Under consideration for publication in Math. Struct. in Comp. Science

Solos in Concert

Cosimo Laneve[1]

Björn Victor[2]

[1] *Dept. of Computer Science, University of Bologna, Italy.* `laneve@CS.UniBO.IT`
[2] *Dept. of Information Technology, Uppsala University, Sweden.* `Bjorn.Victor@it.uu.se`

Received 7 May 2002

We present a calculus of mobile processes without prefix or summation, called the *solos calculus*. Using two different encodings, we show that the solos calculus can express both action prefix and guarded summation. One encoding gives a strong correspondence but uses a match operator; the other yields a slightly weaker correspondence but uses no additional operators. We also show that the expressive power of the solos calculus is still retained by the sub-calculus where actions carry at most two names. On the contrary, expressiveness is lost in the solos calculus without match and with actions carrying at most one name.

1. Introduction

The fusion calculus was introduced by Parrow and Victor (Parrow and Victor, 1998a; Victor and Parrow, 1998; Parrow and Victor, 1998b) as a simplification and extension of the π-calculus (Milner et al., 1992). The simplification is easy to see: the fusion calculus has only one binding operator where the π-calculus has two; in the fusion calculus, input and output are completely symmetric unlike in the π-calculus; and the fusion calculus has only one sensible bisimulation congruence where the π-calculus has three (Parrow and Victor, 1998a). The extension is that the effects of communication need not be local to the recipient process. Furthermore the fusion calculus contains the π-calculus as a proper sub-calculus and thus inherits all its expressive power.

In recent years the *asynchronous* π-calculus (Honda and Tokoro, 1991; Boudol, 1992) has gained interest; here the asymmetry between input and output is further increased by dropping continuations from output prefixes. The resulting calculus has significant expressive power, and is also motivated by practical implementation in distributed systems.

In the fusion calculus it would be unfortunate to break the symmetry between input and output in order to develop an asynchronous subcalculus. Indeed, in this paper we show that continuations may be removed both from inputs and outputs – hereafter called *solos* – without loss of expressive power. More precisely, the fusion calculus of solos, or *solos calculus* for short, where prefixes are replaced by solos and summation is removed, is expressive enough to encode prefixes and guarded summation.

We give two different encodings of the fusion calculus in the solos calculus. One preserves strong barbed bisimulation but uses a match operator; the other yields weak barbed bisimulation but uses no additional operators. Both preserve divergence. A more precise account of our work follows.

In the fusion calculus, as well as in other mobile calculi, the purpose of a prefixing $\alpha \,.\, P$ is to freeze the continuation agent P until the action α has been consumed in a reduction. In other words, every reduction involving P *causally depends* on the reduction involving the prefix α. A second form of causal dependency is that inherent in the match operator: $[x = y]P$ freezes the agent P until x and y are the same. In the fusion calculus, this can be caused by a fusion effect produced in another, parallel, agent. This construction is used in our first encoding.

Apart from these *explicit* causal dependencies, mobile calculi possess another, *implicit* form of causal dependency, which relies on the scope (or restriction) operator (causal dependencies in π-calculus have been studied in several works, such as (Boreale and Sangiorgi, 1998; Degano and Priami, 1995)). Consider for instance the following agent:

$$(v)(\overline{u}\, v \mid v\, y) \;.$$

In this agent, the solo $v\,y$ by no means may react before the solo $\overline{u}\,v$ because the subject name v is bound. When $\overline{u}\,v$ reacts, the scope of v is extended, possibly enabling a reaction with $v\,y$. In our second encoding, we employ this type of causal dependency.

A similar mechanism is used in the encoding of π-calculus into asynchronous π-calculus, where continuations of output prefixes are dropped (Honda and Tokoro, 1991; Boudol, 1992). In particular, π-calculus outputs are translated into outputs carrying local names, in parallel with the continuation prefixed by an input on the local name. However, the asynchronous π-calculus cannot be further simplified by also dropping continuations of input prefixes. The reason is that in π-calculus the communication mechanism is asymmetric, because names carried by the output *replace* the names carried by the input. Thus the information flow is unidirectional and only affects the continuation of the inputs. Therefore the communication mechanism becomes meaningless once input continuations are also removed.

A key of our encodings is the use of *catalyst agents* of the type $(z)u\,zz$, which inputs the same name twice. If some agent in parallel composition with $(z)u\,zz$ sends any two names on u, they will be fused and made indistinguishable everywhere. In one of our encodings such a fusion effect enables an agent guarded by a match construction testing two names for equality; in the other it removes the restriction of a private communication channel, thereby enabling the channel.

An important factor for writing catalyst agents is that the calculus admits solos which carry two names – the *dyadic solos calculus*. We demonstrate that the polyadic solos calculus, where solos can carry arbitrarily many objects, may be encoded into the dyadic one. We also show that the expressiveness of the polyadic solos calculus without match is strictly greater than that of monadic solos.

Related work. Parrow shows in (Parrow, 2000) that in the π-calculus without match, any agent can be encoded as a *concert of trios*, i.e., a parallel composition of possibly replicated prefixes $\alpha_1 . \alpha_2 . \alpha_3$ up to weak open equivalence (Sangiorgi, 1996). He also shows that *duos*, i.e., prefixes nested to depth 2, are not sufficient. In this paper we show that for the fusion calculus, *solos* suffice, i.e., we do not need prefixes at all.

Nestmann and Pierce show in (Nestmann and Pierce, 1996) that input-guarded choice $\sum_i u_i(\widetilde{x_i}) . P_i$ can be encoded in the asynchronous π-calculus without choice, up to coupled bisimulation (Parrow and Sjödin, 1992), and in (Nestmann, 1997) Nestmann gives encodings of separate and mixed guarded choice. While these encodings involve increasingly complex communication protocols, our encodings of separate choice are simpler and work up to stronger equivalences.

In (Palamidessi, 1997) Palamidessi shows that there cannot exist an encoding of mixed choice into the asynchronous π-calculus which is *uniform* and preserves a *reasonable* semantics. A *uniform* encoding is compositional, meaning that $[\![P \mid Q]\!] = [\![P]\!] \mid [\![Q]\!]$, and respects injective renaming of free names. A *reasonable* semantics distinguishes two processes P and Q if the actions in some computation of P are different from those in any computation of Q; in particular it is sensitive to divergence. Nestmann (Nestmann, 1997) argues that these criteria are too strong for practical purposes, and that by allowing a top-level context (but keeping the inner encoding compositional), or relaxing reasonableness, many practically motivated encodings turn out to be "good". Indeed even more theoretically motivated encodings often use top-level contexts, including our second encoding in this paper – it does, however, respect injective renamings. Our first encoding does not need top-level encoding, but is in fact uniform and reasonable in Palamidessi's sense.

Yoshida in (Yoshida, 1998) presents separation results between subcalculi of the asynchronous π-calculus (without match and choice) by means of concurrent combinators, and shows the non-existence of encodings between such subcalculi. Here a concept of *standard encoding* is used, which means that the encoding is homomorphic, respects injective substitutions, and preserves weak observations and reductions. In addition to this the encodings are required to be *message-preserving*, i.e., $[\![\overline{u}\,x]\!] \approx \overline{u}\,x$. While our first encoding is standard by this definition, neither one is message-preserving. Yoshida works with monadic calculi, where the requirement may be quite sensible, but in a polyadic setting this requirement seems very strong, especially when only considering encoded contexts. Yoshida also proves that the *reflexive π-calculus*, a variant of the π-calculus similar to the monadic solos calculus, does not contain a synchronizer agent such as $a(x) . b(y) . \overline{c}\,y$, and is therefore less expressive than the monadic asynchronous π-calculus. In this paper we show that in the polyadic solos calculus such an agent can indeed be encoded, although our encodings are not message-preserving, but rather adds a "protocol header" to the data being sent and received.

Recently, Gardner and Wischik have introduced a *calculus with explicit fusions* of names (Gardner and Wischik, 2000). This calculus is similar to the fusion calculus, but with a local reduction rule. The similarity is strengthened by a fully-abstract translation relating the fusion calculus and the one with explicit fusions. In view of this translation, our expressiveness results, in particular the encodings without the match operator and the result about the monadic sub-calculus, should be easily adapted to the calculus with explicit fusions. In the χ-calculus, a monadic variant of the fusion calculus independently developed in (Fu, 1997), our results cannot be applied, since the χ-calculus is monadic and thus (as we show in Section 5) not expressive enough.

In the action calculus framework, Honda in (Honda, 2000) introduced a monadic *essential π-calculus*, similar to our solos calculus, but does not study the relation to the calculus with prefix operator. In his calculus the interaction $ab \mid \overline{a}c$ results in a *wire* or *open substitution* $[b = c]$, with similarities to the explicit fusions of Gardner and Wischik.

Structure of the paper. The following section introduces our language, its model and the equivalences we use in the rest of the paper. In Section 3, we detail the encodings of prefixes and the proofs of the main results. In Section 4, we extend our results to replication. In Section 5, we study the dyadic solos calculus and the encoding of polyadic solos; we also show that the expressiveness of dyadic solos is strictly greater than the monadic ones. In Section 6, we present encodings of the choice operator. We conclude in Section 7. Additional proofs and examples appear in the appendices.

Acknowledgement. We thank Luca Aceto, Uwe Nestmann, Joachim Parrow, GianLuigi Zavattaro and the anonymous referees for valuable comments and remarks.

2. The Solos Calculus

In this section we first present a subcalculus of the fusion calculus where we use *solos* of the form $u\,\widetilde{x}$ in place of general prefixes of the form $u\,\widetilde{x}\,.\,P$, and leave out summation. We present the syntax and semantics, review the barbed equivalences and congruences, and introduce an auxiliary barbed expansion preorder.

2.1. *Syntax*

We assume an infinite set $\mathcal{N}$ of *names* ranged over by $u, v, \ldots, z$. Names represent communication channels, which are also the values being transmitted in communications. We write $\widetilde{x}$ for a (possibly empty) finite sequence $x_1 \cdots x_n$ of names, and we write $|\widetilde{x}|$ for its length. *Name substitutions*, noted $\{\widetilde{y}/\widetilde{x}\}$ and ranged over by σ, are total functions on $\mathcal{N}$ such that $u \neq \sigma(u)$ for finitely many names u. We use $\mathrm{dom}(\sigma) = \{u : \sigma(u) \neq u\}$ and $\mathrm{ran}(\sigma) = \{\sigma(u) : \sigma(u) \neq u\}$. As usual, $P\{\widetilde{y}/\widetilde{x}\}$ means the simultaneous substitution of names $\widetilde{x}$ in P with names $\widetilde{y}$.

We write $\{\widetilde{x} = \widetilde{y}\}$ for the smallest equivalence relation on $\mathcal{N}$ relating each x_i with y_i, and say that a substitution σ *agrees with* the equivalence φ if for every x, y, $x\,\varphi\,y$ if and only if $\sigma(x) = \sigma(y)$.

Definition 1. The *solos*, ranged over by α, and the *agents*, ranged over by $P, Q, \ldots$, are defined by

$\alpha ::=$	$u\,\widetilde{x}$	(input)	$P ::=$	$\mathbf{0}$	(inaction)
	$\overline{u}\,\widetilde{x}$	(output)		α	(solo)
				$Q \mid R$	(composition)
				$(x)Q$	(scope)
				$[x=y]Q$	(match)

In solos, the name u is the *subject* of the solo, and the names $\widetilde{x}$ are the *objects*. We write a to stand for either u or $\overline{u}$, thus $a\widetilde{x}$ is the general form of a solo. Note that in contrast with input prefixes in the π-calculus, the input solo here does not bind its objects.

A composition allows two solos to interact. We often abbreviate the composition of P_i for $i \in I$, where I is a finite set, with $\prod_{i\in I} P_i$.

The scope $(x)Q$ limits the scope of x to Q; the name x is said to be *bound* in $(x)Q$. This is the only binding operator in solos calculus. We write $(\widetilde{x})P$ for $(x_1)\cdots(x_n)P$, $n \geq 0$. The *free names* in P, denoted $\mathrm{fn}(P)$, are the names in P with a non-bound occurrence.

A match $[x=y]Q$ acts like Q if x and y are the same name. We use M, N to stand for a match operator, and write *match sequence* for a sequence of match operators, ranged over by $\widetilde{M}$, $\widetilde{N}$. We also write $\widetilde{M} \Leftrightarrow \widetilde{N}$ if the conjunction of all matches in $\widetilde{M}$ logically implies all elements in $\widetilde{N}$ and vice versa. We write *labelled nodes* M for the names occurring in M.

2.2. *Reduction semantics*

Following Milner's presentation of π-calculus semantics (Milner, 1993), we first define a structural congruence which equates all agents we will never want to distinguish for any semantic reason, and then use this when giving the operational semantics.

Definition 2. The *structural congruence*, $\equiv$, between agents is the least congruence containing alpha equivalence and satisfying the abelian monoid laws for composition (associativity, commutativity and $\mathbf{0}$ as identity), and the scope laws

$$\begin{gathered}(x)\mathbf{0} \equiv \mathbf{0}, \quad (x)(y)P \equiv (y)(x)P, \quad [x=x]P \equiv P \\ (x)MP \equiv M(x)P, \quad \textit{if } x \notin \textit{labelled nodes } M \\ P \mid (z)Q \equiv (z)(P \mid Q), \quad \textit{if } z \notin \mathrm{fn}(P)\end{gathered}$$

The reduction relation of the solos calculus is the least relation satisfying the rules in Table 1, where structurally equivalent agents are considered the same.

The side-condition of the main reduction rule states that $\widetilde{z}$ contains all but one representative of each equivalence class of $\{\widetilde{x} = \widetilde{y}\}$, and that the substitution maps each equivalence class on that representative. For instance, the reduction

$$(x)(\overline{u}\,x \mid u\,y \mid R) \longrightarrow R\{y/x\}$$

makes the equivalence class $\{x = y\}$, and the substitution $\{y/x\}$ maps x to y. We notice that the above rule may be applied provided the scope of substituted names is *localized* to the term to be reduced. For this reason, the rule also garbage-collects substituted names, since they do not occur anymore in their scope. As a particular case,

$$(\tilde{z})\big(\widetilde{M} u\,\tilde{x} \mid \widetilde{N}\overline{u}\,\tilde{y} \mid R\big) \longrightarrow R\sigma$$

$$\frac{P \longrightarrow P'}{P \mid Q \longrightarrow P' \mid Q} \qquad \frac{P \longrightarrow P'}{(x)P \longrightarrow (x)P'} \qquad \frac{P \equiv Q \quad Q \longrightarrow Q' \quad Q' \equiv P'}{P \longrightarrow P'}$$

Side conditions in the first rule:
$|\tilde{x}| = |\tilde{y}|$, $\widetilde{M} \Leftrightarrow \widetilde{N} \Leftrightarrow \mathsf{true}$,
σ agrees with $\{\tilde{x} = \tilde{y}\}$, $\mathrm{ran}(\sigma) \cap \tilde{z} = \emptyset$, and $\mathrm{dom}(\sigma) = \tilde{z}$.

Table 1. *Reduction rules for the solos calculus.*

$u\,x \mid \overline{u}\,x \mid R \longrightarrow R$ without a scope operator, since the equivalence relation $\{x = x\}$ has only singular equivalence classes, so the resulting substitution is the identity relation.

The separation of binding from input makes it possible to write agents such as the catalysts mentioned in the introduction. When $(x)u\,xx$ is put into a context where an output $\overline{u}\,yz$ occurs with its objects sufficiently bound (i.e., either y or z is bound), they are fused together:

$$(x)u\,xx \mid (y)(\overline{u}\,yz \mid R) \equiv (xy)(u\,xx \mid \overline{u}\,yz \mid R) \longrightarrow R\{z/x, z/y\} = R\{z/y\}$$

assuming $x \notin \mathrm{fn}(R)$.

The effects of a reduction may spread all over the term, according to the placement of bindings with respect to the interacting solos. In particular, effects may be bi-directional, like in

$$(x)(a\,xy \mid P) \mid (z)(\overline{a}\,wz \mid Q) \longrightarrow P\{w/x\} \mid Q\{y/z\}\ .$$

We conclude with a remark about the reduction semantics.

Remark 3. The behaviour of an agent is invariant to the replacement of all solos $u\,\tilde{x}$ with $\overline{u}\,\tilde{x}$, and $\overline{u}\,\tilde{x}$ with $u\,\tilde{x}$ (changing the polarity of solos) because the effects of reductions are fusions of names, rather than substitutions.

2.3. *Behavioural Equivalence*

We will use the standard idea of *barbed bisimulation* developed by Milner and Sangiorgi (Milner and Sangiorgi, 1992a) in the setting of CCS, further investigated in a π-calculus setting (Sangiorgi, 1993), and later used in many other calculi as an intuitive observational equivalence. The idea is that two agents are considered equivalent if their reductions match and they are indistinguishable under global observations.

Definition 4. The *observation relation* is the least relation satisfying the rules below.

$$\begin{array}{ll}
x\widetilde{y} \downarrow x & \\
\overline{x}\widetilde{y} \downarrow x & \\
[x = x]P \downarrow y & \text{if } P \downarrow y \\
(P \mid Q) \downarrow x & \text{if } P \downarrow x \text{ or } Q \downarrow x \\
(x)P \downarrow y & \text{if } P \downarrow y \text{ and } x \neq y
\end{array}$$

We say that P has a *barb* on x if $P \downarrow x$.

Definition 5. A *strong barbed bisimulation* is a symmetric binary relation $\mathcal{S}$ between agents such that $P\ \mathcal{S}\ Q$ implies:

1 If $P \longrightarrow P'$ then $Q \longrightarrow Q'$ and $P'\ \mathcal{S}\ Q'$.
2 If $P \downarrow x$ for some x, then $Q \downarrow x$.

P is strong barbed bisimilar to Q, written $P \stackrel{\cdot}{\sim} Q$, if $P\ \mathcal{S}\ Q$ for some strong barbed bisimulation $\mathcal{S}$. *Strong barbed congruence*, written $\sim$, is the largest barbed bisimulation which is also a congruence.

To define the weak barbed bisimulation and congruence, we change $Q \longrightarrow Q'$ to $Q \longrightarrow^* Q'$ and $Q \downarrow x$ to $Q \longrightarrow^* \downarrow x$ (written $Q \Downarrow x$) in Definition 5.

Definition 6. A *weak barbed bisimulation* is a symmetric binary relation $\mathcal{S}$ between agents such that $P\ \mathcal{S}\ Q$ implies:

1 If $P \longrightarrow P'$ then $Q \longrightarrow^* Q'$ and $P'\ \mathcal{S}\ Q'$.
2 If $P \downarrow x$ for some x, then $Q \Downarrow x$.

P is weak barbed bisimilar to Q, written $P \stackrel{\cdot}{\approx} Q$, if $P\ \mathcal{S}\ Q$ for some weak barbed bisimulation $\mathcal{S}$. *Weak barbed congruence*, written $\approx$, is the largest weak barbed bisimulation which is also a congruence.

Our definitions of strong and weak barbed congruence differ from (Victor and Parrow, 1998). There, $\sim$ and $\approx$ are defined as the largest congruences which are contained in $\stackrel{\cdot}{\sim}$ and $\stackrel{\cdot}{\approx}$, respectively. These definitions do not yield equivalences which are bisimulations. Therefore co-inductive reasoning cannot be used and proofs are usually more difficult. Indeed, in π-calculus and join calculus the two relations coincide, but their equivalence has still not been proved for the fusion calculus. We refer to (Fournet and Gonthier, 1998) and the references therein for a detailed analysis of this issue.

The following proposition collects a couple of immediate consequences of definitions.

Proposition 7.

1 $P \equiv Q$ implies $P \sim Q$.
2 $P \sim Q$ implies $P \approx Q$.
3 (up-to structural congruence) Let $\mathcal{S}$ be a relation that satisfies all the bisimulation clauses of Definition 5 (respectively, Definition 6), after replacing the requirement "$P'\ \mathcal{S}\ Q'$" with "$P' \equiv \mathcal{S} \equiv Q'$". Then $\mathcal{S} \subseteq \sim$ (respectively, $\mathcal{S} \subseteq \approx$).

Proposition 8. Weak barbed congruence is substitution closed, i.e., if $P \approx Q$ then $P\sigma \approx Q\sigma$ for any substitution σ.

Proof. We first prove the proposition for substitutions σ such that $\mathrm{dom}(\sigma) \cap \mathrm{ran}(\sigma) = \emptyset$. To this aim, we observe that the relation

$$\{(C[P\{\widetilde{y}/\widetilde{x}\}], C[(\widetilde{x})(P \mid (z)(\overline{z}\,\widetilde{y} \mid z\,\widetilde{x}))]) \mid \text{for every } C[\,], P, \widetilde{x}, \widetilde{y}, \widetilde{x} \cap \widetilde{y} = \emptyset, z \notin \widetilde{x}, \widetilde{y}\}$$

is a weak barbed congruence up-to structural congruence. Therefore, from $P \approx Q$ we derive $(\widetilde{x})(P \mid (z)(\overline{z}\widetilde{y} \mid z\,\widetilde{x})) \approx (\widetilde{x})(Q \mid (z)(\overline{z}\widetilde{y} \mid z\,\widetilde{x}))$, and by transitivity $P\{\widetilde{y}/\widetilde{x}\} \approx Q\{\widetilde{y}/\widetilde{x}\}$.

Next, we observe that any substitution σ may be decomposed into $\rho \circ \gamma$, such that both $\mathrm{dom}(\rho) \cap \mathrm{ran}(\rho) = \emptyset$ and $\mathrm{dom}(\gamma) \cap \mathrm{ran}(\gamma) = \emptyset$. In particular, ρ maps $\mathrm{dom}(\sigma)$ into names which do not occur into $\mathrm{dom}(\sigma) \cup \mathrm{ran}(\sigma)$ and γ is injective and maps $\mathrm{ran}(\rho)$ into $\mathrm{ran}(\sigma)$. Therefore we may conclude $P\sigma \approx Q\sigma$ by $(P\rho)\gamma \approx (Q\rho)\gamma$. □

We will also make use of an expansion relation (Milner and Sangiorgi, 1992b), an asymmetric form of weak barbed bisimulation where $P \lesssim Q$ means that $P \overset{\cdot}{\approx} Q$ in a way such that P does no more reductions than Q. In the following we write $P \mathrel{\hat{\longrightarrow}} P'$ if $P \longrightarrow P'$ or $P \equiv P'$.

Definition 9. A *weak barbed expansion* is a binary relation $\mathcal{S}$ between agents such that $P \;\mathcal{S}\; Q$ implies:

1 If $P \longrightarrow P'$ then $Q \longrightarrow^{+} Q'$ and $P' \;\mathcal{S}\; Q'$.
2 If $Q \longrightarrow Q'$ then $P \mathrel{\hat{\longrightarrow}} P'$ and $P' \;\mathcal{S}\; Q'$.
3 If $P \downarrow x$ for some x then $Q \Downarrow x$.
4 If $Q \downarrow x$ for some x then $P \downarrow x$.

Q expands P, written $P \lesssim Q$, if $P \;\mathcal{S}\; Q$ for some weak barbed expansion $\mathcal{S}$. We also write $Q \gtrsim P$ instead of $P \lesssim Q$.

3. Encodings of Prefixes

In this section and the following we display the expressiveness of the solos calculus by means of encodings. We first encode the general prefix operator of the fusion calculus using solos. Two such encodings are presented: one using match operators, resulting in a strong operational correspondence with the encoded terms; and one using only solos, scope and parallel composition, yielding a weaker correspondence.

We now add the prefix operator to the calculus by allowing processes of the form $\alpha\,.\,P$. We add two observation rules:

$$x\,\widetilde{y}\,.\,P \downarrow x \qquad\qquad \overline{x}\,\widetilde{y}\,.\,P \downarrow x$$

and the following reduction rule:

$$(\widetilde{z})\big(\widetilde{M}u\,\widetilde{x}\,.\,P \mid \widetilde{N}\overline{u}\,\widetilde{y}\,.\,Q \mid R\big) \longrightarrow (P \mid Q \mid R)\sigma$$

if $|\widetilde{x}| = |\widetilde{y}|$, $\widetilde{M} \Leftrightarrow \widetilde{N} \Leftrightarrow \mathsf{true}$, σ agrees with $\{\widetilde{x} = \widetilde{y}\}$, $\mathrm{ran}(\sigma) \cap \widetilde{z} = \emptyset$ and $\mathrm{dom}(\sigma) = \widetilde{z}$. We call the resulting calculus the *fusion calculus (with prefix)*, f_{pre}. In this calculus we regard a solo $a\widetilde{x}$ as shorthand for the prefix $a\widetilde{x}\,.\,\mathbf{0}$.

$$\begin{aligned}
[\![u\,\widetilde{x}\,.\,P]\!] &\overset{def}{=} (zw)(u\,\widetilde{x}zww \mid [z=w][\![P]\!]) \\
[\![\overline{u}\,\widetilde{x}\,.\,P]\!] &\overset{def}{=} (zw)(\overline{u}\,\widetilde{x}wwz \mid [z=w][\![P]\!]) \\
[\![[x=y]P]\!] &\overset{def}{=} [x=y][\![P]\!] \\
[\![P \mid Q]\!] &\overset{def}{=} [\![P]\!] \mid [\![Q]\!] \\
[\![(x)P]\!] &\overset{def}{=} (x)[\![P]\!]
\end{aligned}$$

Table 2. *Encoding of prefixes using match. z and w are fresh.*

3.1. *Encoding using match*

The encoding of prefixes using match, shown in Table 2, utilizes the fusion power of two interacting solos in combination with the causal dependency of a match continuation on its guard. The encoding of an input prefix $u\,\widetilde{x}\,.\,P$ creates two fresh names z and w. The continuation P of the prefix is guarded by a match operator checking for equality between z and w; being fresh, these are initially different from each other, so P cannot reduce. The input prefix action $u\,\widetilde{x}$ is encoded by a solo with the same subject and polarity, but with three additional objects zww appended to $\widetilde{x}$. An output prefix $\overline{u}\,\widetilde{y}\,.\,Q$ is encoded symmetrically, but with the order of the additional objects changed to wwz. When such input and output solos interact, the result is a fusion of the names z and w on each side of the interaction, thus triggering the continuations P and Q. Increasing polyadicity of names to encode the temporal ordering of prefixes was also used by Parrow in (Parrow, 1995).

To get acquainted with the encoding of Table 2 we detail the interaction of the encoding of two parallel agents:

$$\begin{aligned}
&[\![(x)(u\,x\,.\,P \mid \overline{u}\,y\,.\,Q)]\!] \\
&\overset{def}{=} \quad (x)(\ (zw)(u\,xzww \mid [z=w][\![P]\!]) \\
&\qquad\qquad \mid (zw)(\overline{u}\,ywwz \mid [z=w][\![Q]\!])) \\
&\equiv \quad (xz_1z_2w_1w_2)(\ u\,xz_1w_1w_1 \mid [z_1=w_1][\![P]\!] \\
&\qquad\qquad\qquad\qquad \mid \overline{u}\,yw_2w_2z_2 \mid [z_2=w_2][\![Q]\!]) \\
&\longrightarrow \quad (z_1)(([z_1=w_1][\![P]\!] \mid [z_2=w_2][\![Q]\!])\{y/x, z_1/w_1, z_1/w_2, z_1/z_2\}) \\
&\equiv \quad ([\![P \mid Q]\!])\{y/x\} \\
&= \quad [\![(P \mid Q)\{y/x\}]\!]
\end{aligned}$$

which corresponds exactly to the result of the encoded prefix agents interacting.

This operational correspondence is formalized in the following lemmas:

Lemma 10. For P an agent of f_{pre}, $P \sim [\![P]\!]$.

Proof. It suffices to demonstrate the items below, which follow by structural induction on P or $[\![P]\!]$, or by reduction induction on $P \longrightarrow P'$ or $[\![P]\!] \longrightarrow Q$. We also use $\equiv\subseteq\sim$, by Proposition 7.

1 if $P \equiv P'$ then $[\![P]\!] \equiv [\![P']\!]$;
2 if $P \longrightarrow P'$ then $[\![P]\!] \longrightarrow [\![P']\!]$;

3 if $P \downarrow x$ then $[\![P]\!] \downarrow x$;
4 if $[\![P]\!] \longrightarrow Q$, then $P \longrightarrow P'$ such that $Q \equiv [\![P']\!]$;
5 if $[\![P]\!] \downarrow x$ for some x, then $P \downarrow x$.

$\square$

The encoding $[\![\cdot]\!]$ is adequate with respect to strong barbed congruence, i.e., if $[\![P]\!] \sim [\![Q]\!]$ then $P \sim Q$. This correspondence follows immediately by the following lemma about strong barbed bisimulation.

Lemma 11. For P, Q two agents of f_{pre}, $P \stackrel{\cdot}{\sim} Q$ iff $[\![P]\!] \stackrel{\cdot}{\sim} [\![Q]\!]$.

Proof. By Lemma 10, showing that $\{([\![P]\!], [\![Q]\!]) : P \stackrel{\cdot}{\sim} Q\}$ and $\{(P, Q) : [\![P]\!] \stackrel{\cdot}{\sim} [\![Q]\!]\}$ are barbed bisimulations. $\square$

Theorem 12. For P, Q two agents of f_{pre}, $[\![P]\!] \sim [\![Q]\!]$ implies $P \sim Q$.

The completeness of $[\![\cdot]\!]$ with respect to strong barbed congruence, i.e., $P \sim Q$ implies $[\![P]\!] \sim [\![Q]\!]$, does not hold, since an arbitrary agent can interact with the solo encoding of a prefix action *without* fusing the appropriate names. For instance, $(x)(\overline{u}\,x.\,x) \sim (x)(\overline{u}\,x \mid x)$, while $[\![(x)(\overline{u}\,x.\,x)]\!] \not\sim [\![(x)(\overline{u}\,x \mid x)]\!]$ because the context $u\,abcd \mid [\,]$ separates them.

A result similar to Theorem 12 also holds for $\approx$. In fact, since $\sim \subseteq \approx$, Lemma 10 may be weakened by replacing $\sim$ with $\approx$. Therefore a lemma corresponding to Lemma 11, where $\sim$ is replaced by $\approx$, may be demonstrated. This yields:

Theorem 13. For P, Q two agents of f_{pre}, $[\![P]\!] \approx [\![Q]\!]$ implies $P \approx Q$.

3.2. *Encoding without match*

While the encoding above has very pleasant properties, it may for reasons of minimality be desirable to do without the match operator.

In this section we show that the symmetric form of communication of the solos calculus, together with the *implicit* form of causal dependency described in the introduction, are expressive enough to encode the prefix operator. The encoding is presented in Table 3. The causal dependency used in this encoding is that of a bound subject which cannot interact until the scope has been lifted by a fusion effect. To this aim, the subject of a prefix is encoded by a fresh scoped name w. The whole encoding has a parameter v, and an interaction over this parameter can fuse the fresh name to the original subject, removing the scope of w and eventually enabling a reaction. In order to achieve this we introduce *catalyst agents* $U_v \equiv (z)v\,zzv$ and we add an initial catalyst in the top level encoding $[\!(\cdot)\!]$.

An example illustrating the encoding follows:

$$\begin{array}{ll}
[\!((x)(u\,x\,.\,P \mid \overline{u}\,y\,.\,Q))\!] & \\
\quad \stackrel{def}{=} \quad (vx)\big(\ (v')((w)(w\,xvv' \mid (y)(\overline{v}\,uwy \mid U_y)) \mid [\![P]\!]_{v'}) & \\
\qquad\qquad\quad \mid (v')((w)(\overline{w}\,yv'v \mid (y)(\overline{v}\,uwy \mid U_y)) \mid [\![Q]\!]_{v'}) & \\
\qquad\qquad\quad \mid U_v\big) &
\end{array} \qquad (1)$$

$$\begin{array}{rcl}
U_v & \equiv & (z)v\,zzv \\
[\![u\,\widetilde{x}\,.\,P]\!]_v & \stackrel{def}{=} & (v')((w)(w\,\widetilde{x}vv' \mid (y)(\overline{v}\,uwy \mid U_y)) \mid [\![P]\!]_{v'}) \\
[\![\overline{u}\,\widetilde{x}\,.\,P]\!]_v & \stackrel{def}{=} & (v')((w)(\overline{w}\,\widetilde{x}v'v \mid (y)(\overline{v}\,uwy \mid U_y)) \mid [\![P]\!]_{v'}) \\
[\![[x=y]P]\!]_v & \stackrel{def}{=} & [x=y][\![P]\!]_v \\
[\![(x)P]\!]_v & \stackrel{def}{=} & (x)[\![P]\!]_v \\
[\![P \mid Q]\!]_v & \stackrel{def}{=} & [\![P]\!]_v \mid [\![Q]\!]_v \\
(\![P]\!) & \stackrel{def}{=} & (v)([\![P]\!]_v \mid U_v)
\end{array}$$

Table 3. *Encoding of prefixes without using match. v, w and v' are fresh.*

$$\begin{array}{rll}
\longrightarrow & (vxv'_1v'_2)(\;u\,xvv'_1 \mid [\![P]\!]_{v'_1} \mid U_v & (2)\\
& \qquad \mid (w)(\overline{w}\,yv'_2v(y)(\overline{v}\,uwy \mid U_y)) \mid [\![Q]\!]_{v'_2}) & \\
\longrightarrow & (vxv'_1v'_2)(u\,xvv'_1 \mid [\![P]\!]_{v'_1} \mid \overline{u}\,yv'_2v \mid [\![Q]\!]_{v'_2} \mid U_v) & (3)\\
\longrightarrow & (v)([\![P]\!]_v \mid [\![Q]\!]_v \mid U_v)\{y/x\} & \\
\stackrel{def}{=} & (v)([\![P \mid Q]\!]_v\{y/x\} \mid U_v) & \\
= & (\![(P \mid Q)\{y/x\}]\!) & (4)
\end{array}$$

Initially the solos corresponding to the prefix actions cannot interact, since their subjects are locally scoped. Expanding the definitions we can see at (1) that the catalyst $(z)v\,zzv$ can interact with one of the terms $\overline{v}\,uwy$, thereby changing the subject of the prefix solo, removing the scope. The initial catalyst is consumed when it interacts with the term $\overline{v}\,uwy$, but this interaction also changes the subterm U_y into a new catalyst U_v. This can be used at (2) to remove the scope of the other prefix solo, enabling the interaction between the two prefixes at (3) and producing another new catalyst. Observe that as a side effect of this latter interaction, the continuations $[\![P]\!]_{v'_1}$ and $[\![Q]\!]_{v'_2}$ are now enabled since v'_1 and v'_2 are fused with v. We end up with the desired result (4).

Before proving our main result about $(\![\cdot]\!)$ we require a few basic properties of this mapping. We first point out a basic algebraic property of the encoding $[\![\,\cdot\,]\!]_v$.

Proposition 14. For P an agent of f_{pre}, σ a substitution, and $v \notin \mathrm{fv}(P) \cup \mathrm{dom}(\sigma) \cup \mathrm{ran}(\sigma)$, $[\![P\sigma]\!]_v = [\![P]\!]_v\sigma$.

The following lemma shows the tight correspondence between reductions and observations of P and those of $(\![P]\!)$. The proof is detailed in Appendix A.

Lemma 15. For P an agent of f_{pre}, $P \lesssim (\![P]\!)$.

The correspondence established by Lemma 15 is weaker than Lemma 10 for Table 2 ($\lesssim$ instead of $\sim$). This is because in order to set the proper causal dependencies, every reduction in $P \in f_{\mathrm{pre}}$ amounts to a sequence of (three) reductions in $(\![P]\!)$. Nevertheless, the encoding $(\![\cdot]\!)$ is adequate with respect to weak barbed congruence. To prove this, we need the lemma below.

Lemma 16. For P, Q two agents of f_{pre}, $P \stackrel{\cdot}{\approx} Q$ iff $[\![(P)]\!] \stackrel{\cdot}{\approx} [\![(Q)]\!]$.

Proof. We use the fact that $\{(P,Q) : P \gtrsim\stackrel{\cdot}{\approx}\lesssim Q\}$ and $\{(P,Q) : P \lesssim\stackrel{\cdot}{\approx}\gtrsim Q\}$ are both weak barbed bisimulations. Therefore, by Lemma 15, we derive that $P \lesssim [\![(P)]\!] \stackrel{\cdot}{\approx} [\![(Q)]\!] \gtrsim Q$ and $[\![(P)]\!] \gtrsim P \stackrel{\cdot}{\approx} Q \lesssim [\![(Q)]\!]$ are weak barbed bisimulations. $\square$

Theorem 17. For P, Q two agents of f_{pre}, $[\![(P)]\!] \approx [\![(Q)]\!]$ implies $P \approx Q$.

3.3. *Other translations*

We also considered other translations, which improve the encoding of Table 3 in terms of the number of book-keeping reductions or in terms of channel sorts.

The following translation requires that channels carry only one extra object instead of two, with the added cost of a further book-keeping interaction for every original interaction. We only show the translation rules for prefixes since the others are the same:

$$\begin{array}{rcl}
[\![u\,\widetilde{x}\,.\,P]\!]_v & \stackrel{def}{=} & (wz)(w\,\widetilde{x}z \mid (v')(z\,v'v \mid [\![P]\!]_{v'}) \mid (y)(\overline{v}\,uwy \mid U_y)) \\
[\![\overline{u}\,\widetilde{x}\,.\,P]\!]_v & \stackrel{def}{=} & (wz)(\overline{w}\,\widetilde{x}z \mid (v')(\overline{z}\,vv' \mid [\![P]\!]_{v'}) \mid (y)(\overline{v}\,uwy \mid U_y))
\end{array}$$

Once replication has been added (see Section 4), we can also get rid of the catalyst agents inside the encodings of prefixes of $[\![\,\cdot\,]\!]_v$. Instead we use a replicated catalyst at top level. Catalysts now need only two objects:

$$\begin{array}{rcl}
[\![u\,\widetilde{x}\,.\,P]\!]_v & \stackrel{def}{=} & (wv')(w\,\widetilde{x}vv' \mid [\![P]\!]_{v'} \mid \overline{v}\,uw) \\
[\![\overline{u}\,\widetilde{x}\,.\,P]\!]_v & \stackrel{def}{=} & (wv')(\overline{w}\,\widetilde{x}v'v \mid [\![P]\!]_{v'} \mid \overline{v}\,uw) \\
[\![(P)]\!] & \stackrel{def}{=} & (v)([\![P]\!]_v \mid !\,(z)v\,zz)
\end{array}$$

The processes in Table 3 in any case evaluate the catalyst agents to prepare the term for the initial reductions. These moves are useless if the original term does not reduce. Our last translation allows to remove the initial book-keeping reductions (again we only illustrate the rules for prefixes):

$$\begin{array}{rcl}
(\!(u\,\widetilde{x}\,.\,P)\!) & \stackrel{def}{=} & (z)(u\,\widetilde{x}z \mid (vv')(z\,vv' \mid [\![P]\!]_{v'} \mid v\,zzv)) \\
(\!(\overline{u}\,\widetilde{x}\,.\,P)\!) & \stackrel{def}{=} & (z)(\overline{u}\,\widetilde{x}z \mid (v)(\overline{z}\,vv \mid [\![P]\!]_v))
\end{array}$$

In contrast with $[\![(\cdot)]\!]$, the function $(\!(\,\cdot\,)\!)$ does not introduce any initial catalyst agent and does not rename the subjects of unguarded solos. Solos in continuations are managed as in Table 3.

4. Replication and Recursion

We can add replication to the f_{pre} calculus by introducing the operator $!\,P$ and the structural law $!\,P \equiv P \mid !\,P$. Extending our first encoding using match (Table 2) with $[\![!\,P]\!] =! [\![P]\!]$, Lemmas 10, 11 and Theorem 12 still hold. As an immediate consequence of Lemma 10, we can further strengthen these results by proving preservation of divergence properties.

Theorem 18. For P an agent of f_{pre} with replication, P diverges iff $[\![P]\!]$ diverges.

Extending the second encoding (Table 3) in the same way does not preserve divergence, since $[\!(\,!\,u\,.\,\mathbf{0})\!]$ could reduce indefinitely even though the original term cannot reduce. This is due to the book-keeping reductions which are needed to implement prefixing. However, if we replace replication with guarded recursion, introduced by the structural law $A(\widetilde{y}) \equiv P\{\widetilde{y}/\widetilde{x}\}$ if $A(\widetilde{x}) \overset{def}{=} P$ and $|\widetilde{x}| = |\widetilde{y}|$, where $\widetilde{x}$ are pairwise distinct, and the requirement that process variables only occur under a prefix, then Lemmas 15, 16 and Theorem 17 still hold. In particular, also Propositions 30, 31, and 32 may easily be generalized.

Theorem 19. For P an agent of f_{pre} with guarded recursion, P diverges iff $[\!(P)\!]$ diverges.

Proof. If P diverges, the divergence of $[\!(P)\!]$ is an immediate consequence of Lemma 15. For the *vice versa*, let $(v)(U_v \mid [\!(P)\!]v)$ diverge. Then, by Lemma 33 (in Appendix A), P diverges too. □

Remarkably, the above extension of the solos calculus with replication may be reduced, without losing expressiveness. The next theorem determines how nested replications may be flattened into non-nested replications. As a consequence of the flattening theorem below, we can reduce to consider agents with non-nested replications. This simplifies the theory of the solos calculus, as well as the practice (Laneve et al., 2001). For instance, we have a simple form of canonical representatives of agents:

Remark 20. Every agent in the solos calculus with non-nested replications is structurally congruent to an agent of shape $(\widetilde{x})(P \mid \prod_{i\in I} !\,(\widetilde{y_i})Q_i)$, where P and Q_i, $i \in I$, are parallel composition of solos.

In the π-calculus, Parrow's *concerts of trios* construction (Parrow, 2000) provides a similar flattening: there, the canonical form is the same but with P and Q_i being sequences of prefixes $\alpha_1\,.\,\alpha_2\,.\,\alpha_3$. Again, no nested replication is needed.

Theorem 21. (Flattening)

$$!\,(\widetilde{x})(P \mid !\,Q) \approx (y)(!\,(\widetilde{x})(P \mid \overline{y}\,\widetilde{z}y) \mid !\,(\widetilde{z}v)(y\,\widetilde{z}v \mid \overline{v}\,\widetilde{z}v \mid Q))$$

where $\widetilde{z} = \mathrm{fn}(Q)$ and y, v are fresh names.

Proof. Let

$$\begin{array}{rcl} \mathsf{R}_{P,Q} & = & !\,(\widetilde{x})(P \mid !\,Q) \\ \mathsf{R}'_{P,Q} & = & (y)(!\,(\widetilde{x})(P \mid \overline{y}\,\widetilde{z}y) \mid !\,(\widetilde{z}v)(y\,\widetilde{z}v \mid \overline{v}\,\widetilde{z}v \mid Q)) \end{array}$$

Let $\mathcal{S}$ be a relation containing the pairs $(C[\mathsf{R}\sigma], C[\mathsf{R}'\sigma])$, for every context $C[\cdot]$ and every substitution σ, such that:

— $\mathsf{R} = \mathsf{R}_{P,Q} \mid \prod_{i\in 1..n}(!\,Q\{\widetilde{w_i}/\widetilde{z}\} \mid \prod_{j\in 1..m_i} Q_i\{\widetilde{w_i}/\widetilde{z}\})$;

— $\mathsf{R}' = \mathsf{R}'_{P,Q} \mid \prod_{i\in 1..n}(y)\Big(\overline{y}\,\widetilde{w_i}y \mid !\,(\widetilde{z}v)(y\,\widetilde{z}v \mid \overline{v}\,\widetilde{z}v \mid Q) \mid \prod_{j\in 1..m_i}(\widetilde{z}v)(y\,\widetilde{z}v \mid \overline{v}\,\widetilde{z}v \mid Q_i)\Big)$

for every m, n, $C[\,]$, Q_i, where $\widetilde{z_i} = \mathrm{fn}(Q_i)$, and $y, v, \widetilde{w_i}$ are fresh names.

The relation $\mathcal{S}$ is clearly a congruence. To demonstrate $\mathcal{S}$ is a weak barbed bisimulation up-to structural congruence, we distinguish among the reductions that may occur on each component of the pair, and in each case we simulate them on the other component. There are three cases, according to whether the reduction is inside the hole of the context, inside the context, or involves both the context and the process in the hole. We detail the last case; the former two are easy.

There are three subcases. The first one is when the interacting solos belong to the context and to the agent in the hole, respectively. Let $C[\cdot] = C'[(\widetilde{u})(T \mid \overline{u'}\,\widetilde{u''} \mid [\cdot])]$ and let $C[\mathsf{R}]\ \mathcal{S}\ C[\mathsf{R}']$. There are four possibilities:

1 $P = u'\,\widetilde{v''} \mid P'$. Then

$$\begin{array}{rcl} C[\mathsf{R}] & \longrightarrow\equiv & C'[(\widetilde{v'})(T\sigma \mid P'\sigma \mid [\mathsf{R}_1\sigma])] \\ C[\mathsf{R}'] & \longrightarrow\equiv & C'[(\widetilde{v'})(T\sigma \mid P'\sigma \mid [\mathsf{R}'_1\sigma])] \end{array}$$

where σ agrees with $\{\widetilde{u''} = \widetilde{v''}\}$. Let $C''[\cdot] = C'[(\widetilde{v'})(T\sigma \mid P'\sigma \mid [\cdot])]$. Notice that $C''[\mathsf{R}_1\sigma]\ \mathcal{S}\ C''[\mathsf{R}'_1\sigma]$.

2 $Q = u'\,\widetilde{v''} \mid Q'\{\widetilde{w_i}/\widetilde{z}\}$. Then

$$\begin{array}{rcl} C[\mathsf{R}] & \longrightarrow\equiv & C'[(\widetilde{v'})(T\sigma \mid P\sigma \mid Q'\sigma \mid [\mathsf{R}_1\sigma])] \\ C[\mathsf{R}'] & \longrightarrow^{*}\equiv & C'[(\widetilde{v'})(T\sigma \mid P'\sigma \mid Q'\sigma \mid [(\mathsf{R}'_1 \mid \overline{y}\,\widetilde{w_i}y \mid !\,(\widetilde{z}v)(y\,\widetilde{z}v \mid \overline{v}\,\widetilde{z}v \mid Q))\sigma])] \end{array}$$

where σ agrees with $\{\widetilde{u''} = \widetilde{v''}\}$. Let $C''[\cdot] = C'[(\widetilde{v'})(T\sigma \mid P'\sigma \mid Q'\sigma \mid [\cdot])]$. Notice that $C''[\mathsf{R}_1\sigma]\ \mathcal{S}\ C''[(\mathsf{R}'_1 \mid \overline{y}\,\widetilde{w_i}y \mid !\,(\widetilde{z}v)(y\,\widetilde{z}v \mid \overline{v}\,\widetilde{z}v \mid Q))\sigma]$.

3 $Q\{\widetilde{w_i}/\widetilde{z}\} = u'\,\widetilde{v''} \mid Q'\{\widetilde{w_i}/\widetilde{z}\}$. Similar to cases 1 and 2.

4 $Q_i\{\widetilde{w_i}/\widetilde{z}\} = u'\,\widetilde{v''} \mid Q'_i\{\widetilde{w_i}/\widetilde{z}\}$. Similar to cases 1 and 2.

The second subcase is when the interacting solos are both in the context. Then the reduction may modify the agent in the hole according to a substitution. The conclusion is immediate because $\mathcal{S}$ is closed up-to substitutions of agents in the hole.

The third subcase is when the interacting solos are both in the agent in the hole. It is not difficult to verify that the derivatives of the agents in the holes still have shape R and R' and the context is possibly augmented with a residual of the reduction, as in cases 1 and 2 above. □

5. The dyadic solos calculus

The solos calculus is polyadic, i.e., names carry tuples of values. Polyadicity is crucial in the translations of Tables 2 and 3 to encode the causal dependencies of prefixes. In this section we show that the dyadic sub-calculus, where solos carry at most two values, is as expressive as the full polyadic calculus. This fact is already known in asynchronous π-calculus (Boudol, 1992), and here we rephrase the arguments in the solos calculus. (In the original π-calculus the monadic sub-calculus is sufficient (Milner, 1993).) In addition, we demonstrate that some expressiveness is lost when solos are monadic. All these results concern the so-called well-sorted solos calculus, whose theory is detailed in the next section.

5.1. *A sorting discipline for the solos calculus*

The solos calculus may be equipped with a recursive sorting discipline on names that avoids arity mismatch, in the style of (Milner, 1993). To this aim, let $\mathtt{x}$ be a generic sort-variable, and τ be a *sort* defined by the following grammar:

$$\tau \;::=\; \langle\rangle \;\mid\; \mathtt{x} \;\mid\; \langle\widetilde{\tau}\rangle \;\mid\; \mathsf{rec}\,\mathtt{x}.\,\tau$$

A name u has sort $\langle\tau_1,\cdots,\tau_k\rangle$ if it can carry k objects of sort $\tau_1,\cdots,\tau_k$, respectively. The sort $\mathsf{rec}\,\mathtt{x}.\,\tau$ models recursive sorts. For example, if u has sort $\mathsf{rec}\,\mathtt{x}.\,\langle\mathtt{x}\rangle$, it means that a message as $u\,u$ is well-sorted. As usual, the operator $\mathsf{rec}\,\mathtt{x}.\text{-}$ is a binder, which induces the standard notions of bound and free sort-variables. When we write $\mathsf{rec}\,\mathtt{x}.\,\tau$, we always assume that $\mathtt{x}$ belongs to the free sort-variables in τ and $\mathtt{x}$ occurs inside brackets $\langle\cdot\rangle$. Let τ be *infinite* if

1 $\tau = \mathsf{rec}\,\mathtt{x}.\,\tau'$;
2 or $\tau = \langle\tau_1,\cdots,\tau_k\rangle$ and τ_i is infinite, for some $i \in 1..k$.

The sort τ is called *finite* if it is not infinite. We associate a *weight* to channels with finite sorts, namely the maximum number of nested brackets $\langle\cdot\rangle$ in the sort. Formally let $\mathit{weight}(x) = n$ if the sort of x is finite and the maximum number of nested brackets $\langle\cdot\rangle$ in the sort is n, and $\mathit{weight}(x) = 0$ otherwise, For example, a channel with sort $\langle\langle\langle\rangle\rangle,\langle\rangle\rangle$ has weight 3, while a channel with sort $\langle\mathsf{rec}\,\mathtt{x}.\,\langle\mathtt{x}\rangle,\langle\rangle\rangle$ has weight 0 because its sort is infinite.

An agent is *well-sorted* when

1 names have unique sorts,
2 subjects of solos always carry objects of the right sort, and
3 matching concerns names of the same sort.

The *well-sorted solos calculus* is the sub-calculus where all agents are well-sorted.

Two preliminary propositions gather basic properties of reductions in the well-sorted calculus.

Proposition 22. In the well-sorted solos calculus:

1 reductions fuse names of the same sort;
2 a reduction fuses names of infinite sorts if and only if the subjects of the reduction have infinite sort;
3 a reduction between solos with subjects of weight n fuses names whose sorts are finite and have weight strictly less than n, and *vice versa.*

It is possible to associate an integer to every well-sorted process P, which denotes the maximal weight of channels in P. Let $\mathsf{mweight}[P]$ be the function defined inductively as follows:

$$\begin{array}{rcl}
\mathsf{mweight}[\mathbf{0}] & = & 0 \\
\mathsf{mweight}[\alpha\,u_1\cdots u_k] & = & \mathsf{max}\{\mathit{weight}(\alpha),\mathit{weight}(u_1),\cdots\mathit{weight}(u_k)\} \\
\mathsf{mweight}[P \mid Q] & = & \mathsf{max}\{\mathsf{mweight}[P],\mathsf{mweight}[Q]\} \\
\mathsf{mweight}[(x)P] & = & \mathsf{mweight}[P] \\
\mathsf{mweight}[[x=y]P] & = & \mathsf{max}\{\mathsf{mweight}[P],\mathit{weight}(x),\mathit{weight}(y)\} \\
\mathsf{mweight}[!\,P] & = & \mathsf{mweight}[P]
\end{array}$$

It is routine to check that reductions do not increase the value of $\mathsf{mweight}[\cdot]$.

Proposition 23. For every well-sorted process P, $P \longrightarrow^* P'$ implies $\mathsf{mweight}[P] \geq \mathsf{mweight}[P']$.

5.2. *Reducing polyadicity to dyadicity*

Let $[\![\cdot]\!]$ be the translation from the solos calculus to the dyadic sub-calculus, which is the identity everywhere except for solos which carry at least three values, i.e., $n > 2$:

$$\begin{array}{rcl} [\![u\, x_1 \cdots x_n]\!] & \overset{def}{=} & (z)(u\, x_1 z \mid [\![z\, x_2 \cdots x_n]\!]) \\ [\![\overline{u}\, x_1 \cdots x_n]\!] & \overset{def}{=} & (z)(\overline{u}\, x_1 z \mid [\![\overline{z}\, x_2 \cdots x_n]\!]) \end{array}$$

Note that polyadic solos carrying objects $x_1 \cdots x_n$ are encoded as sequences of dyadic solos. Each solo emits exactly two objects, the first of which is one of the x_i; the second object is a bound name which is used in the encoding of the rest of the sequence (except for the last solo). This second object is used as a subject in another solo, which cannot take part in a reduction until after the first solo has. The definition of $[\![\cdot]\!]$ immediately entails the following lemma.

Lemma 24. If $[\![P]\!] \longrightarrow^* P'$ then for some agent Q

1 $P' \equiv (\widetilde{u})([\![Q]\!] \mid \prod_{i\in I}(z_i)([\![\overline{z_i}\, \widetilde{y_i}]\!] \mid [\![z_i\, \widetilde{x_i}]\!]))$;

2 $P' \longrightarrow^* \equiv (\widetilde{u})[\![Q]\!]\sigma$, where σ agrees with $\{\widetilde{y_i} = \widetilde{x_i} \mid i \in I\}$. This process $(\widetilde{u})[\![Q]\!]\sigma$ is called the *P'-completion.*

The correctness of the encoding is an immediate consequence of the following lemma.

Lemma 25. In the well-sorted solos calculus, $P \stackrel{\bullet}{\approx} Q$ if and only if $[\![P]\!] \stackrel{\bullet}{\approx} [\![Q]\!]$.

Proof. (If direction) By definition of the encoding, we observe that

1 $P \equiv Q$ implies $[\![P]\!] \equiv [\![Q]\!]$;

2 $P \longrightarrow Q$ implies $[\![P]\!] \longrightarrow^* [\![Q]\!]$;

3 $P \downarrow x$ if and only if $[\![P]\!] \downarrow x$.

Let $\mathcal{S}$ be a weak barbed bisimulation between $[\![P]\!]$ and $[\![Q]\!]$. The above three properties immediately entail that $\mathcal{S}' = \{(P', Q') \mid [\![P']\!]\ \mathcal{S}\ [\![Q']\!]\}$ is a weak barbed bisimulation.

(Only-if direction) Let $\mathcal{S}$ be a weak barbed bisimulation such that $P\ \mathcal{S}\ Q$, and let $\mathcal{P}$ and $\mathcal{Q}$ be the set of derivatives of $[\![P]\!]$ and $[\![Q]\!]$, respectively, including $[\![P]\!]$ and $[\![Q]\!]$. Consider the relation in $\mathcal{P} \times \mathcal{Q}$

$$\begin{array}{rl} \mathcal{S}' = \{(P', Q') \mid & P''\ \mathcal{S}\ Q'' \\ & \text{and } [\![P'']\!] \text{ is a } P'\text{-completion} \\ & \text{and } [\![Q'']\!] \text{ is a } Q'\text{-completion}\} \end{array}$$

By definition $[\![P]\!]\ \mathcal{S}'\ [\![Q]\!]$. We demonstrate that $\mathcal{S}'$ is a weak barbed bisimulation up-to structural congruence. We detail the case when the left component reduces; the symmetric case is similar. Let $P'\ \mathcal{S}'\ Q'$ and let $P' \longrightarrow P''$. By Lemma 24(1), $P' \equiv (\widetilde{u})([\![P'_1]\!] \mid \prod_{i\in I}(z_i)([\![\overline{z_i}\, \widetilde{y_i}]\!] \mid [\![z_i\, \widetilde{x_i}]\!]))$. Therefore there are two cases: either (a) $P' \longrightarrow$

P'' is due to a reduction inside $[\![P_1']\!]$, or (b) $P' \longrightarrow P''$ is due to a reduction inside $\prod_{i\in I}(z_i)([\![\overline{z_i}\,\widetilde{y_i}]\!] \mid [\![z_i\,\widetilde{x_i}]\!])$. Case (b) is immediate because $P''\ \mathcal{S}'\ Q'$, by definition of $\mathcal{S}'$. In case (a), let

$$P'' \equiv (\widetilde{u})([\![P_1'']\!] \mid (z)([\![\overline{z}\,\widetilde{y}]\!] \mid [\![z\,\widetilde{x}]\!]) \mid \prod_{i\in I}(z_i)([\![\overline{z_i}\,\widetilde{y_i}]\!] \mid [\![z_i\,\widetilde{x_i}]\!]))$$

and let P_0 and Q_0 be the P' and Q'-completions, respectively. In particular, notice that, by Lemma 24(2), $P_0 \equiv (\widetilde{u})[\![P_1']\!]\sigma$, where σ agrees with $\{\widetilde{x_i} = \widetilde{y_i} \mid i \in I\}$. Next, observe that the reduction $P' \longrightarrow P''$ amounts to a reduction $P_0 \longrightarrow P_0'$ such that P_0' is the P''-completion. Therefore, there is a Q_0' with $Q_0 \longrightarrow^* Q_0'$ and $P_0'\ \mathcal{S}\ Q_0'$. We conclude by remarking $Q' \longrightarrow^* Q_0 \longrightarrow^* Q_0'$ and $P''\ \mathcal{S}'\ [\![Q_0']\!]$. $\square$

Clearly, the above theorem is false when agents are not well-sorted. For instance $(xy)(x\,zyy \mid \overline{x}\,yy \mid y\,) \dot{\approx} \mathbf{0}$, while $[\![(xy)(x\,zyy \mid \overline{x}\,yy \mid y\,)]\!] \not\dot{\approx} [\![\mathbf{0}]\!]$.

Lemma 25 allows us to derive the adequacy of the encoding into the dyadic solos calculus.

Theorem 26. In the well-sorted solos calculus, $[\![P]\!] \approx [\![Q]\!]$ implies $P \approx Q$.

5.3. *Expressiveness of the monadic solos calculus*

The question which naturally arises is whether there is an encoding of polyadic solos into monadic ones, i.e., whether the monadic solos calculus, where names have nullary or monadic sorts, is expressive enough. In the sub-calculus *without* the match operator, we demonstrate that the monadic solos calculus is *not as expressive* as the polyadic one, when the calculus is well-sorted. The proof technique is similar to the one used by Parrow in (Parrow, 2000).

A preliminary lemma conveys a basic property for the inexpressiveness of the monadic solos calculus.

Lemma 27. Let P be a well-sorted term in the monadic solos calculus, $x \in \mathrm{fn}(P)$, and $\mathit{weight}(x) = n$. Let $P \longrightarrow^* P'$ be a minimal derivation such that $P' \downarrow x$, i.e., no other derivation producing a barb on x is shorter. Then every solo reduced in this derivation has subject of finite sort, which is strictly greater than n.

Proof. By induction on the length of the derivation $P \longrightarrow^* P'$. When the length is 0, the lemma is obvious. Otherwise, let $P \longrightarrow Q \longrightarrow^* P'$. By the induction hypothesis, all solos reduced in $Q \longrightarrow^* P'$ have subjects of finite sort, whose weight is strictly greater than n. There are two subcases: either $Q \longrightarrow^* P'$ has length 0, or is greater than 0. We discuss the latter: the former subcase is simpler.

Let $P \equiv (\widetilde{u})(\overline{z}\,v \mid z\,w \mid Q_1)$, where the two solos of the reduction $P \longrightarrow Q$ have been singled out. There are three cases:

i exactly one of the names v, w must occur at least once in solos reduced in $Q \longrightarrow^* P'$;
ii at least one of the names v, w is x;
iii neither (i) nor (ii) is true, i.e., none of the names v, w occur in solos involved in reductions of $Q \longrightarrow^* P'$, and none is x.

In case (i), since subjects of solos reduced in $Q \longrightarrow^* P'$ have finite sort, then by Proposition 22, v, w also have finite sorts with weight greater than n. Therefore z has finite sort and weight greater than n, by the same proposition.

In case (ii), the lemma is an obvious consequence of Proposition 22.

To conclude we demonstrate that case (iii) is vacuous. Indeed, in this case, we show that a term with an x-barb may be derived from P with a shorter derivation, which contradicts the hypothesis that $P \longrightarrow Q \longrightarrow^* P'$ is a minimal derivation. By Remark 20, $P \equiv (\widetilde{u})(\overline{z}\, v \mid z\, w \mid Q' \mid Q'')$, where

1 Q' collects replications and solos,

2 Q'' is a parallel composition of solos, such that Q'' may contain replicas of terms in Q' underneath "!"; these terms result from the structural law $!R \equiv R \mid !R$;

We further constrain Q'' to collect a minimal set of solos such that every reduction in $Q \longrightarrow^* P'$ involves solos in Q'', i.e., $P \longrightarrow Q \equiv (\widetilde{u_0})(Q'\sigma_0 \mid Q''\sigma_0)$ and $Q \longrightarrow^* P'$ is the derivation:

$$\begin{array}{rcl}
(\widetilde{u_0})(Q'\sigma_0 \mid Q''\sigma_0) & \longrightarrow & (\widetilde{u_1})(Q'\sigma_0\sigma_1 \mid Q_1\sigma_0\sigma_1) \\
& \longrightarrow & (\widetilde{u_2})(Q'\sigma_0\sigma_1\sigma_2 \mid Q_2\sigma_0\sigma_1\sigma_2) \\
& \longrightarrow & \cdots \\
& \cdots & \\
& \longrightarrow & (\widetilde{u_k})(Q'\sigma_0\sigma_1\sigma_2\cdots\sigma_k \mid Q_k\sigma_0\sigma_1\sigma_2\cdots\sigma_k) \equiv P'
\end{array}$$

where $Q_k\sigma_0\sigma_1\sigma_2\cdots\sigma_k = (\overline{x}\, y \mid Q'_k)$ or $Q_k\sigma_0\sigma_1\sigma_2\cdots\sigma_k = (x\, y \mid Q'_k)$, for some y and Q'_k, and where $Q_i\sigma_0\cdots\sigma_i \not\downarrow x$ for each $i < k$.

Without loss of generality, let $\sigma_0 = \{v/w\}$. Since in this case solos reduced in the derivation never contain v, by Table 1 we may rewrite the above derivation as follows:

$$\begin{array}{rcl}
(\widetilde{u_0})(\overline{z}\, v \mid z\, w \mid Q' \mid Q'') & \longrightarrow & (\widetilde{u_1})((\overline{z}\, v \mid z\, w \mid Q')\sigma_1 \mid Q_1\sigma_0\sigma_1) \\
& \longrightarrow & (\widetilde{u_2})((\overline{z}\, v \mid z\, w \mid Q')\sigma_1\sigma_2 \mid Q_2\sigma_1\sigma_2) \\
& \longrightarrow & \cdots \\
& \cdots & \\
& \longrightarrow & (\widetilde{u_k})((\overline{z}\, v \mid z\, w \mid Q')\sigma_1\sigma_2\cdots\sigma_k \mid Q_k\sigma_1\sigma_2\cdots\sigma_k)
\end{array}$$

Finally, since $\widetilde{u} = \widetilde{u_0}w$, the second inductive rule of Table 1 yields

$$P \longrightarrow^* (\widetilde{u_k}w)((\overline{z}\, v \mid z\, w \mid Q')\sigma_1\sigma_2\cdots\sigma_k \mid Q_k\sigma_1\sigma_2\cdots\sigma_k)$$

This derivation is shorter than $P \longrightarrow^* P'$, and the final term has a barb on x. This is a contradiction. $\square$

Let us consider an agent A such that,

$$\text{either } A \longrightarrow^* \approx a \text{ or } A \longrightarrow^* \approx a \mid A,$$

namely A is "observationally" terminating by producing a composition of n solos a, for every n, or may not terminate by always emitting a. This nondeterministic behavior may be suitably implemented in the solos calculus by using internal reductions. For example, the above agent A can be defined in the dyadic calculus as follows:

$$(xy)\Big(\overline{x}\, ay \mid !\,(uv)(x\, uv \mid u \mid (w)(\overline{y}\, w \mid \overline{y}\, x \mid v\, w \mid \overline{w}\, ay))\Big)$$

We demonstrate that an agent which is weak barbed congruent to A above is not definable in the well-sorted monadic calculus.

Theorem 28. There is no agent in the well-sorted monadic solos calculus without the match operator which is weak barbed congruent to an agent A such that

$$\text{either } A \longrightarrow^{*} \approx a \text{ or } A \longrightarrow^{*} \approx a \mid A,$$

Proof. By contradiction, we assume there exists such an agent, and let it be P. Since P may produce an infinite number of a, there must be (at least) one subprocess underneath a replication which produces these terms. We consider the case when there is exactly one subprocess. The general case is similar. Therefore

$$P = (\widetilde{u})(P' \mid !\, P'')$$

where P' is a composition of solos. Now observe that

1 $!\, P'' \not\longrightarrow^{*} a \mid Q$. Informally, $!\, P''$ cannot produce, without interacting with P', any solo with free subject a. Otherwise it may produce an infinite amount of such solos, and then it is not possible to derive a finite behaviour of P.

2 Without loss of generality, we assume that the minimal derivation of P yielding a barb a involves solos coming from P' and P''. Formally, it has the following shape (the roles of x and $\overline{x}$, as well as v and w, may be interchanged):

$$\begin{array}{lll}
(\widetilde{u})(P' \mid !\, P'') & \longrightarrow^{*} & \qquad \text{(reductions only involve solos in } P'\text{)} \\
& & (\widetilde{u'})(P'_1 \mid \overline{x}\, v \mid !\, P'')\sigma \\
& \longrightarrow^{*} & \qquad \text{(reductions only involve replicas of } !\, P''\text{)} \\
& & (\widetilde{u''})(P'_1 \mid \overline{x}\, v \mid x\, w \mid P''_1 \mid !\, P'')\sigma' \\
& \longrightarrow & (\widetilde{u'' \setminus w})(P'_1 \mid P''_1 \mid !\, P'')\sigma'\{v/w\} \\
& \longrightarrow^{*} & a \mid Q
\end{array}$$

3 Let τ be the sort of x above. By Lemma 27, τ is finite. By Proposition 22, the term P' must contain at least one solo with subject of sort τ.

4 Next we observe that the production of a finite or infinite number of solos a amounts to produce a finite or infinite number of solos $\overline{x}\, v$. Therefore we must repeat the same argument, now for $\overline{x}\, v$. Again, as in item 1, $\overline{x}\, v$ cannot be produced by reductions inside P'', otherwise it is not possible to derive a finite behaviour of P. Thus, as in 2, without loss of generality, we assume that the minimal derivation of $(\widetilde{u})(P' \mid !\, P'')$ yielding $\overline{x}\, v$ involves solos coming from both P' and P''.

5 This minimal derivation may be written as follows (the roles of y and $\overline{y}$, as well as v' and v'', may be interchanged):

$$\begin{array}{lll}
(\widetilde{u})(P' \mid !\, P'') & \longrightarrow^{*} & \qquad \text{(reductions only involve solos in } P'\text{)} \\
& & (\widetilde{u'})(P'_2 \mid \overline{y}\, v' \mid !\, P'')\sigma \\
& \longrightarrow^{*} & \qquad \text{(reductions only involve replicas of } !\, P''\text{)} \\
& & (\widetilde{u''})(P'_2 \mid \overline{y}\, v' \mid y\, v'' \mid P''_2 \mid !\, P'')\sigma\sigma' \\
& \longrightarrow & (\widetilde{u'' \setminus v''})(P'_2 \mid P''_2 \mid !\, P'')\sigma\sigma'\{v'/v''\} \\
& \longrightarrow & (\widetilde{u'''})(P''' \mid \overline{x}\, v \mid !\, P'')\sigma''
\end{array}$$

Let τ' be the sort of y. By Lemma 27, τ' has finite sort and its weight is strictly greater than the weight of τ. Therefore $\overline{x}\, v$ and $\overline{y}\, v'$ are different solos.

6 The production of a finite or infinite number of solos $\overline{x}\, v$ amounts to produce a finite or infinite number of solos $\overline{y}\, v'$. Therefore we must repeat the same argument, now for $\overline{y}\, v'$.

The conclusion is that P' and a number of replicas of $!\,P''$ must contain an infinite number of different solos with subjects of different finite sorts. This is impossible, by Proposition 23. □

6. Encoding Choice

In this section we present encodings of the choice operator; $P + Q$ allows P or Q to take part in reduction, and discards the other branch when doing so. Here we add the mismatch operator $[x \neq y]P$, which can act like P if x and y are different. We extend the ranges of $M, N, \widetilde{M}$ and $\widetilde{N}$ and the definition of $\widetilde{M} \Leftrightarrow \widetilde{N}$ appropriately, add the reduction rule

$$\frac{P \longrightarrow P',\ x \neq y}{[x \neq y]P \longrightarrow P'}$$

and the observation rule $[x \neq z]P \downarrow y$ if $P \downarrow y$ and $x \neq z$. We also extend our previous encodings homomorphically for the mismatch operator.

Restricting the general choice operator $P+Q$ to guarded choice, $\sum_I \alpha_i \,.\, P_i$, and further requiring that all α_i in a guarded choice have the same polarity (all inputs or all outputs), we can extend the encoding of Table 2 by replacing the encoding of prefixes by the following, where z and w are fresh:

$$\begin{aligned}
\left[\!\left[\sum_I u_i\, \widetilde{x}_i \,.\, P_i \right]\!\right] &\overset{def}{=} (zw) \prod_I [z \neq w](u_i\, \widetilde{x}_i zww \mid [z = w][\![P_i]\!]) \\
\left[\!\left[\sum_I \overline{u_i}\, \widetilde{x}_i \,.\, P_i \right]\!\right] &\overset{def}{=} (zw) \prod_I [z \neq w](\overline{u_i}\, \widetilde{x}_i wwz \mid [z = w][\![P_i]\!])
\end{aligned}$$

The mismatch operator is used in a "test-and-set" construction which tests two names z and w for inequality, and if they are not equal, atomically makes them so. Only one branch of the choice can succeed in doing this. The interaction between an input and an output prefix now not only enables the continuations of the prefixes, but also *disables* the other branches of the choice. Lemma 10 and 11 and Theorems 12 and 18 still hold for this extended encoding.

The encoding of Table 3 can also be extended in a similar way, replacing the encodings of prefixes (v and v' are fresh):

$$\begin{aligned}
\left[\!\left[\sum_I u_i\, \widetilde{x}_i \,.\, P_i \right]\!\right]_v &\overset{def}{=} (v') \prod_I [v \neq v'](w)(w\, \widetilde{x}_i vv' \mid [\![P_i]\!]_{v'} \mid (y)(\overline{v}\, u_i wy \mid U_y)) \\
\left[\!\left[\sum_I \overline{u_i}\, \widetilde{x}_i \,.\, P_i \right]\!\right]_v &\overset{def}{=} (v') \prod_I [v \neq v'](w)(\overline{w}\, \widetilde{x}_i v'v \mid [\![P_i]\!]_{v'} \mid (y)(\overline{v}\, u_i wy \mid U_y))
\end{aligned}$$

Appendix B illustrates this encoding by an example. Lemmas 15 and 11 and Theorem 17 as well as Theorem 19 hold also for this encoding.

As seen above, the mismatch operator is very powerful in the solos calculus. While mismatch distributes over prefixes in the π-calculus, it does not in the fusion calculus (or its encoding into the solos calculus). This makes it powerful enough for the test-and-set construction used in the encoding above; however, such a construction is difficult to implement in a distributed setting, since all component systems must agree at one point in time that the names are distinct. Therefore, the mismatch operation can be implemented, but at the cost of synchronizing all the processes underneath the mismatch, which may be in different locations. For a tightly coupled system, the cost is lower, and indeed most multiprocessor systems implement some sort of test-and-set instruction.

7. Conclusions

In this paper we have shown that the expressive power of the fusion calculus is still retained by the sub-calculus without action prefix and summation – the *solos calculus*. In addition, the expressive power does not change by restriction to dyadic solos, while expressiveness strictly decreases in the monadic calculus without match. The results of this paper are collected in the table below.

Encodings	Results
fusion calculus $\downarrow$ solos calculus	$[\![\cdot]\!]$ encodes prefixes with matches and is adequate - wrt $\sim$ (Theorem 12) and - wrt $\approx$ (Theorem 13) $[\![(\cdot)]\!]$ encodes prefixes without matches and is adequate - wrt $\approx$ (Theorem 17)
polyadic solos calculus $\downarrow$ dyadic solos calculus	$[\![\![\cdot]\!]\!]$ is adequate in the well-sorted calculus - wrt $\approx$ (Theorem 26)
dyadic solos calculus $\not\downarrow$ monadic solos calculus	No encoding is possible in the well-sorted calculus without match (Theorem 28)

Some open questions still remain. For instance we conjecture that the encoding of the polyadic calculus into the dyadic one is also complete. Merro has studied similar encodings for the asynchronous π-calculus (Merro, 2000). However his proof technique uses labelled bisimulation and the correspondence of this equivalence with weak barbed congruence, and this correspondence has still not been proved in the fusion calculus. Furthermore, we conjecture that the unsorted monadic solos calculus has strictly lower expressive power than the fusion calculus. Finally, we also leave open the question whether there is a compositional encoding (in the sense of (Palamidessi, 1997)) of the fusion calculus into the solos calculus without the match operator.

The simplicity of the solos calculus and the results in this paper suggest its use as a test suite for experimenting with implementations, theories and models of mobile programming languages. In this respect, the paper (Laneve et al., 2001), describing a graphical formalism for solos, is a first contribution towards an implementation. Concerning distributed implementations, a promising formalism seems to be the *solos calculus with explicit fusions*, in the style of (Gardner and Wischik, 2000).

References

Boreale, M. and Sangiorgi, D. (1998). A fully abstract semantics for causality in the pi-calculus. *Acta Informatica*, 35(5):353–400. An extended abstract appeared in the *Proceedings of STACS '95*, volume 900 of *LNCS*. A long version appeared as report ECS–LFCS–94–297, University of Edinburgh.

Boudol, G. (1992). Asynchrony and the π-calculus (note). Rapport de Recherche 1702, INRIA Sophia-Antipolis.

Cleaveland, R., editor (1992). *Proc. of CONCUR '92*, volume 630 of *LNCS*. Springer.

Degano, P. and Priami, C. (1995). Causality for mobile processes. In Fülöp, Z. and Gécseg, F., editors, *Proceedings of ICALP '95*, volume 944 of *LNCS*, pages 660–671. Springer.

Fournet, C. and Gonthier, G. (1998). A hierarchy of equivalences for asynchronous calculi. In (Larsen et al., 1998), pages 844–855.

Fu, Y. (1997). A proof-theoretical approach to communication. In Degano, P., Gorrieri, R., and Marchetti-Spaccamela, A., editors, *Proceedings of ICALP '97*, volume 1256 of *LNCS*, pages 325–335. Springer.

Gardner, P. and Wischik, L. (2000). Explicit fusions. In *Proc. of Mathematical Foundations of Computer Science (MFCS)*, volume 1893 of *LNCS*. Springer.

Honda, K. (2000). Elementary structures for process theory (1): Sets with renaming. *Mathematical Structures in Computer Science*, 10(5):617–663.

Honda, K. and Tokoro, M. (1991). An object calculus for asynchronous communication. In America, P., editor, *Proceedings of ECOOP '91*, volume 512 of *LNCS*, pages 133–147. Springer.

Laneve, C., Parrow, J., and Victor, B. (2001). Solo diagrams. In Kobayashi, N. and Pierce, B. C., editors, *Proc. of TACS 2001*, volume 2215 of *LNCS*, pages 127–144. Springer.

Larsen, K. G., Skyum, S., and Winskel, G., editors (1998). *25th Colloquium on Automata, Languages and Programming (ICALP) (Aalborg, Denmark)*, volume 1443 of *LNCS*. Springer.

Merro, M. (2000). Locality and polyadicity in asynchronous name-passing calculi. In Tiuryn, J., editor, *Proceedings of the 2 Int. Conf. on Foundations of Software Science and Computation Structures (FoSSaCS 2000)*, volume 1784 of *LNCS*, pages 238–251. Springer.

Milner, R. (1993). The polyadic π-calculus: A tutorial. In Bauer, F. L., Brauer, W., and Schwichtenberg, H., editors, *Logic and Algebra of Specification*, volume 94 of *Series F*. NATO ASI, Springer. Available as Technical Report ECS-LFCS-91-180, University of Edinburgh, October 1991.

Milner, R., Parrow, J., and Walker, D. (1992). A calculus of mobile processes, part I/II. *Journal of Information and Computation*, 100:1–77.

Milner, R. and Sangiorgi, D. (1992a). Barbed bisimulation. In Kuich, W., editor, *Proc. of ICALP '92*, volume 623 of *LNCS*, pages 685–695. Springer.

Milner, R. and Sangiorgi, D. (1992b). The problem of "weak bisimulation up-to". In (Cleaveland, 1992).

Nestmann, U. (1997). What is a 'good' encoding of guarded choice? In Palamidessi, C. and Parrow, J., editors, *Proc. of EXPRESS '97*, volume 7 of *ENTCS*. Elsevier Science Publishers. Revised version accepted (1998) for *Journal of Information and Computation.*

Nestmann, U. and Pierce, B. C. (1996). Decoding choice encodings. In Montanari, U. and Sassone, V., editors, *Proc. of CONCUR '96*, volume 1119 of *LNCS*, pages 179–194. Springer.

Palamidessi, C. (1997). Comparing the expressive power of the synchronous and the asynchronous π-calculus. In *Proc. of POPL '97*, pages 256–265. ACM.

Parrow, J. (1995). Interaction diagrams. *Nordic Journal of Computing*, 2:407–443.

Parrow, J. (2000). Trios in concert. In Plotkin, G., Stirling, C., and Tofte, M., editors, *Proof, Language and Interaction: Essays in Honour of Robin Milner*, Foundations of Computing, pages 621–637. MIT Press.

Parrow, J. and Sjödin, P. (1992). Multiway synchronization verified with coupled bisimulation. In (Cleaveland, 1992), pages 518–533.

Parrow, J. and Victor, B. (1998a). The fusion calculus: Expressiveness and symmetry in mobile processes. In *Proc. of LICS '98.* IEEE, Computer Society Press.

Parrow, J. and Victor, B. (1998b). The tau-laws of fusion. In Sangiorgi, D. and de Simone, R., editors, *Proc. of CONCUR '98*, volume 1466 of *LNCS*, pages 99–114. Springer.

Sangiorgi, D. (1993). *Expressing Mobility in Process Algebras: First-Order and Higher-Order Paradigms.* PhD thesis, LFCS, University of Edinburgh. CST-99-93 (also published as ECS-LFCS-93-266).

Sangiorgi, D. (1996). A theory of bisimulation for the π-calculus. *Acta Informatica*, 33:69–97.

Victor, B. and Parrow, J. (1998). Concurrent constraints in the fusion calculus. In (Larsen et al., 1998), pages 455–469.

Yoshida, N. (1998). Minimality and separation results on asynchronous mobile processes: Representability theorem by concurrent combinators. Technical Report 5/98, School of Cognitive and Computing Sciences, University of Sussex, UK. An extended abstract appeared in *Proc. of CONCUR'98*, LNCS 1466.

Appendix A. Proofs of Section 3

We begin by defining a standard form of derivatives of $[\![(P)]\!]$.

Definition 29. An agent Q *belongs to the* $[\![P]\!]_v$*-family* ($v \notin \mathrm{fv}(P)$) if

1 $P \equiv (\tilde{z})\big(\prod_{i\in I} u_i\,\tilde{x}_i.R_i \mid \prod_{j\in J} \overline{u_j}\,\tilde{x}_j.R_j \mid R\big)$, and

2 $Q \equiv (\tilde{z})\Big(\prod_{i\in I}(v_i)(u_i\,\tilde{x}_i v v_i \mid [\![R_i]\!]_{v_i}) \;\Big|\; \prod_{j\in J}(v_j)(\overline{u_j}\,\tilde{x}_j v_j v \mid [\![R_j]\!]_{v_j}) \;\Big|\; [\![R]\!]_v\Big)$

where, for every $k \in I \cup J$,

1 v, v_k are different,

2 $\{v, v_k\} \cap (\mathrm{fv}(R_k) \cup \{u_k, \widetilde{x_k}\}) = \emptyset$.

The next proposition guarantees that derivatives of $[\![(P)]\!]$ are actually $[\![(\cdot)]\!]$-reducts.

Proposition 30. Let $Q \in [\![P]\!]_v$-family. If $P \longrightarrow P'$ then $(v)(U_v \mid Q) \longrightarrow^{+}\equiv (v)(U_v \mid Q')$ and $Q' \in [\![P']\!]_v$-family.

Proof. Since $P \equiv (\widetilde{z})\big(\prod_{i\in I} u_i\,\widetilde{x_i}.R_i \mid \prod_{j\in J} \overline{u_j}\,\widetilde{x_j}.R_j \mid R\big)$, the transition $P \longrightarrow P'$ is due to two sub-processes which fall in one of the three cases below:

1 the sub-processes are $u_h\,\widetilde{x_h}.R_h$ and $\overline{u_k}\,\widetilde{x_k}.R_k$, for some h,k;
2 exactly one of the two subprocesses is $u_h\,\widetilde{x_h}.R_h$ or $\overline{u_h}\,\widetilde{x_h}.R_h$, for some $h \in I \cup J$;
3 the two subprocesses belong to R.

We demonstrate the three subcases separately.

1 The transition $P \longrightarrow P'$ may be rewritten as follows:

$$\begin{array}{rcl} P & \equiv & (\widetilde{z})(u_h\,\widetilde{x_h}.R_h \mid \overline{u_k}\,\widetilde{x_k}.R_k \mid \prod_{i\in I\setminus h} u_i\,\widetilde{x_i}.R_i \mid \prod_{j\in J\setminus k} \overline{u_j}\,\widetilde{x_j}.R_j \mid R) \\ & \longrightarrow & (\widetilde{z'})(R_h \mid R_k \mid \prod_{i\in I\setminus h} u_i\,\widetilde{x_i}.R_i \mid \prod_{j\in J\setminus k} \overline{u_j}\,\widetilde{x_j}.R_j \mid R)\sigma \\ & \equiv & (\widetilde{z'})(\prod_{i\in I\setminus h}(u_i\,\widetilde{x_i}.R_i)\sigma \mid \prod_{j\in J\setminus k}(\overline{u_j}\,\widetilde{x_j}.R_j)\sigma \mid R\sigma \mid R_h\sigma \mid R_k\sigma) \\ & \equiv & P' \end{array}$$

where $\mathrm{dom}(\sigma) \subseteq (\widetilde{x_h} \cup \widetilde{x_k})$ and $\mathrm{ran}(\sigma) = \widetilde{z} \setminus \widetilde{z'}$. On the other side we have:

$$\begin{array}{rcl} (v)(U_v \mid Q) & \equiv & (v)(U_v \mid (\widetilde{z})\Big((v_h)(u_h\,\widetilde{x_h}vv_h \mid [\![R_h]\!]_{v_h}) \;\Big|\; (v_k)(\overline{u_k}\,\widetilde{x_k}v_kv \mid [\![R_k]\!]_{v_k}) \\ & & \quad \Big|\; \prod_{i\in I\setminus h}(v_i)(u_i\,\widetilde{x_i}vv_i \mid [\![R_i]\!]_{v_i}) \\ & & \quad \Big|\; \prod_{j\in J\setminus k}(v_j)(\overline{u_j}\,\widetilde{x_j}v_jv \mid [\![R_j]\!]_{v_j}) \;\Big|\; [\![R]\!]_v\Big)) \\ & \longrightarrow & (v)(U_v \mid (\widetilde{z'})\Big([\![R_h]\!]_{v_h} \;\Big|\; [\![R_k]\!]_{v_k} \\ & & \quad \Big|\; \prod_{i\in I\setminus h}(v_i)(u_i\,\widetilde{x_i}vv_i \mid [\![R_i]\!]_{v_i}) \\ & & \quad \Big|\; \prod_{j\in J\setminus k}(v_j)(\overline{u_j}\,\widetilde{x_j}v_jv \mid [\![R_j]\!]_{v_j}) \\ & & \quad \Big|\; [\![R]\!]_v\Big)\sigma\{v/v_h\}\{v/v_k\}) \\ & \equiv & (v)(U_v \mid (\widetilde{z'})\Big(\prod_{i\in I\setminus h}(v_i)(u_i\,\widetilde{x_i}vv_i \mid [\![R_i]\!]_{v_i}) \\ & & \quad \Big|\; \prod_{j\in J\setminus k}(v_j)(\overline{u_j}\,\widetilde{x_j}v_jv \mid [\![R_j]\!]_{v_j}) \\ & & \quad \Big|\; [\![R]\!]_v \mid [\![R_h]\!]_v \mid [\![R_k]\!]_v\Big)\sigma) \\ & = & (v)(U_v \mid (\widetilde{z'})\Big(\prod_{i\in I\setminus h}(v_i)(u_i\,\widetilde{x_i}vv_i \mid [\![R_i]\!]_{v_i})\sigma \\ & & \quad \Big|\; \prod_{j\in J\setminus k}(v_j)(\overline{u_j}\,\widetilde{x_j}v_jv \mid [\![R_j]\!]_{v_j})\sigma \\ & & \quad \Big|\; [\![R\sigma]\!]_v \mid [\![R_h\sigma]\!]_v \mid [\![R_k\sigma]\!]_v\Big)) \\ & \equiv & (v)(U_v \mid Q') \end{array}$$

where the last-but-one step is due to Proposition 14. It is immediate that $Q' \in [\![P']\!]_v$-family.

2 We assume $h \in I$; the case $h \in J$ being similar. Let $R \equiv (\widetilde{z'})(\overline{u}\,\widetilde{x}.R' \mid R'')$; the transition $P \longrightarrow P'$ may be rewritten as follows:

$$\begin{array}{rcl} P & \equiv & (\widetilde{z}\widetilde{z'})(u_h\,\widetilde{x_h}.R_h \mid \overline{u}\,\widetilde{x}.R' \mid \prod_{i\in I\setminus h} u_i\,\widetilde{x_i}.R_i \mid \prod_{j\in J} \overline{u_j}\,\widetilde{x_j}.R_j \mid R'') \\ & \longrightarrow & (\widetilde{z''})(R_h \mid R' \mid \prod_{i\in I\setminus h} u_i\,\widetilde{x_i}.R_i \mid \prod_{j\in J} \overline{u_j}\,\widetilde{x_j}.R_j \mid R'')\sigma \\ & \equiv & (\widetilde{z''})(\prod_{i\in I\setminus h}(u_i\,\widetilde{x_i}.R_i)\sigma \mid \prod_{j\in J}(\overline{u_j}\,\widetilde{x_j}.R_j)\sigma \mid R\sigma \mid R'\sigma \mid R''\sigma) \\ & \equiv & P' \end{array}$$

where $\mathrm{dom}(\sigma) \subseteq (\widetilde{x_h} \cup \widetilde{x})$ and $\mathrm{ran}(\sigma) = (\widetilde{z} \cup \widetilde{z'}) \setminus \widetilde{z''}$. On the other side we have:

$$\begin{array}{rcl}
(v)(U_v \mid Q) & \equiv & (v)(U_v \mid (\widetilde{z}\widetilde{z'})\Big((v_h)(u_h\, \widetilde{x_h} v v_h \mid \llbracket R_h \rrbracket_{v_h}) \;\Big|\; \llbracket \overline{u}\, \widetilde{x}.R' \rrbracket_v \\
& & \quad \Big|\; \prod_{i \in I \setminus h} (v_i)(u_i\, \widetilde{x}_i v v_i \mid \llbracket R_i \rrbracket_{v_i}) \\
& & \quad \Big|\; \prod_{j \in J} (v_j)(\overline{u_j}\, \widetilde{x}_j v_j v \mid \llbracket R_j \rrbracket_{v_j}) \;\Big|\; \llbracket R'' \rrbracket_v\Big)) \\
& \longrightarrow & (v)(U_v \mid (\widetilde{z}\widetilde{z'})\Big((v_h)(u_h\, \widetilde{x_h} v v_h \mid \llbracket R_h \rrbracket_{v_h}) \;\Big|\; (v')(\overline{u}\, \widetilde{x} v' v \mid \llbracket R' \rrbracket_{v'}) \\
& & \quad \Big|\; \prod_{i \in I \setminus h} (v_i)(u_i\, \widetilde{x}_i v v_i \mid \llbracket R_i \rrbracket_{v_i}) \\
& & \quad \Big|\; \prod_{j \in J} (v_j)(\overline{u_j}\, \widetilde{x}_j v_j v \mid \llbracket R_j \rrbracket_{v_j}) \\
& & \quad \Big|\; \llbracket R'' \rrbracket_v\Big)) \\
& \longrightarrow & (v)(U_v \mid (\widetilde{z''})\Big(\llbracket R_h \rrbracket_{v_h} \mid \llbracket R' \rrbracket_{v'} \\
& & \quad \prod_{i \in I \setminus h} (v_i)(u_i\, \widetilde{x}_i v v_i \mid \llbracket R_i \rrbracket_{v_i}) \\
& & \quad \Big|\; \prod_{j \in J} (v_j)(\overline{u_j}\, \widetilde{x}_j v_j v \mid \llbracket R_j \rrbracket_{v_j}) \\
& & \quad \Big|\; \llbracket R'' \rrbracket_v\Big)\sigma\{v/v_i\}\{v/v'\}) \\
& \equiv & (v)(U_v \mid (\widetilde{z'})\Big(\prod_{i \in I \setminus h} (v_i)(u_i\, \widetilde{x}_i v v_i \mid \llbracket R_i \rrbracket_{v_i}) \\
& & \quad \Big|\; \prod_{j \in J} (v_j)(\overline{u_j}\, \widetilde{x}_j v_j v \mid \llbracket R_j \rrbracket_{v_j}) \\
& & \quad \Big|\; \llbracket R'' \rrbracket_v \mid \llbracket R_h \rrbracket_v \mid \llbracket R' \rrbracket_v\Big)\sigma) \\
& = & (v)(U_v \mid (\widetilde{z'})\Big(\prod_{i \in I \setminus h} (v_i)(u_i\, \widetilde{x}_i v v_i \mid \llbracket R_i \rrbracket_{v_i})\sigma \\
& & \quad \Big|\; \prod_{j \in J} (v_j)(\overline{u_j}\, \widetilde{x}_j v_j v \mid \llbracket R_j \rrbracket_{v_j})\sigma \\
& & \quad \Big|\; \llbracket R''\sigma \rrbracket_v \mid \llbracket R_h\sigma \rrbracket_v \mid \llbracket R'\sigma \rrbracket_v\Big)) \\
& \equiv & (v)(U_v \mid Q')
\end{array}$$

where, as before, the last-but-one step is due to Proposition 14. It is immediate that $Q' \in \llbracket P' \rrbracket_v$-family.

3 Similar to the previous cases.

$\square$

Proposition 31. Let $Q \in \llbracket P \rrbracket_v$-family. If $(v)(U_v \mid Q) \longrightarrow Q'$ then $Q' \equiv (v)(U_v \mid Q'')$ and $P \stackrel{\wedge}{\longrightarrow} P'$, such that $Q'' \in \llbracket P' \rrbracket_v$-family.

Proof. Since $Q \equiv (\widetilde{z})\Big(\prod_{i \in I} (v_i)(u_i\, \widetilde{x}_i v v_i \mid \llbracket R_i \rrbracket_{v_i}) \;\Big|\; \prod_{j \in J} (v_j)(\overline{u_j}\, \widetilde{x}_j v_j v \mid \llbracket R_j \rrbracket_{v_j}) \;\Big|\; \llbracket R \rrbracket_v\Big)$, there are two cases, according to

1 the reduction is due to a pair $u_h\, \widetilde{x_h} v v_h$, $\overline{u_k}\, \widetilde{x_k} v_k v$, for some $h \in I$, $k \in J$;
2 the reduction is due to the agent U_v and a sub-agent of $\llbracket R \rrbracket_v$.

The proof is similar to Proposition 30. We notice that, in the second case, P does not perform any reduction. $\square$

Proposition 32. Let $Q \in \llbracket P \rrbracket_v$-family. Then

1 if $P \downarrow x$, for some x, then $(v)(U_v \mid Q) \Downarrow x$;
2 if $(v)(U_v \mid Q) \downarrow x$, for some x, the $P \downarrow x$.

Proof. Both cases are straightforward consequences the of definition of $\llbracket P\rrbracket_v$ and $\llbracket P\rrbracket_v$-family. $\square$

We are now in a position to demonstrate Lemma 15. For legibility, we restate the lemma.

Lemma 15. For P an agent of f_{pre}, $P \lesssim \llparenthesis P\rrparenthesis$.

Proof. By Propositions 30, 31, and 32, the relation

$$\{(P,Q) \mid Q \equiv (v)(U_v \mid Q') \text{ and } Q' \in \llbracket P\rrbracket_v\text{-family and } v \notin \text{fv}(P)\}$$

is a weak barbed expansion containing the pairs $(P, \llparenthesis P\rrparenthesis)$. $\square$

Next, we consider the solos calculus where replication is replaced by guarded recursion, as discussed in Section 4. The following lemma relates the divergence of an agent with guarded recursion to the divergence of its encoding $\llparenthesis\cdot\rrparenthesis$.

Lemma 33. Let P be an agent of f_{pre} with guarded recursion. For every $Q \in \llbracket P\rrbracket_v$-family, if $(v)(U_v \mid Q)$ diverges then P diverges, too.

Proof. We notice that the agent $(v)(U_v \mid \llbracket Q\rrbracket_v)$ performs a finite number of different reductions involving the catalyst agent U_v because recursion is guarded. Therefore, if $(v)(U_v \mid Q)$ diverges, there is a finite prefix of the divergent derivation of shape $(v)(U_v \mid Q) \longrightarrow^*\equiv (v)(U_v \mid Q') \longrightarrow\equiv (v)(U_v \mid Q'')$, where $Q, Q' \in \llbracket P\rrbracket_v$-family and the last reduction does not involve the catalyst agent. Then, by Proposition 31, there is P' such that $P \longrightarrow P'$, and $Q'' \in \llbracket P'\rrbracket_v$-family. We conclude by observing that $(v)(U_v \mid Q'')$ also diverges. $\square$

Appendix B. Example reduction of choice encoding

In this appendix we illustrate the second encoding of the choice operator (page 20) by an example.

$$
\begin{array}{lll}
\multicolumn{3}{l}{\llparenthesis\ (x_1x_2)\big((u\,x_1\,.\,P_1 + u\,x_2\,.\,P_2) \mid (\overline{u}\,y_1\,.\,Q_1 + \overline{u}\,y_2\,.\,Q_2)\big)\ \rrparenthesis} \\
\overset{def}{=} & (vx_1x_2)\big(U_v \mid (v')(& [v \neq v'](w)(w\,x_1vv' \mid \llbracket P_1\rrbracket_{v'} \mid (y)(\overline{v}\,uwy \mid U_y)) \\
& & \mid [v \neq v'](w)(w\,x_2vv' \mid \llbracket P_2\rrbracket_{v'} \mid (y)(\overline{v}\,uwy \mid U_y))) \\
& \quad \mid (v')(& [v \neq v'](w)(\overline{w}\,y_1v'v \mid \llbracket Q_1\rrbracket_{v'} \mid (y)(\overline{v}\,uwy \mid U_y)) \\
& & \mid [v \neq v'](w)(\overline{w}\,y_2v'v \mid \llbracket Q_2\rrbracket_{v'} \mid (y)(\overline{v}\,uwy \mid U_y)))\big) \\
\longrightarrow & (vx_1x_2)\big(U_v \mid (v')(& [v \neq v'](u\,x_1vv' \mid \llbracket P_1\rrbracket_{v'}) \\
& & \mid [v \neq v'](w)(w\,x_2vv' \mid \llbracket P_2\rrbracket_{v'} \mid (y)(\overline{v}\,uwy \mid U_y))) \\
& \quad \mid (v')(& [v \neq v'](w)(\overline{w}\,y_1v'v \mid \llbracket Q_1\rrbracket_{v'} \mid (y)(\overline{v}\,uwy \mid U_y)) \\
& & \mid [v \neq v'](w)(\overline{w}\,y_2v'v \mid \llbracket Q_2\rrbracket_{v'} \mid (y)(\overline{v}\,uwy \mid U_y)))\big) \\
\longrightarrow & (vx_1x_2)\big(U_v \mid (v')(& [v \neq v'](u\,x_1vv' \mid \llbracket P_1\rrbracket_{v'}) \\
& & \mid [v \neq v'](w)(w\,x_2vv' \mid \llbracket P_2\rrbracket_{v'} \mid (y)(\overline{v}\,uwy \mid U_y))) \\
& \quad \mid (v')(& [v \neq v'](\overline{u}\,y_1v'v \mid \llbracket Q_1\rrbracket_{v'}) \\
& & \mid [v \neq v'](w)(\overline{w}\,y_2v'v \mid \llbracket Q_2\rrbracket_{v'} \mid (y)(\overline{v}\,uwy \mid U_y)))\big)
\end{array}
$$

In these initial reductions, for legibility we always move the new catalyst created by $U_y\{v/y\}$ to top level. Already after these reductions, an interaction between two of the summands can take place. To make it more interesting, we let all summands interact with the catalyst:

$$\begin{aligned} \longrightarrow\longrightarrow \quad (vx_1x_2)\big(U_v \mid (v')(\; & [v \neq v'](u\,x_1vv' \mid [\![P_1]\!]_{v'}) \\ & \mid [v \neq v'](u\,x_2vv' \mid [\![P_2]\!]_{v'})) \\ \mid (v')(\; & [v \neq v'](\overline{u}\,y_1v'v \mid [\![Q_1]\!]_{v'}) \\ & \mid [v \neq v'](\overline{u}\,y_2v'v \mid [\![Q_2]\!]_{v'}))\big) \end{aligned} \tag{5}$$

Here, any of the two summands of each parallel component of the original agent can interact with any of the summands of the other component. (Note that all mismatch conditions are false, since v' is fresh in both components.) We arbitrarily choose the first summand of each:

$$\begin{aligned} \longrightarrow \quad (vx_2)\big(U_v \mid (\; & [\![P_1]\!]_v \\ & \mid [v \neq v](u\,x_2vv \mid [\![P_2]\!]_v)) \\ \mid (\; & [\![Q_1]\!]_v \\ & \mid [v \neq v](\overline{u}\,y_2vv \mid [\![Q_2]\!]_v))\big)\{y_1/x_1\} \end{aligned}$$

Since $[v \neq v]P \sim \mathbf{0}$ for all v and P we can remove the components corresponding to the discarded summands, and then we can also garbage collect the binding of x_2:

$$\begin{aligned} \sim \quad & (v)\big(U_v \mid [\![P_1]\!]_v \mid [\![Q_1]\!]_v\big)\{y_1/x_1\} \\ \overset{def}{=} \quad & (\![P_1 \mid Q_1]\!)\{y_1/x_1\} \end{aligned}$$

This is exactly the result we were expecting.

Note that this encoding does not work for mixed guarded choice, where the prefixes in a summation can be both inputs and outputs. This is easy to see from the agent in (5) above, where, if the original summation terms were of mixed polarity, their encoding would at this point allow interaction between the terms of each choice.