcal{S}}_{ES} U'$ with $\iota(\alpha) \cup \text{bn}(\alpha) \subseteq \mathcal{N}$, then equally $R \xrightarrow{\alpha}_{ES} R'$ for some $R' \equiv^{\mathcal{S}} U'$.

Proof. With induction on the α-depth of the inference of $U \xrightarrow{\alpha}^{\mathcal{S}}_{ES} U'$, with a nested induction on the number of applications of rule **ALPHA** at the end of that inference.

- Suppose $U \xrightarrow{\alpha}^{\mathcal{S}}_{ES} U'$ is derived by rule **EARLY-INPUT**. Then $R = x(y).P$ and $U = x(w).V$ with $V \equiv^{\mathcal{S}} P\{w/y\}^3$ by Lemma 52. Moreover, $\alpha = xz$ and $y, w, z \in \mathcal{N}$. So $U' = V\{z/w\}^{\mathcal{S}} \equiv^{\mathcal{S}} P\{w/y\}^3\{z/w\}^{\mathcal{S}} \equiv^{\mathcal{S}} P\{z/y\}^{\mathcal{S}}$ by (20). By **EARLY-INPUT** from $\pi_{ES}(\mathcal{N})$, $R \xrightarrow{\alpha}_{ES} R' := P\{z/y\}$. By Lemma 43 $\text{FN}(P) \subseteq \mathcal{N}$. By (12) $R' \equiv^{\mathcal{S}} P\{z/y\}^{\mathcal{S}} \equiv^{\mathcal{S}} U'$.
- Suppose $U \xrightarrow{\alpha}^{\mathcal{S}}_{ES} U'$ is derived by rule **IDE**. Then $R = U = A(\vec{y})$. Let $A(\vec{x}) \stackrel{\text{def}}{=} P$. Then $P\{\vec{y}/\vec{x}\}^{\mathcal{S}} \xrightarrow{\alpha}^{\mathcal{S}}_{ES} U'$. As $P \in T_{\pi_{ES}(\mathcal{N})}$, and $\{y_1, \ldots, y_n\} = \text{fn}(R) \subseteq \mathcal{N}$, also $P\{\vec{y}/\vec{x}\} \in T_{\pi_{ES}(\mathcal{N})}$. Using (12), $P\{\vec{y}/\vec{x}\} \equiv^{\mathcal{S}} P\{\vec{y}/\vec{x}\}^{\mathcal{S}}$. So by induction $P\{\vec{y}/\vec{x}\} \xrightarrow{\alpha}_{ES} R'$ for some $R' \equiv^{\mathcal{S}} U'$. Consequently, $R \xrightarrow{\alpha}_{ES} R'$ by rule **IDE** from $\pi_{ES}(\mathcal{N})$.
- Suppose $U \xrightarrow{\alpha}^{\mathcal{S}}_{ES} U'$ is derived by **E-S-CLOSE**. Then $R = P|Q$, $U = V|W$, $P \equiv^{\mathcal{S}} V \xrightarrow{M\bar{x}(z)}^{\mathcal{S}}_{ES} V'$, $Q \equiv^{\mathcal{S}} W \xrightarrow{Nvz}^{\mathcal{S}}_{ES} W'$, $\alpha = [x{=}v]MN\tau$ and $U' = (\nu z)(V'|W')$ for some $z \notin \text{FN}(Q)$. Pick $w \in \mathcal{N} \backslash \text{FN}(R)$. Now $V \xrightarrow{M\bar{x}(w)}^{\mathcal{S}}_{ES} \equiv^{\mathcal{S}} V'\{w/z\}$ and $W \xrightarrow{Nvw}^{\mathcal{S}}_{ES} \equiv^{\mathcal{S}} W'\{w/z\}$ by Lemmas 40 and 39. Moreover, the inferences of these transitions have a smaller α-depth than that of $R \xrightarrow{\alpha}^{\mathcal{S}}_{ES} U'$. So $P \xrightarrow{M\bar{x}(w)}_{ES} P' \equiv^{\mathcal{S}} V'\{w/z\}$ and $Q \xrightarrow{Nvw}_{ES} Q' \equiv^{\mathcal{S}} W'\{w/z\}$ by induction. Applying rule **E-S-CLOSE**, $R \xrightarrow{\alpha}_{ES} R' := (\nu w)(P'|Q')$. Lemma 36 yields $\text{FN}(R') \subseteq \text{FN}(R) \not\ni w$. Using this, one obtains $R' \equiv^{\mathcal{S}} (\nu w)(V'\{w/z\}|W'\{w/z\}) \equiv^{\mathcal{S}} (\nu z)(V'|W') = U'$.
- All other cases are trivial. In the case of **E-S-COM** apply Lemma 35 to obtain $y \in \mathcal{N}$. ∎

Theorem 10: $P \mathrel{\bullet\!\!\sim} \text{id}(P)$ for any $P \in \text{T}_{\pi_{ES}(\mathcal{N})}$.

Proof. It suffices to show that the symmetric closure of

$$\mathscr{R} := \{(R, U) \mid R \in \text{T}_{\pi_{ES}(\mathcal{N})}, U \in \text{T}_{\pi^{\mathcal{S}}_{ES}(\mathcal{N})} \wedge R \equiv^{\mathcal{S}} U\}$$

is a strong barbed bisimulation. Let $(R, U) \in \mathscr{R}$.

Let $R \xrightarrow{\tau}_{ES} R'$. By Lemma 54, $R \xrightarrow{\tau}^{\mathcal{S}}_{ES} U'$ for some $U' \equiv^{\mathcal{S}} R'$. So $U \xrightarrow{\tau}^{\mathcal{S}}_{ES} U'$ by rule **ALPHA** of $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$.

Let $U \xrightarrow{\tau}^{\mathcal{S}}_{ES} U'$. By Lemma 55, $\exists R'.\ R \xrightarrow{\tau}_{ES} R' \equiv^{\mathcal{S}} U'$.

Let $U{\downarrow_b}$ with $b \in \mathcal{Z} \cup \overline{\mathcal{Z}}$. Then $U \xrightarrow{by}^{\mathcal{S}}_{ES} U'$ or $U \xrightarrow{b(y)}^{\mathcal{S}}_{ES} U'$ for some y and U', using the definition of O in Section IV. By Lemmas 38 and 40 I may assume, without loss of generality, that $y \in \mathcal{N}$ in case of input or bound output actions. So $R \xrightarrow{by}_{ES} R'$ or $R \xrightarrow{b(y)}_{ES} R'$ by Lemma 55. Thus $R{\downarrow_b}$.

The implication $R{\downarrow_b} \Rightarrow U{\downarrow_b}$ proceeds likewise. ∎

C. The elimination of ALPHA

Let $\xrightarrow{\alpha}^{\mathcal{S}}_{\bullet}$ be the transition relation on T_π generated by the rules of Table VIII. Here $\{z/\!\!/y\}$ denotes the substitution σ with $dom(\sigma) = \{y, z\}$, $\sigma(y) = z$ and $\sigma(z) = y$.[13] Compared to the operational semantics of $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$, rule **ALPHA** is omitted, but applications of this rule are incorporated in rules **RES-ALPHA** and **SYMB-OPEN-ALPHA**.[14] Lemmas 56 and 57 below say that up to strong bisimilarity the transitions relations $\xrightarrow{\alpha}^{\mathcal{S}}_{ES}$ and $\xrightarrow{\alpha}^{\mathcal{S}}_{\bullet}$ of $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$ are equivalent.

Lemma 56: If $R \xrightarrow{\alpha}^{\mathcal{S}}_{\bullet} R'$ then equally $R \xrightarrow{\alpha}^{\mathcal{S}}_{ES} \equiv^{\mathcal{S}} R'$.

Proof. With induction on the inference of $R \xrightarrow{\alpha}^{\mathcal{S}}_{\bullet} R'$. The only nontrivial cases are when $R \xrightarrow{\alpha}^{\mathcal{S}}_{\bullet} R'$ is derived by rule **RES-ALPHA** or **SYMB-OPEN-ALPHA**.

By (12), $P\{z/\!\!/y\} \equiv^{\mathcal{S}} P\{z/y\}$ when $z \notin \text{FN}((\nu y)P)$. Hence $(\nu y)P \equiv^{\mathcal{S}} (\nu z)(P\{z/y\}) \equiv^{\mathcal{S}} (\nu z)(P\{z/\!\!/y\})$. Using this, each application of rule **RES-ALPHA** can be mimicked by an application of **RES** followed by one of **ALPHA**.

Now suppose $R \xrightarrow{\alpha}^{\mathcal{S}}_{\bullet} R'$ is derived by **SYMB-OPEN-ALPHA**. Then $R = (\nu y)P$, $\alpha = M\bar{x}(z)$, $y \neq x$, $z \notin \text{FN}(R)$, $y \notin \text{n}(M)$, $P \xrightarrow{M\bar{x}y}^{\mathcal{S}}_{\bullet} P'$ and $R' = P'\{z/\!\!/y\}$. By induction $P \xrightarrow{M\bar{x}y}^{\mathcal{S}}_{ES} \equiv^{\mathcal{S}} P'$. By Lemma 35 $\text{n}(M) \cup \{x\} \subseteq \text{FN}(P) \backslash \{y\} = \text{FN}(R) \not\ni z$. So by Lemma 37 $P\{z/\!\!/y\} \xrightarrow{M\bar{x}z}^{\mathcal{S}}_{ES} \equiv^{\mathcal{S}} P'\{z/\!\!/y\}$. By **SYMB-OPEN** $(\nu z)(P\{z/\!\!/y\}) \xrightarrow{M\bar{x}(z)}^{\mathcal{S}}_{ES} \equiv^{\mathcal{S}} P'\{z/\!\!/y\} = R'$. So by rule **ALPHA** $R \xrightarrow{\alpha}^{\mathcal{S}}_{ES} \equiv^{\mathcal{S}} R'$. ∎

Lemma 57: If $R \equiv^{\mathcal{S}} U$ and $R \xrightarrow{\alpha}^{\mathcal{S}}_{ES} R'$ then equally $U \xrightarrow{\alpha}^{\mathcal{S}}_{\bullet} U'$ for some U' with $R' \equiv^{\mathcal{S}} U'$.

Proof. With induction on the α-depth of the inference of $R \xrightarrow{\alpha}^{\mathcal{S}}_{ES} R'$, with a nested induction on the number of applications of rule **ALPHA** at the end of the inference.

- The cases that $R \xrightarrow{\alpha}^{\mathcal{S}}_{ES} R'$ is derived by rule **TAU** or **OUTPUT** are trivial.
- Suppose $R \xrightarrow{\alpha}^{\mathcal{S}}_{ES} R'$ is derived by **EARLY-INPUT**. Then $R = x(y).P$, $\alpha = xz$ and $R' = P\{z/y\}^{\mathcal{S}}$. By Lemma 52 $U = x(w).V$ for some $V \equiv^{\mathcal{S}} P\{w/y\}^3$. Consequently, $U \xrightarrow{\alpha}^{\mathcal{S}}_{\bullet} V\{z/w\}^{\mathcal{S}}$. Using (13) and (20) $V\{z/w\}^{\mathcal{S}} \equiv^{\mathcal{S}} P\{w/y\}^3\{z/w\}^{\mathcal{S}} \equiv^{\mathcal{S}} P\{z/y\}^{\mathcal{S}} = R'$.

[13] In rule **RES-ALPHA**, thanks to the side condition $z \notin \text{FN}((\nu y)P)$, using (12), $P\{z/\!\!/y\} \equiv^{\mathcal{S}} P\{z/y\}$. For this reason, the versions with $\{z/\!\!/y\}$ and $\{z/y\}$ are equivalent, up to strong bisimilarity. The use of $\{z/\!\!/y\}$ instead of $\{z/y\}$ makes the substitution surjective, which will be needed in Section D.

The same can be said about **SYMB-OPEN-ALPHA**, since $\text{FN}((\nu y)P') \subseteq \text{FN}((\nu y)P)$ by Lemma 36.

[14] Besides the insignificant difference of $\{z/\!\!/y\}$ versus $\{z/y\}$, rule **SYMB-OPEN-ALPHA** differs from **SYMB-ALPHA-OPEN** in the more restrictive side condition $z \notin \text{FN}((\nu y)P)$ versus $z \notin \text{FN}((\nu y)P')$. Whereas the two rules are interchangeable up to strong barbed bisimilarity, they are not up to strong bisimilarity, for the transition $(\nu y)(\bar{x}y.\mathbf{0} + \bar{z}x.\mathbf{0}) \xrightarrow{\bar{x}(\bar{z})} \mathbf{0}$ is not derivable from Table VIII. I need **SYMB-OPEN-ALPHA** to obtain a transition relation that is strongly bisimilar with the one of $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$.

TABLE VIII
EARLY SYMBOLIC STRUCTURAL OPERATIONAL SEMANTICS OF THE π-CALCULUS WITHOUT RULE **ALPHA**

TAU: $M\tau.P \xrightarrow{M\tau} P$

OUTPUT: $M\bar{x}y.P \xrightarrow{M\bar{x}y} P$

EARLY-INPUT: $Mx(y).P \xrightarrow{Mxz} P\{z/y\}^{\mathcal{S}}$

SUM: $$\frac{P \xrightarrow{\alpha} P'}{P+Q \xrightarrow{\alpha} P'}$$

SYMB-MATCH: $$\frac{P \xrightarrow{\alpha} P'}{[x=y]P \xrightarrow{[x=y]\alpha} P'}$$

IDE: $$\frac{P\{\vec{y}/\vec{x}\}^{\mathcal{S}} \xrightarrow{\alpha} P'}{A(\vec{y}) \xrightarrow{\alpha} P'} \quad (A(\vec{x}) \stackrel{\text{def}}{=} P)$$

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

E-S-COM: $$\frac{P \xrightarrow{M\bar{x}y} P',\ Q \xrightarrow{Nvy} Q'}{P|Q \xrightarrow{[x=v]MN\tau} P'|Q'}$$

E-S-CLOSE: $$\frac{P \xrightarrow{M\bar{x}(z)} P',\ Q \xrightarrow{Nvz} Q'}{P|Q \xrightarrow{[x=v]MN\tau} (\nu z)(P'|Q')} \quad (z \notin \mathrm{FN}(Q))$$

RES-ALPHA: $$\frac{P\{z/\!\!/y\} \xrightarrow{\alpha} P'}{(\nu y)P \xrightarrow{\alpha} (\nu z)P'} \quad \begin{pmatrix} z \notin \mathrm{FN}((\nu y)P) \\ z \notin \mathrm{n}(\alpha) \end{pmatrix}$$

SYMB-OPEN-ALPHA: $$\frac{P \xrightarrow{M\bar{x}y} P'}{(\nu y)P \xrightarrow{M\bar{x}(z)} P'\{z/\!\!/y\}} \quad \begin{pmatrix} y \neq x \\ z \notin \mathrm{FN}((\nu y)P) \\ y \notin \mathrm{n}(M) \end{pmatrix}$$

- Suppose $R \xrightarrow{\alpha}{}^{\mathcal{S}}_{ES} R'$ is derived by **IDE**. Then $R = A(\vec{y})$, so $U = R$. Let $A(\vec{x}) \stackrel{\text{def}}{=} P$. Then $P\{\vec{y}/\vec{x}\}^{\mathcal{S}} \xrightarrow{\alpha}{}^{\mathcal{S}}_{ES} R'$. By induction $P\{\vec{y}/\vec{x}\}^{\mathcal{S}} \xrightarrow{\alpha}{}^{\mathcal{S}}_{\bullet} U'$ for some U' with $R' \equiv^{\mathcal{S}} U'$. By **IDE** $U \xrightarrow{\alpha}{}^{\mathcal{S}}_{\bullet} U'$.
- Suppose $R \xrightarrow{\alpha}{}^{\mathcal{S}}_{ES} R'$ is derived by **PAR**. Then $R = P|Q$, $P \xrightarrow{\alpha}{}^{\mathcal{S}}_{ES} P'$, $R' = P'|Q$ and $\mathrm{bn}(\alpha) \cap \mathrm{FN}(Q) = \emptyset$. Thus $U = V|W$ with $P \equiv^{\mathcal{S}} V$, $Q \equiv^{\mathcal{S}} W$ and $\mathrm{bn}(\alpha) \cap \mathrm{FN}(W) = \emptyset$. By induction $V \xrightarrow{\alpha}{}^{\mathcal{S}}_{\bullet} V'$ for some V' with $P' \equiv^{\mathcal{S}} V'$. So $U = V|W \xrightarrow{\alpha}{}^{\mathcal{S}}_{\bullet} V'|W$ by rule **PAR**, and $R' \equiv^{\mathcal{S}} V'|W$.
- The cases that $R \xrightarrow{\alpha}{}^{\mathcal{S}}_{\bullet} R'$ is derived by rule **E-S-COM**, **E-S-CLOSE**, **SUM** or **SYMB-MATCH** are (also) trivial.
- Suppose $R \xrightarrow{\alpha}{}^{\mathcal{S}}_{ES} R'$ is derived by rule **RES**. Then $R = (\nu z)P$, $P \xrightarrow{\alpha}{}^{\mathcal{S}}_{ES} P'$, $z \notin \mathrm{n}(\alpha)$ and $R' = (\nu z)P'$. So $U = (\nu y)V$ for some $y \notin \mathrm{FN}((\nu z)P)$ and $V \equiv^{\mathcal{S}} P\{y/z\}$, using Lemma 52. Employing (13), (15) and Lemma 49, $V\{z/\!\!/y\} \equiv^{\mathcal{S}} P\{y/z\}\{z/\!\!/y\} \equiv^{\mathcal{S}} P\{z/\!\!/z\} \equiv^{\mathcal{S}} P$. So by induction $V\{z/\!\!/y\} \xrightarrow{\alpha}{}^{\mathcal{S}}_{\bullet} V'$ for some $V' \equiv^{\mathcal{S}} P'$. By (5) $z \notin \mathrm{FN}((\nu z)P) = \mathrm{FN}((\nu y)(P\{y/z\})) = \mathrm{FN}((\nu y)V)$. Hence $U = (\nu y)V \xrightarrow{\alpha}{}^{\mathcal{S}}_{\bullet} (\nu z)V'$ by rule **RES-ALPHA**. Moreover, $R' = (\nu z)P' \equiv^{\mathcal{S}} (\nu z)V'$.
- Suppose $R \xrightarrow{M\bar{x}(z)}{}^{\mathcal{S}}_{ES} R'$ is derived by **SYMB-OPEN**. Then $R = (\nu z)P$, $P \xrightarrow{M\bar{x}z}{}^{\mathcal{S}}_{ES} R'$, $z \neq x$ and $z \notin \mathrm{n}(M)$. Thus $U = (\nu y)V$ for $y \notin \mathrm{FN}((\nu z)P)$ and $V \equiv^{\mathcal{S}} P\{y/z\}$, using Lemma 52. Now $P\{y/z\} \xrightarrow{M\bar{x}y}{}^{\mathcal{S}}_{ES} R'\{y/z\}$ by Lemma 37. Moreover, the α-depth of the inference of $P\{y/z\} \xrightarrow{M\bar{x}y}{}^{\mathcal{S}}_{ES} R'\{y/z\}$ equals that of $P \xrightarrow{M\bar{x}z}{}^{\mathcal{S}}_{ES} R'$, which is smaller than that of $R \xrightarrow{M\bar{x}(z)}{}^{\mathcal{S}}_{ES} R'$. By induction $V \xrightarrow{M\bar{x}y}{}^{\mathcal{S}}_{\bullet} V'$ for some $V' \equiv^{\mathcal{S}} R'\{y/z\}$. By (5), $z \notin \mathrm{FN}(R) = \mathrm{FN}(U)$. Employing Lemma 35, $\mathrm{n}(M) \cup \{x\} \subseteq \mathrm{FN}(P) \backslash \{z\} = \mathrm{FN}(R) \not\ni y$. Consequently, $U = (\nu y)V \xrightarrow{M\bar{x}(z)}{}^{\mathcal{S}}_{\bullet} V'\{z/\!\!/y\}$ by **SYMB-OPEN-ALPHA**. Moreover, $V'\{z/\!\!/y\} \equiv^{\mathcal{S}} R'\{y/z\}\{z/\!\!/y\} \equiv^{\mathcal{S}} R'$ by (11)–(13) and (15), using that $y \notin \mathrm{FN}(R) \supseteq \mathrm{FN}(R') \backslash \{z\}$.
- Suppose $R \xrightarrow{\alpha}{}^{\mathcal{S}}_{ES} R'$ is derived by **ALPHA**. Then there is a $Q \equiv^{\mathcal{S}} R$ such that $Q \xrightarrow{\alpha}{}^{\mathcal{S}}_{ES} R'$ is derived by a simpler proof. So $U \equiv^{\mathcal{S}} Q$ and by induction $U \xrightarrow{\alpha}{}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} R'$. ■

Combining Lemmas 56 and 57 yields the following result, stating that rule **ALPHA** is not needed on top of the rules of Table VIII.

Corollary 3: If $R \equiv^{\mathcal{S}} U$ and $R \xrightarrow{\alpha}{}^{\mathcal{S}}_{\bullet} R'$ then equally $U \xrightarrow{\alpha}{}^{\mathcal{S}}_{\bullet} U'$ for some U' with $R' \equiv^{\mathcal{S}} U'$.

Together, Lemmas 56 and 57 imply that the relation $\equiv^{\mathcal{S}}$ is a strong (barbed) bisimulation between $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$ expressions equipped with the semantics of Table VI, amended as described in Sections A and B, and $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$ expressions equipped with the semantics of Table VIII. Hence, up to $\rightsquigarrow$ it doesn't matter whether which of these semantics one employs.

Although I will not need this in this paper, a trivial adaptation of the above proofs shows that also the early and early symbolic semantics of the π-calculus, displayed in Tables IV and VI can be equivalently adapted by dropping rule **ALPHA** at the cost of strengthening **RES** and **OPEN** (or **SYMB-OPEN**) into **RES-ALPHA** and **OPEN-ALPHA** (or **SYMB-OPEN-ALPHA**).

D. Replacing substitution by relabelling

Let $\pi^{\rho\bullet}_{ES}(\mathcal{N})$ be the variant of the π-calculus with surjective substitutions $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$, enriched with a postfix-written relabelling operator $[\sigma]$ for each surjective substitution σ. Its operational semantics is given by the rules of Table VIII, together with the rule **RELABELLING**

$$\frac{P \xrightarrow{\alpha} P'}{P[\sigma] \xrightarrow{\alpha[\sigma]} P'[\sigma]} \quad (\mathrm{bn}(\alpha[\sigma]) \cap \mathrm{FN}(P[\sigma]) = \emptyset)$$

for the relabelling operators, except that the substitutions $\{z/y\}^{\mathcal{S}}$ and $\{\vec{y}/\vec{x}\}^{\mathcal{S}}$ that appear in rules **EARLY-INPUT** and **IDE** are replaced by applications of the relabelling operators $[\{z/y\}^{\mathcal{S}}]$ and $[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]$, respectively, and the substitution $\{z/\!\!/y\}$ that appears in rules **RES-ALPHA** and **SYMB-OPEN-ALPHA** is replaced by the relabelling operator $[\{z/\!\!/y\}]$. Moreover, rules **PAR** and **E-S-CLOSE** gain side conditions $\mathrm{bn}(\alpha) \cap \mathrm{FN}(P) = \emptyset$ and $z \notin \mathrm{FN}(P)$, respectively.[15] Except for **IDE**, $\pi^{\rho\bullet}_{ES}(\mathcal{N})$ has the

[15]These side conditions are not essential—they could be left out here, or added to the original π-calculus, without ill effects—but are included to obtain a stronger and simpler version of Lemmas 67 and 71 in Sections F and G.

semantics of Table IX. The function FN, used in several rules, is extended to $\pi^{\rho\bullet}_{ES}(\mathcal{N})$ by $\text{FN}(P[\sigma]) := \{x[\sigma] \mid x \in \text{FN}(P)\}$. Recall that names are chosen from $\mathcal{N} \uplus \mathcal{S}$, except that in $x(y).P$ one always has $y \in \mathcal{N}$. Moreover, in defining equations $A(\vec{x}) \stackrel{\text{def}}{=} P$ I require that $x_1, \ldots, x_n \in \mathcal{N}$, and $\text{fn}(P) \subseteq \{x_1, \ldots, x_n\}$.[16]

I will show that the identity $\text{id} : \text{T}_{\pi^{\mathcal{S}}_{ES}(\mathcal{N})} \to \text{T}_{\pi^{\rho\bullet}_{ES}(\mathcal{N})}$ is a valid translation from $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$ to $\pi^{\rho\bullet}_{ES}(\mathcal{N})$. As justified in Section C, at the side of $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$ I will use the transition relation $\longrightarrow^{\mathcal{S}}_{\bullet}$ generated by Table VIII. Lemmas 56 and 57 imply that Lemmas 39–42 also hold for $\longrightarrow^{\mathcal{S}}_{\bullet}$. I will denote the transition relation of $\pi^{\rho\bullet}_{ES}(\mathcal{N})$ simply by $\stackrel{\alpha}{\longrightarrow}$.

For P a $\pi^{\rho\bullet}_{ES}(\mathcal{N})$ process, let $\widehat{P}$ be the $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$ process obtained from P by recursively replacing each subterm $Q[\sigma]$ by $Q\sigma$, and each agent identifier A by $A\hat{}$. Here $A\hat{}$ is a fresh agent identifier with defining equation $A\hat{}(\vec{x}) \stackrel{\text{def}}{=} \widehat{P}$ when $A(\vec{x}) \stackrel{\text{def}}{=} P$ was the defining equation of A. Note that $\text{FN}(\widehat{P}) = \text{FN}(P)$ for each $P \in \text{T}_{\pi^{\rho\bullet}_{ES}(\mathcal{N})}$.

Lemma 58: If $R \stackrel{\beta}{\longrightarrow} R'$ then equally $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R'}$.

Proof. By induction of the inference of $R \stackrel{\beta}{\longrightarrow} R'$.

- Suppose $R \stackrel{\beta}{\longrightarrow} R'$ is derived by **TAU**. Then $R = \tau.P$, $\beta = \tau$ and $R' = P$. Moreover, $\widehat{R} = \tau.\widehat{P}$ and $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \widehat{R'}$.
- The case that $R \stackrel{\beta}{\longrightarrow} R'$ stems from **OUTPUT** goes likewise.
- Suppose $R \stackrel{\beta}{\longrightarrow} R'$ is derived by **EARLY INPUT**. Then $R = x(y).P$, $\beta = xz$ and $R' = P[\{z/y\}^{\mathcal{S}}]$. Moreover, $\widehat{R} = x(y).\widehat{P}$ and $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \widehat{P}\{z/y\}^{\mathcal{S}} = \widehat{P[\{z/y\}^{\mathcal{S}}]} = \widehat{R'}$.
- Suppose $R \stackrel{\beta}{\longrightarrow} R'$ is derived by **SUM**. Then $R = P + Q$ and $P \stackrel{\beta}{\longrightarrow} R'$. Now $\widehat{R} = \widehat{P} + \widehat{Q}$. So $\widehat{P} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R'}$ by induction. Hence, by **SUM**, $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R'}$.
- Suppose $R \stackrel{\beta}{\longrightarrow} R'$ is derived by **SYMB-MATCH**. Then $R = [x{=}y]P$, $P \stackrel{\alpha}{\longrightarrow} R'$ and $\beta = [x{=}y]\alpha$. Now $\widehat{R} = [x{=}y]\widehat{P}$. By induction $\widehat{P} \stackrel{\alpha}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R'}$. By **SYMB-MATCH**, $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R'}$.
- Suppose $R \stackrel{\beta}{\longrightarrow} R'$ is derived by **IDE**. Then $R = A(\vec{y})$ with $A(\vec{x}) \stackrel{\text{def}}{=} P$ and $P[\{\vec{y}/\vec{x}\}^{\mathcal{S}}] \stackrel{\beta}{\longrightarrow} R'$. Now $\widehat{R} = A\hat{}(\vec{y})$ with $A\hat{}(\vec{x}) \stackrel{\text{def}}{=} \widehat{P}$. Moreover, $\widehat{P[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]} = \widehat{P}\{\vec{y}/\vec{x}\}^{\mathcal{S}}$. By induction $\widehat{P}\{\vec{y}/\vec{x}\}^{\mathcal{S}} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R'}$. By **IDE**, $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R'}$.
- Suppose $R \stackrel{\beta}{\longrightarrow} R'$ is derived by **PAR**. Then $R = P|Q$, $P \stackrel{\beta}{\longrightarrow} P'$, $\text{bn}(\beta) \cap \text{FN}(Q) = \emptyset$ and $R' = P'|Q$. Now $\widehat{R}{=}\widehat{P}|\widehat{Q}$, $\text{bn}(\beta){\cap}\text{FN}(\widehat{Q}) = \emptyset$ and $\widehat{R'}{=}\widehat{P'}|\widehat{Q}$. By induction $\widehat{P} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{P'}$. Thus $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R'}$, by **PAR**.
- Suppose $R \stackrel{\beta}{\longrightarrow} R'$ is derived by **E-S-COM**. Then $R = P|Q$, $P \stackrel{M\bar{x}y}{\longrightarrow} P'$, $Q \stackrel{Nvy}{\longrightarrow} Q'$, $R' = P'|Q'$, and $\beta = [x{=}v]MN\tau$. Now $\widehat{R} = \widehat{P}|\widehat{Q}$ and $\widehat{R'} = \widehat{P'}|\widehat{Q'}$. By induction $\widehat{P} \stackrel{M\bar{x}y}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{P'}$ and $\widehat{Q} \stackrel{Nvy}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{Q'}$. Thus $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R'}$, by **E-S-COM**.
- Suppose $R \stackrel{\beta}{\longrightarrow} R'$ is derived by **E-S-CLOSE**. Then $R = P|Q$, $P \stackrel{M\bar{x}(z)}{\longrightarrow} P'$, $Q \stackrel{Nvz}{\longrightarrow} Q'$, $R' = (\nu z)(P'|Q')$, $z \notin \text{FN}(Q)$ and $\beta = [x{=}v]MN\tau$. Now $\widehat{R} = \widehat{P}|\widehat{Q}$, $z \notin \text{FN}(\widehat{Q})$ and $\widehat{R'} = (\nu z)(\widehat{P'}|\widehat{Q'})$. By induction $\widehat{P} \stackrel{M\bar{x}(z)}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{P'}$ and $\widehat{Q} \stackrel{Nvz}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{Q'}$. Thus $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R'}$, by **E-S-CLOSE**.

[16] Optionally, one could furthermore require that no relabelling operators may occur in defining equations; this would simplify the definition of $\widehat{P}$.

- Suppose $R \stackrel{\beta}{\longrightarrow} R'$ is derived by **RES-ALPHA**. Then $R = (\nu y)P$, $P[\{z/\!\!/y\}] \stackrel{\beta}{\longrightarrow} P'$, $R' = (\nu z)P'$, $z \notin \text{FN}(R)$ and $z \notin \text{n}(\beta)$. Now $\widehat{R} = (\nu y)\widehat{P}$, $z \notin \text{FN}(\widehat{R})$ and $\widehat{R'} = (\nu z)\widehat{P'}$. By induction $\widehat{P}\{z/\!\!/y\} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{P'}$. Hence, $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R'}$.
- Suppose $R \stackrel{\beta}{\longrightarrow} R'$ is derived by **SYMB-OPEN-ALPHA**. Then $R{=}(\nu y)P$, $\beta{=}M\bar{x}(z)$, $P \stackrel{M\bar{x}y}{\longrightarrow} P'$, $R'{=}P'[\{z/\!\!/y\}]$, $y \notin x$, $z \notin \text{FN}(R)$ and $y \notin \text{n}(M)$. Now $\widehat{R} = (\nu y)\widehat{P}$, $z \notin \text{FN}(\widehat{R})$ and $\widehat{R'} = \widehat{P'}\{z/\!\!/y\}$. By induction $\widehat{P} \stackrel{M\bar{x}y}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{P'}$. Hence, $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R'}$ by **SYMB-OPEN-ALPHA**.
- Suppose $R \stackrel{\beta}{\longrightarrow} R'$ is derived by **RELABELLING**. Then $R = P[\sigma]$, $P \stackrel{\alpha}{\longrightarrow} P'$, $R' = P'[\sigma]$, $\beta = \alpha[\sigma]$ and $\text{bn}(\beta) \cap \text{FN}(R){=}\emptyset$. By induction $\widehat{P} \stackrel{\alpha}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{P'}$. As $\text{FN}(R){=}\text{FN}(\widehat{R}) = \text{FN}(\widehat{P}\sigma)$, by Lemma 41 $\widehat{R} = \widehat{P}\sigma \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{P'}\sigma = \widehat{R'}$. ■

Lemma 59: If $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} U$ with $\text{bn}(\beta) \cap \text{FN}(R) = \emptyset$ then equally $R \stackrel{\beta}{\longrightarrow} R'$ for some R' with $\widehat{R'} \equiv^{\mathcal{S}} U$.

Proof. By induction on the depth of the inference of $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} U$, with a nested induction on the number of topmost relabelling operators in R.

- First suppose $R = P[\sigma]$. Then $\widehat{R} = \widehat{P}\sigma$. By Lemma 42 $\widehat{P} \stackrel{\alpha}{\longrightarrow}^{\mathcal{S}}_{\bullet} V$ for some α and V with $\alpha[\sigma]{=}\beta$ and $V\sigma{\equiv^{\mathcal{S}}}U$. Moreover, the depth of the inference of $\widehat{P} \stackrel{\alpha}{\longrightarrow}^{\mathcal{S}}_{\bullet} V$ is the same as that of $\widehat{P}\sigma \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} U$. As $\text{bn}(\alpha[\sigma]){\cap}\text{FN}(P[\sigma]) = \emptyset$, also $\text{bn}(\alpha) \cap \text{FN}(P) = \emptyset$. So by induction $P \stackrel{\alpha}{\longrightarrow} P'$ for some P' with $\widehat{P'} \equiv^{\mathcal{S}} V$. By **RELABELLING** $R \stackrel{\beta}{\longrightarrow} P'[\sigma]$. Furthermore one has $\widehat{P'[\sigma]} = \widehat{P'}\sigma \equiv^{\mathcal{S}} V\sigma \equiv^{\mathcal{S}} U$.
  Henceforth I suppose that R is not of the form $P[\sigma]$.
- Suppose $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} U$ is derived by rule **E-S-COM**. Then $R = P|Q$, $\widehat{R} = \widehat{P}|\widehat{Q}$, $\beta = [x{=}v]MN\tau$, $\widehat{P} \stackrel{M\bar{x}y}{\longrightarrow}^{\mathcal{S}}_{\bullet} V$, $\widehat{Q} \stackrel{Nvy}{\longrightarrow}^{\mathcal{S}}_{\bullet} W$ and $U = V|W$. By induction $P \stackrel{M\bar{x}y}{\longrightarrow} P'$ and $Q \stackrel{Nvy}{\longrightarrow} Q'$ for some P' and Q' with $\widehat{P'} \equiv^{\mathcal{S}} V$ and $\widehat{Q'} \equiv^{\mathcal{S}} W$. By **E-S-COM** $R \stackrel{\beta}{\longrightarrow} P'|Q' \equiv^{\mathcal{S}} V|W = U$.
- Suppose $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} U$ is derived by rule **RES-ALPHA**. Then $R = (\nu y)P$, $\widehat{R} = (\nu y)\widehat{P}$, $\widehat{P}\{z/\!\!/y\} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} V$, $z \notin \text{FN}(\widehat{R})$, $z \notin \text{n}(\beta)$ and $U = (\nu z)V$. As $\text{bn}(\beta) \cap \text{FN}((\nu y)P) = \emptyset$ and $z{\notin}\text{n}(\beta){\cup}\text{FN}((\nu y)P)$, also $\text{bn}(\beta){\cap}\text{FN}(P[\{z/\!\!/y\}]){=}\emptyset$. So by induction $P[\{z/\!\!/y\}] \stackrel{\beta}{\longrightarrow} P'$ for some P' such that $\widehat{P'} \equiv^{\mathcal{S}} V$. Now $R \stackrel{\beta}{\longrightarrow} (\nu z)P'$ by **RES-ALPHA**. Moreover, $\widehat{(\nu z)P'} = (\nu z)\widehat{P'} \equiv^{\mathcal{S}} (\nu z)V = U$.
- The cases that $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} U$ is derived by **TAU**, **OUTPUT**, **EARLY-INPUT**, **SUM**, **SYMB-MATCH**, **IDE**, **PAR** or **SYMB-OPEN-ALPHA** are also trivial.
- Suppose $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} U$ is derived by rule **E-S-CLOSE**. Then $R = P|Q$, $\widehat{R} = \widehat{P}|\widehat{Q}$, $\beta = [x{=}v]MN\tau$, $\widehat{P} \stackrel{M\bar{x}(z)}{\longrightarrow}^{\mathcal{S}}_{\bullet} V$, $\widehat{Q} \stackrel{Nvz}{\longrightarrow}^{\mathcal{S}}_{\bullet} W$, $z \notin \text{FN}(\widehat{Q})$ and $U = (\nu z)(V|W)$. Pick $w \notin \text{FN}(\widehat{R}) \cup \text{FN}(U)$. Then $\widehat{P} \stackrel{M\bar{x}(w)}{\longrightarrow}^{\mathcal{S}}_{\bullet} V' \equiv^{\mathcal{S}} V\{w/z\}$ and $\widehat{Q} \stackrel{Nvw}{\longrightarrow}^{\mathcal{S}}_{\bullet} W' \equiv^{\mathcal{S}} W\{w/z\}$ by Lemmas 40 and 39. Moreover, the inferences of these transitions have a smaller depth than that of $\widehat{R} \stackrel{\beta}{\longrightarrow}^{\mathcal{S}}_{\bullet} U$. So by induction $P \stackrel{M\bar{x}(w)}{\longrightarrow} P'$ and $Q \stackrel{Nvw}{\longrightarrow} Q'$ for some P' and Q' with $\widehat{P'} \equiv^{\mathcal{S}} V'$ and $\widehat{Q'} \equiv^{\mathcal{S}} W'$. Hence $R \stackrel{\beta}{\longrightarrow} (\nu w)(P'|Q')$ by **E-S-CLOSE**. Moreover,

$$\widehat{(\nu w)(P'|Q')} = (\nu w)(\widehat{P'}|\widehat{Q'}) \equiv^{\mathcal{S}} (\nu w)(V'|W') \equiv^{\mathcal{S}} (\nu w)((V|W)\{w/z\}) \equiv^{\mathcal{S}} (\nu z)(V|W) = U.$$

■

Let $\mathscr{T}_\rho$ be the identity translation from $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$ to $\pi^{\rho\bullet}_{ES}(\mathcal{N})$.

Theorem 11: $\mathscr{T}_\rho(P) \,\dot{\sim}\, P$ for any $P \in \mathrm{T}_{\pi^{\mathcal{S}}_{ES}(\mathcal{N})}$.

Proof. It suffices to show that the symmetric closure of

$$\mathscr{R} := \{(R,U) \mid R \in \mathrm{T}_{\pi^{\rho\bullet}_{ES}(\mathcal{N})}, U \in \mathrm{T}_{\pi^{\mathcal{S}}_{ES}(\mathcal{N})} \wedge \widehat{R} \equiv^{\mathcal{S}} U\}$$

is a strong barbed bisimulation, as it contains $(\mathscr{T}_\rho(P), P)$.

Let $(R,U) \in \mathscr{R}$. Let $R \xrightarrow{\tau} R'$. Then $\widehat{R} \xrightarrow{\tau}{}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R}'$ by Lemma 58 and $U \xrightarrow{\tau}{}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R}'$ by Corollary 3. So $U \xrightarrow{\tau}{}^{\mathcal{S}}_{\bullet} U'$ for some U' with $(R',U') \in \mathscr{R}$.

Let $U \xrightarrow{\tau}{}^{\mathcal{S}}_{\bullet} U'$. Then $\widehat{R} \xrightarrow{\tau}{}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} U'$ by Corollary 3. By Lemma 59 $R \xrightarrow{\tau} R'$ for some R' with $\widehat{R}' \equiv^{\mathcal{S}} U'$.

Let $U{\downarrow_b}$ with $b \in \mathcal{Z} \cup \overline{\mathcal{Z}}$. Then $U \xrightarrow{by}{}^{\mathcal{S}}_{\bullet} U'$ or $U \xrightarrow{b(y)}{}^{\mathcal{S}}_{\bullet} U'$ for some y and U', using the definition of O in Section IV. Hence $\widehat{R} \xrightarrow{by}{}^{\mathcal{S}}_{\bullet}$ or $\widehat{R} \xrightarrow{b(y)}{}^{\mathcal{S}}_{\bullet}$ by Corollary 3. In the second case, by Lemma 40 I may assume, without loss of generality, that $y \notin \mathrm{FN}(\widehat{R})$. So $R \xrightarrow{by}{}^{\mathcal{S}}_{ES}$ or $R \xrightarrow{b(y)}{}^{\mathcal{S}}_{ES}$ by Lemma 59. Thus $R{\downarrow_b}$.

The implication $R{\downarrow_b} \Rightarrow U{\downarrow_b}$ proceeds likewise. ∎

The following example shows why the side condition $\mathrm{bn}(\alpha[\sigma]) \cap \mathrm{FN}(P[\sigma]) = \emptyset$ in rule **RELABELLING** is necessary.

Example 11: Let $P = \bar{x}z.x(w).\bar{w}q \mid x(y).(\nu v)\bar{x}v.y(r)$. In the π-calculus, or in $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$, this process can do two successive τ-steps in a row, but not three:

$$P \xrightarrow{\tau} x(w).\bar{w}q \mid (\nu v)\bar{x}v.z(r) \xrightarrow{\tau} (\nu u)(\bar{u}q \mid z(r)) \not\xrightarrow{\tau} .$$

In the above expression the bound name u may be chosen freely from $\mathcal{N}\backslash\{x,z,q\}$. The side condition of rule **SYMB-OPEN-ALPHA** prevents the choices $u = x, z$, since x and z are free in $(\nu v)\bar{x}(v).z(r)$. The side condition of (the symmetric counterpart of) rule **E-S-CLOSE** prevents the choices $u = x, q$.

In a version of $\pi^{\rho\bullet}_{ES}(\mathcal{N})$ in which rule **RELABELLING** does not have the side condition $\mathrm{bn}(\alpha[\sigma]) \cap \mathrm{FN}(P[\sigma]) = \emptyset$, and **E-S-CLOSE** lacks the side condition $z \in \mathrm{FN}(P)$, one obtains $(\nu v)\bar{x}v.y(r) \xrightarrow{\bar{x}(z)} y(r)[z/\!\!/v]$ and thus

$$\big((\nu v)\bar{x}v.y(r)\big)[z/y] \xrightarrow{\bar{x}(z)} y(r)[z/\!\!/v][z/y] \xrightarrow{zq} \mathbf{0}[z/\!\!/v][z/y][q/r]$$

which yields

$$\begin{array}{rcl} P & \xrightarrow{\tau} & x(w).\bar{w}q \mid \big((\nu v)\bar{x}v.y(r)\big)[z/y] \\ & \xrightarrow{\tau} & (\nu z)(\bar{w}q[z/w] \mid y(r)[z/\!\!/v][z/y]) \\ & \xrightarrow{\tau} & (\nu z)(\mathbf{0}[z/w] \mid \mathbf{0}[z/\!\!/v][z/y][q/r]), \end{array}$$

in violation of Theorem 11.

Although $(\nu v)\bar{x}v.y(r) \xrightarrow{\bar{x}(z)}{}^{\mathcal{S}}_{\bullet} y(r)\{z/\!\!/v\}$, one does not have $\big((\nu v)\bar{x}v.y(r)\big)\{z/y\} \xrightarrow{\bar{x}(z)}{}^{\mathcal{S}}_{\bullet} y(r)\{z/\!\!/v\}\{z/y\}$. This shows that the necessity of the side condition $\mathrm{bn}(\alpha[\sigma]) \cap \mathrm{FN}(R\sigma) = \emptyset$ in Lemma 37/41. Hence $\big((\nu v)\bar{x}v.y(r)\big)[z/y] \xrightarrow{\bar{x}(z)} y(r)[z/\!\!/v][z/y]$ violates Lemma 58. Since rule **SYMB-OPEN-ALPHA** is invoked *before* the relabelling $[z/y]$ is applied—in contrast with $\pi^{\mathcal{S}}_{ES}(\mathcal{N})$, where it is invoked after applying the substitution $\{z/y\}$—the choice $u = z$ cannot be avoided. Here the side condition of **RELABELLING** comes to the rescue; it rules out the offending transition because $z \in \mathrm{bn}(\bar{x}(z)) \cap \mathrm{FN}(\big((\nu v)\bar{x}v.y(r)\big)[z/y])$.

E. α-conversion for $\pi^{\rho\bullet}_{ES}(\mathcal{N})$

For $\pi^{\rho\bullet}_{ES}(\mathcal{N})$-processes P and Q write $P \equiv Q$ iff $\widehat{P} \equiv^{\mathcal{S}} \widehat{Q}$. If $P \equiv Q$ then $\mathrm{FN}(P) = \mathrm{FN}(\widehat{P}) = \mathrm{FN}(\widehat{Q}) = \mathrm{FN}(Q)$.

Lemma 60: If $P \equiv Q$ and $P \xrightarrow{\alpha} P'$ with $\mathrm{bn}(\alpha) \cap \mathrm{FN}(Q) = \emptyset$ then equally $Q \xrightarrow{\alpha} Q'$ for some Q' with $P' \equiv Q'$.

Proof. Suppose $P \equiv Q$ and $P \xrightarrow{\alpha} P'$ with $\mathrm{bn}(\alpha) \cap \mathrm{FN}(P) = \emptyset$. Then $\widehat{P} \equiv^{\mathcal{S}} \widehat{Q}$, and $\widehat{P} \xrightarrow{\alpha}{}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{P}'$ by Lemma 58. Therefore, $\widehat{Q} \xrightarrow{\alpha}{}^{\mathcal{S}}_{\bullet} U$ for some $U \equiv^{\mathcal{S}} \widehat{P}'$ by Corollary 3. By Lemma 59 $Q \xrightarrow{\alpha} Q'$ for some Q' with $\widehat{Q}' \equiv^{\mathcal{S}} U$. Now $P' \equiv Q'$. ∎

Lemma 61:

(a) $(\forall x \in \mathrm{FN}(P).\, x[\sigma] = x[\sigma']) \Rightarrow P[\sigma] \equiv P[\sigma']$,
(b) $P[\sigma_1][\sigma_2] \equiv P[\sigma_2 \circ \sigma_1]$,
(c) $P[\epsilon] \equiv P$, where ϵ is the empty substitution,
(d) $(P|Q)[\sigma] \equiv P[\sigma] \mid Q[\sigma]$,
(e) $((\nu y)P)[\sigma] \equiv (\nu y)(P[\sigma])$, when $y \notin dom(\sigma) \cup range(\sigma)$,
(f) $(\nu y)P \equiv (\nu z)(P[\{z/\!\!/y\}])$, when $z \notin \mathrm{FN}((\nu y)P)$,
(g) if $P_1 \equiv Q_1$ and $P_2 \equiv Q_2$ then $P_1|P_2 \equiv Q_1|Q_2$, and
(h) if $P \equiv Q$ then $(\nu y)P \equiv (\nu y)Q$ and $P[\sigma] \equiv Q[\sigma]$.

Proof. (a), (b) and (c) follow immediately from (12), (11) and (15), (d) and (e) from the definition of substitution, (g) and (h) from the congruence property of $\equiv^{\mathcal{S}}$ and (13), and (f) follows from (10), using (a) and (h) to replace $\{z/y\}$ by $\{z/\!\!/y\}$. ∎

The following five lemmas are counterparts for $\pi^{\rho\bullet}_{ES}(\mathcal{N})$ of Lemmas 35, 36, 40, 39 and 38 for $\pi_{ES}(\mathcal{N})$, adapted to avoid non-surjective substitutions $\{w/\!\!/z\}$.

Lemma 62: If $P \xrightarrow{\alpha} Q$ then $\mathrm{n}(\alpha)\backslash(\iota(\alpha) \cup \mathrm{bn}(\alpha)) \subseteq \mathrm{FN}(P)$.

Proof. Immediately from Lemmas 58, 56 and 35. ∎

Lemma 63: If $P \xrightarrow{\alpha} Q$ then $\mathrm{FN}(Q) \subseteq \mathrm{FN}(P) \cup \iota(\alpha) \cup \mathrm{bn}(\alpha)$.

Proof. Immediately from Lemmas 58, 56 and 36. ∎

Lemma 64: If $R \xrightarrow{M\bar{x}(z)} R_z$ and $w \notin \mathrm{FN}(R)$ then equally $R \xrightarrow{M\bar{x}(w)} \equiv R_z[\{w/\!\!/z\}]$.

Proof. Let $R \xrightarrow{M\bar{x}(z)} R_z$. Then $\widehat{R} \xrightarrow{M\bar{x}(z)}{}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R}_z$ by Lemma 58. So $\widehat{R} \xrightarrow{M\bar{x}(z)}{}^{\mathcal{S}}_{ES} U \equiv^{\mathcal{S}} \widehat{R}_z$ by Lemma 56. Using that $w \notin \mathrm{FN}(R) = \mathrm{FN}(\widehat{R})$, by Lemma 40 $\widehat{R} \xrightarrow{M\bar{x}(w)}{}^{\mathcal{S}}_{ES} V$ for a $V \equiv^{\mathcal{S}} U\{w/z\}$. By Lemma 36, $\mathrm{FN}(U)\backslash\{z\} \subseteq \mathrm{FN}(\widehat{R}) \not\ni w$, and therefore $U\{w/\!\!/z\} \equiv^{\mathcal{S}} U\{w/z\}$ by (12). By Lemma 57 $\widehat{R} \xrightarrow{M\bar{x}(w)}{}^{\mathcal{S}}_{\bullet} W$ for a $W \equiv^{\mathcal{S}} V$. By Lemma 59, using that $w \notin \mathrm{FN}(R)$, $R \xrightarrow{M\bar{x}(w)} R'$ for an R' with $\widehat{R}' \equiv^{\mathcal{S}} W$. Using (13) $\widehat{R}' \equiv^{\mathcal{S}} \widehat{R}_z\{w/\!\!/z\} = \widehat{R_z[\{w/\!\!/z\}]}$, and thus $R' \equiv R_z[\{w/\!\!/z\}]$. ∎

Lemma 65: If $R \xrightarrow{Mxz} R_z$ and $z, w \notin \mathrm{FN}(R)$, then equally $R \xrightarrow{Mxw} R_w$ for some R_w with $R_w \equiv R_z[\{w/\!\!/z\}]$.

Proof. The proof is the same as the one of Lemma 64, but using Lemma 39 instead of Lemma 40, thereby also using the precondition $z \notin \mathrm{FN}(R)$. ∎

Lemma 66: If $R \xrightarrow{Mxz} R_z$ and w is a name, then equally $R \xrightarrow{Mxw} R_w$ for some R_w such that $R_w[\sigma] \equiv R_z[\sigma]$ for each surjective substitution σ with $z[\sigma] = w[\sigma]$.

TABLE IX
STRUCTURAL OPERATIONAL SEMANTICS OF THE π-CALCULUS WITH RELABELLING

TAU: $M\tau.P \xrightarrow{M\tau} P$

OUTPUT: $M\bar{x}y.P \xrightarrow{M\bar{x}y} P$

EARLY-INPUT: $Mx(y).P \xrightarrow{Mxz} P[\{z/y\}^{\mathcal{S}}]$

SUM: $$\frac{P \xrightarrow{\alpha} P'}{P+Q \xrightarrow{\alpha} P'}$$

SYMB-MATCH: $$\frac{P \xrightarrow{\alpha} P'}{[x{=}y]P \xrightarrow{[x=y]\alpha} P'}$$

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

PAR: $$\frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q} \begin{pmatrix}\mathrm{bn}(\alpha)\cap \mathrm{FN}(P)=\emptyset\\ \mathrm{bn}(\alpha)\cap \mathrm{FN}(Q)=\emptyset\end{pmatrix}$$

E-S-COM: $$\frac{P \xrightarrow{M\bar{x}y} P',\ Q \xrightarrow{Nvy} Q'}{P|Q \xrightarrow{[x=v]MN\tau} P'|Q'}$$

E-S-CLOSE: $$\frac{P \xrightarrow{M\bar{x}(z)} P',\ Q \xrightarrow{Nvz} Q'}{P|Q \xrightarrow{[x=v]MN\tau} (\nu z)(P'|Q')} \begin{pmatrix}z\notin \mathrm{FN}(P)\\ z\notin \mathrm{FN}(Q)\end{pmatrix}$$

RES-ALPHA: $$\frac{P[\{z/y\}] \xrightarrow{\alpha} P'}{(\nu y)P \xrightarrow{\alpha} (\nu z)P'} \begin{pmatrix}z\notin \mathrm{FN}((\nu y)P)\\ z\notin \mathrm{n}(\alpha)\end{pmatrix}$$

SYMB-OPEN-ALPHA: $$\frac{P \xrightarrow{M\bar{x}y} P'}{(\nu y)P \xrightarrow{M\bar{x}(z)} P'[\{z/y\}]} \begin{pmatrix}y\neq x\\ z\notin \mathrm{FN}((\nu y)P)\\ y\notin \mathrm{n}(M)\end{pmatrix}$$

RELABELLING: $$\frac{P \xrightarrow{\alpha} P'}{P[\sigma] \xrightarrow{\alpha[\sigma]} P'[\sigma]} \begin{pmatrix}\mathrm{bn}(\alpha[\sigma])\cap\\ \mathrm{FN}(P[\sigma])=\emptyset\end{pmatrix}$$

Proof. Let $R \xrightarrow{Mxz} R_z$. Then $\widehat{R} \xrightarrow{Mxz}{}^{\mathcal{S}}_{\bullet} \equiv^{\mathcal{S}} \widehat{R}_z$ by Lemma 58. So $\widehat{R} \xrightarrow{Mxz}{}^{\mathcal{S}}_{ES} U \equiv^{\mathcal{S}} \widehat{R}_z$ by Lemma 56. Pick $v \notin \mathrm{FN}(\widehat{R})$. By Lemma 38 $\widehat{R} \xrightarrow{Mxv}{}^{\mathcal{S}}_{ES} V$ for some V with $U \equiv^{\mathcal{S}} V\{z/v\}$, and by Lemma 39 $\widehat{R} \xrightarrow{Mxw}{}^{\mathcal{S}}_{ES} W$ for some W with $W \equiv^{\mathcal{S}} V\{w/v\}$. By Lemma 57 $\widehat{R} \xrightarrow{Mxw}{}^{\mathcal{S}}_{\bullet} W^\dagger$ for some $W^\dagger \equiv^{\mathcal{S}} W$. By Lemma 59 $R \xrightarrow{Mxw} R_w$ for some R_w with $\widehat{R}_w \equiv^{\mathcal{S}} W^\dagger$.

Now let σ be a surjective substitution with $z[\sigma] = w[\sigma]$. Then $\widehat{R_w[\sigma]} = \widehat{R}_w\sigma \equiv^{\mathcal{S}} V\{w/v\}\sigma = V\{z/v\}\sigma \equiv^{\mathcal{S}} \widehat{R}_z\sigma \equiv^{\mathcal{S}} \widehat{R_z[\sigma]}$ by(11), (12) and (13), so $R_w[\sigma] \equiv R_z[\sigma]$. ■

F. Agent identifiers without parameters

Let $\pi^{\rho}_{ES}(\mathcal{N})$ be the variant of $\pi^{\rho\bullet}_{ES}(\mathcal{N})$ in which agent identifiers may be called only with their own declared names as parameters, i.e. such that $\vec{y} = \vec{x}$ in rule IDE. As a result, the relabelling operator $[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]$ in this rule can be dropped. The operational semantics of $\pi^{\rho}_{ES}(\mathcal{N})$ is displayed in Table IX.

Let $\mathscr{T}_r : \mathrm{T}_{\pi^{\rho\bullet}_{ES}(\mathcal{N})} \to \mathrm{T}_{\pi^{\rho}_{ES}(\mathcal{N})}$ be the compositional translation satisfying $\mathscr{T}_r(A(\vec{y})) := A_r(\vec{x})[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]$, and acting homomorphically on all other operators. Here A_r is a fresh agent identifier with defining equation $A_r(\vec{x}) \stackrel{\text{def}}{=} \mathscr{T}_r(P)$ when $A(\vec{x}) \stackrel{\text{def}}{=} P$ was the defining equation of A. Clearly $\mathrm{FN}(\mathscr{T}_r(P)) = \mathrm{FN}(P)$. I will show that $\mathscr{T}_r$ is a valid translation from $\pi^{\rho\bullet}_{ES}(\mathcal{N})$ to $\pi^{\rho}_{ES}(\mathcal{N})$, up to strong barbed bisimilarity.

Lemma 67: If $R \xrightarrow{\beta} R'$ and $\mathrm{bn}(\beta) \cap \mathrm{FN}(R) = \emptyset$ then $\mathscr{T}_r(R) \xrightarrow{\beta} \mathscr{T}_r(R')$.

Proof. By induction of the inference of $R \xrightarrow{\beta} R'$.

- Suppose $R \xrightarrow{\beta} R'$ is derived by EARLY INPUT. Then $R = x(y).P$, $\beta = xz$ and $R' = P[\{z/y\}^{\mathcal{S}}]$. Now $\mathscr{T}_r(R) = x(y).\mathscr{T}_r(P) \xrightarrow{\beta} \mathscr{T}_r(P)[\{z/y\}^{\mathcal{S}}] = \mathscr{T}_r(R')$.
- Suppose $R \xrightarrow{\beta} R'$ is derived by IDE. Then $R = A(\vec{y})$, $A(\vec{x}) \stackrel{\text{def}}{=} P$ and $P[\{\vec{y}/\vec{x}\}^{\mathcal{S}}] \xrightarrow{\beta} R'$. As $\mathrm{bn}(\beta)\cap\mathrm{FN}(R)=\emptyset$ and $\mathrm{FN}(P[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]) \subseteq \mathrm{FN}(R)$, $\mathrm{bn}(\beta)\cap\mathrm{FN}(P[\{\vec{y}/\vec{x}\}^{\mathcal{S}}])=\emptyset$. By induction
$$\mathscr{T}_r(P)[\{\vec{y}/\vec{x}\}^{\mathcal{S}}] = \mathscr{T}_r(P[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]) \xrightarrow{\beta} \mathscr{T}_r(R').$$
so by RELABELLING $\mathscr{T}_r(P) \xrightarrow{\alpha} P'$ for some α and P' with $\alpha[\{\vec{y}/\vec{x}\}^{\mathcal{S}}] = \beta$ and $P'[\{\vec{y}/\vec{x}\}^{\mathcal{S}}] = \mathscr{T}_r(R')$. By rule IDE from $\pi^{\rho}_{ES}(\mathcal{N})$, $A_r(\vec{x}) \xrightarrow{\alpha} P'$. So by RELABELLING $\mathscr{T}_r(R) = A_r(\vec{x})[\{\vec{y}/\vec{x}\}^{\mathcal{S}}] \xrightarrow{\beta} P'[\{\vec{y}/\vec{x}\}^{\mathcal{S}}] = \mathscr{T}_r(R')$, here using the side condition that $\mathrm{bn}(\beta) \cap \mathrm{FN}(R) = \emptyset$.
- Using that $\mathrm{FN}(\mathscr{T}_r(Q)) = \mathrm{FN}(Q)$ for all Q, all remaining cases are trivial. ■

Lemma 68: If $\mathscr{T}_r(R) \xrightarrow{\beta} U$ then $R \xrightarrow{\beta} R'$ for some R' with $\mathscr{T}_r(R') = U$.

Proof. By induction of the inference of $\mathscr{T}_r(R) \xrightarrow{\beta} U$.

- Suppose $R = A(\vec{y})$. Then $\mathscr{T}_r(R) = A_r(\vec{x})[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]$. By RELABELLING $\mathrm{bn}(\beta)\cap\mathrm{FN}(\mathscr{T}_r(R)) = \emptyset$ and $A_r(\vec{x}) \xrightarrow{\alpha} V$ for some α and V with $\alpha[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]=\beta$ and $V[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]=U$. Let $A(\vec{x}) \stackrel{\text{def}}{=} P$, so that $A_r(\vec{x}) \stackrel{\text{def}}{=} \mathscr{T}_r(P)$. Via rule IDE $\mathscr{T}_r(P) \xrightarrow{\alpha} V$. By induction $P \xrightarrow{\alpha} P'$ for some P' with $\mathscr{T}_r(P') = V$. As $\mathrm{FN}(P[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]) \subseteq \mathrm{FN}(R) = \mathrm{FN}(\mathscr{T}_r(R))$, $\mathrm{bn}(\beta) \cap \mathrm{FN}(P[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]) = \emptyset$. So by RELABELLING $P[\{\vec{y}/\vec{x}\}^{\mathcal{S}}] \xrightarrow{\beta} P'[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]$. By rule IDE from $\pi^{\rho\bullet}_{ES}(\mathcal{N})$, $R \xrightarrow{\beta} P'[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]$. Moreover,
$$\mathscr{T}_r(P'[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]) = \mathscr{T}_r(P')[\{\vec{y}/\vec{x}\}^{\mathcal{S}}] = V[\{\vec{y}/\vec{x}\}^{\mathcal{S}}] = U.$$
- Using that $\mathrm{FN}(\mathscr{T}_r(Q)) = \mathrm{FN}(Q)$ for all Q, all other cases are trivial. ■

Theorem 12: $\mathscr{T}_r(P) \mathrel{\dot{\sim}} P$ for any $P \in \mathrm{T}_{\pi^{\rho\bullet}_{ES}(\mathcal{N})}$.

Proof. It suffices to show that the symmetric closure of
$$\mathscr{R} := \{(R, \mathscr{T}_r(R)) \mid R \in \mathrm{T}_{\pi^{\rho\bullet}_{ES}(\mathcal{N})}\}$$
is a strong barbed bisimulation. This follows from Lemmas 67 and 68, also using Lemma 64 to handle the barbs from R. ■

G. Clash-free processes

For $\mathcal{R}$ a collection of *private names*, disjoint with $\mathcal{N} \uplus \mathcal{S} =: \mathcal{Z}$, I define $\pi^{\rho}_{ES}(\mathcal{N}, \mathcal{R})$ as the variant of $\pi^{\rho}_{ES}(\mathcal{N} \uplus \mathcal{R})$ where in all expressions $x(y).Q$ the name y must be chosen from $\mathcal{N}$,

and in defining equations $A(\vec{x}) \stackrel{\text{def}}{=} P$ one has $x_1, \ldots, x_n \in \mathcal{N}$. The semantics of $\pi^{\rho}_{ES}(\mathcal{N}, \mathcal{R})$ equals that of $\pi^{\rho}_{ES}(\mathcal{N} \uplus \mathcal{R})$.

Clearly, the relation $\equiv$ and all results from Section E are inherited by $\pi^{\rho}_{ES}(\mathcal{N}, \mathcal{R})$.

Definition 22: The set $\text{RN}(P)$ of *restriction-bound* names of a $\pi^{\rho}_{ES}(\mathcal{N}, \mathcal{R})$ process P is inductively defined by

- if $R = (\nu y)P$ then $\text{RN}(R) \supseteq \text{RN}(P) \cup \{y\}$,
- if $R = \tau.P$ or $\bar{x}y.P$ or $x(y).P$ then $\text{RN}(R) \supseteq \text{RN}(P)$,
- if $R = [x{=}y]P$ then $\text{RN}(R) \supseteq \text{RN}(P)$,
- if $R = P[\sigma]$ then $\text{RN}(R) \supseteq \{x[\sigma] \mid x \in \text{RN}(P)\}$,
- if $R = P + Q$ or $P|Q$ then $\text{RN}(R) \supseteq \text{RN}(P) \cup \text{RN}(Q)$,
- if $R = A(\vec{x})$ and $A(\vec{x}) \stackrel{\text{def}}{=} P$ then $\text{RN}(R) \supseteq \text{RN}(P)$.

Each statement $y{\in}\text{RN}(R)$ can be proven using $y{\in}\text{RN}((\nu y)P)$ as an axiom, and each of the six clauses above as proof rules. It follows that in fact one has $=$ wherever $\supseteq$ is written in Definition 22.

Definition 23: The collection $\mathscr{C}$ of *clashing* $\pi^{\rho}_{ES}(\mathcal{N}, \mathcal{R})$ processes is the smallest such that

(i) if $\text{FN}(R) \cap \text{RN}(R) \neq \emptyset$ then $R \in \mathscr{C}$,
(ii) if $R = P|Q$ and $\text{RN}(P) \cap \text{RN}(Q) \neq \emptyset$ then $R \in \mathscr{C}$,
(iii) if $R = (\nu y)P$ and $y \in \text{RN}(P)$ then $R \in \mathscr{C}$,
(iv) if $R = P[\sigma]$ and $y[\sigma] = z[\sigma]$ for different $y, z \in \text{RN}(P)$ then $R \in \mathscr{C}$,
(v) if $P \in \mathscr{C}$ then also $\tau.P \in \mathscr{C}$, $\bar{x}y.P \in \mathscr{C}$, $x(y).P \in \mathscr{C}$, $[x{=}y]P \in \mathscr{C}$, $(\nu y)P \in \mathscr{C}$ and $P[\sigma] \in \mathscr{C}$,
(vi) if $P \in \mathscr{C}$ or $Q \in \mathscr{C}$ then also $P{+}Q \in \mathscr{C}$ and $P|Q \in \mathscr{C}$,
(vii) if $A(\vec{x}) \stackrel{\text{def}}{=} P$ and $P \in \mathscr{C}$ then also $A(\vec{x}) \in \mathscr{C}$,
(viii) and if $\text{RN}(R) \cap \mathcal{Z} \neq \emptyset$ then $R \in \mathscr{C}$.

A process P is *clash-free* if $P \notin \mathscr{C}$.

Now take $\mathcal{R}$ as given in Section IX, where also the relabelling operators $[\ell]$, $[r]$, $[e]$ and $[p_y]$ are defined, with $p_y = \{p/\!\!/y\} \circ e$. Let the translation $\mathscr{T}_{\text{cf}} : \text{T}_{\pi^{\rho}_{ES}(\mathcal{N})} \to \text{T}_{\pi^{\rho}_{ES}(\mathcal{N},\mathcal{R})}$ satisfy

$$\begin{array}{lcl} \mathscr{T}_{\text{cf}}((\nu y)P) & := & (\nu p)(\mathscr{T}_{\text{cf}}(P)[p_y]) \\ \mathscr{T}_{\text{cf}}(P \mid Q) & := & \mathscr{T}_{\text{cf}}(P)[\ell] \parallel \mathscr{T}_{\text{cf}}(Q)[r] \\ \mathscr{T}_{\text{cf}}(A(\vec{x})) & := & A_{\text{cf}}(\vec{x}) \end{array}$$

and act homomorphically on all other operators. Here A_{cf} is a fresh agent identifier with defining equation $A_{\text{cf}}(\vec{x}) \stackrel{\text{def}}{=} \mathscr{T}_{\text{cf}}(P)$ when $A(\vec{x}) \stackrel{\text{def}}{=} P$ was the defining equation of A.

Lemma 69: Let $P \in \text{T}_{\pi^{\rho}_{ES}(\mathcal{N})}$ be a $\pi^{\rho}_{ES}(\mathcal{N})$ process. Then $\text{FN}(\mathscr{T}_{\text{cf}}(P)) = \text{FN}(P) \subseteq \mathcal{Z}$ and $\mathscr{T}_{\text{cf}}(P)$ is clash-free.

Proof. The first statement follows by a straightforward structural induction on P.

Let $\mathcal{R}_0 := \{\wp p \mid \wp \in \{e, \ell, r\}^*\}$; these are the names $\wp p^{\upsilon} \in \mathcal{R}$ with $\upsilon = \varepsilon$. A straightforward induction on the derivation of $y \in \text{RN}(\mathscr{T}_{\text{cf}}(P))$ from the rules of Definition 22 shows that $\text{RN}(\mathscr{T}_{\text{cf}}(P)) \subseteq \mathcal{R}_0$.

Using these facts to handle Requirements (i) and (viii), a straightforward induction on the derivation of $\mathscr{T}_{\text{cf}}(P) \in \mathscr{C}$ from the rules of Definition 23 leads to a contradiction. For (ii) use that $\forall y, z \in \mathcal{R}_0.\ y[\ell] \neq z[r]$. For (iii) use that $z[p_y] \neq p$ for all $z \in \mathcal{R}_0$. For (iv) use that p_y, ℓ and r are injective substitutions, whereas any relabelling operator $[\sigma]$ inherited from $\pi^{\rho}_{ES}(\mathcal{N})$ satisfies $dom(\sigma) \cap \mathcal{R} = \emptyset$. ■

I will show that $\mathscr{T}_{\text{cf}}$ is a valid translation, up to strong barbed bisimilarity.

Lemma 70: $\mathscr{T}_{\text{cf}}((\nu y)P) \equiv (\nu y)\mathscr{T}_{\text{cf}}(P)$ for $P \in \text{T}_{\pi^{\rho}_{ES}(\mathcal{N})}$.

Proof. By definition $\mathscr{T}_{\text{cf}}((\nu y)P) = (\nu p)(\mathscr{T}_{\text{cf}}(P)[p_y])$. Since $\text{FN}(\mathscr{T}_{\text{cf}}(P)){=}\text{FN}(P){\subseteq}\mathcal{Z}$ by Lemma 69, $y \notin \text{FN}(\mathscr{T}_{\text{cf}}(P)[p_y])$. $(\nu p)(\mathscr{T}_{\text{cf}}(P)[p_y]) \equiv (\nu y)(\mathscr{T}_{\text{cf}}(P)[p_y][\{y/\!\!/p\}]) \equiv (\nu y)\mathscr{T}_{\text{cf}}(P)$ by Lemma 61(f,b,a,c,h). ■

Lemma 71: If $R \xrightarrow{\beta} R'$ then $\mathscr{T}_{\text{cf}}(R) \xrightarrow{\beta}{\equiv} \mathscr{T}_{\text{cf}}(R')$.

Proof. By induction of the inference of $R \xrightarrow{\beta} R'$. Since R is a $\pi^{\rho}_{ES}(\mathcal{N})$ process, $\text{n}(\beta) \cup \text{n}(R) \subseteq \mathcal{Z}$.

- Suppose $R \xrightarrow{\beta} R'$ is derived by **EARLY INPUT**. Then $R = x(y).P$, $\beta = xz$ and $R' = P[\{z/y\}^{\mathcal{S}}]$. Now $\mathscr{T}_{\text{cf}}(R) = x(y).\mathscr{T}_{\text{cf}}(P) \xrightarrow{\beta} \mathscr{T}_{\text{cf}}(P)[\{z/y\}^{\mathcal{S}}] = \mathscr{T}_{\text{cf}}(R')$.
- Suppose $R \xrightarrow{\beta} R'$ is derived by **IDE**. Then $R = A(\vec{x})$. Let $A(\vec{x}) \stackrel{\text{def}}{=} P$, so $A_{\text{cf}}(\vec{x}) \stackrel{\text{def}}{=} \mathscr{T}_{\text{cf}}(P)$. Then $P \xrightarrow{\beta} R'$. By induction $\mathscr{T}_{\text{cf}}(P) \xrightarrow{\beta}{\equiv} \mathscr{T}_{\text{cf}}(R')$. Applying rule **IDE**, $\mathscr{T}_{\text{cf}}(R) = A_{\text{cf}}(\vec{x}) \xrightarrow{\beta}{\equiv} \mathscr{T}_{\text{cf}}(R')$.
- The cases that $R \xrightarrow{\beta} R'$ is derived by **TAU**, **OUTPUT**, **SUM** or **SYMB-MATCH** are (also) trivial.
- Suppose $R \xrightarrow{\beta} R'$ is derived by **PAR**. Then $R = P|Q$, $P \xrightarrow{\beta} P'$, $\text{bn}(\beta){\cap}\text{FN}(R) = \emptyset$ and $R' = P'|Q$. By induction $\mathscr{T}_{\text{cf}}(P) \xrightarrow{\beta} V \equiv \mathscr{T}_{\text{cf}}(P')$. As $\text{n}(\beta) \subseteq \mathcal{Z}$, $\beta[\ell] = \beta$. Moreover, $\mathcal{Z} \supseteq \text{FN}(P) = \text{FN}(\mathscr{T}_{\text{cf}}(P)) = \text{FN}(\mathscr{T}_{\text{cf}}(P)[\ell])$ by Lemma 69, so $\text{bn}(\beta) \cap \text{FN}(\mathscr{T}_{\text{cf}}(P)[\ell]) = \emptyset$. Thus, $\mathscr{T}_{\text{cf}}(P)[\ell] \xrightarrow{\beta} V[\ell] \equiv \mathscr{T}_{\text{cf}}(P')[\ell]$ by rule **RELABELLING** and Lemma 61(h). Since $\mathcal{Z} \supseteq \text{FN}(Q) = \text{FN}(\mathscr{T}_{\text{cf}}(Q)) = \text{FN}(\mathscr{T}_{\text{cf}}(Q)[r])$ by Lemma 69, $\text{bn}(\beta){\cap}\text{FN}(\mathscr{T}_{\text{cf}}(Q)[r]){=}\emptyset$. Hence, by rule **PAR**, $\mathscr{T}_{\text{cf}}(R) = \mathscr{T}_{\text{cf}}(P)[\ell]|\mathscr{T}_{\text{cf}}(Q)[r] \xrightarrow{\beta} V[\ell]|\mathscr{T}_{\text{cf}}(Q)[r] \equiv \mathscr{T}_{\text{cf}}(P')[\ell]|\mathscr{T}_{\text{cf}}(Q)[r] = \mathscr{T}_{\text{cf}}(R')$, in the $\equiv$-step using Lemma 61(g).
- Suppose $R \xrightarrow{\beta} R'$ is derived by **E-S-COM**. Then $R = P|Q$, $P \xrightarrow{M\bar{x}y} P'$, $Q \xrightarrow{Nvy} Q'$, $R' = P'|Q'$, and $\beta = [x{=}v]MN\tau$. By induction $\mathscr{T}_{\text{cf}}(P) \xrightarrow{M\bar{x}y} V \equiv \mathscr{T}_{\text{cf}}(P')$ and $\mathscr{T}_{\text{cf}}(Q) \xrightarrow{Nvy} W \equiv \mathscr{T}_{\text{cf}}(Q')$. By rule **RELABELLING** and Lemma 61(h) $\mathscr{T}_{\text{cf}}(P)[\ell] \xrightarrow{M\bar{x}y} V[\ell] \equiv \mathscr{T}_{\text{cf}}(P')[\ell]$ and $\mathscr{T}_{\text{cf}}(Q)[r] \xrightarrow{Nvy} W[r] \equiv \mathscr{T}_{\text{cf}}(Q')[r]$. Thus, by **E-S-COM** and Lemma 61(g) $\mathscr{T}_{\text{cf}}(R) = \mathscr{T}_{\text{cf}}(P)[\ell]|\mathscr{T}_{\text{cf}}(Q)[r] \xrightarrow{\beta} V[\ell]|W[r] \equiv \mathscr{T}_{\text{cf}}(P')[\ell]|\mathscr{T}_{\text{cf}}(Q')[r] = \mathscr{T}_{\text{cf}}(R')$.
- Suppose $R \xrightarrow{\beta} R'$ is derived by **E-S-CLOSE**. Then $R = P|Q$, $P \xrightarrow{M\bar{x}(z)} P'$, $Q \xrightarrow{Nvz} Q'$, $z \notin \text{FN}(R)$, $\beta = [x{=}v]MN\tau$ and $R' = (\nu z)(P'|Q')$. By induction $\mathscr{T}_{\text{cf}}(P) \xrightarrow{M\bar{x}(z)} V \equiv \mathscr{T}_{\text{cf}}(P')$ and $\mathscr{T}_{\text{cf}}(Q) \xrightarrow{Nvz} W \equiv \mathscr{T}_{\text{cf}}(Q')$. Since $z{\notin}\text{FN}(P){=}\text{FN}(\mathscr{T}_{\text{cf}}(P)){=}\text{FN}(\mathscr{T}_{\text{cf}}(P)[\ell])$, by rule **RELABELLING** and Lemma 61(h) one obtains $\mathscr{T}_{\text{cf}}(P)[\ell] \xrightarrow{M\bar{x}(z)} V[\ell] \equiv \mathscr{T}_{\text{cf}}(P')[\ell]$ and likewise $\mathscr{T}_{\text{cf}}(Q)[r] \xrightarrow{Nvz} W[r] \equiv \mathscr{T}_{\text{cf}}(Q')[r]$. Thus, by **E-S-CLOSE** and Lemma 61(g,h) $\mathscr{T}_{\text{cf}}(R){=}\mathscr{T}_{\text{cf}}(P)[\ell]|\mathscr{T}_{\text{cf}}(Q)[r] \xrightarrow{\beta} (\nu z)(V[\ell]|W[r]) \equiv (\nu z)(\mathscr{T}_{\text{cf}}(P')[\ell]|\mathscr{T}_{\text{cf}}(Q')[r]) \equiv \mathscr{T}_{\text{cf}}(R')$. The last step follows by Lemma 70.
- Suppose $R \xrightarrow{\beta} R'$ is derived by **RES-ALPHA**. Then $R =$

$(\nu y)P$, $P[\{z/\!\!/y\}] \xrightarrow{\beta} P'$, $R' = (\nu z)P'$, $z \notin \text{FN}(R)$ and $z \notin \text{n}(\beta)$. By induction $\mathscr{T}_{\text{cf}}(P[\{z/\!\!/y\}]) \xrightarrow{\beta} V \equiv \mathscr{T}_{\text{cf}}(P')$. By Lemma 69 $z \notin \text{FN}(\mathscr{T}_{\text{cf}}(R))$. Using this, Lemma 70, **RES-ALPHA**, Lemma 61(h), and the observation that $\mathscr{T}_{\text{cf}}(P[\{z/\!\!/y\}]) = \mathscr{T}_{\text{cf}}(P)[\{z/\!\!/y\}]$, one obtains $\mathscr{T}_{\text{cf}}(R) \equiv (\nu y)\mathscr{T}_{\text{cf}}(P) \xrightarrow{\beta} (\nu z)V \equiv (\nu z)\mathscr{T}_{\text{cf}}(P') \equiv \mathscr{T}_{\text{cf}}(R')$.
As $P[\{z/\!\!/y\}] \xrightarrow{\beta} P'$ must be derived by rule **RELABELLING**, $\text{bn}(\beta) \cap \text{FN}(P[\{z/\!\!/y\}]) = \emptyset$. Since $z \notin \text{bn}(\beta)$ this implies $\text{bn}(\beta) \cap \text{FN}(R) = \emptyset$. Using this, Lemma 60 yields $\mathscr{T}_{\text{cf}}(R) \xrightarrow{\beta} \equiv \mathscr{T}_{\text{cf}}(R')$.

- Suppose $R \xrightarrow{\beta} R'$ is derived by **SYMB-OPEN-ALPHA**. Then $R = (\nu y)P$, $\beta = M\bar{x}(z)$, $P \xrightarrow{M\bar{x}y} P'$, $y \neq x$, $z \notin \text{FN}(R)$, $y \notin \text{n}(M)$ and $R' = P'[\{z/\!\!/y\}]$. By induction $\mathscr{T}_{\text{cf}}(P) \xrightarrow{M\bar{x}y} V \equiv \mathscr{T}_{\text{cf}}(P')$. By Lemma 70 $\mathscr{T}_{\text{cf}}(R) \equiv (\nu y)\mathscr{T}_{\text{cf}}(P)$. By Lemma 69 $z \notin \text{FN}(R) = \text{FN}(\mathscr{T}_{\text{cf}}(R)) = \text{FN}((\nu y)\mathscr{T}_{\text{cf}}(P))$. So by rule **SYMB-OPEN-ALPHA** $(\nu y)\mathscr{T}_{\text{cf}}(P) \xrightarrow{M\bar{x}(z)} V[\{z/\!\!/y\}]$. By Lemma 60 $\mathscr{T}_{\text{cf}}(R) \xrightarrow{M\bar{x}(z)} \equiv V[\{z/\!\!/y\}]$. Moreover, by Lemma 61(h), $V[\{z/\!\!/y\}] \equiv \mathscr{T}_{\text{cf}}(P')[\{z/\!\!/y\}] = \mathscr{T}_{\text{cf}}(R')$.
- Suppose $R \xrightarrow{\beta} R'$ is derived by **RELABELLING**. Then $R = P[\sigma]$, $P \xrightarrow{\alpha} P'$, $\alpha[\sigma] = \beta$, $\text{bn}(\beta) \cap \text{FN}(R) = \emptyset$ and $R' = P'[\sigma]$. By induction $\mathscr{T}_{\text{cf}}(P) \xrightarrow{\alpha} V \equiv \mathscr{T}_{\text{cf}}(P')$. As $\text{bn}(\beta) \cap \text{FN}(\mathscr{T}_{\text{cf}}(R)) = \emptyset$, by **RELABELLING** and Lemma 61(h) $\mathscr{T}_{\text{cf}}(R) = \mathscr{T}_{\text{cf}}(P)[\sigma] \xrightarrow{\beta} V[\sigma] \equiv \mathscr{T}_{\text{cf}}(P')[\sigma] = \mathscr{T}_{\text{cf}}(R')$. ■

Lemma 72: If $\mathscr{T}_{\text{cf}}(R) \xrightarrow{\beta} U$ with $\iota(\beta) \cup \text{bn}(\beta) \subseteq \mathcal{Z}$ then $R \xrightarrow{\beta} R'$ for some R' with $\mathscr{T}_{\text{cf}}(R') \equiv U$.

Proof. By induction of the depth of the inference of the transition $\mathscr{T}_{\text{cf}}(R) \xrightarrow{\beta} U$. Since R is a $\pi^{\rho}_{ES}(\mathcal{N})$ process, using Lemma 69, $\text{FN}(\mathscr{T}_{\text{cf}}(R)) = \text{FN}(R) \subseteq \mathcal{Z}$.

- The case that $\mathscr{T}_{\text{cf}}(R) \xrightarrow{\beta} U$ is derived by **EARLY-INPUT** proceeds as in the proof of Lemma 71, but using $\iota(\beta) \subseteq \mathcal{N}$.
- Suppose $\mathscr{T}_{\text{cf}}(R) \xrightarrow{\beta} U$ is derived by **IDE**. Then $R = A(\vec{x})$. Let $A(\vec{x}) \stackrel{\text{def}}{=} P$, so $A_{\text{cf}}(\vec{x}) \stackrel{\text{def}}{=} \mathscr{T}_{\text{cf}}(P)$. Then $\mathscr{T}_{\text{cf}}(P) \xrightarrow{\beta} U$. By induction $P \xrightarrow{\beta} R'$ for some R' with $\mathscr{T}_{\text{cf}}(R') \equiv U$. Applying rule **IDE**, $R \xrightarrow{\beta} R'$.
- The cases that $\mathscr{T}_{\text{cf}}(R) \xrightarrow{\beta} U$ is derived by **TAU**, **OUTPUT**, **SUM** or **SYMB-MATCH** are (also) trivial.
- Suppose $\mathscr{T}_{\text{cf}}(R) \xrightarrow{\beta} U$ is derived by **PAR**. Then $R = P|Q$, $\mathscr{T}_{\text{cf}}(R) = \mathscr{T}_{\text{cf}}(P)[\ell]|\mathscr{T}_{\text{cf}}(Q)[r]$, $\mathscr{T}_{\text{cf}}(P) \xrightarrow{\alpha} V$, $\alpha[\ell] = \beta$, $\text{bn}(\beta) \cap \text{FN}(\mathscr{T}_{\text{cf}}(R)) = \emptyset$ and $U = V[\ell]|\mathscr{T}_{\text{cf}}(Q)[r]$. Since $\iota(\beta) \cup \text{bn}(\beta) \subseteq \mathcal{Z}$, also $\iota(\alpha) \cup \text{bn}(\alpha) \subseteq \mathcal{Z}$. Therefore, by induction, $P \xrightarrow{\alpha} P'$ for some P' with $\mathscr{T}_{\text{cf}}(P') \equiv V$. So $\text{n}(\alpha) \subseteq \mathcal{Z}$ and $\alpha = \alpha[\ell] = \beta$. By rule **PAR**, using that $\text{bn}(\beta) \cap \text{FN}(R) = \emptyset$ via Lemma 69, $R = P|Q \xrightarrow{\beta} P'|Q$. By Lemma 61(g,h), $\mathscr{T}_{\text{cf}}(P'|Q) = \mathscr{T}_{\text{cf}}(P')[\ell]|\mathscr{T}_{\text{cf}}(Q)[r] \equiv V[\ell]|\mathscr{T}_{\text{cf}}(Q)[r] = U$.
- Suppose $\mathscr{T}_{\text{cf}}(R) \xrightarrow{\beta} U$ is derived by **E-S-COM**. Then $R = P|Q$, $\mathscr{T}_{\text{cf}}(R) = \mathscr{T}_{\text{cf}}(P)[\ell]|\mathscr{T}_{\text{cf}}(Q)[r]$, $\beta = [x=v]MN\tau$, $\mathscr{T}_{\text{cf}}(P) \xrightarrow{M\bar{x}y} V$, $\mathscr{T}_{\text{cf}}(Q) \xrightarrow{Nvz} W$, $y[\ell] = z[r]$ and $U = V[\ell]|W[r]$. Here I use Lemma 62 to infer that $\text{n}(M) \cup \text{n}(N) \cup \{x, y, v\} \subseteq \mathcal{N}$, so that these names are not affected by $[\ell]$ or $[r]$. It follows that $z = y$. By induction $P \xrightarrow{M\bar{x}y} P$ and $Q \xrightarrow{Nvy} Q'$ for processes P', Q' with $\mathscr{T}_{\text{cf}}(P') \equiv V$ and $\mathscr{T}_{\text{cf}}(Q') \equiv W$. Applying **E-S-COM**, $R \xrightarrow{\beta} P'|Q'$. Moreover, employing Lemma 61(g,h), $\mathscr{T}_{\text{cf}}(P'|Q') = \mathscr{T}_{\text{cf}}(P')[\ell] \mid \mathscr{T}_{\text{cf}}(Q')[r] \equiv V[\ell]|W[r] = U$.
- Suppose $\mathscr{T}_{\text{cf}}(R) \xrightarrow{\beta} U$ is derived by **E-S-CLOSE**. Then $R = P|Q$, $\mathscr{T}_{\text{cf}}(R) = \mathscr{T}_{\text{cf}}(P)[\ell]|\mathscr{T}_{\text{cf}}(Q)[r]$, $\mathscr{T}_{\text{cf}}(P) \xrightarrow{\alpha} V$, $\mathscr{T}_{\text{cf}}(Q) \xrightarrow{\gamma} W$, $\beta = [x=v]MN\tau$, $\alpha[\ell] = M\bar{x}(z)$, $\gamma[r] = Nvz$, $z \notin \text{FN}(\mathscr{T}_{\text{cf}}(R))$ and $U = (\nu z)((V[\ell]|W[r])$. By Lemma 62 $\{x, v\} \cup \text{n}(M) \cup \text{n}(N) \subseteq \text{FN}(\mathscr{T}_{\text{cf}}(R)) \subseteq \mathcal{Z}$. So $\alpha = M\bar{x}(w)$ and $\gamma = Nvu$ with $w[\ell] = u[r] = z$. Note that $w, u \notin \text{FN}(\mathscr{T}_{\text{cf}}(R)) \subseteq \mathcal{Z}$. Pick $q \in \mathcal{Z} \backslash \text{FN}(R)$. By taking $q \neq w, u$ one also has $q \notin \text{FN}(V|W|V[\ell]|W[r])$. By Lemmas 64 and 65 $\mathscr{T}_{\text{cf}}(P) \xrightarrow{M\bar{x}(q)} V' \equiv V\{q/\!\!/w\}$ and $\mathscr{T}_{\text{cf}}(Q) \xrightarrow{Nxq} W' \equiv W\{q/\!\!/w\}$, and the inferences of these transitions have a smaller depth than the one of $\mathscr{T}_{\text{cf}}(R) \xrightarrow{\beta} U$. Therefore, by induction, $P \xrightarrow{M\bar{x}(q)} P'$ and $Q \xrightarrow{Nxq} Q'$ for some P' and Q' with $\mathscr{T}_{\text{cf}}(P') \equiv V'$ and $\mathscr{T}_{\text{cf}}(Q') \equiv W'$. By rule **E-S-CLOSE** $R \xrightarrow{\beta} (\nu q)(P'|Q')$. By Lemma 70 and Lemma 61(g,h,b,a,d,f)

$$\begin{aligned}
\mathscr{T}_{\text{cf}}((\nu q)(P'|Q')) &\equiv (\nu q)\big((\mathscr{T}_{\text{cf}}(P')[\ell] \mid \mathscr{T}_{\text{cf}}(Q')[r])\big) \\
&\equiv (\nu q)\big((V'[\ell] \mid W'[r])\big) \\
&\equiv (\nu q)\big((V[\{q/\!\!/w\}][\ell] \mid W[\{q/\!\!/u\}][r])\big) \\
&\equiv (\nu q)\big((V[\ell][\{q/\!\!/z\}] \mid W[r][\{q/\!\!/z\}])\big)\big] \\
&\equiv (\nu q)\big((V[\ell] \mid W[r])[\{q/\!\!/z\}]\big) \\
&\equiv (\nu z)(V[\ell] \mid W[r]) \equiv U.
\end{aligned}$$

- Suppose R has the form $(\nu y)P$, i.e., $\mathscr{T}_{\text{cf}}(R) \xrightarrow{\beta} U$ is derived by **RES-ALPHA** or **SYMB-OPEN-ALPHA**. By Lemma 70 $\mathscr{T}_{\text{cf}}(R) \equiv (\nu y)\mathscr{T}_{\text{cf}}(P)$, so by Lemma 60 $(\nu y)\mathscr{T}_{\text{cf}}(P) \xrightarrow{\beta} V \equiv U$, and the inference of this transition has the same depth as the one of $\mathscr{T}_{\text{cf}}(R) \xrightarrow{\beta} U$.

  First suppose $(\nu y)\mathscr{T}_{\text{cf}}(P) \xrightarrow{\beta} V$ is derived by **RES-ALPHA**. Then $\mathscr{T}_{\text{cf}}(P[\{z/\!\!/y\}]) = \mathscr{T}_{\text{cf}}(P)[\{z/\!\!/y\}] \xrightarrow{\beta} W$ and $V = (\nu z)W$ for some W and $z \notin \text{FN}(R) \cup \text{n}(\beta)$. So by induction $P[\{z/\!\!/y\}] \xrightarrow{\beta} P'$ for some P' with $\mathscr{T}_{\text{cf}}(P') \equiv W$. By **RES-ALPHA** $R = (\nu y)P \xrightarrow{\beta} (\nu z)P'$. Moreover, by Lemma 70 and Lemma 61(h),

  $$\mathscr{T}_{\text{cf}}((\nu z)P') \equiv (\nu z)\mathscr{T}_{\text{cf}}(P') \equiv (\nu z)W = V \equiv U.$$

  Next suppose $(\nu y)\mathscr{T}_{\text{cf}}(P) \xrightarrow{\beta} V$ is derived by **SYMB-OPEN-ALPHA**. Then $\beta = M\bar{x}(z)$, $\mathscr{T}_{\text{cf}}(P) \xrightarrow{M\bar{x}y} W$, $y \neq x$, $z \notin \text{FN}(R)$, $y \notin \text{n}(M)$ and $V = W[\{z/\!\!/y\}]$. By induction $P \xrightarrow{M\bar{x}y} P'$ for some P' with $\mathscr{T}_{\text{cf}}(P') \equiv W$. By **RES-ALPHA** $R \xrightarrow{\beta} P'[\{z/\!\!/y\}]$. By Lemma 61(h),

  $$\mathscr{T}_{\text{cf}}(P'[\{z/\!\!/y\}]) = \mathscr{T}_{\text{cf}}(P')[\{z/\!\!/y\}] \equiv W[\{z/\!\!/y\}] = V \equiv U.$$

- Suppose $\mathscr{T}_{\text{cf}}(R) \xrightarrow{\beta} U$ is derived by **RELABELLING**. Then $R = P[\sigma]$, $\mathscr{T}_{\text{cf}}(R) = \mathscr{T}_{\text{cf}}(P)[\sigma]$, $\mathscr{T}_{\text{cf}}(P) \xrightarrow{\alpha} V$, $\alpha[\sigma] = \beta$, $\text{bn}(\beta) \cap \text{FN}(R) = \emptyset$ and $U = V[\sigma]$. Since the relabelling operator $[\sigma]$ stems from $\pi^{\rho}_{ES}(\mathcal{N})$, $dom(\sigma) \subseteq \mathcal{Z}$. Hence $\iota(\beta) \cup \text{bn}(\beta) \subseteq \mathcal{Z}$ implies $\iota(\alpha) \cup \text{bn}(\alpha) \subseteq \mathcal{Z}$. By induction $P \xrightarrow{\alpha} P'$ for some P' with $\mathscr{T}_{\text{cf}}(P') \equiv V$. By **RELABELLING** $R = P[\sigma] \xrightarrow{\beta} P'[\sigma]$. Moreover, by Lemma 61(h), $\mathscr{T}_{\text{cf}}(P'[\sigma]) = \mathscr{T}_{\text{cf}}(P')[\sigma] \equiv V[\sigma] = U$. ■

Theorem 13: $\mathscr{T}_{\rm cf}(P) \nsim P$ for any $P \in {\rm T}_{\pi^{\rho}_{ES}(\mathcal{N})}$.

Proof. It suffices to show that the symmetric closure of

$$\mathscr{R} := \{(R,U) \mid R \in {\rm T}_{\pi^{\rho}_{ES}(\mathcal{N})} \wedge U \equiv \mathscr{T}_{\rm cf}(R)\}$$

is a strong barbed bisimulation, as it contains $(P, \mathscr{T}_{\rm cf}(P))$.

Let $(R,U) \in \mathscr{R}$. Let $R \xrightarrow{\tau} R'$. Then $\mathscr{T}_{\rm cf}(R) \xrightarrow{\tau} \equiv \mathscr{T}_{\rm cf}(R')$ by Lemma 71 and $U \xrightarrow{\tau} \equiv \mathscr{T}_{\rm cf}(R')$ by Lemma 60. Thus $U \xrightarrow{\tau} U'$ for some U' with $(R',U') \in \mathscr{R}$.

Let $U \xrightarrow{\tau} U'$. Then $\mathscr{T}_{\rm cf}(R) \xrightarrow{\tau} \equiv U'$ by Lemma 60. By Lemma 72 $R \xrightarrow{\tau} R'$ for some R' with $\mathscr{T}_{\rm cf}(R') \equiv U'$.

Let $U{\downarrow_b}$ with $b \in \mathcal{Z} \cup \overline{\mathcal{Z}}$. Then $U \xrightarrow{by} U'$ or $U \xrightarrow{b(y)} U'$ for some y and U', using the definition of O in Section IV. By Lemmas 66 and 64 I may assume, without loss of generality, that $y \in \mathcal{Z} \backslash {\rm FN}(U)$ in case of input or bound output actions. Hence $\mathscr{T}_{\rm cf}(R) \xrightarrow{by}$ or $\mathscr{T}_{\rm cf}(R) \xrightarrow{b(y)}$ by Lemma 60. So $R \xrightarrow{by}$ or $R \xrightarrow{b(y)}$ by Lemma 72. Thus $R{\downarrow_b}$.

The implication $R{\downarrow_b} \Rightarrow U{\downarrow_b}$ proceeds likewise. ∎

H. Eliminating α-conversion for clash-free processes

Let $\xrightarrow{\alpha}_\bullet$ be the transition relation on ${\rm T}_{\pi^{\rho}_{ES}(\mathcal{N},\mathcal{R})}$ obtained by from the transition relation $\xrightarrow{\alpha}$ by replacing the rules **RES-ALPHA** and **SYMB-OPEN-ALPHA** from Table IX by the rules rules **RES** and **SYMB-OPEN** from Table VI. Thus, the α-conversion implicit in these rules has been removed.

As witnessed by Example 1, on $\pi^{\rho}_{ES}(\mathcal{N},\mathcal{R})$ the transition relation $\xrightarrow{\alpha}_\bullet$ differs essentially from $\xrightarrow{\alpha}$, and fails to derive some crucial transitions. However, I will show that restricted to the clash-free processes within $\pi^{\rho}_{ES}(\mathcal{N},\mathcal{R})$ this transition relation is just as suitable as $\xrightarrow{\alpha}$; up to strong barbed bisimilarity there is no difference. Since the only the clash-free processes in $\pi^{\rho}_{ES}(\mathcal{N},\mathcal{R})$ occur as the target of the previous translation step, the relation $\xrightarrow{\alpha}_\bullet$ may be used in the source of the next.

Lemma 73: If $P \xrightarrow{M\bar{x}(y)}_\bullet Q$ then $y \in {\rm RN}(P)$.

Proof. A trivial induction on the inference of $P \xrightarrow{M\bar{x}(y)}_\bullet Q$. ∎

Lemma 74: If R is clash-free and $R \xrightarrow{\alpha}_\bullet R'$ then $R \xrightarrow{\alpha} \equiv R'$.

Proof. By induction on the derivation of $R \xrightarrow{\alpha}_\bullet R'$.

- Suppose $R \xrightarrow{M\bar{x}(y)}_\bullet R'$ is derived by **SYMB-OPEN**. Then $R = (\nu y)P$, $P \xrightarrow{M\bar{x}y}_\bullet R'$, $y \neq x$ and $y \notin {\rm n}(M)$. By induction $P \xrightarrow{M\bar{x}y} U$ for a process $U \equiv R'$. So by **SYMB-OPEN-ALPHA** $R \xrightarrow{M\bar{x}(y)} U[\{y/\!/y\}]$. Moreover, $U[\{y/\!/y\}] \equiv U \equiv R'$ by Lemma 61(c,a).
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **RES**. Then $R = (\nu y)P$, $P \xrightarrow{\alpha}_\bullet P'$, $R' = (\nu y)P'$ and $y \notin {\rm n}(\alpha)$. For $u \in {\rm bn}(\alpha)$, by Lemma 73 $u \in {\rm RN}(P)$, so by the clash-freedom of P, $u \notin {\rm FN}(P)$. By induction $P \xrightarrow{\alpha} V$ for some $V \equiv P'$. So by **RELABELLING** $P[\{y/\!/y\}] \xrightarrow{\alpha} V[\{y/\!/y\}]$ and by rule **RES-ALPHA** $R \xrightarrow{\alpha} (\nu y)(V[\{y/\!/y\}])$. By Lemma 61(c,a,h) $(\nu y)(V[\{y/\!/y\}]) \equiv (\nu y)V \equiv (\nu y)P' = R'$
- All other cases are trivial with Lemma 61(g,h). ∎

Lemma 75: Let R be clash-free. If $R \xrightarrow{\alpha} R'$, ${\rm bn}(\alpha) = \emptyset$ and $\iota(\alpha) \cap {\rm RN}(R) = \emptyset$, then $R \xrightarrow{\alpha}_\bullet U$ for some U with $R' \equiv U$.

If $R \xrightarrow{M\bar{x}(z)} R'$ then $R \xrightarrow{M\bar{x}(y)}_\bullet U$ for some y and U with $R' \equiv U[\{z/\!/y\}]$ and $z \notin {\rm FN}(U) \backslash \{y\}$.

Proof. By induction on the depth of the derivation of $R \xrightarrow{\alpha} R'$.

- The cases that $R \xrightarrow{\alpha}_\bullet R'$ is derived by **TAU**, **OUTPUT**, **EARLY-INPUT**, **SUM**, **SYMB-MATCH** or **IDE** are trivial.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by **PAR**. Then $R = P|Q$, $P \xrightarrow{\alpha} P'$, ${\rm bn}(\alpha) \cap {\rm FN}(Q) = \emptyset$ and $R' = P'|Q$.
  First assume ${\rm bn}(\alpha) = \emptyset$. Since $\iota(\alpha) \cap {\rm RN}(R) = \emptyset$, also $\iota(\alpha) \cap {\rm RN}(P) = \emptyset$. By induction $P \xrightarrow{\alpha}_\bullet V$ for some V with $P' \equiv V$. By rule **PAR** $R \xrightarrow{\alpha}_\bullet V|Q$. Moreover, $R' = P'|Q \equiv V|Q$ by Lemma 61(g).
  Next assume $\alpha = M\bar{x}(z)$. So $z \notin {\rm FN}(R)$. By induction $P \xrightarrow{M\bar{x}(y)}_\bullet V$ for some y, V with $P' \equiv V[\{z/\!/y\}]$. By Lemma 73, $y \in {\rm RN}(P)$, so $y \notin {\rm FN}(R)$ by the clash-freedom of R. Hence $R \xrightarrow{M\bar{x}(y)}_\bullet V|Q$ by rule **PAR**. Moreover, $R' = P'|Q \equiv V[\{z/\!/y\}]|Q[\{z/\!/y\}] \equiv (V|Q)[\{z/\!/y\}]$ by Lemma 61(c,a,g,d), using that $y, z \notin {\rm FN}(Q)$. Finally, by Lemma 63, ${\rm FN}(V|Q) \backslash \{y\} \subseteq {\rm FN}(R) \not\ni z$.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by rule **E-S-COM**. Then $R = P|Q$, $\alpha = [x{=}v]MN\tau$, $P \xrightarrow{M\bar{x}y} P'$, $Q \xrightarrow{Mvy} Q'$ and $R' = P'|Q'$. By Lemma 62 $y \in {\rm FN}(P)$. So by the clash-freedom of R, $y \notin {\rm RN}(Q)$. By induction $P \xrightarrow{M\bar{x}y}_\bullet V$ and $Q \xrightarrow{Mvy}_\bullet W$ with $P \equiv V$ and $Q' \equiv W$. Hence by **E-S-COM** $R \xrightarrow{\alpha}_\bullet V|W$, and $R' = P'|Q' \equiv V|W$.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by rule **E-S-CLOSE**. Then $R = P|Q$, $\alpha = [x{=}v]MN\tau$, $P \xrightarrow{M\bar{x}(z)} P'$, $Q \xrightarrow{Mvz} Q'$, $z \notin {\rm FN}(R)$ and $R' = (\nu z)(P'|Q')$. By induction, one has $P \xrightarrow{M\bar{x}(y)}_\bullet V$ for some y and V with $P' \equiv V[\{z/\!/y\}]$ and $z \notin {\rm FN}(V) \backslash \{y\}$. By Lemma 73 $y \in {\rm RN}(P)$, so by the clash-freedom of R, $y \notin {\rm RN}(Q)$ and $y \notin {\rm FN}(R)$. By Lemma 65 $Q \xrightarrow{Mvy} W_y$ for some $W_y \equiv Q'[\{y/\!/z\}]$. So by induction $Q \xrightarrow{Mvy}_\bullet W$ for some $W \equiv W_y$. By **E-S-CLOSE** $R \xrightarrow{\alpha}_\bullet (\nu y)(V|W)$. Moreover,

$$\begin{aligned} R' = (\nu z)(P'|Q') &\equiv (\nu z)(V[\{z/\!/y\}] \mid Q'[\{z/\!/y\}][\{z/\!/y\}]) \\ &\equiv (\nu z)(V[\{z/\!/y\}] \mid W_y[\{z/\!/y\}]) \\ &\equiv (\nu z)((V|W)[\{z/\!/y\}]) \\ &\equiv (\nu y)(V|W) \end{aligned}$$

  by Lemma 61(h,a,b,c,h,d,f), in the last step using that $z \notin {\rm FN}(R) \supseteq {\rm FN}((\nu y)(V|W))$ by Lemma 63.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by **RELABELLING**. Then $R = P[\sigma]$, $P \xrightarrow{\beta} P'$, $\beta[\sigma] = \alpha$ and $R' = P'[\sigma]$.
  First assume ${\rm bn}(\alpha) = \emptyset$. Since $\iota(\alpha) \cap {\rm RN}(R) = \emptyset$, also $\iota(\beta) \cap {\rm RN}(P) = \emptyset$. By induction $P \xrightarrow{\beta}_\bullet V$ for some V with $P' \equiv V$. Now $R \xrightarrow{\alpha}_\bullet V[\sigma]$ by **RELABELLING**. Moreover, $R' = P'[\sigma] \equiv V[\sigma]$ by Lemma 61(h).
  Next assume $\alpha = M\bar{x}(z)$ and $\beta = K\bar{q}(w)$. So $w[\sigma] = z$. Then $z \notin {\rm FN}(R)$ by the side-condition of **RELABELLING**. By induction $P \xrightarrow{K\bar{q}(v)}_\bullet V$ for some v and V with $P' \equiv V[\{w/\!/v\}]$. Let $y := v[\sigma]$. By Lemma 73, $v \in {\rm RN}(P)$, so $y \in {\rm FN}(R)$. Hence $y \notin {\rm FN}(R)$ by the clash-freedom of R. So there is no $n \in {\rm FN}(P)$ with $n[\sigma] = y$ or $n[\sigma] = z$. Since ${\rm FN}(V) \subseteq {\rm FN}(P) \cup \{v\}$ by Lemma 63, the name v is the only possible $n \in {\rm FN}(V)$ with $n[\sigma] = y$ or $n[\sigma] = z$.

By rule **RELABELLING** $R \xrightarrow{M\bar{x}(y)}_\bullet V[\sigma]$. Lemma 61(h,b,a) yields $R' = P'[\sigma] \equiv V[\{w/\!\!/v\}][\sigma] \equiv V[\sigma][\{z/\!\!/y\}]$. Finally, by Lemma 63, $\text{FN}(V[\sigma])\backslash\{y\} \subseteq \text{FN}(R) \not\ni z$.

- Suppose $R \xrightarrow{M\bar{x}(z)} R'$ is derived by **SYMB-OPEN-ALPHA**. Then $R = (\nu y)P$, $P \xrightarrow{M\bar{x}y} P'$, $R' = P'[\{z/\!\!/y\}]$, $y \neq x$, $z \notin \text{FN}(R)$ and $y \notin \text{n}(M)$. By induction $P \xrightarrow{M\bar{x}y}_\bullet U$ for a process U with $P' \equiv U$. So by **SYMB-OPEN** $R \xrightarrow{M\bar{x}(y)}_\bullet U$. Moreover, $R' = P'[\{z/\!\!/y\}] \equiv U[\{z/\!\!/y\}]$ by Lemma 61(h). Finally, by Lemma 63, $\text{FN}(U)\backslash\{y\} \subseteq \text{FN}(R) \not\ni z$.
- Suppose $R \xrightarrow{\alpha} R'$ is derived by rule **RES-ALPHA**. Then $R = (\nu y)P$, $P[\{z/\!\!/y\}] \xrightarrow{\alpha} P'$, $R' = (\nu z)P'$ and $z \notin \text{FN}(R) \cup \text{n}(\alpha)$.
  First assume $\text{bn}(\alpha) = \emptyset$. As $\iota(\alpha) \cap \text{RN}(R) = \emptyset$, $y \notin \iota(\alpha)$ and $\iota(\alpha) \cap \text{RN}(P) = \emptyset$. Hence $y \notin \text{n}(\alpha)$ by Lemma 62. By **RELABELLING** $P \xrightarrow{\alpha} V$ for a V with $V[\{z/\!\!/y\}] = P'$. So by induction $P \xrightarrow{\alpha}_\bullet W$ for some $W \equiv V$. So by **RES** $R \xrightarrow{\alpha}_\bullet (\nu y)W$. By Lemma 63 $\text{FN}(V)\backslash\{y\} \subseteq \text{n}(\alpha) \cup \text{FN}(P)\backslash\{y\} = \text{n}(\alpha) \cup \text{FN}(R) \not\ni z$. Hence $R' = (\nu z)P' = (\nu z)(V[\{z/\!\!/y\}]) \equiv (\nu y)V$ by Lemma 61(f).
  Next assume $\alpha = M\bar{x}(w)$. By Lemma 62 $\text{n}(M) \cup \{x\} \subseteq \text{FN}(R) \not\ni y$. So by **RELABELLING** $P \xrightarrow{M\bar{x}(u)} W$ for a W with $W[\{z/\!\!/y\}] = P'$. Here $u := w[\{z/y\}]$. By induction $P \xrightarrow{M\bar{x}(v)}_\bullet V$ for some v and V with $W \equiv V[\{u/\!\!/v\}]$ and $w \notin \text{FN}(V)\backslash\{v\}$. By Lemma 73 $v \in \text{RN}(P)$. So $v \neq y$ by the clash-freedom of R. Hence $R \xrightarrow{M\bar{x}(v)}_\bullet (\nu y)V$ by **RES**. Pick a name $q \notin \text{FN}(R') \cup \text{FN}((\nu y)V) \cup \{w, v\}$. Now

$$\begin{aligned} R' = (\nu z)P' &= (\nu z)(W[\{z/\!\!/y\}]) \\ &\equiv (\nu q)(W[\{z/\!\!/y\}][\{q/\!\!/z\}]) \\ &\equiv (\nu q)(V[\{u/\!\!/v\}][\{z/\!\!/y\}][\{q/\!\!/z\}]) \\ &\equiv (\nu q)(V[\{q/\!\!/y\}][\{w/\!\!/v\}]) \\ &\equiv ((\nu q)(V[\{q/\!\!/y\}]))[\{w/\!\!/v\}] \\ &\equiv ((\nu y)V)[\{w/\!\!/v\}] \end{aligned}$$

  by Lemma 61(f,h,b,a,e,f,h), provided that

$$n[\{u/\!\!/v\}][\{z/\!\!/y\}][\{q/\!\!/z\}] = n[\{q/\!\!/y\}][\{w/\!\!/v\}] \qquad (21)$$

  for all $n \in \text{FN}(V)$. This has to be checked only for $n \in \{y, v\}$, because $q \notin \text{FN}(V)\backslash\{y\}$, $w \notin \text{FN}(V)\backslash\{v\}$, u is either w or z, and $z \notin \text{FN}(R) \supseteq \text{FN}(V)\backslash\{y, v\}$ by Lemma 63. To check (21) I recall that $y \neq v \neq q \neq w \neq z$ and consider two cases.
  - Let $w \neq y$. Then $u = w$, and (21) holds for $n = y, v$.
  - Let $w = y$. Then $u = z \neq y \neq q$ and again (21) holds for $n = y, v$.

  Finally, $w \notin \text{FN}((\nu y)V)\backslash\{v\}$. ■

Below a substitution σ is called *clash-free* on a $\pi^\rho_{ES}(\mathcal{N}, \mathcal{R})$ process P iff $\text{RN}(P) \cap (dom(\sigma) \cup range(\sigma)) = \emptyset$.

Observation 4: If P is clash-free and σ is clash-free on P, then $P[\sigma]$ is clash-free and $\text{RN}(P[\sigma]) = \text{RN}(P)$.

Lemma 76: If $x(y).P$ is clash-free and $z \notin \text{RN}(x(y).P)$ then substitution $\{z/y\}^{\mathcal{S}}$ from **EARLY-INPUT** is clash-free on P.

Proof. By definition $dom(\{z/y\}^{\mathcal{S}}) \subseteq \mathcal{Z}$ and $range(\{z/y\}^{\mathcal{S}}) \subseteq \mathcal{Z} \cup \{z\}$. Since $x(y).P$ is clash-free, so is process P by Definition 23(v), and $\text{RN}(P) = \text{RN}(x(y).P)$ by Definition 22. Hence $\text{RN}(P) \subseteq \mathcal{R}$ and $z \notin \text{RN}(x(y).P) = \text{RN}(P)$. ■

Lemma 77: If $R \xrightarrow{\alpha}_\bullet R'$ with $\iota(\alpha) \cap \text{RN}(R) = \emptyset$ and R is clash-free, then so is R'. Moreover, $\text{RN}(R') \subseteq \text{RN}(R)$ and $\text{bn}(\alpha) \cap \text{RN}(R') = \emptyset$.

Proof. By induction on the derivation of $R \xrightarrow{\alpha}_\bullet R'$. Since R is clash-free, $\text{RN}(R) \subseteq \mathcal{R}$.

- By Definition 23(iv-v) a subexpression P of a clash-free process R is also clash-free, and by Definition 22 $\text{RN}(P) \subseteq \text{RN}(R)$. Using this, the cases that $R \xrightarrow{\alpha}_\bullet R'$ is derived by **TAU**, **OUTPUT**, **SUM** or **SYMB-MATCH** are trivial.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **EARLY-INPUT**. Then $R = x(y).P$, $\alpha = xz$ with $z \notin \text{RN}(R)$, and $R' = P[\{z/y\}^{\mathcal{S}}]$. By Definition 23(v) P is clash-free and by Definition 22 $\text{RN}(P) = \text{RN}(R)$. By Lemma 76 and Observation 4 R' is clash-free and $\text{RN}(R') = \text{RN}(R)$. Moreover, $\text{bn}(\alpha) = \emptyset$.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **IDE**. Then $R = A(\vec{x})$. Let $A(\vec{x}) \stackrel{\text{def}}{=} P$. Then $P \xrightarrow{\alpha}_\bullet R'$. Process P is clash-free by Definition 23(vii), and $\text{RN}(P) = \text{RN}(R)$ by Definition 22. By induction $\text{RN}(R') \subseteq \text{RN}(P) = \text{RN}(R)$, process R' is clash-free, and $\text{bn}(\alpha) \cap \text{RN}(R') = \emptyset$.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **PAR**. Then $R = P|Q$, $P \xrightarrow{\alpha}_\bullet P'$ and $R' = P'|Q$. Since $\iota(\alpha) \cap \text{RN}(R) = \emptyset$ and $\text{RN}(P) \subseteq \text{RN}(R)$, also $\iota(\alpha) \cap \text{RN}(P) = \emptyset$. Using that P is clash-free, by induction P' is clash-free and $\text{RN}(P') \subseteq \text{RN}(P)$. Moreover, $\text{bn}(\alpha) \cap \text{RN}(P') = \emptyset$. Now $\text{RN}(R') = \text{RN}(P') \cup \text{RN}(Q) \subseteq \text{RN}(P) \cup \text{RN}(Q) = \text{RN}(R)$.

  Let $z \in \text{bn}(\alpha)$. Then $z \in \text{RN}(P)$ by Lemma 73, so by the clash-freedom of R one has $z \notin \text{RN}(Q)$. Moreover, $z \notin \text{RN}(P')$ by the above, so $z \notin \text{RN}(R')$.

  To show that R' is clash-free, using that P' and Q are clash-free, it suffices to show (i) $\text{FN}(R') \cap \text{RN}(R') = \emptyset$ and (ii) $\text{RN}(P') \cap \text{RN}(Q) = \emptyset$. The latter follows since $\text{RN}(P) \cap \text{RN}(Q) = \emptyset$ by the clash-freedom of R. For the former, Lemma 63 yields $\text{FN}(R') \subseteq \text{FN}(R) \cup \iota(\alpha) \cup \text{bn}(\alpha)$. Moreover $\text{FN}(R) \cap \text{RN}(R) = \emptyset$ by the clash-freedom of R, $\iota(\alpha) \cap \text{RN}(R) = \emptyset$ by assumption, and $\text{bn}(\alpha) \cap \text{RN}(R') = \emptyset$ as derived above. This entails (i).
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by rule **E-S-COM**. Then $R = P|Q$, $\alpha = [x{=}v]MN\tau$, $P \xrightarrow{M\bar{x}y}_\bullet P'$, $Q \xrightarrow{Mvy}_\bullet Q'$ and $R' = P'|Q'$. By Lemma 62 $y \in \text{FN}(P)$, so $y \notin \text{RN}(Q)$ by the clash-freedom of R. Using that P and Q are clash-free, by induction P' and Q' are clash-free, $\text{RN}(P') \subseteq \text{RN}(P)$ and $\text{RN}(Q') \subseteq \text{RN}(Q)$. It follows that $\text{RN}(R') = \text{RN}(P') \cup \text{RN}(Q') \subseteq \text{RN}(P) \cup \text{RN}(Q) = \text{RN}(R)$.

  To show that R' is clash-free, using that P' and Q' are clash-free, it suffices to show (i) $\text{FN}(R') \cap \text{RN}(R') = \emptyset$ and (ii) $\text{RN}(P') \cap \text{RN}(Q') = \emptyset$. The latter follows since $\text{RN}(P) \cap \text{RN}(Q) = \emptyset$ by the clash-freedom of R. For the former, Lemma 63 yields $\text{FN}(R') \subseteq \text{FN}(R)$. As $\text{FN}(R) \cap \text{RN}(R) = \emptyset$ by the clash-freedom of R, this entails (i).
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by rule **E-S-CLOSE**. Then $R = P|Q$, $\alpha = [x{=}v]MN\tau$, $P \xrightarrow{M\bar{x}(z)}_\bullet P'$, $Q \xrightarrow{Mvz}_\bullet Q'$

and $R' = (\nu z)(P'|Q')$. By Lemma 73 $z \in \text{RN}(P)$, so $y \notin \text{RN}(Q)$ by the clash-freedom of R. Using that P and Q are clash-free, by induction P' and Q' are clash-free, $\text{RN}(P') \subseteq \text{RN}(P)$, $\text{RN}(Q') \subseteq \text{RN}(Q)$ and $z \notin \text{RN}(P')$. It follows that $\text{RN}(R') = \text{RN}(P') \cup \text{RN}(Q') \cup \{z\} \subseteq \text{RN}(P) \cup \text{RN}(Q) = \text{RN}(R)$.

To show that R' is clash-free, using that P' and Q' are clash-free, it suffices to show (i) $\text{FN}(R') \cap \text{RN}(R') = \emptyset$, (ii) $\text{RN}(P') \cap \text{RN}(Q') = \emptyset$ and (iii) $z \notin \text{RN}(P'|Q')$. That (i) and (ii) hold follows exactly as in the case of **E-S-COM** above. For (iii), using that $z \in \text{RN}(P)$ and $z \notin \text{RN}(P')$, in case $z \in \text{RN}(P'|Q')$ then $z \in \text{RN}(Q') \subseteq \text{RN}(Q)$, contradicting the clash-freedom of R.

- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **RES**. Then $R = (\nu y)P$, $P \xrightarrow{\alpha}_\bullet P'$, $y \notin \text{n}(\alpha)$ and $R' = (\nu y)P'$. Since $\iota(\alpha) \cap \text{RN}(R) = \emptyset$ and $\text{RN}(P) \subseteq \text{RN}(R)$, also $\iota(\alpha) \cap \text{RN}(P) = \emptyset$. Using that P is clash-free, by induction P' is clash-free and $\text{RN}(P') \subseteq \text{RN}(P)$. Moreover, $\text{bn}(\alpha) \cap \text{RN}(P') = \emptyset$. Now $\text{RN}(R') = \text{RN}(P') \cup \{y\} \subseteq \text{RN}(P) \cup \{y\} = \text{RN}(R)$.

  As $\text{bn}(\alpha) \cap \text{RN}(P') = \emptyset$ and $y \notin \text{n}(\alpha)$, $\text{bn}(\alpha) \cap \text{RN}(R') = \emptyset$.

  To show that R' is clash-free, using that P' is clash-free, it suffices to show (i) $\text{FN}(R') \cap \text{RN}(R') = \emptyset$ and (iii) $y \notin \text{RN}(P')$. The latter follows since $y \notin \text{RN}(P)$ by the clash-freedom of R. The former follows exactly as in the case of **PAR**.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by rule **SYMB-OPEN**. Then $R = (\nu y)P$, $\alpha = M\bar{x}(y)$, $P \xrightarrow{M\bar{x}y}_\bullet R'$, $y \neq x$ and $y \notin \text{n}(M)$. Using that P is clash-free, by induction R' is clash-free and $\text{RN}(R') \subseteq \text{RN}(P) \subseteq \text{RN}(R)$. Since R is clash-free, $y \notin \text{RN}(P) \supseteq \text{RN}(R')$, so $\text{bn}(\alpha) \cap \text{RN}(R') = \emptyset$.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by rule **RELABELLING**. Then $R = P[\sigma]$, $P \xrightarrow{\beta}_\bullet P'$, $\beta[\sigma] = \alpha$, $\text{bn}(\alpha) \cap \text{FN}(R) = \emptyset$ and $R' = P'[\sigma]$. Since $\iota(\alpha) \cap \text{RN}(R) = \emptyset$, also $\iota(\alpha) \cap \text{RN}(P) = \emptyset$. Using that P is clash-free, by induction P' is clash-free, $\text{RN}(P') \subseteq \text{RN}(P)$ and $\text{bn}(\alpha) \cap \text{RN}(P') = \emptyset$. It follows that $\text{RN}(R') \subseteq \text{RN}(R)$.

  Suppose $z \in \text{bn}(\alpha) \cap \text{RN}(R')$. Then there are $v \in \text{bn}(\beta)$ and $w \in \text{RN}(P')$ with $v[\sigma] = w[\sigma]$. As $\text{bn}(\alpha) \cap \text{RN}(P') = \emptyset$ one has $v \neq w$,. By Lemma 73 $v \in \text{RN}(P)$. Moreover, $w \in \text{RN}(P)$. This contradicts Condition (iv) of the clash-freedom of R.

  To show that R' is clash-free, using that P' is clash-free, it suffices to show (i) $\text{FN}(R') \cap \text{RN}(R') = \emptyset$ and (iv) $y[\sigma] \neq z[\sigma]$ for different $y, z \in \text{RN}(P')$. The latter follows from the same condition for $\text{RN}(P)$. The former follows exactly as in the case of **PAR**. ∎

Let $\mathscr{T}_\bullet$ be the identity translation from the clash-free processes in $\pi^\rho_{ES}(\mathcal{N}, \mathcal{R})$ equipped with the standard transition relation $\xrightarrow{\alpha}$ to clash-free processes in $\pi^\rho_{ES}(\mathcal{N}, \mathcal{R})$ equipped with the alternative transition relation $\xrightarrow{\alpha}_\bullet$. The results above imply that $\mathscr{T}_\bullet$ is valid up to strong barbed bisimilarity.

Theorem 14: $\mathscr{T}_\bullet(P) \mathrel{\dot{\leftrightarrow}} P$ for any clash-free $P \in \text{T}_{\pi^\rho_{ES}(\mathcal{N},\mathcal{R})}$.

Proof. It suffices to show that the symmetric closure of

$$\mathscr{R} := \{(U, R) \mid U, R \in \text{T}_{\pi^\rho_{ES}(\mathcal{N},\mathcal{R})} \wedge U \equiv R \wedge R \text{ clash-free}\}$$

is a strong barbed bisimulation. Take $(R, U) \in \mathscr{R}$.

Let $U \xrightarrow{\tau} U'$. By Lemma 60 $R \xrightarrow{\tau} R^\dagger$ for some $R^\dagger \equiv U'$. So $R \xrightarrow{\tau}_\bullet R'$ for some $R' \equiv R^\dagger$ by Lemma 75. By Lemma 77 R' is clash-free. Hence $(U', R') \in \mathscr{R}$.

Let $R \xrightarrow{\tau}_\bullet R'$. Then $R \xrightarrow{\tau} R^\dagger$ for some $R^\dagger \equiv R'$ by Lemma 74. By Lemma 60 $U \xrightarrow{\tau} U'$ for some $U' \equiv R^\dagger$. By Lemma 77 R' is clash-free. Hence $(U', R') \in \mathscr{R}$.

Let $U{\downarrow_b}$ with $b \in \mathcal{Z} \cup \overline{\mathcal{Z}}$. Then $U \xrightarrow{by} U'$ or $U \xrightarrow{b(y)} U'$ for some y and U', using the definition of O in Section IV. By Lemmas 66 and 64 I may assume, without loss of generality, that $y \notin \text{RN}(R)$ in case of an input action by, and $y \notin \text{FN}(R)$ in case of a bound output action $b(y)$. Hence $R \xrightarrow{by}$ or $R \xrightarrow{b(y)}$ by Lemma 60 and $R \xrightarrow{by}_\bullet$ or $R \xrightarrow{b(y)}_\bullet$ by Lemma 75. Thus $R{\downarrow_b}$.

The implication $R{\downarrow_b} \Rightarrow U{\downarrow_b}$ proceeds likewise. ∎

I. Eliminating restriction

Let $\pi^\dagger_{ES}(\mathcal{N}, \mathcal{R})$ be the variant of $\pi^\rho_{ES}(\mathcal{N}, \mathcal{R})$ without restriction operators. Hence there is no need for rules **RES** and **SYMB-OPEN**. Since the resulting semantics cannot generate transitions labelled $M\bar{x}(z)$, rule **E-S-CLOSE** can be dropped as well. Moreover, $\text{bn}(\alpha) = \emptyset$ for all transition labels α. So the side condition of rules **PAR** and **RELABELLING** can be dropped too. Hence one is left with Table IX without the orange part.

In $\pi^\dagger_{ES}(\mathcal{N}, \mathcal{R})$ I also drop the restriction that $\text{fn}(P) \subseteq \{x_1, \ldots, x_n\}$ in defining equations $A(\vec{x}) \stackrel{\text{def}}{=} P$. Thus $A(\vec{x})$ can just as well be denoted A.[17] On $\pi^\dagger_{ES}(\mathcal{N}, \mathcal{R})$ I don't use FN.

The fifth step $\mathscr{T}_\nu$ of my translation goes from $\pi^\rho_{ES}(\mathcal{N}, \mathcal{R})$ to $\pi^\dagger_{ES}(\mathcal{N}, \mathcal{R})$. It simply drops all restriction operators. It is defined compositionally by $\mathscr{T}_\nu((\nu y)P) = \mathscr{T}_\nu(P)$ and $\mathscr{T}_\nu(A(\vec{x})) = A_\nu(\vec{x})$, where A_ν is a fresh agent identifier with defining equation $A_\nu(\vec{x}) \stackrel{\text{def}}{=} \mathscr{T}_\nu(P)$ when $A(\vec{x}) \stackrel{\text{def}}{=} P$ was the defining equation of A; the translation $\mathscr{T}_\nu$ acts homomorphically on all other constructs.

Although this translation in not valid in general, I proceed to prove its validity for clash-free processes. By Theorem 14 I may use the transition relation $\xrightarrow{\alpha}_\bullet$ on $\pi^\rho_{ES}(\mathcal{N}, \mathcal{R})$.

For α an action in $\pi^\rho_{ES}(\mathcal{N}, \mathcal{R})$ one defines the *debinding* of α by $\langle\alpha\rangle := \alpha$ if α has the form $M\tau$, Mxz or $M\bar{x}y$, and $\langle M\bar{x}(y)\rangle := M\bar{x}y$.

Lemma 78: If $R \xrightarrow{\alpha}_\bullet R'$, then $\mathscr{T}_\nu(R) \xrightarrow{\langle\alpha\rangle} \mathscr{T}_\nu(R')$.

Proof. By induction on the derivation of $R \xrightarrow{\alpha}_\bullet R'$.

- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by rule **EARLY-INPUT**. Then $R = x(y).P$, $\alpha = xz$, $R' = P[\{z/y\}^{\mathcal{S}}]$ and $\mathscr{T}_\nu(R) = x(y).\mathscr{T}_\nu(P)$. Moreover, $\mathscr{T}_\nu(R) \xrightarrow{\langle\alpha\rangle} \mathscr{T}_\nu(P)[\{z/y\}^{\mathcal{S}}] = \mathscr{T}_\nu(P[\{z/y\}^{\mathcal{S}}])$.
- The cases that $R \xrightarrow{\alpha}_\bullet R'$ is derived by **TAU** or **OUTPUT** are even more trivial.

[17] Here I assume that all sets $\mathcal{K}_n$ are disjoint, i.e., the same π-calculus agent identifier A does occur with multiple arities. When this assumption is not met, an arity-index at the $\pi^\dagger_{ES}(\mathcal{N}, \mathcal{R})$ identifier A is needed.

- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **IDE**. Then $R = A(\vec{x})$, say with $A(\vec{x}) \stackrel{\text{def}}{=} P$, and therefore $P \xrightarrow{\alpha}_\bullet R'$. Moreover $\mathscr{T}_\nu(R) = A_\nu(\vec{x})$, with A_ν defined by $A_\nu(\vec{x}) \stackrel{\text{def}}{=} \mathscr{T}_\nu(P)$. Now $\mathscr{T}_\nu(P) \xrightarrow{\langle\alpha\rangle} \mathscr{T}_\nu(R')$ by induction. With rule **IDE** one infers $\mathscr{T}_\nu(R) \xrightarrow{\langle\alpha\rangle} \mathscr{T}_\nu(R')$.
- The cases that $R \xrightarrow{\alpha}_\bullet R'$ is derived by **SUM**, **SYMB-MATCH**, **PAR** or **E-S-COM** are trivial.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **E-S-CLOSE**. Then $R = P|Q$, $\alpha = [x{=}v]M N\tau$, $P \xrightarrow{M\bar{x}(z)}_\bullet P'$, $Q \xrightarrow{Nvz}_\bullet Q'$, $R' = (\nu z)(P'|Q')$ and $\mathscr{T}_\nu(R) = \mathscr{T}_\nu(P)|\mathscr{T}_\nu(Q)$. Now $\mathscr{T}_\nu(P) \xrightarrow{M\bar{x}z} \mathscr{T}_\nu(P')$ and $\mathscr{T}_\nu(Q) \xrightarrow{Nvz} \mathscr{T}_\nu(Q')$ by induction. Thus $\mathscr{T}_\nu(R) \xrightarrow{\langle\alpha\rangle} \mathscr{T}_\nu(P')|\mathscr{T}_\nu(Q') = \mathscr{T}_\nu(R')$ by application of rule **E-S-COM**.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **RES**. Then $R = (\nu y)P$, $R' = (\nu y)P'$, $P \xrightarrow{\alpha}_\bullet P'$. Now $\mathscr{T}_\nu(P) \xrightarrow{\langle\alpha\rangle} \mathscr{T}_\nu(P')$ by induction. So $\mathscr{T}_\nu(R) = \mathscr{T}_\nu(P) \xrightarrow{\langle\alpha\rangle} \mathscr{T}_\nu(P') = \mathscr{T}_\nu(R')$.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **SYMB-OPEN**. Then $R = (\nu y)P$, $\alpha = M\bar{x}(y)$ and $P \xrightarrow{M\bar{x}y}_\bullet R'$. By induction $\mathscr{T}_\nu(P) \xrightarrow{M\bar{x}y} \mathscr{T}_\nu(R')$. So $\mathscr{T}_\nu(R) = \mathscr{T}_\nu(P) \xrightarrow{\langle\alpha\rangle} \mathscr{T}_\nu(R')$.
- Suppose $R \xrightarrow{\alpha}_\bullet R'$ is derived by **RELABELLING**. Then $R = P[\sigma]$, $P \xrightarrow{\beta}_\bullet P'$, $\beta[\sigma] = \alpha$ and $R' = P'[\sigma]$. By induction $\mathscr{T}_\nu(P) \xrightarrow{\langle\beta\rangle} \mathscr{T}_\nu(P')$. Hence $\mathscr{T}_\nu(R) = \mathscr{T}_\nu(P)[\sigma] \xrightarrow{\langle\alpha\rangle} \mathscr{T}_\nu(P')[\sigma] = \mathscr{T}_\nu(R')$. ■

The next lemma makes use of the set $\mathrm{no}(\beta)$ of *non-output* names of an action β in $\pi^\dagger_{ES}(\mathcal{N},\mathcal{R})$. Here $\mathrm{no}(\beta) := \mathrm{n}(\beta)$ if β has the form $M\tau$ or Mxz, whereas $\mathrm{no}(M\bar{x}y) := \mathrm{n}(M)\cup\{x\}$.

Lemma 79: If R is clash-free and $\mathscr{T}_\nu(R) \xrightarrow{\beta} U$, where $\mathrm{no}(\beta)\cap \mathrm{RN}(R) = \emptyset$, then $R \xrightarrow{\alpha}_\bullet R'$ for some α and R' with $\langle\alpha\rangle = \beta$ and $\mathscr{T}_\nu(R') = U$.

Proof. By induction on the derivation of $\mathscr{T}_\nu(R) \xrightarrow{\beta} U$, and a nested structural induction on R.

- The cases $R = \tau.P$ and $R = \bar{x}y.P$ are trivial.
- Let $R = x(y).P$. Then $\mathscr{T}_\nu(R) = x(y).\mathscr{T}_\nu(P)$. Hence $\beta = xz$ and $U = \mathscr{T}_\nu(P)[\{z/y\}^{\mathcal{S}}]$. Furthermore, $R \xrightarrow{xz}_\bullet P[\{z/y\}^{\mathcal{S}}]$ and $\mathscr{T}_\nu(P[\{z/y\}^{\mathcal{S}}) = \mathscr{T}_\nu(P)[\{z/y\}^{\mathcal{S}}] = U$.
- Let $R = A(\vec{x})$ with $A(\vec{x}) \stackrel{\text{def}}{=} P$. Then $\mathscr{T}_\nu(R) = A_\nu(\vec{x})$ with $A_\nu(\vec{x}) \stackrel{\text{def}}{=} \mathscr{T}_\nu(P)$. So $\mathscr{T}_\nu(P) \xrightarrow{\beta} U$. Process P is clash-free by Definition 23(vii), and $\mathrm{RN}(P) = \mathrm{RN}(R)$ by Definition 22. So $\mathrm{no}(\beta)\cap\mathrm{RN}(P[\{\vec{y}/\vec{z}\}^{\mathcal{S}}]) = \emptyset$. Thus, by induction, $P \xrightarrow{\alpha}_\bullet R'$ for some α and R' with $\langle\alpha\rangle = \beta$ and $\mathscr{T}_\nu(R') = U$. Now $R \xrightarrow{\alpha}_\bullet R'$ by **IDE**.
- The cases that R is $\mathbf{0}$, $P+Q$ or $[x{=}y]P$ are trivial.
- Let $R = P[\sigma]$. Then $\mathscr{T}_\nu(R) = \mathscr{T}(P)[\sigma]$ and $\mathscr{T}(P) \xrightarrow{\delta} V$ for some δ with $\delta[\sigma] = \beta$, and some V with $V[\sigma] = U$. By definition P is clash-free. Considering that $\mathrm{RN}(R) \supseteq \{y[\sigma] \mid y \in \mathrm{RN}(P)\}$, one has $\mathrm{no}(\delta)\cap\mathrm{RN}(P) = \emptyset$. So by induction $P \xrightarrow{\gamma}_\bullet P'$ for some γ and P' with $\langle\gamma\rangle = \delta$ and $\mathscr{T}_\nu(P') = V$. Let $\alpha := \gamma[\sigma]$. Then $\langle\alpha\rangle = \langle\gamma[\sigma]\rangle = \langle\gamma\rangle[\sigma] = \delta[\sigma] = \beta$. In case $z \in \mathrm{bn}(\alpha)$ then $z = w[\sigma]$ with $w \in \mathrm{fn}(\gamma)$, so $w \in \mathrm{RN}(P)$ by Lemma 73 and hence $z \in \mathrm{RN}(R)$, so the clash-freedom of R implies $z \notin \mathrm{FN}(R)$. Therefore, by **RELABELLING**, $R = P[\sigma] \xrightarrow{\alpha}_\bullet P'[\sigma]$, and $\mathscr{T}_\nu(P'[\sigma]) = \mathscr{T}_\nu(P')[\sigma] = V[\sigma] = U$.
- Let $R = P|Q$. Then $\mathscr{T}_\nu(R) = \mathscr{T}_\nu(P)|\mathscr{T}_\nu(Q)$. Suppose $\mathscr{T}_\nu(R) \xrightarrow{\beta} U$ is derived by **PAR**. Then $\mathscr{T}_\nu(P) \xrightarrow{\beta} V$ and $U = V|\mathscr{T}_\nu(Q)$. By definition P is clash-free. Since $\mathrm{RN}(P) \subseteq \mathrm{RN}(P|Q)$, $\mathrm{no}(\beta)\cap\mathrm{RN}(P) = \emptyset$. So by induction $P \xrightarrow{\alpha}_\bullet P'$ for some α and P' with $\langle\alpha\rangle{=}\beta$ and $\mathscr{T}_\nu(P') = V$. By Lemma 73 $\mathrm{bn}(\alpha) \subseteq \mathrm{RN}(P) \subseteq \mathrm{RN}(R)$, so $\mathrm{bn}(\alpha)\cap\mathrm{FN}(R) = \emptyset$ by the clash-freedom of R. Thus $R \xrightarrow{\alpha}_\bullet P'|Q$ by **PAR**, and $\mathscr{T}_\nu(P'|Q) = V|\mathscr{T}_\nu(Q) = U$.
  Now suppose $\mathscr{T}_\nu(R) \xrightarrow{\beta} U$ is derived by **E-S-COM**. Then $\beta = [x{=}v]M N\tau$, $\mathscr{T}_\nu(P) \xrightarrow{M\bar{x}y} V$, $\mathscr{T}_\nu(Q) \xrightarrow{Nvy} W$ and $U = V|W$. By definition P and Q are clash-free. Since $\mathrm{RN}(P) \subseteq \mathrm{RN}(P|Q)$, $\mathrm{no}(M\bar{x}y)\cap\mathrm{RN}(P) = \emptyset$. So by induction either $P \xrightarrow{M\bar{x}y}_\bullet P'$ or $P \xrightarrow{M\bar{x}(y)}_\bullet P'$ for some P' with $\mathscr{T}_\nu(P') = V$.
  In the first case $y \in \mathrm{FN}(P) \subseteq \mathrm{FN}(R)$ by Lemma 62, so $y \notin \mathrm{RN}(R) \supseteq \mathrm{RN}(Q)$ by the clash-freedom of R. Hence also $\mathrm{no}(Nvy)\cap\mathrm{RN}(Q) = \emptyset$. By induction $Q \xrightarrow{Nvy}_\bullet Q'$ for some Q' with $\mathscr{T}_\nu(Q') = W$. So $R \xrightarrow{\alpha}_\bullet P'|Q'$ by **E-S-COM**, and $\mathscr{T}_\nu(P'|Q') = V|W = U$.
  In the second case $y \in \mathrm{RN}(P) \subseteq \mathrm{RN}(R)$ by Lemma 73, so $y \notin \mathrm{FN}(R)\cup\mathrm{RN}(Q)$ by the clash-freedom of R. Hence $\mathrm{no}(Nvy)\cap\mathrm{RN}(Q) = \emptyset$. By induction $Q \xrightarrow{Nvy}_\bullet Q'$ for some Q' with $\mathscr{T}_\nu(Q') = W$. So $R \xrightarrow{\alpha}_\bullet (\nu y)(P'|Q')$ by **E-S-CLOSE**, and $\mathscr{T}_\nu((\nu y)(P'|Q')) = \mathscr{T}_\nu(P'|Q') = V|W = U$.
- Finally, let $R = (\nu y)P$. Then $\mathscr{T}_\nu(R) = \mathscr{T}_\nu(P)$. Moreover, $\mathrm{no}(\beta)\cap\mathrm{RN}(P) = \emptyset$. By induction, $P \xrightarrow{\alpha}_\bullet P'$ for some α and P' with $\langle\alpha\rangle = \beta$ and $\mathscr{T}_\nu(P') = U$.
  In case $y \notin \mathrm{n}(\alpha)$, $R \xrightarrow{\alpha}_\bullet (\nu y)P'$ by **RES**. Moreover, $\mathscr{T}_\nu((\nu y)P') = \mathscr{T}_\nu(P') = U$.
  In case $y \in \mathrm{n}(\alpha) = \mathrm{n}(\beta)$, using that $\mathrm{RN}(R) \ni y \notin \mathrm{no}(\beta)$, β must have the form $M\bar{x}y$ with $y \neq x$ and $y \notin \mathrm{n}(M)$. So α is either $M\bar{x}(y)$ or $M\bar{x}y$. If $\alpha = M\bar{x}(y)$ then $y \in \mathrm{RN}(P)$ by Lemma 73, contradicting the clash-freedom of R. So $\alpha = M\bar{x}y$. Now $R \xrightarrow{M\bar{x}(y)}_\bullet P'$ by **SYMB-OPEN**. ■

Theorem 15: If P is clash-free then $\mathscr{T}_\nu(P) \mathrel{\dot{\sim}} P$.

Proof. It suffices to show that the symmetric closure of

$$\mathscr{R} := \{(R, \mathscr{T}_\nu(R)) \mid R \text{ in } \pi^\rho_{ES}(\mathcal{N},\mathcal{R}) \text{ is clash-free}\}.$$

is a strong barbed bisimulation. So suppose R is clash-free.

Let $R \xrightarrow{\tau}_\bullet R'$. Then $\mathscr{T}_\nu(R) \xrightarrow{\tau} \mathscr{T}_\nu(R')$ by Lemma 78. Moreover, R' is clash-free by Lemma 77, so $R' \mathscr{R} \mathscr{T}_\nu(R')$.

Let $\mathscr{T}_\nu(R) \xrightarrow{\tau} U'$. Then, by Lemma 79, $R \xrightarrow{\tau}_\bullet R'$ for some R' with $\mathscr{T}_\nu(R') = U'$. Moreover, R' is clash-free by Lemma 77, so $R' \mathscr{R} U'$.

Now let $R{\downarrow_b}$ with $b \in \mathcal{Z}\cup\overline{\mathcal{Z}}$. Then $R \xrightarrow{by}_\bullet$ or $R \xrightarrow{b(y)}_\bullet$ for some y, using the definition of O in Section IV. So $\mathscr{T}_\nu(R) \xrightarrow{by}$ by Lemma 78. Thus $\mathscr{T}_\nu(R){\downarrow_b}$.

Finally, let $\mathscr{T}_\nu(R){\downarrow_b}$ with $b = x \in \mathcal{Z}$ or $b = \bar{x}$ with $x \in \overline{\mathcal{Z}}$. Then $\mathscr{T}_\nu(R) \xrightarrow{by} U'$ for some y and U'. Since R is clash-free, $\mathrm{RN}(R) \subseteq \mathcal{R}$, so $x \notin \mathrm{RN}(P)$. By Lemma 66 I may assume that if $b = x$ then $y \notin \mathrm{RN}(R)$. Hence $\mathrm{no}(by)\cap\mathrm{RN}(R) = \emptyset$. Thus, by Lemma 79, $R \xrightarrow{by}_\bullet R'$ or $R \xrightarrow{b(y)}_\bullet R'$ for some R'. Hence $R{\downarrow_b}$. ■

J. The last step

The language $\pi^{\dagger}_{ES}(\mathcal{N},\mathcal{R})$ can almost be recognised as an instance of $\text{CCS}^{\text{trig}}_{\gamma}$. Let Act be the set of all actions $M\tau$, $M\bar{x}y$ and Mxy with names from $\mathcal{H}$. As parameters of $\text{CCS}^{\text{trig}}_{\gamma}$ I take $\mathcal{K}$ to be the disjoint union of all the sets $\mathcal{K}_n$ for $n \in \mathbb{N}$, of n-ary agent identifiers from the chosen instance of the π-calculus, and $\mathscr{A} := Act\backslash\{\tau\}$. The set $\mathscr{S} \subseteq \mathscr{A}$ of synchronisations consists of all actions $M\tau$ with $M \neq \varepsilon$. The communication function $\gamma : (\mathscr{A}\backslash\mathscr{S})^2 \rightharpoonup \mathscr{S} \cup \{\tau\}$ is given by $\gamma(M\bar{x}y, Nvy) = [x{=}v]MN\tau$, and its commutative variant. Now the parallel composition of $\pi^{\dagger}_{ES}(\mathcal{N},\mathcal{R})$ turns out to be the same as for this instance of $\text{CCS}^{\text{trig}}_{\gamma}$. Likewise, the silent and output prefixes are instances of $\text{CCS}^{\text{trig}}_{\gamma}$ prefixing, and the agent identifiers of $\pi^{\dagger}_{ES}(\mathcal{N},\mathcal{R})$ are no different from $\text{CCS}^{\text{trig}}_{\gamma}$ agent identifiers. However, the input prefix of $\pi^{\dagger}_{ES}(\mathcal{N},\mathcal{R})$ does not occur in $\text{CCS}^{\text{trig}}_{\gamma}$. Yet, one can identify $Mx(y).P$ with $\sum_{z\in\mathcal{H}} Mxz.(P[\{z/y\}^{\mathcal{S}}])$, for both processes have the very same outgoing transitions. The $\pi^{\dagger}_{ES}(\mathcal{N},\mathcal{R})$ matching operator is no different from the triggering operator of MEIJE or $\text{CCS}^{\text{trig}}_{\gamma}$ (see Section XI): both rename only the first actions their argument process can perform, namely by adding a single match $[x{=}y]$ in front of each of them—this match is suppressed when $x{=}y$.

This yields to the following translation from $\pi^{\dagger}_{ES}(\mathcal{N},\mathcal{R})$ to $\text{CCS}^{\text{trig}}_{\gamma}$:

$$\begin{array}{lcl}
\mathscr{T}_\gamma(\mathbf{0}) & := & \mathbf{0}\\
\mathscr{T}_\gamma(M\tau.P) & := & M\tau.\mathscr{T}_\gamma(P)\\
\mathscr{T}_\gamma(M\bar{x}y.P) & := & M\bar{x}y.\mathscr{T}_\gamma(P)\\
\mathscr{T}_\gamma(Mx(y).P) & := & \sum_{z\in\mathcal{H}} Mxz.\big(\mathscr{T}_\gamma(P)[\{z/y\}]\big)\\
\mathscr{T}_\gamma([x{=}y]P) & := & [x{=}y]{\Rightarrow}\mathscr{T}_\gamma(P)\\
\mathscr{T}_\gamma(P \mid Q) & := & \mathscr{T}_\gamma(P)\|\mathscr{T}_\gamma(Q)\\
\mathscr{T}_\gamma(P+Q) & := & \mathscr{T}_\gamma(P)+\mathscr{T}_\gamma(Q)\\
\mathscr{T}_\gamma(A) & := & A
\end{array}$$

where the $\text{CCS}^{\text{trig}}_{\gamma}$ defining equations $A = \mathscr{T}_\gamma(P)$ of agent identifiers A are inherited verbatim from $\pi^{\dagger}_{ES}(\mathcal{N},\mathcal{R})$.

Here the use of the triggering operator can be avoided by restricting attention to the π-calculus with implicit matching. For that language the clause for $\mathscr{T}_\gamma([x{=}y]P)$ can be dropped, at the expense of the addition of the blue Ms above, which are absent when dealing with the full π-calculus.

Theorem 16: $\mathscr{T}_\gamma(P) \leftrightarrows P$ for each $\pi^{\dagger}_{ES}(\mathcal{N},\mathcal{R})$ process P.

Proof. Trivial. ∎

Putting all steps of my translation from $\pi_L(\mathcal{N})$ to $\text{CCS}^{\text{trig}}_{\gamma}$ together, I obtain

$$\begin{array}{lcl}
\mathscr{T}(\mathbf{0}) & := & \mathbf{0}\\
\mathscr{T}(M\tau.P) & := & M\tau.\mathscr{T}(P)\\
\mathscr{T}(M\bar{x}y.P) & := & M\bar{x}y.\mathscr{T}(P)\\
\mathscr{T}(Mx(y).P) & := & \sum_{z\in\mathcal{H}} Mxz.\big(\mathscr{T}(P)[\{z/y\}^{\mathcal{S}}]\big)\\
\mathscr{T}((\nu y)P) & := & \mathscr{T}(P)[p_y]\\
\mathscr{T}([x{=}y]P) & := & [x{=}y]{\Rightarrow}\mathscr{T}(P)\\
\mathscr{T}(P \mid Q) & := & \mathscr{T}(P)[\ell] \parallel \mathscr{T}(Q)[r]\\
\mathscr{T}(P+Q) & := & \mathscr{T}(P)+\mathscr{T}(Q)\\
\mathscr{T}(A(\vec{y})) & := & A[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]
\end{array}$$

where the CCS_γ agent identifier A has the defining equation $A = \mathscr{T}(P)$ when $A(\vec{x}) \stackrel{\text{def}}{=} P$ was the defining equation of the $\pi_L(\mathcal{N})$ agent identifier A. Abbreviating $[\{z/y\}^{\mathcal{S}}]$ by $[z/y]$ and $[\{\vec{y}/\vec{x}\}^{\mathcal{S}}]$ by $[\vec{y}/\vec{x}]$, this is the translation presented in Section IX.