A behavioural theory for a π-calculus with preorders

Daniel Hirschkoff, Jean-Marie Madiot, Xian Xu

A behavioural theory for a π-calculus with preorders
(long version)

Daniel Hirschkoff[1], Jean-Marie Madiot[1], and Xian Xu[2]

[1] ENS Lyon, Université de Lyon, CNRS, INRIA, France,
[2] East China University of Science and Technology, China

Abstract. We study the behavioural theory of $\pi\mathtt{P}$, a π-calculus featuring restriction as the only binder. In contrast with calculi such as Fusions and Chi, reduction in $\pi\mathtt{P}$ generates a preorder on names rather than an equivalence relation. We present two characterisations of barbed congruence in $\pi\mathtt{P}$: the first is based on a compositional LTS, and the second is an axiomatisation. The results in this paper bring out basic properties of $\pi\mathtt{P}$, mostly related to the interplay between the restriction operator and the preorder on names.

1 Introduction

The π-calculus expresses mobility via name passing, and has two binders: the input prefix binds the value to be received, and restriction is used to delimit the scope of a private name. The study of Fusions [13], Chi [6], Explicit Fusions [16] and Solos [10] has shown that using restriction as the only binder is enough to express name passing. In such calculi (which, reusing a terminology from [8], we shall refer to as *fusion calculi*), the bound input prefix, $a(x).P$, is dropped in favour of free input, $ab.P$. This yields a pleasing symmetry between input and output prefixes; moreover, one can encode bound input in terms of free input as $(\nu x)ax.P$.

In [8], the analysis of capability types [14] for fusion calculi has shown that an important modification to existing fusion calculi is necessary in order to be able to define meaningful type systems, beyond the simple discipline of sorting. This has led to the introduction of $\pi\mathtt{P}$, a π-calculus with preorders. Like existing fusion calculi, $\pi\mathtt{P}$ features restriction as the only binder, and free input and output prefixes. The calculus differs however from existing name-passing calculi. The most important difference is that interaction does not have the effect of equating (or *fusing*) two names, but instead generates an *arc* process, according to the following reduction relation:

$$\overline{c}a.P \mid cb.Q \quad \longrightarrow \quad a\not b \mid P \mid Q \ .$$

The arc $a\not b$ expresses the fact that anything that can be done using name b can be done using a as well (but not the opposite): we say that a is *above* b. Arcs induce a preorder relation on names, which can evolve along reductions.

Arcs can modify interaction possibilities: in presence of a/b, a is above b, hence a process emitting on b can also make an output transition along channel a. In general, an output on channel c can interact with an input on d provided c and d are *joinable*, written $c \curlyvee d$, which means that there is some name that is above both c and d according to the preorder relation To formalise these observations, the operational semantics exploits *conditions*, which are either of the form $b \leqslant a$ (a is above b), or $a \curlyvee b$ (a and b are joinable).

[8] shows that capabilities make sense in a fusion calculus if preorders replace equivalence relations ('fusions'). Accordingly, [8] presents a type system with input/output types and subtyping for $\pi\mathtt{P}$. Name preorders have an impact on how processes express behaviours. It is necessary, beyond types, to understand the consequences of introducing this new construct, whose behaviour does not seem to be reducible to known situations. [8] defines barbed congruence, written $\simeq$, and presents some laws for $\simeq$, suggesting that the behavioural theory of $\pi\mathtt{P}$ differs from the one of other fusion calculi. As an illustration, consider the following law, which is valid in $\pi\mathtt{P}$ (and in the π-calculus):

$$\overline{a}(x).\overline{b}(y).(\overline{x} \mid y) \;\simeq\; \overline{a}(x).\overline{b}(y).(\overline{x}.y + y.\overline{x}) \;\; .$$

$\overline{x}$ (resp. y) stands for an output (resp. input) where the value being transmitted is irrelevant. In Fusions, unlike in the π-calculus, the process that creates successively two fresh names x and y cannot prevent the context from equating ("fusing") x and y; hence, in order for the equivalence to hold, it is necessary to add a third summand on the right, $[x = y]\tau$. In $\pi\mathtt{P}$ we could *program* the possibility for the context to equate x and y, by changing the continuation process $\overline{x} \mid y$. This example suggests that $\pi\mathtt{P}$ gives a better control on restricted names. (Note that a different approach is adopted in [3,4] to address this question.)

In order to gain a better understanding of such phenomena, the main purpose of the present work is to deepen the study of the behavioural theory of $\pi\mathtt{P}$, in an untyped setting. We define a structural Labelled Transition System (LTS) for $\pi\mathtt{P}$, and show that the induced notion of bisimilarity, written $\sim$, characterises $\simeq$ (Section 3). It can be noted that [8] presents a characterisation of barbed congruence, using an LTS that is rather ad hoc, because it is based on the definition of the reduction relation. Unlike the latter, the LTS we present here is *structural*. The LTS reveals interesting aspects of interaction in $\pi\mathtt{P}$.

An important observation is related to the interplay between arcs and the restriction operator. It is for instance possible for a process to react to an input offer on some channel, say c, without being actually able to perform an output on c. This is the case for $P_0 \triangleq (\nu a)(\overline{a}v \mid a/c)$, as $P_0 \mid cu$ reduces to v/u (P_0 could do an output on c if the arc a/c was replaced with c/a). This phenomenon leads to the addition of a new type of labels in the LTS, corresponding to what we call *protected actions*. As expected, protected actions correspond to observables in the reduction-based semantics supporting the definition of $\simeq$.

Arc processes do not have transitions, but, as mentioned above, they induce conditions involving names, which in turn influence the behaviour of processes. Accordingly, strong bisimilarity, $\sim$, not only tests transitions, but also has a clause to guarantee that related processes entail the same conditions. The LTS

includes a label $[\varphi]\tau$, expressing "conditional synchronisation". $[\varphi]\tau$ transitions have a special treatment in the definition of $\sim$.

We provide a second characterisation of barbed congruence, by presenting a set of laws that define an axiomatisation of $\simeq$ (Section 4). Algebraic laws help analysing the behaviour of the constructs of the calculus and their interplay.

We discuss the main ideas in the axiomatisation. In a rather expected way, the prefixes in the calculus correspond to the labels of the LTS. This means in particular that we include prefixes for protected inputs and outputs. One of the salient aspects of the axiomatisation is the existence of a *state* component in processes, corresponding to the preorder induced by arcs. To manipulate the state component of processes, several laws in the axiomatisation can be used to rewrite prefixes using arcs, which makes it possible to express all transitions induced by a prefixed process. Another set of laws express persistence of the state component in processes (the state can only be extended along computation).

Moreover, the restriction operator prevents the state from being globally shared in general: for instance, in process P_0 above, name a can be used instead of c, but is only known inside the scope of (νa). All in all, the handling of restriction in our axiomatisation requires more care than is usually the case, due to the necessity to express the "view" that subprocesses have on the preorder of names. As we show in Section 4, this has an influence on the overall structure of the presentation of the axiomatisation.

Beyond the LTS and the axiomatisation, which are the main contributions of this paper, we present two results that are related to our study of $\simeq$ in $\pi\mathtt{P}$:

- Protected actions can be viewed as *cocapabilities*, in the sense of capability types. We define an i/o type system with cocapabilities for $\pi\mathtt{P}$.
- We show that our approach for the axiomatisation of $\pi\mathtt{P}$ can be adapted to provide an axiomatisation of behavioural equivalence for the calculus of Explicit Fusions, which features a fusion construct for processes.

The paper describes our results and sketches the most important proofs. We refer to [9] for a more detailed presentation of the technical details. Related work is discussed along the paper, where it is relevant.

2 $\pi\mathtt{P}$: Reduction-Based Semantics

2.1 The Calculus: Preorders and Processes

We consider a countable set of names $a, b, c, \ldots, x, y, \ldots$, and define conditions (φ), extended names (α, β), prefixes (π) and processes (P, Q) as follows:

$$\varphi ::= a \leqslant b \;\big|\; a \curlyvee b \qquad \alpha, \beta ::= a \;\big|\; \{a\} \qquad \pi ::= \alpha(x) \;\big|\; \overline{\alpha}(x) \;\big|\; [\varphi]\tau$$
$$P, Q ::= P \mid Q \;\big|\; (\nu a)P \;\big|\; a/b \;\big|\; \textstyle\sum_{i \in I} \pi_i.P_i$$

In a sum process, we let I range over a finite set of integers. 0 is the inactive process, defined as the empty sum. We use S to range over *sum processes* of the form $\sum_i \pi_i.P_i$, and write $\pi.P \in S$ if $\pi.P$ is a summand of S. We sometimes

decompose sum processes using the binary sum operator, writing, e.g., $S_1 + S_2$ (in particular, $S + 0 = S$). We abbreviate $\pi.0$ as π, and write $\alpha(x).P$ simply as $\alpha.P$ when the transmitted name is not relevant, and similarly for $\overline{\alpha}$. (νa) binds a in the restricted process, and prefixes $\alpha(x)$ $\overline{\alpha}(x)$ bind x in the continuation process. The set of free names of P, $\mathsf{fn}(P)$, is defined in the usual way. $P\{b/a\}$ is the process obtained by substituting a with b in P, in a capture-avoiding way.

There are two forms of φ conditions: $a \leqslant b$ is read "b is above a" and $a \curlyvee b$ is read "a and b are joinable". In both cases, we have $\mathsf{n}(\varphi) = \{a, b\}$. We explain below how we extend relations $\leqslant$ and $\curlyvee$ to extended names. When $\mathsf{n}(\varphi) = \{a\}$, we say that φ is *reflexive* and we abbreviate $[\varphi]\tau$ as τ. Condition $b \leqslant a$ is ensured by the arc process $a\!\!/\!\!b$.

In a prefix $\alpha(x)$ or $\overline{\alpha}(x)$, we say that extended name α is in subject position, while x is in object position. As discussed in Section 1, extended names include *protected names*, of the form $\{a\}$, which can be used in subject position only. A prefix of the form $[\varphi]\tau$ is called a *conditional* τ, while other prefixes are called *visible*. Bound and free names for prefixes are given by: $\mathsf{bn}([\varphi]\tau) = \emptyset$ and $\mathsf{bn}(\alpha(x)) = \mathsf{bn}(\overline{\alpha}(x)) = \{x\}$, $\mathsf{fn}([\varphi]\tau) = \mathsf{n}(\varphi)$, $\mathsf{fn}(\alpha(x)) = \mathsf{fn}(\overline{\alpha}(x)) = \mathsf{n}(\alpha)$ with $\mathsf{n}(a) = \mathsf{n}(\{a\}) = \{a\}$.

We use an overloaded notation, and define processes representing conditions:

$$a \curlyvee b \;\triangleq\; (\nu u)(u\!\!/\!\!a \mid u\!\!/\!\!b) \qquad\qquad a \leqslant b \;\triangleq\; b\!\!/\!\!a \;\;.$$

Γ ranges over sets of at most two conditions. We define $\Gamma \vdash \varphi$, meaning that Γ implies φ, and $P \rhd \Gamma$, meaning that P entails φ for all $\varphi \in \Gamma$:

$$\vdash\text{-REFL}\ \frac{}{\Gamma \vdash a \leqslant a} \qquad \vdash\text{-IN}\ \frac{\varphi \in \Gamma}{\Gamma \vdash \varphi} \qquad \vdash\text{-MIRROR}\ \frac{\Gamma \vdash b \curlyvee a}{\Gamma \vdash a \curlyvee b} \qquad \vdash\text{-TRANS}\ \frac{\Gamma \vdash a \leqslant b \quad \Gamma \vdash b \leqslant c}{\Gamma \vdash a \leqslant c} \qquad \vdash\text{-JOIN}\ \frac{\Gamma \vdash a \leqslant b \quad \Gamma \vdash c \leqslant b}{\Gamma \vdash a \curlyvee c} \qquad \vdash\text{-EXTJOIN}\ \frac{\Gamma \vdash a \leqslant b \quad \Gamma \vdash b \curlyvee c}{\Gamma \vdash a \curlyvee c}$$

$$\rhd\text{-ARC}\ \frac{}{a\!\!/\!\!b \rhd b \leqslant a} \qquad \rhd\text{-COMBINE}\ \frac{P \rhd \Gamma \quad \Gamma \vdash \varphi}{P \rhd \varphi} \qquad \rhd\text{-PAR-L}\ \frac{P \rhd \varphi}{P \mid Q \rhd \varphi} \qquad \rhd\text{-PAR-R}\ \frac{Q \rhd \varphi}{P \mid Q \rhd \varphi} \qquad \rhd\text{-RES}\ \frac{P \rhd \varphi \quad a \notin \mathsf{n}(\varphi)}{(\nu a)P \rhd \varphi}$$

As an example, the reader might check that $(\nu u)(u\!\!/\!\!a \mid u\!\!/\!\!b) \mid b\!\!/\!\!c \rhd a \curlyvee c$.

Notation 1 *We write $\Phi(P) = \{\varphi \mid P \rhd \varphi\}$ the set of conditions entailed by P.*

2.2 Reduction Semantics and Barbed Congruence

Definition 1. *Structural congruence, written $\equiv$, is the smallest congruence satisfying α-conversion for restriction, and the following axioms:*

$$P \mid 0 \equiv P \qquad (P \mid Q) \mid R \equiv P \mid (Q \mid R) \qquad P \mid Q \equiv Q \mid P$$

$$(\nu a)0 \equiv 0 \qquad (\nu c)(\nu d)P \equiv (\nu d)(\nu c)P \qquad (\nu a)(P \mid Q) \equiv (\nu a)P \mid Q \text{ if } a \notin \mathsf{fn}(Q)$$

$$\sum_{i \in I} \pi_i.P_i \equiv \sum_{i \in I} \pi_{\sigma(i)}.P_{\sigma(i)} \;\; \sigma \text{ a permutation of } I$$

Relations $\equiv$ and $\rhd$ are used to define the reduction of processes. We rely on $\rhd$ to infer that two processes interact on joinable (extended) names. This allows us to introduce reduction-closed barbed congruence, along the lines of [8].

Definition 2 (Reduction). $\mapsto$ *is the relation defined by the following rules: (we write* $\equiv\mapsto\equiv$ *for relation composition):*

$$\frac{\overline{\alpha}(x).P \in S_1 \quad \beta(y).Q \in S_2 \quad R \rhd \alpha \curlyvee \beta \quad x \neq y}{R \mid S_1 \mid S_2 \;\mapsto\; R \mid (\nu xy)(x/y \mid P \mid Q)} \qquad \begin{array}{l} a \curlyvee \{b\} = \{b\} \curlyvee a = a \leqslant b \\ \{a\} \curlyvee \{b\} = \text{undefined} \end{array}$$

$$\frac{[\varphi]\tau.P \in S \qquad R \rhd \varphi}{R \mid S \;\mapsto\; R \mid P} \qquad \frac{P \mapsto P'}{P \mid R \mapsto P' \mid R} \qquad \frac{P \mapsto P'}{(\nu a)P \mapsto (\nu a)P'} \qquad \frac{P \equiv \mapsto \equiv P'}{P \mapsto P'}$$

We choose as in [8] a generic definition of barbs, uniform over usual process calculi, instead of the more syntactic characterisation.

Definition 3 (Barbs). *The barb* $\downarrow_{\overline{a}}$ *holds for a process* P *if* $P \mid a(x).\omega \mapsto P'$*, where* P' *is a process in which* ω *is unguarded, and* ω *is a special name that does not appear in* P*. We define similarly the barb* $\downarrow_a$*, using the tester* $\overline{a}(x).\omega$*.*

Definition 4. *Barbed congruence is the largest congruence* $\simeq$ *satisfying:*

- *if* $P \downarrow_a$ *and* $P \simeq Q$ *then* $Q \downarrow_a$*, and similarly for* $\downarrow_{\overline{a}}$*, and*
- *if* $P \mapsto P'$ *and* $P \simeq Q$ *then for some* Q'*,* $P' \simeq Q'$ *and* $Q \mapsto Q'$*.*

The remainder of the paper is devoted to the presentation of two characterisations of $\simeq$. We first comment on the definition of $\pi\mathtt{P}$ given above.

[8] discusses an alternative version of reduction, called "eager", where arcs can rewrite prefixes, yielding, e.g., $d/c \mid c(x).P \mapsto d/c \mid d(x).P$. [8] explains why the present semantics is more compelling (for instance $a(x).a(y)$ would not be equivalent to $a(x) \mid a(y)$ in the eager version).

Our calculus lacks prefixes for free input and output. These can be encoded using bound prefixes, as follows (name x is chosen fresh):

$$[ab.P] \;\triangleq\; (\nu x)a(x).(P \mid x/b) \qquad [\overline{a}b.P] \;\triangleq\; (\nu x)\overline{a}(x).(P \mid b/x) \;\;.$$

In a calculus having both kind of prefixes, we could show that the above encoded processes behave like primitive free prefixes.

Moreover, it is possible to encode protected prefixes, as follows (u is fresh):

$$[\{\overline{a}\}(x).P] \;\triangleq\; (\nu u)(u/a \mid \overline{u}(x).P) \qquad [\{a\}(x).P] \;\triangleq\; (\nu u)(u/a \mid u(x).P) \;\;.$$

Remark 5 (Comparison with [8]). Unlike the version of $\pi\mathtt{P}$ presented in [8], our calculus has sum and primitive bound prefixes. Including such constructs is customary when studying axiomatisations (bound prefixes are often present as soon as the calculus has restriction — see e.g. [12,15]). We also have primitive protected prefixes, to make the technical development of Section 4 simpler. As discussed above, these are encodable in $\pi\mathtt{P}$. The same holds for free prefixes, which are *not* included in the syntax. An LTS for $\pi\mathtt{P}$ based on free prefixes is definable, at the cost of a more complex technical development [9].

3 A Labelled Transition System for $\pi\mathtt{P}$

3.1 LTS and Bisimilarity

The LTS defines transitions $P \xrightarrow{\mu} P'$, where the grammar for the labels, μ, is the same as the one for prefixes π. We comment on the rules, given on Figure 1.

The first two rules correspond to the firing of visible prefixes. The transition involves a fresh name x, upon which the participants in a communication "agree". Name y remains local, via the installation of an arc, according to the directionality of the prefix. (Adopting a rule with no arc installation would yield a more complex definition of $\sim$). The rule for the $[\varphi]\tau$ prefix is self explanatory. The rule describing communication follows the lines of the corresponding rule for $\mapsto$; no arc is installed (but arcs are introduced in the prefix rules).

The three rules mentioning $\rhd$ are called *preorder rules*. The two preorder rules for visible actions exploit $\leqslant$, which is defined for extended names (as we did for $\curlyvee$ above). Note that the condition involving $\rhd$ is *the same* in these two rules. For instance, if $P \xrightarrow{a(x)} P'$ and $P \rhd a \leqslant c$, we can derive $P \xrightarrow{c(x)} P'$. In this case, $P \rhd a \leqslant \{c\}$ is also derivable, hence $P \xrightarrow{\{c\}(x)} P'$. We can also check that P_0 from Section 1 satisfies $P_0 \xrightarrow{\overline{\{c\}}(x)}$, because $\overline{a}v \mid a\!\!/c \rhd c \curlyvee a$ (and hence $a \leqslant \{c\}$).

The other preorder rule can be used to modify conditional τs involved in a transition. As an example, let $P_1 \triangleq (\overline{a}(x).Q \mid n\!\!/u) \mid (u(y).R \mid n\!\!/a)$. Process P_1 can perform a τ transition: the two arcs can, intuitively, let the output at a and the input at u interact at name n. Technically, this can be derived by inferring a $\xrightarrow{[a \curlyvee u]\tau}$ transition (from the output on the left and the input on the right). The latter can then be turned into a τ transition, exploiting the fact that *the whole process* entails $a \curlyvee u$. Finally, the congruence rules are as expected.

Definition 6 ($\sim$). *A symmetric relation $\mathcal{R}$ is a bisimulation if $P \mathcal{R} Q$ implies:*

- *If $P \rhd \varphi$ then $Q \rhd \varphi$.*
- *If $P \xrightarrow{\alpha(x)} P'$, with $x \notin \mathsf{fn}(Q)$, then there is Q' such that $Q \xrightarrow{\alpha(x)} Q'$ and $P' \mathcal{R} Q'$; we impose the same condition with $\overline{\alpha}$ instead of α.*
- *If $P \xrightarrow{[\varphi]\tau} P'$ then there is Q' such that $Q \xrightarrow{[\varphi]\tau} Q'$ and $P' \mid \varphi \mathcal{R} Q' \mid \varphi$.*

Bisimilarity, written $\sim$, is the greatest bisimulation.

This definition is reminiscent of the efficient bisimulation from [16]. Note in particular the necessity to add φ in parallel in the $\xrightarrow{[\varphi]\tau}$ case.

3.2 The Characterisation Theorem

On the $\vdash^{\imath}$ relation We define below $\vdash^{\imath}$, an atomic decomposition of $\vdash$ to be able to easily work proof inductions. The idea is that $\Gamma \vdash \varphi$ is equivalent to $\psi \vdash^{\imath}_{\psi_1} \circ \ldots \circ \vdash^{\imath}_{\psi_n} \varphi$ for some reflexive ψ and for some $\psi_1, \ldots, \psi_n \in \Gamma$, where $\vdash^{\imath}_{\psi}$ is the relation $\{(\varphi, \zeta) \mid \varphi, \psi \vdash \zeta\}$.

$$\frac{x \notin \mathsf{fn}(\alpha) \cup \mathsf{fn}(P)}{\alpha(y).P \xrightarrow{\alpha(x)} (\nu y)(x/y \mid P)}\ {\to}\text{-IN} \qquad \frac{x \notin \mathsf{fn}(\alpha) \cup \mathsf{fn}(P)}{\overline{\alpha}(y).P \xrightarrow{\overline{\alpha}(x)} (\nu y)(y/x \mid P)}\ {\to}\text{-OUT} \qquad \frac{}{[\varphi]\tau.P \xrightarrow{[\varphi]\tau} P}\ {\to}\text{-TAU}$$

$$\frac{P \xrightarrow{\overline{\alpha}(x)} P' \qquad Q \xrightarrow{\beta(x)} Q'}{P \mid Q \xrightarrow{[\alpha \curlyvee \beta]\tau} (\nu x)(P' \mid Q')}\ {\to}\text{-COMM-L} \qquad \frac{P \xrightarrow{[\varphi_2]\tau} P' \quad P \rhd \Gamma \quad \Gamma, \varphi_1 \vdash \varphi_2}{P \xrightarrow{[\varphi_1]\tau} P'}\ {\to}\text{-TAU-}\rhd$$

$$\frac{P \xrightarrow{\alpha(x)} P' \quad P \rhd \alpha \leqslant \beta}{P \xrightarrow{\beta(x)} P'}\ {\to}\text{-IN-}\rhd \qquad \frac{P \xrightarrow{\overline{\alpha}(x)} P' \quad P \rhd \alpha \leqslant \beta}{P \xrightarrow{\overline{\beta}(x)} P'}\ {\to}\text{-OUT-}\rhd$$

$$\begin{array}{l} a \leqslant \{b\} = a \curlyvee b \\ \{a\} \leqslant \{b\} = b \leqslant a \\ \{a\} \leqslant \ \ b \ \ = \text{undefined} \end{array}$$

$$\frac{P \xrightarrow{\mu} P' \qquad a \notin \mathsf{fn}(\mu)}{(\nu a)P \xrightarrow{\mu} (\nu a)P'}\ {\to}\text{-RES} \qquad \frac{P \xrightarrow{\mu} P' \qquad \mathsf{bn}(\mu) \cap \mathsf{fn}(Q) = \emptyset}{P \mid Q \xrightarrow{\mu} P' \mid Q}\ {\to}\text{-PAR-L} \qquad \frac{\pi_i.P_i \xrightarrow{\mu} P'}{\sum_i \pi_i.P_i \xrightarrow{\mu} P'}\ {\to}\text{-SUM}$$

Fig. 1. LTS for $\pi\mathtt{P}$. Symmetric versions of the two rules involving $\mid$ are omitted.

Definition 7. *The predicate $\varphi_1, \varphi_2 \vdash^i \varphi_3$ is defined as follows:*

$$\frac{}{a \leqslant b, b \leqslant c \vdash^i a \leqslant c} \qquad \frac{}{a \leqslant c, b \leqslant c \vdash^i a \curlyvee b} \qquad \frac{}{a \curlyvee b, c \leqslant a \vdash^i c \curlyvee b} \qquad \frac{}{a \curlyvee b, c \leqslant b \vdash^i a \curlyvee c}$$

$$\frac{}{b \leqslant c, a \leqslant b \vdash^i a \leqslant c} \qquad \frac{}{b \leqslant c, a \leqslant c \vdash^i a \curlyvee b} \qquad \frac{}{c \leqslant a, a \curlyvee b \vdash^i c \curlyvee b} \qquad \frac{}{c \leqslant b, a \curlyvee b \vdash^i a \curlyvee c}$$

Note that the second line is the symmetric variant of the first.

Lemma 8. *If $\varphi_1, \varphi_2 \vdash^i \varphi$ and $\varphi_{21}, \varphi_{22} \vdash^i \varphi_2$ then there exists φ' such that either:*

- *$\varphi_1, \varphi_{21} \vdash^i \varphi'$ and $\varphi', \varphi_{22} \vdash^i \varphi$, or*
- *$\varphi_1, \varphi_{22} \vdash^i \varphi'$ and $\varphi', \varphi_{21} \vdash^i \varphi$.*

Equivalent formulation: if $\varphi_{21}, \varphi_{22} \vdash^i \varphi_2$ then $\vdash^i{}_{\varphi_2} \subseteq (\vdash^i{}_{\varphi_{21}} \vdash^i{}_{\varphi_{22}} \cup \vdash^i{}_{\varphi_{22}} \vdash^i{}_{\varphi_{21}})$.

Proof. Finite and exhaustive case analysis on the premises; the proof has been mechanized.

The relations $\vdash$ and $\vdash^i$ mean the same:

Lemma 9. *If $\Gamma \vdash \varphi$ then $\vdash^i{}_{\varphi} \subseteq \vdash^i{}_{\psi_1} \ldots \vdash^i{}_{\psi_n} \vdash^i{}_{\zeta}$ for some $\psi_1, \ldots, \psi_n \in \Gamma$ and some reflexive ζ.*

Proof. By induction on $\Gamma \vdash \varphi$. Lemma 8 provides us with a helpful notion of composability that covers the rules $\vdash$-TRANS, $\vdash$-JOIN, $\vdash$-EXTJOIN. □

Corollary 10. *If $\varphi_1, \varphi_2 \vdash^i \varphi_3$ then $\varphi_1, \varphi_2 \vdash \varphi_3$. Conversely, if $\Gamma \vdash \varphi$ then $\psi \vdash^i{}_{\psi_1} \ldots \vdash^i{}_{\psi_n} \varphi$ for some reflexive ψ and for some $\psi_1, \ldots, \psi_n \in \Gamma$.*

Proof. The first part is trivial: each rule for $\vdash^i$ is simulated by (three) rules for $\vdash$. The second part follows from Lemma 9.

Notation 2 *We write $\varphi_1 \vdash^i{}_{\varphi_2} \varphi_3$ whenever $\varphi_1, \varphi_2 \vdash^i \varphi_3$ and we write $\vdash^i{}_P$ for the union of all $\vdash^i{}_\varphi$ for $P \rhd \varphi$.*

Intuitively, $\vdash^i{}_P$ denotes all the implications between conditions that hold because of the conditions that are entailed by process P in one step.

Lemma 11. $\vdash^i{}_{P|Q} \subseteq (\vdash^i{}_P \cup \vdash^i{}_Q)^*$.

Proof. We write $\mathcal{R}$ for the relation $\vdash^i{}_P \cup \vdash^i{}_Q$. We prove by induction on $P \mid Q \rhd \varphi_2$ that for all φ_1 and φ_3, if $\varphi_1, \varphi_2 \vdash^i \varphi_3$ then $\varphi_1 \mathcal{R}^* \varphi_3$. There are three cases. For $\rhd$-PAR-L we know that $P \rhd \varphi_2$ so $\varphi_1 \vdash^i{}_P \varphi_3$ and for $\rhd$-PAR-R we get the same way $\varphi_1 \vdash^i{}_Q \varphi_3$. The case for the rule $\rhd$-COMBINE is handled using Lemma 9. □

Lemma 12. *If $\varphi_1 (\vdash^i{}_P)^* \varphi_3$ then $\varphi_1 \vdash^i{}_P \varphi_2 \vdash^i{}_P \varphi_3$ for some φ_2 such that $\mathsf{n}(\varphi_2) \subseteq \mathsf{n}(\varphi_1) \cup \mathsf{n}(\varphi_3)$.*

Proof. (Sketch). Suppose $\varphi_1 \vdash^i{}_P \ldots \vdash^i{}_P \varphi_3$. One obtains φ_3 from φ_1 by appending ψ's such that $P \rhd \psi$ at the left or at the right of φ_1. We can in fact append all left ψs at first (an operation that can be done in one $\vdash^i{}_P$ step), and once it is done, appending all right ψs (the other $\vdash^i{}_P$ step).

Lemma 13. *Suppose $\varphi_1 \vdash^i{}^*_P \varphi_4$ and $\mathsf{n}(\varphi_i) = \{a_i, b_i\}$. For some φ_2 and φ_3 we have $\varphi_1 \vdash^i{}_P \varphi_2 \vdash^i{}_P \varphi_4$ and $\varphi_1 \vdash^i{}_P \varphi_3 \vdash^i{}_P \varphi_4$ such that $\mathsf{n}(\varphi_2) = \{a_1, b_4\}$ and $\mathsf{n}(\varphi_3) = \{a_4, b_1\}$.*

Proof. As in the proof of Lemma 12, but we care about whether we do left-right or right-left.

Lemma 14. *If $0 \rhd \varphi$ then φ is reflexive, and for all P, $P \rhd \varphi$.*

Proof. By induction on the derivation of $0 \rhd \varphi$: the only possible rule is $\rhd$-COMBINE, which can apply for P as well as for 0.

Lemma 15. *If $P \rhd \varphi$ and there is $a \in \mathsf{n}(\varphi) \setminus \mathsf{fn}(P)$, then φ is reflexive.*

Proof. We prove by induction on the derivation of $P \rhd \varphi$ that $\mathsf{n}(\varphi) = \{a\}$ for each of those φ. We have only one interesting case, for the $\rhd$-COMBINE rule: $P \rhd \varphi_1, \ldots, P \rhd \varphi_n$ and $\varphi_1, \ldots, \varphi_n \vdash \varphi$. We can suppose w.l.o.g. that Γ only contains φs that have been effectively used by one of the rules for $\vdash$.

By induction hypothesis, each time a appear in a φ_i, $\mathsf{n}(\varphi_i) = \{a\}$. Hence, as each rule for $\vdash$ propagates a into each premise, it will be the same for all the premises, leading at the end to the fact all φ_i are reflexive for a, and hence $\mathsf{n}(\varphi) = \{a\}$. □

A context C is a process with one occurrence of the hole; replacing the hole with P yields a process written $C[P]$.

Lemma 16. *If $\Phi(P) \subseteq \Phi(Q)$ then $\Phi(C[P]) \subseteq \Phi(C[Q])$.*

Proof. By induction on $C[P] \rhd \varphi'$ we prove $C[Q] \rhd \varphi'$, the case for $\rhd$-COMBINE is trivial. In all other cases we make a straightforward case analysis on C: it cannot be a sum, and if it is $C' \mid R$ with the rule $\rhd$-PAR-L (or $R \mid C'$ with $\rhd$-PAR-R, or $\nu a C'$ with $\rhd$-RES) then the induction hypothesis is enough. It is immediate if $C = C' \mid R$ with the rule $\rhd$-PAR-R (or $R \mid C'$ with $\rhd$-PAR-L), and if $C = []$ then we use $P \rhd \varphi \Rightarrow Q \rhd \varphi$.

Lemma 17. *If $P \rhd \varphi$ then $\Phi(P \mid \varphi) = \Phi(P)$.*

Proof. By induction on a derivation of $P \mid \varphi \rhd \psi$ for any ψ, we prove $P \rhd \psi$. The $\rhd$-PAR-L case is trivial, the $\rhd$-PAR-R case uses the fact $P \rhd \psi$, and the $\rhd$-COMBINE case is trivial.

Lemma 18. *If $P \equiv Q$ and $P \rhd \varphi$ then $Q \rhd \varphi$.*

Proof. We prove by induction on the derivation of $P \equiv Q$ that $\Phi(P) = \Phi(Q)$: this generalisation is necessary because symmetry is among the axioms of $\equiv$.

In most cases of this induction – the main exceptions being the first two items below – we prove that $\forall \varphi, P \rhd \varphi \Rightarrow Q \rhd \varphi$ and $\forall \varphi, Q \rhd \varphi \Rightarrow P \rhd \varphi$ separately, using an induction on the judgement $- \rhd \varphi$. The case for $\rhd$-COMBINE is immediate so we focus on when the last rule is another one.

1. The congruence rule is handled using Lemma 16.
2. The equivalence rules follow from the fact that $\Leftrightarrow$ is itself an equivalence.
3. $P \mid 0 \equiv P$: the induction on $P \mid 0 \rhd \varphi$ goes as follows: each $\rhd$-PAR-L rule is removed, and each $\rhd$-PAR-R rule is replaced by a proof of $0 \rhd \varphi \Rightarrow P \rhd \varphi$ (Lemma 14).
4. $P \equiv P \mid 0$: no induction needed, only an application of $\rhd$-PAR-L.
5. $(P \mid Q) \mid R \equiv P \mid (Q \mid R)$: the last rule is $\rhd$-PAR-R or $\rhd$-PAR-L. In the former case, we know $R \rhd \varphi$ so two applications of $\rhd$-PAR-R are needed to prove $P \mid (Q \mid R) \rhd \varphi$. In the case of rule $\rhd$-PAR-R, we have $P \mid Q \rhd \varphi$; we perform another induction on $P \mid Q \rhd \varphi$ to establish $P \mid (Q \mid R) \rhd \varphi$.
6. $P \mid (Q \mid R) \equiv (P \mid Q) \mid R$: similar proof.
7. $P \mid Q \equiv Q \mid P$: the last rule is either $\rhd$-PAR-L ($P \rhd \varphi$) or $\rhd$-PAR-R ($Q \rhd \varphi$), and we apply respectively $\rhd$-PAR-R or $\rhd$-PAR-L.
8. $Q \mid P \equiv P \mid Q$: same proof.
9. $(\nu a)P \equiv (\nu b)P\{b/a\}$ when $b \notin \mathsf{fn}(P)$: the last rule, and the one we apply, is $\rhd$-RES and we end up proving that if $P \rhd \varphi$ then $P\{b/a\} \rhd \varphi$ when $a, b \notin \mathsf{n}(\varphi)$. More generally we can prove $P \rhd \varphi \Rightarrow P_\sigma \rhd \varphi_\sigma$ for all (even non injective) substitution σ since $\rhd$ and $\vdash$ are oblivious to substitutions.
10. $(\nu b)P\{b/a\} \equiv (\nu a)P$: similar proof.
11. $(\nu ab)P \equiv (\nu ba)P$: the last rule is $\rhd$-RES. We know $(\nu b)P \rhd \varphi$ with $a \notin \mathsf{n}(\varphi)$. By induction on $(\nu b)P \rhd \varphi$ we can prove $P \rhd \varphi$, from which we can get $(\nu a)P$ and then $(\nu ba)P$.
12. $(\nu ba)P \equiv (\nu ab)P$: same proof.

13. $(\nu a P) \mid Q \equiv \nu a(P \mid Q)$ with $a \notin \mathsf{fn}(Q)$. If the last rule is $\rhd$-PAR-R, coming from $Q \rhd \varphi$, in which case we conclude easily. If the last rule is $\rhd$-PAR-L, coming from $\nu a P \rhd \varphi$, we get $P \rhd \varphi$ (like in the proof of the previous case) from which $(\nu a)(P \mid Q) \rhd \varphi$ follows. It is worth noting that if $a \in \mathsf{n}(\varphi)$, φ must be trivial ($a \curlyvee a$ or $a \leqslant a$).
14. $\nu a(P \mid Q) \equiv (\nu a P) \mid Q$ with $a \notin \mathsf{fn}(Q)$. This is the most difficult case in the proof of the present lemma. Intuitively, we need to put together all contributions from P so that they are independent of Q.
    First, as before, from $\nu a(P \mid Q) \rhd \varphi$ we get $P \mid Q \rhd \varphi$.
    We treat the case where $\varphi = b_0 \curlyvee c_0$ (the cases $b_0 \leqslant c_0$ and $c_0 \leqslant b_0$ are simpler). The derivation of $P \mid Q \rhd \varphi$ at hand is built up using a certain number of instances of rule $\rhd$-COMBINE involving processes P and Q, which we summarise as follows: there exist names $b_0, \ldots, b_n$ and $c_0, \ldots, c_m$ such that:
    - either $P \rhd b_n \curlyvee c_m$ or $Q \rhd b_n \curlyvee c_m$;
    - $b_i \leqslant b_{i+1}$ is derived from P or Q, i.e. either $P \rhd b_i \leqslant b_{i+1}$ or $Q \rhd b_i \leqslant b_{i+1}$, and the same holds for $c_j \leqslant c_{j+1}$.

    We show that we can construct a derivation where name a is not used, which means that we eliminate occurrences of a in the b_is and the c_js. For this, we proceed as follows:
    (a) we replace every derivation of the shape $Q \rhd \gamma$, where γ is a formula involving the b_is or the c_js, and $a \in \mathsf{n}(\gamma)$, with a derivation of $P \rhd \gamma$: this is possible thanks to Lemma 15 (because $a \notin \mathsf{fn}(Q)$, so γ must be reflexive).
    We obtain this way the property that whenever $Q \rhd \gamma$ for some γ involving the b_is or the c_js, $a \notin \mathsf{n}(\gamma)$.
    (b) Therefore, if $b_i = a$ for some i, we have $i > 0$ since $a \notin \mathsf{n}(\phi)$, and $P \rhd b_{i-1} \leqslant b_i = a$.
    If $i < n$, we also have $P \rhd a \leqslant b_{i+1}$, which allows us to deduce, using rule $\rhd$-COMBINE, $P \rhd b_{i-1} \leqslant b_{i+1}$. We thus got rid of an occurrence of a in the b_is.
    If $i = n$, since $P \rhd b_n \curlyvee c_m$ and $a = b_n$, we deduce $P \rhd b_{n-1} \curlyvee c_m$. Also in this case, we got rid of an occurrence of a in the b_is.
    (c) We proceed like this as long as a appears in the b_is, and do the same to get rid of all occurrences of a among the c_js.

    We are thus left with a set of formulas γ such that: (i) either $P \rhd \gamma$ or $Q \rhd \gamma$, and (ii) $a \notin \mathsf{n}(\gamma)$. Moreover, these formulas can be combined to yield $\varphi = b_0 \curlyvee c_0$. Whenever we have a derivation of $P \rhd \gamma$, we can use rule $\rhd$-RES to obtain a derivation of $(\nu a)P \rhd \gamma$. We then transform all derivations of formulas from either $(\nu a)P$ or Q into the corresponding derivation from $(\nu a)P \mid Q$, using either rule $\rhd$-PAR-L or $\rhd$-PAR-R.
    We conclude by applying rule $\rhd$-COMBINE $n + m$ times, to finally derive $(\nu a)P \mid Q \rhd \varphi$.

This concludes the induction on the derivation of $P \equiv Q$. □

Definition 19. *We define a relation $\sqsubseteq^{\varphi}$ between labels as follows: (i) $\alpha_1(x) \sqsubseteq^{\varphi} \alpha_2(x)$ and $\overline{\alpha}_1(x) \sqsubseteq^{\varphi} \overline{\alpha}_2(x)$ when $\varphi = \alpha_2 \leqslant \alpha_1$, and (ii) $[\varphi_1]\tau \sqsubseteq^{\varphi} [\varphi_2]\tau$ when $\varphi_1, \varphi \vdash \varphi_2$. We write $\sqsubseteq_P$ for the smallest preorder containing all $\sqsubseteq^{\varphi}$ when $P \rhd \varphi$.*

Lemma 20. *If $\varphi_1, \varphi_2 \vdash \varphi_3$ then $(\sqsubseteq^{\varphi_3}) \subseteq (\sqsubseteq^{\varphi_1} \sqsubseteq^{\varphi_2}) \cup (\sqsubseteq^{\varphi_2} \sqsubseteq^{\varphi_1})$.*

Proof. We suppose $\mu \sqsubseteq^{\varphi_3} \mu'$. If μ is a conditional τ, Lemma 8 is enough to conclude. Otherwise, we make an exhaustive case analysis on $\varphi_1, \varphi_2 \vdash \varphi_3$ and $\sqsubseteq^{\varphi_3}$, there a quite a few, but easy, cases. The proof has been mechanized, but we write below the main cases after removing some symmetric cases.

1. $a \leqslant b, b \leqslant c \vdash a \leqslant c$: from $\mu \sqsubseteq^{a \leqslant c} \mu'$ we know that either (we ignore when μ, μ' are output labels, we also write $\{a\}$ instead of $\{a\}(x)$):

$$\begin{array}{rcl} c \sqsubseteq^{a \leqslant c} a & \rightsquigarrow & c \sqsubseteq^{b \leqslant c} b \sqsubseteq^{a \leqslant b} a \\ \{a\} \sqsubseteq^{a \leqslant c} \{c\} & \rightsquigarrow & \{a\} \sqsubseteq^{a \leqslant b} \{b\} \sqsubseteq^{b \leqslant c} \{c\} \\ \{c\} \sqsubseteq^{a \leqslant c} a & \rightsquigarrow & \{c\} \sqsubseteq^{b \leqslant c} b \sqsubseteq^{a \leqslant b} a \\ \{a\} \sqsubseteq^{a \leqslant c} c & \rightsquigarrow & \{a\} \sqsubseteq^{a \leqslant b} \{b\} \sqsubseteq^{b \leqslant c} c \end{array}$$

2. $a \leqslant c, b \leqslant c \vdash a \curlyvee b$:

$$\begin{array}{rcl} \{a\} \sqsubseteq^{a \curlyvee b} b & \rightsquigarrow & \{a\} \sqsubseteq^{a \leqslant c} c \sqsubseteq^{b \leqslant c} b \\ \{b\} \sqsubseteq^{a \curlyvee b} a & \rightsquigarrow & \{b\} \sqsubseteq^{b \leqslant c} c \sqsubseteq^{a \leqslant c} a \end{array}$$

3. $a \curlyvee b, c \leqslant a \vdash c \curlyvee b$:

$$\begin{array}{rcl} \{c\} \sqsubseteq^{c \curlyvee b} b & \rightsquigarrow & \{c\} \sqsubseteq^{c \leqslant a} \{a\} \sqsubseteq^{a \curlyvee b} b \\ \{b\} \sqsubseteq^{c \curlyvee b} c & \rightsquigarrow & \{b\} \sqsubseteq^{a \curlyvee b} a \sqsubseteq^{c \leqslant a} c \end{array}$$

4. $a \curlyvee b, c \leqslant b \vdash a \curlyvee c$:

$$\begin{array}{rcl} \{c\} \sqsubseteq^{a \curlyvee c} a & \rightsquigarrow & \{c\} \sqsubseteq^{c \leqslant b} \{b\} \sqsubseteq^{a \curlyvee b} a \\ \{a\} \sqsubseteq^{a \curlyvee c} c & \rightsquigarrow & \{a\} \sqsubseteq^{a \curlyvee b} b \sqsubseteq^{c \leqslant b} c \end{array}$$

Lemma 21. $\sqsubseteq_{P|Q} \subseteq (\sqsubseteq_P \cup \sqsubseteq_Q)^*$.

Proof. By definition we know that $\mu \sqsubseteq^{\varphi} \mu'$ for some φ such that $P \mid Q \rhd \varphi$. First, remark that the cases mirror ($a \curlyvee b \sqsubseteq^{\varphi} b \curlyvee a$) and discard ($\psi \sqsubseteq^{\varphi} \psi$) are handled immediately.

There exists a trivial ψ such that $\psi \vdash_{P|Q} \varphi$. By Lemma 11 we know that $\vdash_{P|Q} \subseteq (\vdash_P \cup \vdash_Q)^*$, namely there is a sequence $\zeta_1, \zeta_2, \ldots, \zeta_n$ (with $\psi = \zeta_1$ and $\varphi = \zeta_n$) such that $\zeta_1 \vdash_{R_1} \ldots \zeta_{n-1} \vdash_{R_{n-1}} \zeta_n$ with $R_i \in \{P, Q\}$. Since $\zeta_i \vdash_{R_i} \zeta_{i+1}$ we know that $\sqsubseteq^{\zeta_{i+1}} \subseteq (\sqsubseteq^{\zeta_i} \cup \sqsubseteq_{R_i})^*$ by Lemma 20.

We write $\mathcal{R}$ for $\sqsubseteq_P \cup \sqsubseteq_Q$. Hence $\sqsubseteq^{\zeta_i} \subseteq \mathcal{R}^*$ implies $\sqsubseteq^{\zeta_{i+1}} \subseteq \mathcal{R}^*$, since $\sqsubseteq_{R_i} \subseteq \mathcal{R}$. Since ψ is trivial $\sqsubseteq^{\zeta_1} = \sqsubseteq_{\psi} \subseteq \sqsubseteq_P$, we conclude by induction on n: $\sqsubseteq^{\zeta_n} = \sqsubseteq_{\varphi} \subseteq \mathcal{R}^*$.

Proof. This breaks into proofs of $\vdash_{P|Q} \subseteq (\vdash_P \cup \vdash_Q)^*$ where $\vdash_R$ is the relation $\{(\varphi_1, \varphi_3) \mid \varphi_1, \varphi_2 \vdash \varphi_3, R \rhd \varphi_2\}$ and of $\varphi_1 \vdash_R \varphi_3$ implies $\sqsubseteq^{\varphi_3} \subseteq (\sqsubseteq^{\varphi_1} \cup \sqsubseteq_R)^*$. □

Lemma 22 (Composition of extended $\leqslant$ and $\curlyvee$). *On extended names, $\leqslant$ is transitive and $\curlyvee$ can be extended on the left (or on the right), but cannot in general be derived from $\leqslant$.*

1. *If $\alpha \leqslant \beta$ and $\beta \leqslant \gamma$ are defined, so is $\alpha \leqslant \gamma$ and $\alpha \leqslant \beta, \beta \leqslant \gamma \vdash \alpha \leqslant \gamma$.*
2. *If $\alpha \leqslant \beta$ and $\beta \curlyvee \gamma$ are defined, so is $\alpha \curlyvee \gamma$ and $\alpha \leqslant \beta, \beta \curlyvee \gamma \vdash \alpha \curlyvee \gamma$.*
3. *In general $\alpha \leqslant \beta$ and $\gamma \leqslant \beta$ do not imply $\alpha \curlyvee \gamma$.*

Proof. The first two items needs a careful case analysis:

1. to prove $\alpha \leqslant \beta, \beta \leqslant \gamma \vdash \alpha \leqslant \gamma$, we have four cases for α,β,γ:
   (a) a,b,c: then $a \leqslant b, b \leqslant c \vdash a \leqslant c$ by transitivity of $\leqslant$,
   (b) a,b,$\{c\}$: then $a \leqslant b, b \curlyvee c \vdash a \curlyvee c$ by left extension of $\curlyvee$,
   (c) a,$\{b\}$,$\{c\}$: then $a \curlyvee b, c \leqslant b \vdash a \curlyvee c$ by right extension of $\curlyvee$,
   (d) $\{a\}$,$\{b\}$,$\{c\}$: then $b \leqslant a, b \leqslant c \vdash c \leqslant a$ by transitivity of $\leqslant$.
2. to prove $\alpha \leqslant \beta, \beta \curlyvee \gamma \vdash \alpha \curlyvee \gamma$, we have 4 well-defined cases for α,β,γ:
   (a) a,b,c: then $a \leqslant b, b \curlyvee c \vdash a \curlyvee c$ by left extension of $\curlyvee$,
   (b) a,b,$\{c\}$: then $a \leqslant b, b \leqslant c \vdash a \leqslant c$ by transitivity of $\leqslant$,
   (c) a,$\{b\}$,c: then $a \curlyvee b, c \leqslant b \vdash a \curlyvee c$ by right extension of $\curlyvee$,
   (d) a,$\{b\}$,$\{c\}$: impossible since $\{b\} \curlyvee \{c\}$ is undefined,
   (e) $\{a\}$,b,c: impossible since $\{a\} \leqslant b$ is undefined,
   (f) $\{a\}$,b,$\{c\}$: impossible since $\{a\} \leqslant b$ is undefined,
   (g) $\{a\}$,$\{b\}$,c: then $b \leqslant a, c \leqslant b \vdash c \leqslant a$ by transitivity of $\leqslant$,
   (h) $\{a\}$,$\{b\}$,$\{c\}$: impossible since $\{b\} \curlyvee \{c\}$ is undefined.

And finally a counterexample for (3) is when $\alpha = \beta = \gamma = \{a\}$ since we always have $\{a\} \leqslant \{a\}$ but $\{a\} \curlyvee \{a\}$ is undefined. □

Corollary 23. *If $\alpha(x) \sqsubseteq_P \alpha_1(x)$ and $\alpha \curlyvee \beta$ is defined, then $\alpha_1 \curlyvee \beta$ is defined and $[\alpha \curlyvee \beta]\tau \sqsubseteq_P [\alpha_1 \curlyvee \beta]\tau$.*

Proof. Remark that $\alpha(x) \sqsubseteq_P \alpha_1(x)$ is equivalent to $P \rhd \alpha_1 \leqslant \alpha$ and by definition of $\sqsubseteq_P$, it is enough to prove $\alpha \curlyvee \beta, \alpha_1 \leqslant \alpha \vdash \alpha_1 \curlyvee \beta$ which is an instance of the Lemma 22.

Remark 24. For visible labels, $\sqsubseteq_P$ needs to be closed neither by transitivity nor by reflexivity, as the first item of Lemma 22 indicates.

Lemma 25. *If $A \rhd \varphi$ and A is a parallel composition of preorder processes, then there are some $\varphi_1, \ldots, \varphi_n \in A$ and a reflexive ψ such that $\sqsubseteq^{\varphi} = \sqsubseteq^{\psi} \sqsubseteq^{\varphi_1} \ldots \sqsubseteq^{\varphi_n}$.*

Proof. First remark that if ψ_1 and ψ_2 are reflexive, then there is some ψ such that $\sqsubseteq^{\psi_1} \sqsubseteq^{\psi_2} \subseteq \sqsubseteq^{\psi}$. Also remark that for all φ, if ψ_1 is reflexive, then there is a reflexive ψ_2 for which $\sqsubseteq^{\varphi} \sqsubseteq^{\psi_1} \subseteq \sqsubseteq^{\psi_2} \sqsubseteq^{\varphi}$. It is now enough to prove the weaker statement with reflexive ψs interleaving with the φ_is. Moreover this weaker statement is transitive so we can just use induction on $A \rhd \varphi$ now:

- $a \not| b \rhd b \leqslant a$: trivial ($\sqsubseteq^{b \leqslant a} \subseteq \sqsubseteq^{b \leqslant a}$).
- the case for $\rhd$-PAR-* is trivial as well,

– $\rhd$-COMBINE needs an induction of its own:
  - $\vdash$-REFL $A \rhd a \leqslant a$: φ is then indeed reflexive
  - $\vdash$-IN $\varphi \in A$: indeed $\sqsubseteq^{\varphi} \subseteq \sqsubseteq^{\varphi}$,
  - $\vdash$-MIRROR $\sqsubseteq^{a \curlyvee b} = \sqsubseteq^{b \curlyvee a}$.
  - $\vdash$-TRANS since $a \leqslant b, b \leqslant c \vdash a \curlyvee c$ we use Lemma 20.
  - $\vdash$-JOIN same as above with $a \leqslant b, c \leqslant b \vdash a \curlyvee c$
  - $\vdash$-EXTJOIN same with $a \leqslant b, b \curlyvee c \vdash a \curlyvee c$. □

Lemma 26. *If $\mu \sqsubseteq_{P|Q} \mu'$ and $a \notin \mathsf{fn}(\mu), \mathsf{fn}(\mu'), \mathsf{fn}(Q)$ then $\mu \sqsubseteq_{(\nu a P)|Q} \mu'$.*

Proof. We use use Lemma 21 to get a sequence $\mu_0, \mu_1, \ldots, \mu_n$ with $\mu = \mu_0$ and $\mu' = \mu_n$ and $\mu_i \sqsubseteq^{\varphi_i} \mu_{i+1}$ with for each i, $P \rhd \varphi_i$ or $Q \rhd \varphi_i$. Suppose $a \in \mathsf{fn}(\varphi_i)$ and $Q \rhd \varphi_i$, then wlog. since φ_i is trivial and we can suppose that in fact $P \rhd \varphi_i$. We then regroup all subsequences $\varphi_i, \ldots, \varphi_j$ entailed by P together (and we do the same with Q) to obtain after some reindexing: $\mu_{j_1} \sqsubseteq_P \mu_{j_2} \sqsubseteq_Q \mu_{j_3} \sqsubseteq_P \ldots$ such that for all k, $a \notin \mathsf{fn}(\mu_{j_k})$. Then we can write in fact $\mu_{j_1} \sqsubseteq_{\nu a P} \mu_{j_2} \sqsubseteq_Q \mu_{j_3} \sqsubseteq_{\nu a P} \ldots$ and we can conclude that $\mu \sqsubseteq_{(\nu a P)|Q} \mu'$.

Lemma 27. *If $\mu \sqsubseteq_{P|Q} \mu'$ and $a \notin \mathsf{fn}(\mu), \mathsf{fn}(Q)$ then for some λ such that $a \notin \mathsf{fn}(\lambda)$, $\mu \sqsubseteq_{(\nu a P)|Q} \lambda \sqsubseteq_P \mu'$.*

Proof. As in the proof of Lemma 26 we get $\mu = \mu_{j_1} \sqsubseteq_P \mu_{j_2} \sqsubseteq_Q \ldots \sqsubseteq_P \mu_{j_n} = \mu'$ except a may appear in $\mu' = \mu_{j_n}$, but not in any other $\mu = \mu_{j_k}$ if $k < n$. We choose $\lambda = \mu_{j_{n-1}}$, the rest is the same.

Lemma 28. *If $\varphi_1 = \alpha \curlyvee \gamma$ is defined and $\varphi_2 \vdash^{i}_{P} \varphi_1$, then either*

1. *for some β, $\beta(x) \sqsubseteq^{\psi} \alpha(x)$ and $\varphi_2 = \beta \curlyvee \gamma$, or*
2. *for some β, $\beta(x) \sqsubseteq^{\psi} \gamma(x)$ and $\varphi_2 = \alpha \curlyvee \beta$ (this case being unnecessary if $\mathsf{fn}(\alpha) \not\subseteq \mathsf{n}(\varphi_2)$).*

Lemma 29. *If $P \xrightarrow{\mu} P'$ and $\eta \sqsubseteq_P \mu$ then $P \xrightarrow{\eta} P'$. Conversely, whenever $P \xrightarrow{\eta} P'$ there exists μ such that $\eta \sqsubseteq_P \mu$ and $P \xrightarrow{\mu} P'$, of which we can have a proof of smaller size that does not end with a preorder rule.*

Proof. Both direction are proved by simple inductions.

Notation 3 *We adopt the following notation in writing derivations, to denote either an application of the first part of Lemma 29 (i.e. an application of several preorder rules), or a recursive case analysis on the rules deriving $P \xrightarrow{\mu} P'$ until a non-preorder rule is reached, by application of the second part of Lemma 29.*

$$\dfrac{\dfrac{P \xrightarrow{\mu'} P'}{[\mu \sqsubseteq_P \mu']}}{P \xrightarrow{\mu} P'}$$

Lemma 30 justifies the fact that we can use the shortcut τ for a $[\varphi]\tau$ transition with "some reflexive φ" as well as "any reflexive φ".

Lemma 30. *The following are equivalent:*

1. $P \xrightarrow{[\psi]\tau} P'$ *for some reflexive* ψ,
2. $P \xrightarrow{[\psi]\tau} P'$ *for all reflexive* ψ,
3. $P \xrightarrow{[\varphi]\tau} P'$ *and there is a name* $a \in \mathsf{n}(\varphi) \setminus \mathsf{fn}(P)$.

Proof. First remark that if ψ is reflexive then for all Γ, $\Gamma \vdash \psi$, which means $[\varphi]\tau \sqsubseteq_P [\psi]\tau$ for any φ, so we only need to prove $(iii) \Rightarrow (i)$ which is equivalent to saying $[\psi]\tau \sqsubseteq_P [\varphi]\tau$ for some reflexive ψ, which in turn follows from Lemma 15. □

Lemma 31. *If* $(\nu a)P \xrightarrow{\mu} P_1$ *then* $P_1 = (\nu a)P'$ *for some* P' *such that* $P \xrightarrow{\mu} P'$.

Proof. Using Lemma 29 we know that $(\nu a)P \xrightarrow{\mu_1} P_1$ for some μ_1 such that $\mu \sqsubseteq_{\nu a P} \mu_1$, and that this transition is coming from a reduction $P \xrightarrow{\mu_1} P'$ with $P_1 = (\nu a)P'$. Since $\Phi(\nu a P) \subseteq \Phi(P)$ we know $\mu \sqsubseteq_P \mu_1$ so we can derive $P \xrightarrow{\mu} P'$.

We are now ready to prove that transitions commute with $\equiv$: The next lemma follows from an analysis similar to the proof of Lemma 18. Lemma 41 is the most complex case in the proof of congruence of $\sim$, for which Lemmas 18 and 41 are needed. Lemma 42 is useful for the completeness proof.

Lemma 32. *If* $P \equiv Q$ *and* $P \xrightarrow{\mu} P'$ *then* $Q \xrightarrow{\mu} \equiv P'$.

Proof. More generally, we prove by induction on the derivation of $P \equiv Q$ that for all μ, $((P \xrightarrow{\mu} P' \Rightarrow Q \xrightarrow{\mu} \equiv P')$ and $(Q \xrightarrow{\mu} Q' \Rightarrow P \xrightarrow{\mu} \equiv Q'))$.
Then in most cases we begin by an induction on the derivation of $P \xrightarrow{\mu} P'$. We factor out the cases corresponding to preorder rules using Lemma 29 twice: once to get $\mu_1 \sqsubseteq_P \mu$ and $P \xrightarrow{\mu_1} P'$ on which we can use the induction hypothesis to prove $Q \xrightarrow{\mu_1} Q' \equiv P'$, and once to get $Q \xrightarrow{\mu} Q'$, since $\mu_1 \sqsubseteq_Q \mu$ (Lemma 18). Hence, we can always assume that the last rule is not a preorder rule and therefore decomposes the structure of P.

1. Congruence: we suppose the result holds for $P \equiv Q$, and we prove it for $C[P] \equiv C[Q]$. Assuming $(\forall \mu,\ P \xrightarrow{\mu} P' \Rightarrow Q \xrightarrow{\mu} \equiv P')$ we prove, by induction on $C[P] \xrightarrow{\mu} P_1$, that $C[Q] \xrightarrow{\mu} Q_1 \equiv P_1$. We assume the last rule is deconstructing $C[P]$ into smaller parts (as if it were an induction on C):
   (a) $C = []$: immediate.
   (b) $C = \Sigma \pi_i.R_i + \rho.C'$: if a π_i branch is chosen, P_1 does not depend on P and $C[Q] \xrightarrow{\mu} P_1$. If the ρ branch is chosen $P_1 = (\nu x)(\varphi \mid C'[P])$ where φ is an arc. Then $C[Q] \xrightarrow{\mu} (\nu x)(\varphi \mid C'[Q]) \equiv P_1$.
   (c) $C = \nu a C'$: immediate (the side condition is the same, and if $P_1 \equiv Q_1$ then $\nu a P_1 \equiv \nu a Q_1$).
   (d) $C = C' \mid R$ and the last rule is PAR (or symmetrically): we obtain $C'[P] \xrightarrow{\mu} P_2$ with $P_1 = P_2 \mid R$ and by induction $C'[Q] \xrightarrow{\mu} Q_2 \equiv P_2$. Using PAR, $C'[Q] \mid R \xrightarrow{\mu} Q_2 \mid R \equiv P_1$.

(e) $C = R \mid C'$ and the last rule is PAR (or symmetrically): same, but easier.

(f) $C = C' \mid R$ or $C = R \mid C'$ and the last rule is one of the communication rules, e.g. $\rightarrow$-COMM-L:
$C'[P] \xrightarrow{\overline{a}(x)} P_2$ and $R \xrightarrow{c(x)} R'$ with $P_1 = (\nu x)(P_2 \mid R')$. By induction, $C'[Q] \xrightarrow{\overline{a}(x)} Q_2 \equiv P_2$ so by applying $\rightarrow$-COMM-L again, $C'[Q] \mid R \xrightarrow{[a \curlyvee c]\tau} \equiv (\nu x)(P_2 \mid R')$.

2. Equivalence properties: reflexivity and transitivity are straightforward; symmetry is handled by the generality of the statement.
3. $P \mid 0 \equiv P$: straightforward: the last rule must be $\rightarrow$-PAR-L.
4. $P \equiv P \mid 0$: straightforward.
5. $(P \mid Q) \mid R \equiv P \mid (Q \mid R)$: there are a lot of cases, of which the first below is the only trivial one:

(a) $\rightarrow$-PAR-R: $R \xrightarrow{\mu} R'$ so two applications of $\rightarrow$-PAR-R are enough to get $P \mid (Q \mid R) \xrightarrow{\mu} \equiv (P \mid Q) \mid R'$.

(b) $\rightarrow$-PAR-L, preorder rules and Lemma 29 to get $\mu \sqsubseteq_{P|Q} \mu_1$, then a $\rightarrow$-PAR-R rule to get $Q \xrightarrow{\mu_1} Q'$. Getting back $P \mid (Q \mid R) \xrightarrow{\mu_1} P \mid (Q' \mid R)$ is easy, what is more tricky is to get right the label μ: for that we use again Lemma 29 thanks to the fact that $\mu \sqsubseteq_{P|(Q|R)} \mu_1$, in turn implied by $\mu \sqsubseteq_{P|Q} \mu_1$ (since $\Phi(P \mid Q) \subseteq \Phi((P \mid Q) \mid R) = \Phi(P \mid (Q \mid R))$ by rule $\triangleright$-PAR-L and Lemma 18).

$$\dfrac{\dfrac{\dfrac{Q \xrightarrow{\mu_1} Q'}{P \mid Q \xrightarrow{\mu_1} P \mid Q'}\,\text{PAR-R}}{\dfrac{\left[\mu \sqsubseteq_{P|Q} \mu_1\right]}{P \mid Q \xrightarrow{\mu} P \mid Q'}}}{(P \mid Q) \mid R \xrightarrow{\mu} (P \mid Q') \mid R}\,\text{PAR-L} \quad \rightsquigarrow \quad \dfrac{\dfrac{\dfrac{Q \xrightarrow{\mu_1} Q'}{Q \mid R \xrightarrow{\mu_1} Q' \mid R}\,\text{PAR-L}}{\dfrac{P \mid (Q \mid R) \xrightarrow{\mu_1} P \mid (Q' \mid R)}{\left[\mu \sqsubseteq_{P|(Q|R)} \mu_1\right]}}\,\text{PAR-R}}{P \mid (Q \mid R) \xrightarrow{\mu} P \mid (Q' \mid R)}\,.$$

(c) $\rightarrow$-PAR-L, preorder rules and Lemma 29 to get $\mu \sqsubseteq_{P|Q} \mu_1$, then a $\rightarrow$-PAR-R rule to get $P \xrightarrow{\mu_1} P'$. Only the top of the proof tree is modified (from Q to P), the rest of the derivation is the same.

(d) $\rightarrow$-COM-L rule ($\varphi = \alpha \curlyvee \gamma$), preorder rules on the left premise to get $\mu = \alpha(x) \sqsubseteq_{P|Q} \alpha_1(x)_1$, then a $\rightarrow$-PAR-R rule to get $Q \xrightarrow{\alpha_1(x)} Q'$. By Lemma 23 we can have $\varphi_1 = \alpha_1 \curlyvee \gamma$ such that $[\varphi]\tau \sqsubseteq_{P|Q} [\varphi_1]\tau$ from which we conclude as previously $[\varphi]\tau \sqsubseteq_{P|(Q|R)} [\varphi_1]\tau$ to get the final transition from Lemma 29. Same proof for $\rightarrow$-COM-R.

$$\dfrac{\dfrac{\dfrac{Q \xrightarrow{\alpha_1(x)} Q'}{P \mid Q \xrightarrow{\alpha_1(x)} P \mid Q'}\,\text{PAR-R}}{\dfrac{\left[\alpha(x) \sqsubseteq_{P|Q} \alpha_1(x)\right]}{P \mid Q \xrightarrow{\alpha(x)} P \mid Q'}} \quad R \xrightarrow{\overline{\gamma}(x)} R'}{(P \mid Q) \mid R \xrightarrow{[\varphi]\tau} \nu x((P \mid Q') \mid R')}\,\text{COM-L} \quad \rightsquigarrow \quad \dfrac{\dfrac{\dfrac{Q \xrightarrow{\alpha_1(x)} Q' \qquad R \xrightarrow{\overline{\gamma}(x)} R'}{Q \mid R \xrightarrow{[\varphi_1]\tau} \nu x(Q' \mid R')}\,\text{COM-L}}{\dfrac{P \mid (Q \mid R) \xrightarrow{[\varphi_1]\tau} P \mid \nu x(Q' \mid R')}{\left[[\varphi]\tau \sqsubseteq_{P|(Q|R)} [\varphi_1]\tau\right]}}\,\text{PAR-R}}{P \mid (Q \mid R) \xrightarrow{[\varphi]\tau} P \mid \nu x(Q' \mid R')}\,.$$

(e) $\rightarrow$-COM-* rule, preorder rules, then a $\rightarrow$-PAR-L rule: it is the same proof as above: the transition with the label $[\varphi_1]\tau$ from $P \mid (Q \mid R)$ is easy to obtain, the rest is the same.

6. $P \mid (Q \mid R) \equiv (P \mid Q) \mid R$: similar proof.
7. $P \mid Q \equiv Q \mid P$: the last rule is either:
   (a) $\rightarrow$-PAR-R or $\rightarrow$-PAR-L, that complement each other, or
   (b) $\rightarrow$-COM-L rules, resulting in a process of the form $(\nu x)(P' \mid Q')$ with $P \xrightarrow{\alpha(x)} P'$, $Q \xrightarrow{\overline{\gamma}(x)} Q'$ and $\mu = [\alpha \curlyvee \gamma]\tau$. The application of the rule $\rightarrow$-COM-R will give the transition $[\gamma \curlyvee \alpha]\tau$ which is convertible to μ through the rule $\rightarrow$-TAU-$\triangleright$ since $\alpha \curlyvee \gamma \vdash \gamma \curlyvee \alpha$.
8. $Q \mid P \equiv P \mid Q$: same proof.
9. $(\nu a)P \equiv (\nu b)P\{b/a\}$ when $b \notin \mathsf{fn}(P)$: this case boils down to a proof of if $P \xrightarrow{\mu} P'$ then $P_\sigma \xrightarrow{\mu_\sigma} P'_\sigma$ where σ is an injective substitution (whose domain does not intersect with $\mathsf{bn}(\mu)$).
10. $(\nu b)P\{b/a\} \equiv (\nu a)P$: similar proof.
11. $(\nu ab)P \equiv (\nu ba)P$: last rule is $\rightarrow$-RES, we get $(\nu b)P \xrightarrow{\mu} P_1$, then using Lemma 29 we get $(\nu b)P \xrightarrow{\mu_1} (\nu b)P'$ and $P \xrightarrow{\mu_1} P'$ with $\mu \sqsubseteq_P \mu_1$, and again with Lemma 29 we get $P \xrightarrow{\mu} P'$ and apply $\rightarrow$-RES twice.
12. $(\nu ba)P \equiv (\nu ab)P$: same proof.
13. $(\nu aP) \mid Q \equiv \nu a(P \mid Q)$: the last non-preorder rule can be of two kinds:
   (a) the last rule is a $\rightarrow$-PAR-* rule. If Q does the transition, we are done. If νaP does the transition μ, then we get a transition μ_1 with $\mu \sqsubseteq_{\nu aP} \mu_1$ from P using Lemma 29. $P \mid Q$ can also make the transition μ_1, and since $\mu \sqsubseteq_{\nu aP} \mu_1$ implies $\mu \sqsubseteq_{\nu a(P|Q)} \mu_1$ we get the transition μ from $\nu a(P \mid Q)$.
   (b) If the last rule is a $\rightarrow$-COM-* rule then $\mu = [\varphi]\tau$, and νaP does some transition μ to what must be $\nu aP'$ with $P \xrightarrow{\mu_1} P'$ and $\mu \sqsubseteq_{\nu aP} \mu_1$. Composing P with Q (in the case of $\rightarrow$-COM-*) yields a transition $[\varphi_1]\tau$ with $[\varphi_1]\tau \sqsubseteq_{\nu aP} [\varphi]\tau$ which implies $[\varphi_1]\tau \sqsubseteq_{\nu a(P|Q)} [\varphi]\tau$, so Lemma 29 is again used twice.
14. $\nu a(P \mid Q) \equiv (\nu aP) \mid Q$: in this case, again, one must pay close attention to interference between restriction and preorder judgements. From the transition μ from $\nu a(P \mid Q)$ we get the same transition from $P \mid Q$, but after an application of Lemma 29 we have $\mu \sqsubseteq_{P|Q} \mu_1$ with a transition μ_1 that can be broken into either $P \xrightarrow{\mu_1} P'$ or $Q \xrightarrow{\mu_1} Q'$ (in the case of a $\rightarrow$-PAR-* rule), or both $P \xrightarrow{\mu} P'$ and $Q \xrightarrow{\chi} Q'$ (in the case of a $\rightarrow$-COM-* rule). The problem is, we have to recover all the contributions from P in such a way that νaP can achieve as much. Notably μ or μ_1 may mention a. We treat the three cases separately:
   (a) the case of the $\rightarrow$-PAR-R rule has two subcases. First when $a \notin \mathsf{fn}(\mu_1)$, it boils down to Lemma 26. If μ_1 is a visible action then the object of μ_1 must stay fresh, but we know already that $\mathsf{obj}(\mu_1) \cap \mathsf{fn}(P) = \emptyset$ which implies $\mathsf{obj}(\mu_1) \cap \mathsf{fn}(\nu aP) = \emptyset$.
   The case $a \in \mathsf{fn}(\mu_1)$ is not possible when μ_1 is a visible action because then, $\mathsf{fn}(\mu_1) \subseteq \mathsf{fn}(Q)$ and $a \notin \mathsf{fn}(Q)$. In the case μ_1 is of the form $[\varphi_1]\tau$, we apply Lemma 30 to get that $Q \xrightarrow{\tau} Q'$ and thus $(\nu aP) \mid Q \xrightarrow{\tau} (\nu aP) \mid$

Q' from which we can easily derive any transition between the same processes with any label of the form $[\varphi]\tau$ (and μ must be of this form).

$$\dfrac{\dfrac{\dfrac{\dfrac{Q \xrightarrow{\mu_1} Q'}{P \mid Q \xrightarrow{\mu_1} P \mid Q'}}{\left[\mu \sqsubseteq_{P|Q} \mu_1\right]}}{P \mid Q \xrightarrow{\mu} P \mid Q' \quad a \notin \mathsf{n}(\mu)}}{(\nu a)(P \mid Q) \xrightarrow{\mu} (\nu a)(P \mid Q')} \quad \rightsquigarrow \quad \dfrac{\dfrac{\dfrac{Q \xrightarrow{\mu_1} Q'}{(\nu a P) \mid Q \xrightarrow{\mu_1} (\nu a P) \mid Q'}}{\left[\mu \sqsubseteq_{(\nu a P)|Q} \mu_1\right]}}{(\nu a P) \mid Q \xrightarrow{\mu} (\nu a P) \mid Q'}$$

(b) The case of rule $\rightarrow$-PAR-L: the action μ_1, coming originally from P, may contain a but we use Lemma 27 to obtain a transition along λ, to which we can apply Q's original role in the transformation of μ_1 into μ (combined to the role of $\nu a P$).
The following derivations illustrate the reasoning exposed above.

$$\dfrac{\dfrac{\dfrac{\dfrac{P \xrightarrow{\mu_1} P'}{P \mid Q \xrightarrow{\mu_1} P' \mid Q}}{\left[\mu \sqsubseteq_{P|Q} \mu_1\right]}}{P \mid Q \xrightarrow{\mu} P' \mid Q \quad a \notin \mathsf{n}(\mu)}}{(\nu a)(P \mid Q) \xrightarrow{\mu} (\nu a)(P' \mid Q)} \quad \rightsquigarrow \quad \dfrac{\dfrac{\dfrac{\dfrac{P \xrightarrow{\mu_1} P'}{\left[\lambda \sqsubseteq_P \mu_1\right]}}{P \xrightarrow{\lambda} P' \quad a \notin \mathsf{n}(\lambda)}}{\nu a P \xrightarrow{\lambda} \nu a P'}}{(\nu a P) \mid Q \xrightarrow{\lambda} (\nu a P') \mid Q}}{\left[\mu \sqsubseteq_{(\nu a P)|Q} \lambda\right]}}{(\nu a P) \mid Q \xrightarrow{\mu} (\nu a P') \mid Q}$$

(c) The case of the $\rightarrow$-COM-L rule ($\rightarrow$-COM-R being symmetric): this time $\mu = [\varphi]\tau$ is obtained from some $[\varphi_1]\tau$ such that $[\varphi]\tau \sqsubseteq_{P|Q} [\varphi_1]\tau$ (as usual, using Lemma 29) and $[\varphi_1]\tau$ is obtained from two transitions: $P \xrightarrow{\alpha(x)} P'$ and $Q \xrightarrow{\overline{\gamma}(x)} Q'$ with $\varphi = \alpha \curlyvee \gamma$. We write μ for $\alpha(x)$ and χ for $\overline{\gamma}(x)$.
Lemma 27 allows us to decompose $\sqsubseteq_{P|Q}$ into $[\varphi]\tau \sqsubseteq_{(\nu a P)|Q} [\varphi_2]\tau \sqsubseteq_P [\varphi_1]\tau$. We use Lemma 13 to decompose $\varphi_2 \vdash_P^* \varphi_1$ into $\varphi_2 \vdash_P \varphi_3 \vdash_P \varphi_1$ such that $\mathsf{fn}(\mu) \cap \mathsf{n}(\varphi_3) = \emptyset$. We apply Lemma 28 to get β such that $\beta(x) \sqsubseteq_P \alpha(x)$ and $\varphi_3 = \beta \curlyvee \gamma$. Since $a \notin \mathsf{n}(\varphi_3)$, we know that $a \notin \mathsf{n}(\beta)$, so we can apply the restriction rule on the transition along $\beta(x)$. Furthermore, since $\varphi_2 \vdash_P \varphi_3$ and a is in none of both, we know by transitivity that $[\varphi]\tau \sqsubseteq_{(\nu a P)|Q} [\varphi_3]\tau$. We write η for $\beta(x)$ below:

$$\dfrac{\dfrac{\dfrac{\dfrac{P \xrightarrow{\mu} P' \qquad Q \xrightarrow{\chi} Q'}{P \mid Q \xrightarrow{[\varphi_1]\tau} (\nu x)(P' \mid Q')}}{\left[[\varphi]\tau \sqsubseteq_{P|Q} [\varphi_1]\tau\right]}}{P \mid Q \xrightarrow{[\varphi]\tau} (\nu x)(P' \mid Q') \quad a \notin \mathsf{n}(\varphi)}}{(\nu a)(P \mid Q) \xrightarrow{[\varphi]\tau} (\nu a)(\nu x)(P' \mid Q')} \quad \rightsquigarrow \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{P \xrightarrow{\mu} P'}{[\eta \sqsubseteq_P \mu]}}{P \xrightarrow{\eta} P' \quad a \notin \mathsf{n}(\eta)}}{\nu a P \xrightarrow{\eta} \nu a P' \qquad Q \xrightarrow{\chi} Q'}}{(\nu a P) \mid Q \xrightarrow{[\varphi_3]\tau} (\nu x)((\nu a P') \mid Q')}}{\left[[\varphi]\tau \sqsubseteq_{(\nu a P)|Q} [\varphi_3]\tau\right]}}{(\nu a P) \mid Q \xrightarrow{[\varphi]\tau} (\nu x)((\nu a P') \mid Q')}$$

This concludes the induction.

Lemma 33. *If $P \rhd \varphi$ then $P \mid \varphi \sim P$.*

Proof. We prove that $\mathcal{R} = \{(P \mid \varphi, P) \mid P \rhd \varphi\}$ is a simulation ($\mathcal{R}^{-1}$ is clearly one). Suppose $P \mid \varphi \xrightarrow{\mu} P_1$. Using Lemma 29 we get μ_1 such that $\mu \sqsubseteq_{P\mid\varphi} \mu_1$, and a proof of $P \mid \varphi \xrightarrow{\mu_1} P'$ with a non preorder rule as last rule. This rule must be a PAR rule coming from P so in fact $P_1 = P' \mid \varphi$ with $P \xrightarrow{\mu_1} P'$. Since $\Phi(P \mid \varphi) = \Phi(P)$ by Lemma 17 we know that $\mu \sqsubseteq_P \mu_1$ and we apply Lemma 29 again to get $P \xrightarrow{\mu} P'$ and indeed $P' \mid \varphi \; \mathcal{R} \; P'$.

Lemma 34. *$\equiv$ is a bisimulation.*

Proof. Almost direct consequence of Lemmas 18 and 32. The correspondence between transitions is exact, hence the only non-trivial case is for the $[\varphi]\tau$ transition because one would want $P' \mid \varphi \equiv Q' \mid \varphi$ instead of just $P' \equiv Q'$. This holds because $\equiv$ is a congruence.

Definition 35 (Bisimulation up to $\sim$). *A relation $\mathcal{R}$ is a bisimulation up to bisimilarity if it validates the clauses of the usual bisimulation, except when requiring that $P_1 \; \mathcal{R} \; Q_1$ we only require $P_1 \sim P_2$ and $Q_1 \sim Q_2$ with $P_2 \; \mathcal{R} \; Q_2$.*

Lemma 36. *If $\mathcal{R}$ is a bisimulation up to bisimilarity, then $\mathcal{R} \subseteq \sim$.*

Proof. We prove $\sim\mathcal{R}\sim$ is a bisimulation. The only unusual transition μ is when $\mu = [\varphi]\tau$, but from $P \sim P_1 \; \mathcal{R} \; Q_1 \sim Q$ and $P \xrightarrow{\mu} P'$ we know:

- that $P_1 \xrightarrow{\mu} P_1'$ and $(P' \mid \varphi) \sim (P_1' \mid \varphi)$ from the $\sim$ game,
- that $Q_1 \xrightarrow{\mu} Q_1'$ and $(P_1' \mid \varphi) \sim\mathcal{R}\sim (Q_1' \mid \varphi)$ from the $\mathcal{R}$ up-to game,
- that $P \xrightarrow{\mu} P'$ and $(Q_1' \mid \varphi) \sim (Q' \mid \varphi)$ from the second $\sim$ game.

So the case for a $[\varphi]\tau$ transition is no more difficult than usual: we conclude by transitivity of $\sim$.

Lemma 37. *$P \sim Q$ implies $(\nu a)P \sim (\nu a)Q$ for all a.*

Proof. We prove $\mathcal{R} \triangleq \{(\nu aP, \nu aQ) \mid P \sim Q\}$ is a bisimulation up to bisimilarity (Lemma 36), which is relatively easy using Lemma 31. The only interesting case is for a conditional τ transition. If $\nu aP \xrightarrow{[\varphi]\tau} \nu aP'$ then $P \xrightarrow{[\varphi]\tau} P'$ and using $\sim$, $Q \xrightarrow{[\varphi]\tau} Q'$ with $(P' \mid \varphi) \sim (Q' \mid \varphi)$. Since we want to relate $(\nu a)P' \mid \varphi$ and $(\nu a)Q' \mid \varphi$ we $\sim$-rewrite them into $(\nu a)(P' \mid \varphi)$ and $(\nu a)(Q' \mid \varphi)$ (using Lemma 34) which are indeed related through $\mathcal{R}$. The condition about $\rhd$ is ensured by compositionality of $\rhd$ (Lemma 16).

Definition 38 (Bisimulation up to $\sim$ and ν). *A relation $\mathcal{R}$ is a bisimulation up to restriction and bisimilarity if it validates the clauses of the usual bisimulation, except that the outcomes of the transitions, P_1 and Q_1, are requested to satisfy $P_1 \sim (\nu\widetilde{a})P_2$ and $Q_1 \sim (\nu\widetilde{a})Q_2$ with $P_2 \; \mathcal{R} \; Q_2$, where $\widetilde{a}$ stands for a (possibly empty) tuple of names.*

Lemma 39. *If $\mathcal{R}$ is a bisimulation up to restriction and bisimilarity then $\mathcal{R} \subseteq \sim$.*

Proof. We write $\mathcal{R}^\nu$ for $\{(\nu\tilde{a})P, (\nu\tilde{a})Q) \mid P\ \mathcal{R}\ Q\}$. We know that $\mathcal{R} \subseteq \mathcal{S} \triangleq \sim\mathcal{R}^\nu\sim$ so it is enough to prove that $\mathcal{S}$ is a bisimulation. Suppose $P \sim (\nu\tilde{a})P_1\ \mathcal{R}^\nu\ (\nu\tilde{a})Q_1 \sim Q$ with $P_1\ \mathcal{R}\ Q_1$. We start from a transition $P \xrightarrow{\mu} P'$, we use the bisimulation game on $\sim$ and Lemma 31 to deduce $(\nu\tilde{a})P_1 \xrightarrow{\mu} (\nu\tilde{a})P_1'$ with $P_1 \xrightarrow{\mu} P_1'$ and $(P' \mid A) \sim ((\nu\tilde{a})P_1' \mid A)$ with $A = \varphi$ if $\mu = [\varphi]\tau$ or $A = 0$ otherwise. Using the bisimulation up-to on $\mathcal{R}$ we obtain $Q_1 \xrightarrow{\mu} Q_1'$ with $(P_1' \mid A) \sim (\nu\tilde{c})P_2\ \mathcal{R}^\nu\ (\nu\tilde{c})Q_2 \sim (Q_1' \mid A)$ for some $P_2\ \mathcal{R}\ Q_2$. We also get $(\nu\tilde{a})Q_1 \xrightarrow{\mu} (\nu\tilde{a})Q_1'$ from which using the game on the second $\sim$, $Q \xrightarrow{\mu} Q'$ with $((\nu\tilde{a})Q_1' \mid A) \sim (Q' \mid A)$.

We now compose what we have: $(P' \mid A) \sim ((\nu\tilde{a})P_1' \mid A)$ and $(P_1' \mid A) \sim ((\nu\tilde{c})P_2)$. We can get from the latter $((\nu\tilde{a})P_1' \mid A) \sim ((\nu\tilde{ac})P_2)$ using Lemmas 34 and 37, and then compose them using transitivity of $\sim$; similarly for Q: $(Q' \mid A) \sim ((\nu\tilde{ac})Q_2)$. Since $P_2\ \mathcal{R}\ Q_2$, the pair of $P' \mid A$ and $Q' \mid A$ is in $\mathcal{S}$.

Definition 40 (Bisimulation up to $\sim$ and σ). *A relation $\mathcal{R}$ is a bisimulation up to bisimilarity and injective substitution if it validates the clauses of the usual bisimulation, except that the outcomes of the transitions, P_1 and Q_1, are requested to satisfy $P_1 \sim P_{2\sigma}$ and $Q_1 \sim Q_{2\sigma}$ with $P_2\ \mathcal{R}\ Q_2$, where σ stands for an injective name substitution.*

Proof. We prove $\sim\ \mathcal{R}_\sigma\ \sim$ is a bisimulation where $\mathcal{R}_\sigma$ stands for $\{(P_\sigma, Q_\sigma) \mid P\ \mathcal{R}\ Q$ and σ is injective$\}$.

Lemma 41. *$P \sim Q$ implies $P \mid R \sim Q \mid R$ for all R.*

Proof. We show that

$$\mathcal{R} \triangleq \{(P \mid R, Q \mid R) \mid P \sim Q\}$$

is a bisimulation up to restriction and bisimilarity. The condition about induced φ can be dealt with using Lemma 16: since $(\forall\psi,\ P \rhd \psi \Leftrightarrow Q \rhd \psi)$ by definition of $\sim$, we have $P \mid R \rhd \varphi$ implies $Q \mid R \rhd \varphi$ and symmetrically.

Now suppose $P \mid R \xrightarrow{\mu} P_1$ with $\mathsf{bn}(\mu) \cap \mathsf{fn}(P, Q, R) = \emptyset$. We can suppose by Lemma 29 that for some μ_1 with $\mu \sqsubseteq_{P|R} \mu_1$, $P \mid R \xrightarrow{\mu_1} P_1$, and that the last rule applied is not a preorder rule. Since $\Phi(P \mid R) = \Phi(Q \mid R)$, $\mu \sqsubseteq_{Q|R} \mu_1$, and it is enough to get $Q \mid R \xrightarrow{\mu_1} Q_1$ to prove $Q \mid R \xrightarrow{\mu} Q_1$. There are two cases in the bisimulation game, depending on μ_1:

Proof. We prove $\{(P \mid R, Q \mid R) \mid P \sim Q\}$ is a bisimulation up to restriction and bisimilarity, and (thanks to Lemmas 16 and 29) we only focus on the clause about $[\varphi]\tau$ transitions, where Lemma 29 gives φ_1 s.t. $[\varphi]\tau \sqsubseteq_{P|R} [\varphi_1]\tau$ and $Q \mid R$..

We prove that $\mathcal{R} = \{(P \mid R, Q \mid R) \mid P \sim Q\}$ is a bisimulation up to restriction and bisimilarity. Lemma 16 handles the first clause about conditions, and Lemma 29 the second one about visible transitions. Lemma 29 also helps handling a $[\varphi]\tau$ transition from $P \mid R$ to P_1 by saying it comes from a $[\varphi_1]\tau$

transition from P, or R, or both, and such that $[\varphi]\tau \sqsubseteq_{P|R} [\varphi_1]\tau$. From that, getting a transition $[\varphi]\tau$ from $Q \mid R$ to Q_1 is easy, but one must relate $P_1 \mid \varphi$ and $Q_1 \mid \varphi$ in $\mathcal{R}$ using three different assumptions:

1. $R \xrightarrow{[\varphi_1]\tau} R'$. Then $P \mid R' \mid \varphi \sim \mathcal{R} \sim Q \mid R' \mid \varphi$. (up to $\sim$)
2. $P \xrightarrow{[\varphi_1]\tau} P'$. Then $P' \mid \varphi_1 \sim Q' \mid \varphi_1$, that implies $P' \mid R \mid \varphi \sim \mathcal{R} \sim Q' \mid R \mid \varphi$. Indeed φ_1 is absorbed by φ since $P \mid Q \mid \varphi \rhd \varphi_1$. (up to $\sim$)
3. P and R synchronised into $(\nu x)(P' \mid R')$. Then $P' \sim Q'$ and we can relate $(\nu x)(P' \mid R') \mid \varphi$ to $(\nu x)(Q' \mid R') \mid \varphi$ (up to $\sim$ and ν). □

Lemma 42. *For any P, Q and φ, $P \xrightarrow{[\varphi]\tau} Q$ iff $P \mid \varphi \xrightarrow{\tau} Q \mid \varphi$.*

Theorem 43 (Characterisation). *$P \simeq Q$ iff $P \sim Q$.*

Proof (Sketch). The proof follows a standard pattern, and exploits the lemmas stated above. For completeness, we define tester processes, corresponding to the three clauses in Def. 6. The first one is handled using $[\varphi]\tau.\overline{w}$. We use φ for the second clause (by Lemma 42). To test, e.g., $\xrightarrow{\{a\}(x)}$, we use $\overline{a}(y).(z/y \mid \overline{w} \mid w)$. □

3.3 Protected Names and Cocapabilities

Protected prefixes act as cocapabilities, by expressing the ability to react to an interaction offer from the context (cf. P_0 in Section 1). We can exploit this idea to define an extension of i/o-types [14,8] for $\pi\mathtt{P}$, by introducing *cocapability types* $\overline{\mathtt{o}}T$ and $\overline{\mathtt{i}}T$.

We refer to Appendix B for a description of this type system.

4 Axiomatisation

We now move to the axiomatisation of $\sim$, which is given by the laws of Figure 2.

The overall strategy to define the axiomatisation differs from the standard approach. This is mainly due to the handling of restriction, and its interplay with the state component defined by the toplevel preorder processes. Intuitively, pushing restrictions under prefixes can be done only if the toplevel prefixes of a process have somehow been saturated. As a consequence, restriction must be handled together with prefixes and sums. We first show how equational laws allow us to relate processes at toplevel only, and then derive an axiomatisation for full processes.

4.1 Preliminaries

As is usually the case in axiomatisations, we first study the processes consisting only of sums, leaving the treatment of parallel composition for later.

We make two remarks, before embarking in the presentation of the equational laws for sums. The first remark is that in $\pi\mathtt{P}$, we have to deal with preorder processes, that act somehow like a persistent state, and are used in stating

the equational laws. The second remark is that the treatment of restriction in the axiomatisation requires some particular care, essentially because of the presence of preorder processes. Moreover, contrarily to what is usually the case, the restriction has to be dealt with together with prefixes and sums, as we show below.

We first focus on processes without restriction, for which the axiomatisation can be presented using simpler laws. The ideas presented in this case are at work, in a somewhat more complicated fashion, when handling restriction in Section 4.4.

We introduce a dedicated notation to factor out the preorder part in processes.

Notations and Terminology. We use A to range over processes that consist of φ processes only, which we call *preorder processes*. We often view such processes as multisets of conditions. We use notation A, P to denote a process that can be written, *using the monoid laws for* $|$, as $A \mid P$, where P does not contain toplevel arcs. Note that A may contain restrictions, corresponding to the definition of join processes (given towards the end of Section 2.1), but extrusion is not allowed to decompose a process as A, P. Note that we could have adopted a more complex definition, in order to decompose, for instance, $(\nu u)(u/a \mid u/b \mid c/d \mid \overline{n}m.P)$ into $a \curlyvee b \mid c/d, \overline{n}m.P$ (which involves an application of name extrusion), but this simpler definition is sufficient for our purposes.

Accordingly, A, S stands for $A \mid S$, where S is a sum process, and 0 is written $\emptyset, 0$.

We use the following notion of measure to reason by induction on processes.

Definition 44 (Measure on processes). *Given a $\pi\mathtt{P}$ process P, we define $|P|$ as the maximum number of prefixes in summands of P, i.e., $|\sum_i \pi_i.P_i| = \max_i (1 + |P_i|)$ (hence $|0| = 0$), $|(\nu a)P| = |P|$, $|P \mid Q| = |P| + |Q|$, and $|a/b| = 0$.*

We write $\vdash P = Q$ whenever P and Q can be related by equational reasoning using the laws of Figure 2.

We omit the laws for equational reasoning (equivalence, substitutivity). We will reason up to these laws, and up to the standard laws expressing that $|$ and $+$ obey the laws of commutative monoids, and that $+$ is idempotent, in the remainder.

As usual, expansion allows us to rewrite the parallel composition of two sum processes into a sum, the third summand describing synchronisation in $\pi\mathtt{P}$.

Note that α-conversion for input prefixes follows from Laws L26 and L24, by deriving the following equalities (and similarly for the other visible prefixes):

$a(y).P \overset{L24}{=} a(x).(\nu y)(x/y \mid P) \overset{L26}{=} a(x).(\nu y')(x/y' \mid P\{y'/y\}) \overset{L24}{=} a(y').P\{y'/y\}$.

The remaining laws are more specific to $\pi\mathtt{P}$, and are analysed below, in Propositions 46, 49 and 52. The latter results are used to establish completeness of the axiomatisation.

The laws are sound:

Lemma 45. *The laws of Figure 2 relate bisimilar processes.*

Proof (Sketch). For laws 30-36, we establish a "saturation property", expressing the fact that when erasing a preorder process φ or a prefix π that mentions a, we generate all processes φ or π could induce. The other laws are easy. □

Standard structural laws

L1	$(P_1 \mid P_2) \mid P_3 = P_1 \mid (P_2 \mid P_3)$		L4	$(S_1 + S_2) + S_3 = S_1 + (S_2 + S_3)$	
L2	$P_1 \mid P_2 = P_2 \mid P_1$		L5	$S_1 + S_2 = S_2 + S_1$	
L3	$P_1 \mid 0 = P_1$		L6	$\pi.P + \pi.P = \pi.P$	

Expansion law (we can suppose $x \neq y$, $\mathsf{bn}(\pi_i) \notin \mathsf{fn}(T)$, $\mathsf{bn}(\rho_j) \notin \mathsf{fn}(S)$.)

L7 $\sum_i \pi_i.P_i \mid \sum_j \rho_j.R_j = \sum_i \pi_i.(P_i \mid T) + \sum_j \rho_j.(S \mid R_j) + \sum_{i,j} [\alpha \curlyvee \beta]\tau.(\nu xy)(x/y \mid P_i \mid R_j)$ when $\alpha \curlyvee \beta$ is defined and $\{\pi_i, \rho_j\} = \{\overline{\alpha}(x), \beta(y)\}$

Laws for preorder processes

L8	$a \leqslant b \mid b \leqslant c = a \leqslant b \mid b \leqslant c \mid a \leqslant c$	L9	$a \leqslant b \mid c \leqslant b = a \leqslant b \mid c \leqslant b \mid a \curlyvee c$
L10	$a \leqslant b \mid b \curlyvee c = a \leqslant b \mid b \curlyvee c \mid a \curlyvee c$	L11	$a \leqslant a = 0$

Laws for prefixes (counterparts of Laws L16-L19 for output are omitted)

L12	$[\varphi]\tau.P = [\varphi]\tau.(\varphi \mid P)$	L13	$\varphi, S + \pi.P = \varphi, S + \pi.(\varphi \mid P)$
L14	$[a \leqslant a]\tau.P = [b \curlyvee b]\tau.P$	L15	$[a \curlyvee b]\tau.P = [a \curlyvee b]\tau.P + [a \leqslant b]\tau.P$
L16	$a(x).P = a(x).P + \{a\}(x).P$	L17	$[a \curlyvee b]\tau.P = [a \curlyvee b]\tau.P + [b \curlyvee a]\tau.P$
L18	$b/a, S + a(x).P = b/a, S + a(x).P + b(x).P$		
L19	$a/b, S + \{a\}(x).P = a/b, S + \{a\}(x).P + \{b\}(x).P$		
L20	$b/a, S + [a \leqslant c]\tau.P = b/a, S + [a \leqslant c]\tau.P + [b \leqslant c]\tau.P$		
L21	$a/b, S + [c \leqslant a]\tau.P = a/b, S + [c \leqslant a]\tau.P + [c \leqslant b]\tau.P$		
L22	$b/a, S + [a \curlyvee c]\tau.P = b/a, S + [a \curlyvee c]\tau.P + [b \curlyvee c]\tau.P$		
L23	$b/a, S + [a \curlyvee c]\tau.P = b/a, S + [a \curlyvee c]\tau.P + [c \leqslant b]\tau.P$		
L24	$\alpha(y).P = \alpha(x).(\nu y)(x/y \mid P)$ if $x \notin \mathsf{fn}(P)$		
L25	$\overline{\alpha}(y).P = \overline{\alpha}(x).(\nu y)(y/x \mid P)$ if $x \notin \mathsf{fn}(P)$		

Laws for restriction (counterparts of Laws L31 and L32 for output are omitted; $a \leqslant b \in A^{\neq}$ stands for $a \leqslant b \in A$ and $a \neq b$, and similarly for $a \curlyvee b$.)

L26	$(\nu b)P = (\nu a)(P\{a/b\})$ if $a \notin \mathsf{fn}(P)$	L27	$(\nu c)(\nu d)P = (\nu d)(\nu c)P$
L28	$P \mid (\nu a)Q = (\nu a)(P \mid Q)$ if $a \notin \mathsf{fn}(P)$	L29	$(\nu a)0 = 0$

L30 $(\nu a)A = \{b \leqslant c \mid b \leqslant a,\ a \leqslant c \in A^{\neq}\} \uplus \{b \curlyvee c \mid b \leqslant a,\ c \leqslant a \in A^{\neq}\} \uplus \{b \curlyvee c \mid a \curlyvee c,\ b \leqslant a \in A^{\neq}\} \uplus \{\varphi \in A \mid a \notin \mathsf{n}(\varphi)\}$

L31 $(\nu a)(A,\ S + a(x).P) = (\nu a)\big(A,\ S + \sum_{a \leqslant b \in A^{\neq}} b(x).(\nu a)(A \mid P) + \sum_{\substack{b \leqslant a \in A^{\neq} \\ \vee\, a \curlyvee b \in A^{\neq}}} \{b\}(x).(\nu a)(A \mid P)\big)$

L32 $(\nu a)(A,\ S + \{a\}(x).P) = (\nu a)\big(A,\ S + \sum_{b \leqslant a \in A^{\neq}} \{b\}(x).(\nu a)(A \mid P)\big)$

L33 $(\nu a)(A,\ S + [a \leqslant c]\tau.P) = (\nu a)\big(A,\ S + \sum_{a \leqslant b \in A^{\neq}} [b \leqslant c]\tau.(\nu a)(A \mid P)\big)$ $a \neq c$

L34 $(\nu a)(A,\ S + [c \leqslant a]\tau.P) = (\nu a)\big(A,\ S + \sum_{b \leqslant a \in A^{\neq}} [c \leqslant b]\tau.(\nu a)(A \mid P)\big)$ $a \neq c$

L35 $(\nu a)(A,\ S + [a \curlyvee c]\tau.P) = (\nu a)\big(A,\ S + \sum_{a \leqslant b \in A^{\neq}} [b \curlyvee c]\tau.(\nu a)(A \mid P) + \sum_{\substack{b \leqslant a \in A^{\neq} \\ \vee\, a \curlyvee b \in A^{\neq}}} [c \leqslant b]\tau.(\nu a)(A \mid P)\big)$ $a \neq c$

L36 $(\nu a)(A,\ S + \pi.P) = (\nu a)\big(A,\ S + \pi.(\nu a)(A \mid P)\big)$ $a \notin \mathsf{n}(\pi)$

Fig. 2. An axiomatisation of $\sim$

4.2 Laws for preorder processes.

Laws L8-L11 are used to *saturate* preorder processes, as expressed by the following result.

Proposition 46. *If $A_1, S_1 \sim A_2, S_2$, then there exists $A^\star$ such that $\vdash A_i, S_i = A^\star, S_i$ $(i = 1, 2)$, and $A^\star = \prod\{\varphi \mid \varphi$ not reflexive and $A_1 \rhd \varphi\}$.*

(Note that we could have picked A_2 instead of A_1 above.)

Proof. We define a rewriting relation on preorder processes, and write $A \overset{\Upsilon}{\mapsto} A'$ whenever A' is obtained from A using one of the laws L8-L11, oriented from left to right, as a rewrite rule modulo associativity and commutativity of parallel composition. We furthermore impose that no reflexive condition is added in a rewrite step, nor a condition that is already contained in the preorder process.

We then prove the following three properties about $\overset{\Upsilon}{\mapsto}$:

1. If $A \overset{\Upsilon}{\mapsto} A'$, then $A \sim A'$: this is a consequence of Lemma 45 (or, alternatively, by Lemma 33).
2. For any (finite) A, there is no infinite $\overset{\Upsilon}{\mapsto}$-chain emanating from A.
   Indeed, the rules defining $\overset{\Upsilon}{\mapsto}$ do not introduce any new name. Moreover, a new arc or join can only be added if it is not already present. Since there are finitely many conditions built on a finite set of names, $\overset{\Upsilon}{\mapsto}$ terminates.
3. Suppose A is a $\overset{\Upsilon}{\mapsto}$-normal form. Then, for any non-reflexive φ, if $A \rhd \varphi$, then φ appears in A (which we write $\varphi \in A$).
   This follows by induction on the derivation of $A \rhd \varphi$. The only interesting case is for the $\rhd$-COMBINE rule, i.e. we know that $A \rhd \Gamma$ and $\Gamma \vdash \varphi$. We conclude by associating to rules $\vdash$-TRANS, $\vdash$-JOIN and $\vdash$-EXTJOIN laws L8, L9 and L10 respectively.

The observations above entail the expected property. □

We say that A is a *saturated preorder process* whenever $A^\star \equiv A$. We use $A^\star$ to range over such processes. We can remark that even if A contains only arcs, $A^\star$ may contain restrictions, because of induced conditions involving Υ.

4.3 Laws for Prefixes and Sums

We comment on the laws for prefixes in Figure 2. We first remark that rewriting a sum process using these laws yields a sum process, and the same holds for the laws for preorder processes.

Law L12 expresses the fact that the condition φ should be enforced after a $[\varphi]\tau$ transition (see the third clause of the definition of $\sim$, Definition 6). Law L13 propagates φs in depth, expressing the persistence of condition processes (φ).

Law L14 is used to equate all plain τ prefixes.

Laws L24-L25 are analogous to law L12, because they are related to how the process is observed after the prefix has been fired (see transition rules IN, OUT, PR-I and PR-O, as well as Fact 1).

Laws L18-L23 describe how arcs act on prefixes, by triggering new interaction possibilities.

The next lemma relates transitions of sum processes and the laws for prefixes. In the statement, we say that two prefixes π and π' *only differ in their bound names* whenever either π and π' are visible actions of the same kind, in which case they have the same subject name, or they are a conditional τ, in which case $\pi = \pi'$.

Lemma 47. *If $A, S \xrightarrow{\mu} A, P$ then $\vdash A, S = A, S + \pi.Q$ for some π and Q such that μ and π only differ in their bound names and $\pi.Q \xrightarrow{\mu} P$.*

Proof. Suppose S is of the form $S_1 + \pi'.Q$ and that the transition $S \xrightarrow{\mu} P$ is coming from from $\pi'.Q \xrightarrow{\mu} P$ and in fact (Lemma 29) from $\pi'.Q \xrightarrow{\mu'} P$ such that $\mu \sqsubseteq_{A,S} \mu'$. Since $\Phi(A, S) = \Phi(A)$ we know also that $\mu \sqsubseteq_A \mu'$.

Then we have directly $\pi \sqsubseteq_A \pi'$. We prove by induction on $\pi \sqsubseteq_A \pi'$ that for all $S, \vdash A, S + \pi'.Q = A, S + \pi'.Q + \pi.Q$.

- Reflexivity of $\sqsubseteq$ is handled by the fact that $+$ is idempotent.
- Transitivity (e.g. $\pi_3 \sqsubseteq_A \pi_2 \sqsubseteq_A \pi_1$) is handled by monoid laws for $+$. We write Q_i for $\pi_i.Q$ below. We know by induction that:
  (1) $\vdash A, S + Q_1 = A, S + Q_1 + Q_2$ (for all S) and
  (2) $\vdash A, S + Q_2 = A, S + Q_2 + Q_3$ (for all S). Then:

$$\begin{aligned}
&\vdash\ A, S + Q_1 =_{(1)}\\
&\vdash\ A, S + Q_1 + Q_2 =_{(\equiv)}\\
&\vdash\ A, (S + Q_1) + Q_2 =_{(2)}\\
&\vdash\ A, (S + Q_1) + Q_2 + Q_3 =_{(\equiv)}\\
&\vdash\ A, (S + Q_3) + Q_1 + Q_2 =_{(1)}\\
&\vdash\ A, (S + Q_3) + Q_1 =_{(\equiv)}\\
&\quad\ A, S + Q_1 + Q_3
\end{aligned}$$

  so now we can only have to prove it for $\pi \sqsubseteq^{\varphi} \pi'$ when $A \rhd \varphi$. Note that using the reasoning about we can work up to transitivity.
- We now decompose $\sqsubseteq^{\varphi}$ when $A \rhd \varphi$. Lemma 25 tells us[3] there are some $\varphi_1, \ldots, \varphi_n \in A$ and a reflexive ψ such that $\sqsubseteq^{\varphi} = \sqsubseteq^{\psi} \sqsubseteq^{\varphi_1} \ldots \sqsubseteq^{\varphi_n}$ so in fact we only need to prove the result when φ is actually in A or when it is reflexive:
  - φ is reflexive and $\alpha(x) \sqsubseteq^{a \leqslant a} \alpha(x)$: nothing to do (+ idempotent)
  - φ is reflexive and $\{a\}(x) \sqsubseteq^{a \curlyvee a} a(x)$: Law L16
  - $\varphi \in A$ and $\alpha(x) \sqsubseteq^{\beta \leqslant \alpha} \beta(x)$: this yields several cases:
    - $a(x) \sqsubseteq^{b \leqslant a} b(x)$: Law L18
    - $\{a\}(x) \sqsubseteq^{a \leqslant b} \{b\}(x)$: Law L19
    - $\{a\}(x) \sqsubseteq^{a \curlyvee b} b(x)$: decompose $a \curlyvee b$ back into $u/a \mid u/b$ (Law L30) then from $b(x)$ get $u(x)$ by L18, then $\{u\}(x)$ (L16) and then $\{a\}(x)$ (L19).
  - φ is reflexive and $[\varphi_1]\tau \sqsubseteq^{\varphi} [\varphi_2]\tau$: either + idempotent or L15 is sufficient.
  - $\varphi \in A$ and $[\varphi_1]\tau \sqsubseteq^{\varphi} [\varphi_2]\tau$ when $\varphi_1, \varphi \vdash \varphi_2$: this yields several cases again, we can break $\vdash$ into compositions of $\vdash^{\downarrow}$, again reasoning up to transitivity:

[3] in fact, the ψ is not absolutely necessary, but we make this analysis using a more precise version of $\sqsubseteq^{\cdot}$ using $\vdash^{\downarrow}$ instead of $\vdash$ so the case analysis is less verbose

* $[a \leqslant b]\tau \sqsubseteq^{b \leqslant c} [a \leqslant c]\tau$: instance of L21
* $[a \leqslant c]\tau \sqsubseteq^{b \leqslant c} [a \curlyvee b]\tau$: instance of L23 (using L17)
* $[a \curlyvee b]\tau \sqsubseteq^{c \leqslant a} [c \curlyvee b]\tau$: instance of L22
* $[a \curlyvee b]\tau \sqsubseteq^{c \leqslant b} [a \curlyvee c]\tau$: instance of L22 (using L17)
* $[b \leqslant c]\tau \sqsubseteq^{a \leqslant b} [a \leqslant c]\tau$: instance of L20
* $[b \leqslant c]\tau \sqsubseteq^{a \leqslant c} [a \curlyvee b]\tau$: instance of L23
* $[c \leqslant a]\tau \sqsubseteq^{a \curlyvee b} [c \curlyvee b]\tau$: same as below
* $[c \leqslant b]\tau \sqsubseteq^{a \curlyvee b} [a \curlyvee c]\tau$: from $a \curlyvee b$ we can get some u/a and u/b. Then, from $[a \curlyvee c]\tau$ we can add the summand $[u \curlyvee c]\tau$ (L22), then $[c \leqslant u]\tau$ (L15), then $[c \leqslant b]\tau$ (L21). □

Laws L15-L23 can be used to "saturate" the topmost prefixes in sums. We express this using the equivalence below, and rely on Lemma 47 to prove Prop. 49:

Definition 48 (Head sum normal form, $\asymp_h$). *Given two sum processes S and T, we write $S \prec_h T$ whenever for any summand $\pi.P$ of S, there exists a summand $\pi.Q$ of T with $\pi.P \sim \pi.Q$. We let $S \asymp_h T$ stand for $S \prec_h T \wedge T \prec_h S$.*

Proposition 49. *Whenever $A^\star, S_1 \sim A^\star, S_2$, where S_1, S_2 are two sum processes, there are S_1', S_2' s.t. $\vdash A^\star, S_i = A^\star, S_i'$ (for $i = 1, 2$) and $S_1' \asymp_h S_2'$.*

Proof. We first use law L13 to replicate $A^\star$ under all prefixes in S_1 and S_2, which is useful later in the proof. We therefore suppose that for any summand $\pi.P$ of S_1 or S_2, $P = A^\star \mid P_0$ for some P_0.

We prove the following property:

$$\begin{array}{l} A^\star, S_1 \sim A^\star, S_2 \\ \pi.P \in S_1 \end{array} \quad \Rightarrow \quad \exists Q \quad \begin{array}{l} \vdash A^\star, S_2 = A^\star, S_2 + \pi.Q \\ \pi.P \sim \pi.Q \end{array} \tag{1}$$

by running the bisimulation game with a label μ such that π and μ differ only on their object (that should be fresh in μ): $\pi.P \xrightarrow{\mu} P'$, yielding $A^\star, S_1 \xrightarrow{\mu} A^\star \mid P'$; the game returns a transition $A^\star, S_2 \xrightarrow{\mu} A^\star \mid Q'$. Using Lemma 47 we get Q such that $A^\star, S_2 = A^\star, S_2 + \pi.Q$ and $\pi.Q \xrightarrow{\mu} Q'$. We now have to prove that $\pi.P \sim \pi.Q$. There are two cases:

1. if μ is a visible action, then, by definition of $\sim$, we have $A^\star \mid P' \sim A^\star \mid Q'$. We can now observe that $P' \sim A^\star \mid P'$ and $Q' \sim A^\star \mid Q'$, because $A^\star$ has been replicated under prefixes. We thus deduce $P' \sim Q'$, which, by congruence, gives $\mu.P' \sim \mu.Q'$. Using the appropriate law among L24-25, we deduce $\pi.P \sim \mu.P'$ and $\pi.Q \sim \mu.Q'$, which implies $\pi.P \sim \pi.Q$.
2. if $\mu = [\varphi]\tau$ then $\pi = \mu$ and $P' = P$, $Q' = Q$. The bisimulation game yields $\varphi \mid A^\star \mid P' \sim \varphi \mid A^\star \mid Q'$ and by the same reasoning as before, $\varphi \mid P' \sim \varphi \mid Q'$. By congruence $[\varphi]\tau.(\varphi \mid P') \sim [\varphi]\tau.(\varphi \mid Q')$, and by L12, $[\varphi]\tau.P' \sim [\varphi]\tau.Q'$ i.e. $\pi.P \sim \pi.Q$.

We have now (1). Equation (1) implies that $\vdash A^\star, S_2 = A^\star, S_2 + T_2$ with $S_1 \prec_h S_2 + T_2$ and $T_2 \prec_h S_1$.

Using the symmetry on $S_2 + T_2$ (and not on S_2), we get T_1 such that $\vdash A^\star, S_1 = A^\star, S_1 + T_1$ with $S_2 + T_2 \prec_h S_1 + T_1$ and $T_1 \prec_h S_2 + T_2$. Since $S_1 \prec_h S_2 + T_2$ as well, we can conclude $S_1 + T_1 \prec_h S_2 + T_2$ and thus $S_1' \asymp_h S_2'$ with $\vdash S_i' = S_i + T_i$. □

Remark 50 (On the definition of $\asymp_h$). In the definition of $\prec_h$, we impose $\pi.P \sim \pi.Q$, and not simply $P \sim Q$. The equivalence induced by the choice of the latter condition would indeed be too discriminating. To see why, consider $P_0 = a(x).c\not x$ and $Q_0 = a(x).0$. Obviously, $c\not x \not\sim 0$. On the other hand, we have $P_0 \sim Q_0$: after a $\xrightarrow{a(y)}$ transition on both sides, we must compare $(\nu x)(c\not x \mid y\not x)$ and $(\nu x)(y\not x)$, and both are bisimilar to 0. In order to derive $\vdash P_0 = Q_0$, we rely on the following property, which explains the shape of laws L24,L25: $a(y).P \sim a(y).Q$ iff $(\nu y)(x\not y \mid P) \sim (\nu y)(x\not y \mid Q)$.

A similar reasoning can be done for the other kinds of visible prefixes.

4.4 Laws for Restriction

Laws L26-L29 are standard. The other laws are used to "push" restrictions inside processes. Law L30 is used to eliminate a restriction on a name a in a preorder process, by propagating the information expressed by all φs that mention a. Intuitively, L30 is used after laws L31-L36 have been used to erase all prefixes mentioning the restricted name a, pushing the restriction on a inwards. The latter laws describe a kind of "synchronous application" of the prefix laws seen above.

Below are some equalities that are derivable using the laws for restriction:

$$(\nu a)(b\not a \mid a\not c) = b\not c \qquad (\nu a)(a\not b \mid \overline{a}(x).P) = \overline{\{b\}}(x).P \qquad \overline{a}(x).x = \overline{a}(x).\{x\}$$
$$(\nu a)(S{+}a(x).P) = (\nu a)S \qquad (\nu a)(S{+}b(x).P) = (\nu a)(S{+}b(x).(\nu a)P) \text{ if } a \neq b$$

It can be noted that axiomatisations often treat restriction separately, by first focusing on a restriction-free calculus. In $\pi\mathtt{P}$, because of preorders, we cannot in general push restrictions on top of sum processes, so the situation is more complex.

It can be noted that we cannot rely on a law like $(\nu a)(S_1 + S_2) = (\nu a)S_1 + (\nu a)S_2$, for two reasons. On the one hand, we do not allow sum of restricted processes. On the other hand, a restricted process has in general a preorder component in $\pi\mathtt{P}$, so the left hand side of the equation would rather look like $(\nu a)(A, S_1 + S_2)$, and we need to exploit the information contained in A when pushing restrictions under prefixes.

We first present a technical lemma about $\rhd$.

Lemma 51. *Suppose $a \neq b$.*

1. *If $A \rhd a \leqslant b$ then for some $x \neq a$, $a \leqslant x \in A$ and $A \rhd x \leqslant b$.*
2. *If $A \rhd b \leqslant a$ then for some $x \neq a$, $x \leqslant a \in A$ and $A \rhd b \leqslant x$.*
3. *If $A \rhd a \curlyvee b$ then for some $x \neq a$,*
   (a) $a \leqslant x \in A$ and $A \rhd x \curlyvee b$, or
   (b) $\{a \curlyvee x, x \curlyvee a, x \leqslant a\} \cap A \neq \emptyset$ and $A \rhd b \leqslant x$.

Proof. By induction on the derivation of $A \rhd \cdot$.

Proposition 52 (Pushing restriction inwards).
For any A, S, where A is a preorder process and S is a sum process, and for any set of names $\widetilde{a}$, there exist A' and S', where S' is a sum process and A' is a preorder process such that $\vdash (\nu\widetilde{a})(A, S) = A', S'$ and $|(\nu\widetilde{a})(A, S)| \geq |A', S'|$.

Proof. Using name extrusion, we pull all toplevel restrictions of A, S in order to derive $\vdash A, S = (\nu\widetilde{a})(A_0, S_0)$, for some A_0, S_0 without toplevel restriction.

We then reason by induction over the number of names in $\widetilde{a}$. We apply laws L31-L36 from left to right, until name a does not appear free in any topmost prefix of the sum. At that point, since the restriction on a has been pushed under prefixes, a has no free occurrence in the sum. We can thus use name intrusion, so that law L30 can be applied to get rid of the restriction on a on the preorder part of the process.

This operation is iterated until restrictions are pushed under all prefixes, and law L29 can be used to get rid of the restriction. □

It is often the case that axiomatisations for strong bisimilarity are first given for a calculus without restriction and parallel composition, and then extended to a calculus enriched with these operators. In our case, as discussed above, restriction cannot be handled separately from prefixes. We can however prove a result for the calculus without parallel composition.

4.5 Axiomatisation of Strong Bisimilarity

The results presented so far can be put together to establish an axiomatisation on the subcalculus of $\pi\mathtt{P}$ in which parallel composition is only used to compose preorder processes.

We first present some observations, that follow from the shape of the prefix rules in Figure 1.

Fact 1

1. *For any P and Q, $a(y).P \sim a(y).Q$ iff $(\nu y)(x/y \mid P) \sim (\nu y)(x/y \mid Q)$.*
2. *For any P and Q, $\overline{a}(y).P \sim \overline{a}(y).Q$ iff $(\nu y)(y/x \mid P) \sim (\nu y)(y/x \mid Q)$.*
3. *For any P and Q, $\{a\}(y).P \sim \{a\}(y).Q$ iff $(\nu y)(y/x \mid P) \sim (\nu y)(y/x \mid Q)$.*
4. *For any P and Q, $\{\overline{a}\}(y).P \sim \{\overline{a}\}(y).Q$ iff $(\nu y)(x/y \mid P) \sim (\nu y)(x/y \mid Q)$.*

The grammar $P ::= A, \sum_i \pi_i.P_i \mid (\nu a)P$ defines what we call $|$*-free processes*: only arcs are composed, and the non-preorder part is a sum.

Proposition 53 (Characterisation on the calculus without parallel composition). *For all $|$-free processes P and Q, $P \sim Q$ iff $\vdash P = Q$.*

Proof. The right to left implication follows from Lemma 45 and congruence of $\sim$.

Suppose now $P \sim Q$. We reason by induction on $|P| + |Q|$.

By Proposition 52, there are sum-only processes P_0, Q_0 with no toplevel restriction such that $\vdash P = P_0$ and $\vdash Q = Q_0$.

We then reason up to associativity and commutativity of parallel composition to write $\vdash P_0 = A_1, S_1$ and $\vdash Q_0 = A_2, S_2$. We have $A_1, S_1 \sim A_2, S_2$, which gives, by Proposition 46, $\vdash A_i, S_i = A^\star, S_i$ for $i = 1, 2$, for some $A^\star$.

We can then apply Proposition 49 to deduce $\vdash A^\star, S_i = A^\star, S'_i$, for $i = 1, 2$, for some S'_1, S'_2 s.t. $S'_1 \asymp_h S'_2$.

To sum up, we have proved until now $\vdash P = A^\star, S'_1$, $\vdash Q = A^\star, S'_2$, and $S'_1 \asymp_h S'_2$.

We now prove, by induction over the number of summands in S'_1, that for any such summand $\pi.T_1$, there is a summand $\pi.T_2$ in S'_2 s.t. $\vdash \pi.T_1 = \pi.T_2$. Once this will be proved, we shall establish the same way the symmetrical property, which will allow us to deduce $\vdash S'_1 = S'_2$.

Suppose then $\pi.T_1$ is a summand of S'_1.

We reason by case analysis on the shape of π, and suppose $\pi = a(x)$. We know, since $S'_1 \asymp_h S'_2$, that there is a summand $a(x).T_2$ of S'_2 such that $a(x).T_1 \sim a(x).T_2$. We have $\vdash a(x).T_i = a(y).(\nu x)(y/x \mid T_i)$, for $i = 1, 2$, by Law L24, induction hypothesis (on $(\nu x)(y/x \mid T_1) \sim (\nu x)(y/x \mid T_2)$) and congruence using the context $a(y).[]$.

Moreover, since $a(x).T_1 \sim a(x).T_2$, we know, by Fact 1, that $a(y).(\nu x)(y/x \mid T_1) \sim a(y).(\nu x)(y/x \mid T_2)$. This allows us to rely on the induction hypothesis to show $\vdash a(y).(\nu x)(y/x \mid T_1) = a(y).(\nu x)(y/x \mid T_2)$, which gives us, as announced $\vdash \pi.T_1 = \pi.T_2$.

The other cases for the shape of π are treated similarly. □

We now move to the full calculus, by taking into account parallel composition. As is usually the case, this relies on a law for expansion.

The following result states that any process is equivalent to a process that can be written without using parallel composition. Intuitively, such a process can be viewed as a "sum-only process with restriction".

We can remark that expanding processes involves the introduction of restrictions (in the third sum law L7).

Lemma 54. *For any P, there exists a $|$-free process Q s.t. $\vdash P = Q$.*

Proof. First write $\vdash P = A, P_1$ using the monoid laws for parallel composition. Then, by induction on $|P_1|$, we build S such that $\vdash P_1 = S$ and $|P_1| = |S|$ using law L7. □

We can then extend Proposition 53 to the whole $\pi\mathtt{P}$ calculus:

Theorem 55 (Axiomatisation of $\sim$ in $\pi\mathtt{P}$).
For all $\pi\mathtt{P}$ processes P and Q, $P \sim Q$ iff $\vdash P = Q$.

Proof. The theorem follows by Proposition 53 and Lemma 54.

Remark 56 (Discussion about normal forms). The proofs of the results in this section suggest that we can define a strategy to apply the rules of Figure 2, in order to rewrite a $\pi\mathtt{P}$ process P to its *normal form*, $\mathsf{nf}(P)$, so that $P \sim Q$ iff $\mathsf{nf}(P) = \mathsf{nf}(Q)$.

For preorder processes, the saturated form is a normal form for $\sim$: if $A_1 \sim A_2$, then, by Proposition 46, $\vdash A_1^\star = A_2^\star$. By contrast, the proof of Proposition 49 does not compute a canonical form for sum processes. For instance, from the equivalence

$$b/a \mid c/a \mid \overline{a}(x).0 \sim b/a \mid c/a \mid \overline{a}(x).0 + \overline{b}(x).0 \ ,$$

Proposition 49 rewrites these processes into $b/a \mid c/a \mid \overline{a}(x).0 + \overline{b}(x).0$, but not into $b/a \mid c/a \mid \overline{a}(x).0 + \overline{b}(x).0 + \overline{c}(x).0$, which could be seen as a normal form for $\sim$, obtained by saturating the sum. Actually, the normal form could even be

$$b/a \mid c/a \mid \overline{a}(x).0 + \overline{b}(x).0 + \overline{c}(x).0 + \overline{\{a\}}(x).0 + \overline{\{b\}}(x).0 + \overline{\{c\}}(x).0 \ ,$$

by virtue of several applications of (the counterpart for output of) law L16 with π_1 an output prefix and π_2 a protected output.

We leave the rigorous description of this normalisation procedure for future work.

4.6 Adapting our Axiomatisation to Explicit Fusions

We can reuse the ideas presented above to describe an axiomatisation for barbed congruence in Explicit Fusions (EF, [16]). EF feature *fusion processes*, of the form $a{=}b$, which can equate names via $\equiv$: we have $a{=}b \mid P \equiv a{=}b \mid P\{b/a\}$.

We refer to Appendix C for a description of the axiomatisation.

5 Conclusions and Future Work

Working with an preorder on names, using arc processes, has an influence on the behavioural theory of $\pi\mathtt{P}$, notably through the interplay between arcs and restrictions. The preorder relation between names is represented explicitly in $\pi\mathtt{P}$ processes, using arcs. We do not see any natural "implicit version" of $\pi\mathtt{P}$, mimicking the relation between Explicit Fusions and Fusions, whereby the extension of the preorder along a communication would not generate an arc.

The stateful nature of the preorder component of $\pi\mathtt{P}$ processes can be related to *frames* in the applied π-calculus [1] and *assertions* in Psi-calculi [2]. Liu and Lin's proof system for applied π [11] is rather different from our axiomatisation for $\pi\mathtt{P}$, but has in common the "state component" of processes. One could expect that the same would hold for an equational presentation of bisimilarity in Psi-calculi, possibly relying on some hypotheses on the assertions in order for an axiomatisation to be definable.

Among the natural extensions of this work is the study of the weak version of behavioural equivalence. One fundamental use of types is to help reasoning about processes. With the behavioural theory in the untyped setting in place,

we could study *typed behavioural equivalence*, building on the type system of [8]. Works like [14,5] should be relevant in this direction.

The behavioural theory of $\pi\mathtt{P}$ is based on an operational account. An intriguing question is the construction of a denotational model for $\pi\mathtt{P}$, and the comparison with known model for π and Fusions.

Acknowledgements. We thank Davide Sangiorgi and Fu Yuxi for useful discussions about behavioural equivalence in $\pi\mathtt{P}$. This work has been supported by project ANR 12IS02001 PACE and NSF of China (61261130589).

References

1. M. Abadi and C. Fournet. Mobile values, new names, and secure communication. In *Proc. of POPL*, pages 104–115. ACM, 2001.
2. J. Bengtson, M. Johansson, J. Parrow, and B. Victor. Psi-calculi: Mobile processes, nominal data, and logic. In *LICS*, page 39–48. IEEE, 2009.
3. M. Boreale, M. G. Buscemi, and U. Montanari. D-fusion: A distinctive fusion calculus. In *Proc. APLAS*, volume 3302 of *LNCS*, pages 296–310. Springer, 2004.
4. M. Boreale, M. G. Buscemi, and U. Montanari. A general name binding mechanism. In *Proc. TGC*, volume 3705 of *LNCS*, pages 61–74. Springer, 2005.
5. Y. Deng and D. Sangiorgi. Towards an algebraic theory of typed mobile processes. *Theor. Comput. Sci.*, 350(2-3):188–212, 2006.
6. Y. Fu. The χ-calculus. In *APDC*, pages 74–81. IEEE Computer Society, 1997.
7. P. Gardner and L. Wischik. Explicit fusions. In *MFCS*, volume 1893 of *LNCS*, pages 373–382. Springer, 2000.
8. D. Hirschkoff, J.-M. Madiot, and D. Sangiorgi. Name-passing calculi: From fusions to preorders and types. In *LICS*, pages 378–387. IEEE Computer Society, 2013.
9. D. Hirschkoff, J.-M. Madiot, and X. Xu. Long version of this paper. Available from `http://madiot.org`.
10. C. Laneve and B. Victor. Solos in concert. *Mathematical Structures in Computer Science*, 13(5):657–683, 2003.
11. J. Liu and H. Lin. Proof system for applied pi calculus. In *Proc. IFIP TCS*, volume 323 of *IFIP Advances in Inf. and Comm. Technol.*, pages 229–243. Springer, 2010.
12. J. Parrow and D. Sangiorgi. Algebraic theories for name-passing calculi. *Inf. Comput.*, 120(2):174–197, 1995.
13. J. Parrow and B. Victor. The fusion calculus: expressiveness and symmetry in mobile processes. In *LICS*, pages 176 –185. IEEE, 1998.
14. B. C. Pierce and D. Sangiorgi. Typing and subtyping for mobile processes. *Mathematical Structures in Computer Science*, 6(5):409–453, 1996.
15. D. Sangiorgi and D. Walker. *The Pi-Calculus: a theory of mobile processes*. Cambridge University Press, 2001.
16. L. Wischik and P. Gardner. Strong bisimulation for the explicit fusion calculus. In *Proc. of FoSSaCS*, volume 2987 of *LNCS*, pages 484–498, 2004.

Appendix

A A LTS for a Variant of $\pi\mathtt{P}$ with Free Prefixes

We call $\pi\mathtt{P}_{\mathrm{F}}$ the variant of $\pi\mathtt{P}$ defined by the following grammar, that includes primitive free prefixes (the grammar differs from the one of $\pi\mathtt{P}$ only in the definition of prefixes):

$$\varphi ::= a \leqslant b \mid a \curlyvee b \qquad \alpha, \beta ::= a \mid \{a\} \qquad \pi ::= \alpha b \mid \overline{\alpha} b \mid [\varphi]\tau$$

$$P, Q ::= P \mid Q \mid (\nu a)P \mid a\!\!/\!\!b \mid \sum_{i \in I} \pi_i.P_i$$

We present below an LTS similar to the one of the π-calculus as far as restriction is concerned: there are now scope rules. In the following, $\hat{a}$, $\hat{b}$ range over sets of names that can be either a singleton or empty. When this set is $\{a\}$, we write $(\nu\hat{a})$ for (νa) and otherwise for nothing.

Labels: $\mu ::= \overline{\alpha} b \mid \alpha b \mid (\nu b)\overline{\alpha} b \mid (\nu b)\alpha b \mid [\varphi]\tau$.

$$\frac{}{\pi.P \xrightarrow{\pi}_{\mathrm{F}} P} \to_F\text{-PREF} \qquad \frac{P \xrightarrow{[\varphi_2]\tau}_{\mathrm{F}} P' \quad P \rhd \Gamma \quad \Gamma, \varphi_1 \vdash \varphi_2}{P \xrightarrow{[\varphi_1]\tau}_{\mathrm{F}} P'} \to_F\text{-TAU-}\rhd$$

$$\frac{P \xrightarrow{(\nu\hat{c})\alpha c}_{\mathrm{F}} P' \quad P \rhd \alpha \leqslant \beta}{P \xrightarrow{(\nu\hat{c})\beta c}_{\mathrm{F}} P'} \to_F\text{-IN-}\rhd \qquad \frac{P \xrightarrow{(\nu\hat{c})\overline{\alpha} c}_{\mathrm{F}} P' \quad P \rhd \alpha \leqslant \beta}{P \xrightarrow{(\nu\hat{c})\overline{\beta} c}_{\mathrm{F}} P'} \to_F\text{-OUT-}\rhd$$

$$\frac{P \xrightarrow{(\nu\hat{b})\overline{\alpha} b}_{\mathrm{F}} P' \quad Q \xrightarrow{(\nu\hat{d})\gamma d}_{\mathrm{F}} Q' \quad \emptyset = \hat{b} \cap \hat{d} = \hat{b} \cap \mathsf{fn}(Q) = \hat{d} \cap \mathsf{fn}(P)}{P \mid Q \xrightarrow{[\alpha \curlyvee \gamma]\tau}_{\mathrm{F}} (\nu\hat{b})(\nu\hat{d})(b\!\!/\!\!d \mid P' \mid Q')} \to_F\text{-COM-L}$$

$$\frac{P \xrightarrow{\mu}_{\mathrm{F}} P' \qquad a \notin \mathsf{n}(\mu)}{(\nu a)P \xrightarrow{\mu}_{\mathrm{F}} (\nu a)P'} \to_F\text{-RES} \qquad \frac{P \xrightarrow{\mu}_{\mathrm{F}} P' \qquad \mu \in \{\gamma a, \overline{\gamma} a\}}{(\nu a)P \xrightarrow{(\nu a)\mu}_{\mathrm{F}} P'} \to_F\text{-OPEN}$$

$$\frac{P \xrightarrow{\mu}_{\mathrm{F}} P' \qquad \mathsf{bn}(\mu) \cap \mathsf{fn}(Q) = \emptyset}{P \mid Q \xrightarrow{\mu}_{\mathrm{F}} P' \mid Q} \to_F\text{-PAR-L} \qquad \frac{\pi_i.P_i \xrightarrow{\mu}_{\mathrm{F}} P'}{\sum_i \pi_i.P_i \xrightarrow{\mu}_{\mathrm{F}} P'} \to_F\text{-SUM}$$

The LTS above induces the following notion of bisimilarity:

Definition 57 (F-bisimilarity). *A symmetric relation $\mathcal{R}$ is an F-bismimulation iff $P \mathcal{R} Q$ implies*

1. *for all φ, $P \rhd \varphi$ implies $Q \rhd \varphi$;*

2. *if* $P \xrightarrow{(\nu\hat{b})\overline{\alpha}b}_{\mathrm{F}} P'$, *there is* Q', c *and* $\hat{c}$ *s.t.*
$Q \xrightarrow{(\nu\hat{c})\overline{\alpha}c}_{\mathrm{F}} Q'$ *and* $(\nu\hat{b})(P' \mid b/f) \mathrel{\mathcal{R}} (\nu\hat{c})(Q' \mid c/f)$ *with* f *fresh;*
3. *if* $P \xrightarrow{(\nu\hat{b})\alpha b}_{\mathrm{F}} P'$, *there is* Q', c *and* $\hat{c}$ *s.t.*
$Q \xrightarrow{(\nu\hat{c})\alpha c}_{\mathrm{F}} Q'$ *and* $(\nu\hat{b})(P' \mid f/b) \mathrel{\mathcal{R}} (\nu\hat{c})(Q' \mid f/c)$ *with* f *fresh;*
4. *if* $P \xrightarrow{[\varphi]\tau}_{\mathrm{F}} P'$, *there is* Q' *s.t.* $Q \xrightarrow{[\varphi]\tau}_{\mathrm{F}} Q'$ *and* $P' \mid \varphi \mathrel{\mathcal{R}} Q' \mid \varphi$.

F-bisimilarity, written $\sim_{\mathrm{F}}$*, is the greatest F-bisimulation.*

Remark 58. First one could feel that if $P \xrightarrow{\overline{a}b}_{\mathrm{F}} P'$ and $P \sim Q$ and Q answered as $Q \xrightarrow{\overline{a}c}_{\mathrm{F}} Q'$, then we should have required that $Q' \rhd c \leqslant b$. This is a fair requirement, and is already taken care of by the condition $P' \mid b/f \sim Q' \mid c/f$. Indeed, $(P' \mid b/f) \rhd f \leqslant b$ so we must have $(Q' \mid c/f) \rhd f \leqslant b$, and the only way it can entail $f \leqslant b$ is through c (since f is fresh), so we must have $Q' \rhd c \leqslant b$.

Proposition 59. *Define* $[\alpha(x).P] = (\nu x)\alpha x.[P]$ *and* $[\overline{\alpha}(x).P] = (\nu x)\overline{\alpha}x.[P]$. *For any* P, Q, $P \sim Q$ *iff* $[P] \sim_{\mathrm{F}} [Q]$.

Proof. In all the following, we assume x is fresh. We also use P, Q, R instead of the same letter for the different clauses to ease reference to them. We first prove a tight operational correspondence from $\pi\mathtt{P}$ to $\pi\mathtt{P}_{\mathrm{F}}$:

- if $P \rhd \varphi$ then $[P] \rhd \varphi$,
- if $P \xrightarrow{\overline{\alpha}(x)} P'$ then for some y, P_2: $P' \equiv (\nu y)(y/x \mid P_2)$ and $[P] \xrightarrow{(\nu y)\overline{\alpha}y}_{\mathrm{F}} [P_2]$,
- if $Q \xrightarrow{\alpha(x)} Q'$ then for some z, Q_2: $Q' \equiv (\nu z)(x/z \mid Q_2)$ and $[Q] \xrightarrow{(\nu z)\alpha z}_{\mathrm{F}} [Q_2]$,
- if $R \xrightarrow{[\varphi]\tau} R'$ then $[R] \xrightarrow{[\varphi]\tau}_{\mathrm{F}} [R_2]$ for some R_2 such that $R_2 \sim R'$.

and another from $\pi\mathtt{P}_{\mathrm{F}}$ back to $\pi\mathtt{P}$ (note that the object y is always bound, and that x can be universally quantified, especially it can be f):

- if $[P] \rhd \varphi$ then $P \rhd \varphi$,
- if $[P] \xrightarrow{(\nu y)\overline{\alpha}y}_{\mathrm{F}} P_1$ then for some P_2, $P \xrightarrow{\overline{\alpha}(x)} \equiv (\nu y)(y/x \mid P_2)$ and $P_1 = [P_2]$,
- if $[Q] \xrightarrow{(\nu z)\alpha z}_{\mathrm{F}} Q_1$ then for some Q_2, $Q \xrightarrow{\alpha(x)} \equiv (\nu z)(x/z \mid Q_2)$ and $Q_1 = [Q_2]$,
- if $[R] \xrightarrow{[\varphi]\tau}_{\mathrm{F}} R_1$ then $R_1 = [R_2]$ for a R' such that $R_2 \sim R'$ and $R \xrightarrow{[\varphi]\tau} R'$.

These two correspondences describes the basic steps necessary to prove that $\{([P], [Q]) \mid P \sim Q\}$ is a F-bisimulation and that $\{(P, Q) \mid [P] \sim_F [Q]\}$ is a bisimulation up to bisimilarity, to finally prove $P \sim Q$ iff $[P] \sim_{\mathrm{F}} [Q]$.

The correspondence from $\pi\mathtt{P}$ to $\pi\mathtt{P}_{\mathrm{F}}$ goes without problem. The $R_2 \sim R'$ condition comes from the fact $(\nu x)(y/x \mid x/z)$ must be transformed into the bisimilar process y/z when composing P' and Q' into $(\nu x)(P' \mid Q')$:

$$\begin{aligned} R' = (\nu x)(P' \mid Q') &\equiv (\nu x)((\nu z)(x/z \mid Q_2) \mid (\nu y)(y/x \mid P_2)) \\ &\equiv (\nu yz)((\nu x)(y/x \mid x/z) \mid Q_2 \mid P_2) \\ &\sim (\nu yz)(y/z \mid Q_2 \mid P_2) = R_2 \end{aligned}$$

In fact, the same reasoning (and the same equation) apply when proving the operational correspondence from $\pi\mathtt{P}_{\mathrm{F}}$ back to $\pi\mathtt{P}$. □

Types

$$T ::= 1 \mid iT \mid oT \mid \sharp T \mid \overline{\mathtt{o}}T \mid \overline{\mathtt{i}}T$$

Subtyping

$$\frac{}{\sharp T \leq iT} \qquad \frac{}{\sharp T \leq oT} \qquad \frac{}{oT \leq \overline{\mathtt{i}}T} \qquad \frac{}{iT \leq \overline{\mathtt{o}}T}$$

$$\frac{T \leq U}{iT \leq iU} \qquad \frac{T \leq U}{oU \leq oT} \qquad \frac{T \leq U}{\overline{\mathtt{o}}T \leq \overline{\mathtt{o}}U} \qquad \frac{T \leq U}{\overline{\mathtt{i}}U \leq \overline{\mathtt{i}}T}$$

Typing rules

$$\frac{\Gamma \vdash a : oT \qquad \Gamma, x : T \vdash P}{\Gamma \vdash \overline{a}(x).P} \qquad \frac{\Gamma \vdash a : iT \qquad \Gamma, x : T \vdash P}{\Gamma \vdash a(x).P}$$

$$\frac{\Gamma \vdash a : \overline{\mathtt{o}}T \qquad \Gamma, x : T \vdash P}{\Gamma \vdash \overline{\{a\}}(x).P} \qquad \frac{\Gamma \vdash a : \overline{\mathtt{i}}T \qquad \Gamma, x : T \vdash P}{\Gamma \vdash \{a\}(x).P} \qquad \frac{\Gamma(a) \leq \Gamma(b)}{\Gamma \vdash a\!/\!b}$$

$$\frac{\Gamma \vdash P}{\Gamma \vdash [\varphi]\tau.P} \qquad \frac{\Gamma \vdash P \qquad \Gamma \vdash Q}{\Gamma \vdash P \mid Q} \qquad \frac{\Gamma \vdash P \qquad \Gamma \vdash Q}{\Gamma \vdash P + Q}$$

Fig. 3. A type system for capabilities and cocapabilities

B A Type System with Cocapabilities

Figure 3 presents a type for $\pi\mathtt{P}$ with *cocapabilities*. Cocapability types are of the form $\overline{\mathtt{o}}T$ and $\overline{\mathtt{i}}T$, and are used to type protected prefixes.

We have the expected property of type preservation:

Proposition 60 (Subject reduction). *If $\Gamma \vdash P$ and $P \mapsto P'$, then $\Gamma \vdash P'$.*

As with capabilities, a typing hypothesis of the form $a : \overline{\mathtt{i}}T$ provides only the right to perform a protected input at a. It is possible, for instance, to derive

$$a : \overline{\mathtt{i}}T, b : oT \vdash \{a\}(x).\overline{b}x.0 \ .$$

Note that it is *not* possible to derive a similar judgement using the encoding of protected prefixes:

$$a : \overline{\mathtt{i}}T, b : oT \not\vdash (\nu u)(u\!/\!a \mid u(x).\overline{b}x.0) \ .$$

We leave for future work the investigation of a typed behavioural equivalence based on this type system (intuitively, types can help reasoning about the behaviour processes by constraining the observation capabilities of contexts).

Another direction worth exploring is a finer study of the type system in a version of $\pi\mathtt{P}$ with free prefixes, along the lines of [8]. In particular, we expect that usages of protected names correspond to *negative polarities* (while all names in subject positions are positive in [8]).

C An Axiomatisation for Explicit Fusions

We show how our ideas can be adapted to the calculus of Explicit Fusions [7]. Accordingly, we adopt a presentation of the calculus that follows the lines of $\pi\mathtt{P}$ as we have introduced it.

The grammar of prefixes, conditions and processes is as follows:

$$\varphi ::= a{=}b \qquad \pi ::= \overline{a}(x) \mid a(x) \mid [a{=}b]\tau \qquad P ::= P|Q \mid \nu a P \mid \Sigma_i \pi_i.P_i \mid a{=}b$$

As in $\pi\mathtt{P}$, we adopt primitive bound prefixes. Free prefixes can be encoded: $[ab.P] = (\nu u)a(u).(u{=}b \mid P)$. Note in passing that fusions can be represented in $\pi\mathtt{P}$, encoding $a{=}b$ with $a{/}b \mid b{/}a$.

The following definition is adapted from the *efficient bisimulation* of [16]. The LTS is defined according to the approach in Figure 1, except in their LTS $P \xrightarrow{\tau} P'$ does not necessarily imply that $P \xrightarrow{[a=b]\tau} P'$ for every a and b. So we have to change the third clause of the definition of bisimulation to take this into account. The way we handle objects does not matter, as the resulting bisimilarity is the same.

Definition 61 ($\sim_{\mathrm{EF}}$). *An* efficient bisimulation *is a symmetric relation $\mathcal{R}$ such that if $P\mathcal{R}Q$ then:*

1. *$P \rhd a{=}b$ iff $Q \rhd a{=}b$;*
2. *$P \xrightarrow{\mu} P'$ implies $Q \xrightarrow{\mu} Q'$ for some Q' s.t. $P'\mathcal{R}Q'$, for $\mu \neq [\varphi]\tau$;*
3. *$P \xrightarrow{[a=b]\tau} P'$ implies $a{=}b \mid Q \xrightarrow{\tau} Q'$ and $a{=}b \mid P'\mathcal{R}Q'$.*

We write $\sim_{\mathrm{EF}}$ for the largest efficient bisimulation.

We write $\vdash_{\mathrm{EF}} P = Q$ if the equality can be derived using equational reasoning from the laws of Figure 4.

The axiomatisation is considerably simpler than in $\pi\mathtt{P}$. The stateful component of processes encodes an equivalence relation on names. This makes it possible to reason locally, whereby a form of global reasoning is necessary in the laws of Figure 2 for $\pi\mathtt{P}$.

The fact that fusions are symmetric makes protected names unnecessary and simplifies greatly the handling of restriction. Indeed, laws L31-L35 can be dealt with using the simpler law $(\nu a)\sum_i \pi_i.P_i = \sum_{i|a\notin \mathtt{n}(\pi_i)} \pi_i.(\nu a)P_i$.

Lemma 62. *For any P, a, b, we can derive $\vdash_{\mathrm{EF}} P \mid a{=}b = P\{a{/}b\} \mid a{=}b$.*

Proposition 63. *For any P, Q, we have $P \sim_{\mathrm{EF}} Q$ iff $\vdash_{\mathrm{EF}} P = Q$.*

Standard structural laws.

$$P \mid (Q \mid R) = (P \mid Q) \mid R \qquad P \mid Q = Q \mid P \qquad P \mid 0 = P$$

$$(\nu a)(P \mid Q) = P \mid (\nu a)Q \;\text{ and }\; (\nu b)P = (\nu a)P\{a/b\} \text{ when } a \notin \mathsf{fn}(P)$$

$$(\nu a)\sum_i \pi_i.P_i = \sum_{i \mid a \notin \mathsf{n}(\pi_i)} \pi_i.(\nu a)P_i$$

Laws for fusions.

$$a{=}a = 0 \qquad a{=}b \mid a{=}c = a{=}b \mid a{=}c \mid b{=}c \qquad a{=}b = b{=}a \qquad (\nu a)a{=}b = 0$$

Laws for prefixes.

$$a{=}b \mid S + \pi.P = a{=}b \mid S + \pi.(a{=}b \mid P) \qquad [a{=}b]\tau.P = [a{=}b]\tau.(a{=}b \mid P)$$

$$\pi.P + \pi.P = \pi.P \qquad [a{=}b]\tau.P = [b{=}a]\tau.P \qquad a{=}b \mid [a{=}c]\tau.P = a{=}b \mid [b{=}c]\tau.P$$

$$a{=}b \mid S + a(x).P = a{=}b \mid S + b(x).P \qquad a{=}b \mid S + \overline{a}(x).P = a{=}b \mid S + \overline{b}(x).P$$

Expansion law.

$$\begin{aligned}\textstyle\sum_i \pi_i.P_i \;\mid\; \sum_j \rho_j.R_j = {} & \textstyle\sum_i \pi_i.(P_i \mid T) + \sum_j \rho_j.(S \mid R_j) \\ & + \textstyle\sum_{i,j} [a{=}b]\tau.(\nu x)(P_i \mid R_j) \\ & \text{where } \{\pi_i, \rho_j\} = \{\overline{a}(x), b(x)\}\end{aligned}$$

Fig. 4. Axiomatisation for the Explicit Fusions calculus