Submitted for publication

Old Names for Nu

Lucian Wischik, University of Bologna. lu@wischik.com

January 2004

Abstract. We describe a broadcast implementation of the pi calculus and prove it correct.
Surprisingly, the biggest proof challenge was the *nu* command, also called restriction. In typical implementations of process calculi (Pict, chemical abstract machine) the command allocates a new communication channel. In a broadcast implementation it need only generate a fresh name. Both techniques amount to discarding top-level restrictions and dispensing with scope extrusion; they yield a very simple multiset or chemical solution semantics. In contrast, the pi calculus and the join calculus retain restriction and scope extrusion. We prove that, for pi, the two treatments of *nu* yield the same behavioural semantics (are fully abstract).

1 Introduction

This paper leads to a broadcast implementation of the pi calculus, and a proof of its correctness. Before reaching the implementation (Section 3), we make a considerable detour to consider the pi calculus' *nu* command. This is necessary because there is a gap between its implementation and its theory, which has not previously been settled formally. The following is an example execution of a pi program. It is taken from Pict [18], but could equally be taken from CML [1, 20] or Klaim [7] or Facile [16] or the Chemical Abstract Machine [2]. States are written for example as $x, y : P|Q|R$ – this indicates that two channels x and y exist, and that three threads P, Q and R are running in parallel.

$$
\begin{array}{lll}
 & x, u:\ \nu x.(\overline{u}\,x) \mid u(y).(\overline{u}\,x|P) & \\
\rightarrow & x', x, u:\ \overline{u}\,x' \mid u(y).(\overline{u}\,x|P) & \textit{create fresh channel } x'\textit{; apply } \{x'/x\} \\
\rightarrow & x', x, u:\ \overline{x}\,'x \mid P\{x'/y\} & \textit{rendezvous on channel } u
\end{array}
$$

Consider the first step of the execution:

$$\nu x.P \mid R \ \rightarrow\ P\{x'/x\} \mid R, \quad x' \text{ fresh} \qquad \textit{new}$$

This rule interprets the *nu* command by creating a new channel – equivalently, generating a fresh name. (The ν binds more tightly than | does). We might call this a *global semantics*, since the rule applies to the entire system and generates a globally fresh name.

However, for pi the global semantics is never actually used (except by the implementors of Pict...). Instead, in the traditional pi calculus semantics, the execution trace is as follows:

$$\begin{array}{lll} u(y).(\overline{x}\,y|P) \mid \nu x.(\overline{u}\,x) & & \\ \quad \equiv_\pi u(y).(\overline{x}\,y|P) \mid \nu x'.(\overline{u}\,x') & & \textit{alpha-rename } x \textit{ to } x' \\ \quad \equiv_\pi \nu x'.(u(y).(\overline{x}\,y|P) \mid \overline{u}\,x') & & \textit{extrude scope of } x' \\ \quad \rightarrow_\pi \nu x'.(\overline{x}\,x' \mid P\{x'/y\}) & & \textit{rendezvous on channel } u \end{array}$$

Here, *nu* is not executed. It is retained, and allows for alpha-renaming. The powerful technique of *scope extrusion* allows its scope to expand arbitrarily. Scope extrusion is part of the reversible 'structural congruence' $\equiv_\pi$.

The global semantics has some attractive features compared to this traditional pi: the global semantics is how the implementations actually work; in the absence of scope extrusion it allows de-Bruijn notation for the pi calculus; and it allows an elegant *multiset* model where a term is just a multiset of elements $\overline{u}\,\widetilde{x}.P$, $u(\widetilde{x}).P$, $\nu x.P$. (This is like a chemical soup [2] but where all terms are already fully heated and no cooling is needed). One naturally wonders whether the global and traditional semantics are equivalent. 'Certainly,' might say Fournet and Gonthier [9], who use a global semantics in the first half of their paper on the join calculus and switch to the traditional semantics for the second half with only the briefest of mentions. 'Sort of' might say Turner [22], who makes proofs that at least a few operational semantics are preserved. 'Clearly not' remarked some colleagues, on the grounds that the rule *new* above allows an observation on x'. A chief contribution of this paper (Section 2) is to confirm formally that the two semantics are indeed equivalent – *fully abstract* up to barbed congruence. This then makes it easier to prove correct an implementation.

The result actually has some complexity. For instance, the traditional pi semantics has the context-closure axiom

$$\frac{P \;\rightarrow_\pi\; P'}{P|Q \;\rightarrow_\pi\; P'|Q}$$

But the same inference cannot be made in the global semantics, as shown by the counter-example

$$\frac{\nu x.\overline{u}\,x \;\rightarrow\; \overline{u}\,x'}{\nu x.\overline{u}\,x \mid x'().P \;\not\rightarrow\; \overline{u}\,x' \mid x'().P}$$

Intuitively it *should* be possible to somehow make the same inference, by indicating that the name x' should be fresh not just with respect to where it was created but also with respect to possible contexts.

Several related *fresh* approaches have been proposed; see Table 1. They keep freshness information, but their primary goal is a finite form of labelled transition system – where for instance the emission of a bound name $\nu x.\overline{u}\,x.P \xrightarrow{\overline{u}\,(x')} P\{x'/x\}$ can be treated just through a single representative x' rather than all x'. But unlike the intuition in the previous paragraph, and unlike the global semantics, these fresh approaches only use their freshness annotations for input and output

$\nu x.P\|R \rightarrow_i P\{^{x'}/_x\}\|R$	*Global* semantics, where the execution of the command generates a fresh name $x' \notin \mathrm{fn}(\nu x.P\|R)$. This is a global rule, which includes the rest of the system R. The notation $x, y, z : P$ is sometimes used to keep a list of all names that exist. Used for instance in Pict [18], CML [1, 20], Facile [16] and the Chemical Abstract Machine [2].
$(\!(\widetilde{x})\!)\, P$	Again a global semantics, but where the set $\widetilde{x}$ is the list of all names that have so far been created during execution. Used in [14] and [15], and as a proof technique in this paper (Definition2).
$\mathsf{N}x : P \xrightarrow{\mu}_n P'$	
$(\theta x)P \xrightarrow{\mu}_\theta (\theta x)P'$	*Fresh semantics* The Gabbay-Pitts operator N [13, 11, 12] and the abstraction θ [3] indicate that the name x is locally fresh in this transition. But when transition is composed with others, the quantifier ensures that it keeps fresh. The final case [17] uses a permutation ρ to 'make space' for new name that has been emitted.
$\nu x.(\overline{u}\, x\|P) \xrightarrow{\overline{u}\,(\text{-})}_{hd} P\rho$	
$P\|\nu x.Q \equiv_\pi \nu x.(P\|Q)$	The traditional pi calculus has scope extrusion and alpha-renaming as part of its *structural congruence* $\equiv_\pi$, rather than as part of reaction. This gives a purely local semantics, as an alternative to global notions of freshness.

Table 1: Nu approaches

transitions rather than rendezvous. For rendezvous, they revert to restrictions as in the normal pi calculus, with the concomitant scope extrusion. (If [17] were adapted to allow τ to be deduced from input and output transitions as is normal, then it would avoid restrictions like the global semantics; but this adaption seems impossible).

Our proof uses a different freshness approach. We give an intermediate calculus which has exactly the same operation as the global semantics, but which keeps a top-level record $(\!(\widetilde{x})\!)$ of which names have been generated so far in the entire history of the computation: the *'old names'*. With the history, all axioms in the pi calculus can be expressed as indirect inferences in our fresh calculus. The results then lift over to the global semantics – even though the indirect inference lemmas do not.

We remark that the operations in the other fresh approaches do not coincide with the global semantics, and so their results cannot be lifted into the global semantics. We also remark that they use labelled transitions; we use barbed bisimulation instead, which is easier to apply to diverse implementation models. For instance, in Section 3 we give a broadcast implementation and prove it correct. Broadcast uses entirely different labelled transitions, but its barbs are just the same as those of the pi calculus (Definition 5).

In normal barbed bisimulation, all free names are considered observable ('to have barbs'). This works because the set of free names does not normally increase. By contrast, in the global pi calculus, the set of free names can increase and decrease. For this reason we parameterise observation just on the set of names that were initially free – so, on page 1, the x' is not observable. Similarly the reflexive chemical machine [10] is parameterised for observation on just a single distinguished name *test*; it turns out that this is sufficient. Sewell [21] also uses a finite parameterisation $\{keyboard, mouse, screen\}$, although his set of free names never decreases and so the parameterisation is not actually necessary.

Sewell's work, like that of Turner [22], is concerned with proving the correctness of the Pict implementation [18] of the pi calculus. Pict is a deterministic single-machine implementation, isolated except through interaction with the user. Because it is deterministic, it is necessarily not bisimilar with the (non-deterministic) pi calculus. Turner instead just proves that the machine is 'valid' (never makes reactions disallowed by the pi calculus) and 'non-blocking' (if the calculus admits some reaction, the machine also admits some reaction). Sewell extends the results to a may-testing preorder. These limited results are appropriate for a single isolated machine such as Pict, as Sewell argues. But they are no longer appropriate as we move to distributed implementations and use the pi calculus to program mobile devices or write web services. A distributed implementation really is non-deterministic, due to network vagaries, and so a full bisimulation result is possible. Moreover, a web service or other component might be placed in arbitrary contexts, not just run in isolation, so bisimulation congruence rather than just bisimulation is important. For example, in the sequel [23] we use distributed atomic transactions as our implementation. Non-blocking on its own is not enough to prove the *non-triviality* required of a transaction, but bisimulation is. Bisimulation on its own is not enough to prove compositionality of transactions, but bisimulation congruence is.

By using fresh names and eliminating scope extrusion, we obtain an appealing multiset model for the pi calculus (Table 2). A different form of multiset was previously given by Engelfriet [8]. He retains restrictions, so that his barbs match those of the pi calculus; and he uses scope extrusion to assume a normal form in which all restrictions are extruded to the top level. Actually it is not possible to extrude restrictions through a replication, so Engelfriet first replaces any replicated term $!P$ with an infinite number of copies of P (each of which do then allow restrictions to be pushed out). The significance of our current work is to show that *top-level restrictions are unnecessary*, and so the multiset can remain finite.

Our idea is that, in general, traditional co-inductive techniques will be used at compile-time to prove that an original program is bisimilar to some optimised version of itself; then use Theorem 1 to prove that the two are bisimilar also in a global semantics; then use some implementation-correctness result (such as Proposition 6 for our broadcast implementation) to prove that two are also bisimilar in some particular implementation. The reason for using global semantics is that it simplifies the implementation-correctness results.

Global pi calculus has terms P and contexts C as follows. We identify terms $P \equiv Q$ up to commutativity and associativity of $|$, with $\mathbf{0}$ as identity.

$$\begin{array}{lcl} P & ::= & \overline{u}\,\widetilde{x}.P \;\mid\; u(\widetilde{x}).P \;\mid\; !u(\widetilde{x}).P \;\mid\; \nu x.P \;\mid\; P|P \;\mid\; \mathbf{0} \\ C & ::= & \overline{u}\,\widetilde{x}.C \;\mid\; u(\widetilde{x}).C \;\mid\; !u(\widetilde{x}).C \;\mid\; \nu x.C \;\mid\; C|P \;\mid\; P|C \;\mid\; \_ \end{array}$$

Reaction relation is

$$(\nu x.P) \mid R \;\rightarrow\; P\{x'/x\} \mid R \qquad x' \notin \mathrm{fn}(\nu x.P|R) \qquad \text{(new)}$$

$$\overline{u}\,\widetilde{y}.P \mid u(\widetilde{x}).Q \mid R \;\rightarrow\; P \mid Q\{\widetilde{y}/\widetilde{x}\} \mid R \qquad \text{(react)}$$

$$\overline{u}\,\widetilde{y}.P \mid !u(\widetilde{x}).Q \mid R \;\rightarrow\; P \mid Q\{\widetilde{y}/\widetilde{x}\} \mid !u(\widetilde{x}).Q \mid R \qquad \text{(replication)}$$

Barbs (observations) are as follows: x is observable in P, written $P \downarrow x$, when

$$\overline{u}\,\widetilde{x}.P \downarrow u \qquad u(\widetilde{x}).P \downarrow u \qquad !u(\widetilde{x}).P \downarrow u \qquad P \mid Q \downarrow u \text{ if } P \downarrow u \text{ or } Q \downarrow u$$

Multisets are an equivalent formulation of the global pi calculus: a term P is a multiset of elements $\overline{u}\,\widetilde{x}.P$, $u(\widetilde{x}).P$, $!u(\widetilde{x}).P$ and $\nu x.P$; *new* operates on one element of the multiset, *reaction* on two; observation $P \downarrow u$ is when at least one element is observable on u.

Barbed bisimulation with respect to $\widetilde{z}$ is as follows. Write $\xrightarrow{\widetilde{z}}$ for $\rightarrow$ such that no $x' \in \widetilde{z}$ is ever chosen fresh by *new*. Write $P \downarrow^{\widetilde{z}} u$ when $P \downarrow u$ and $u \in \widetilde{z}$. Write $\stackrel{\widetilde{z}}{\Rightarrow}$ for $\xrightarrow{\widetilde{z}}{}^{*}$ and $\Downarrow^{\widetilde{z}}$ for $\stackrel{\widetilde{z}}{\Rightarrow}\downarrow^{\widetilde{z}}$. Then barbed bisimulation with respect to $\widetilde{z}$ is the largest symmetric relation $\stackrel{\widetilde{z}}{\approx}$ such that whenever $P \stackrel{\widetilde{z}}{\approx} Q$ then (1) $P \downarrow^{\widetilde{z}} u$ implies $Q \Downarrow^{\widetilde{z}} u$ and (2) $P \xrightarrow{\widetilde{z}} P'$ implies $Q \stackrel{\widetilde{z}}{\Rightarrow} \stackrel{\widetilde{z}}{\approx} P'$.

Barbed congruence written $P \approx Q$ holds if for all contexts C, then $C[P] \stackrel{\widetilde{z}}{\approx} C[Q]$ for $\widetilde{z} = \mathrm{fn}(C[P]|C[Q])$.

Table 2: The global pi calculus. Bound names $\mathrm{bn}(P)$ and free names $\mathrm{fn}(P)$ are standard: x is bound in $\nu x.P$ and $\widetilde{x}$ is bound in $u(\widetilde{x}).P$ and $!u(\widetilde{x}).P$. Note that reduction is closed only under associativity and commutativity of $|$: it is *not* closed under scope extrusion, alpha-renaming or contexts.

2 Global pi calculus

The global pi calculus is defined in Figure 2. Note that terms in this calculus have exactly the same syntax as those in the pi calculus; the difference is in their operational semantics. Strikingly, the global pi calculus allow terms to be considered purely as multisets – this is an important simplification.

We illustrate the global calculus with some examples. The step $\nu x.\overline{x} \rightarrow \overline{x}\,'$ creates a new name (it holds for any fresh x'). The example from page 1 is

$$\begin{array}{lcll} \nu x.\overline{u}\,x \mid u(y).(\overline{u}\,x|P) & \rightarrow & \overline{u}\,x' \mid u(y).(\overline{u}\,x|P) & \textit{create fresh channel } x' \\ & \rightarrow & \overline{x}\,'x \mid P\{x'/y\} & \textit{react} \end{array}$$

Traditional pi calculus has terms P and contexts C as in Table 2. We identify terms $P \equiv_\pi Q$ as follows:

$$P|\mathbf{0} \equiv_\pi P \qquad P|Q \equiv_\pi Q|P \qquad P|(Q|R) \equiv_\pi (P|Q)|R \qquad !P \equiv_\pi P \mid !P$$

$$\nu x.\nu y.P \equiv_\pi \nu y.\nu x.P \qquad \nu x.\mathbf{0} \equiv_\pi \mathbf{0}$$

$$x \notin \mathrm{fn}\, P \text{ implies } \nu x.(P|Q) \equiv_\pi P|\nu x.Q \text{ and } \nu y.P \equiv_\pi \nu x.P\{^x/_y\}$$

$$\frac{P \equiv_\pi Q}{Q \equiv_\pi P} \qquad \frac{P \equiv_\pi Q}{\nu x.P \equiv_\pi \nu x.Q} \qquad \frac{P \equiv_\pi Q}{P|R \equiv_\pi Q|R} \qquad \frac{P \equiv_\pi Q \quad Q \equiv_\pi R}{P \equiv_\pi R}$$

Reaction relation is as follows:

$$\overline{u}\,\widetilde{y}.P \mid u(\widetilde{x}).Q \;\rightarrow_\pi\; P \mid Q\{\widetilde{y}/\widetilde{x}\}$$

$$\frac{P \rightarrow_\pi P'}{\nu x.P \rightarrow_\pi \nu x.P'} \qquad \frac{P \rightarrow_\pi P'}{P|Q \rightarrow_\pi P'|Q} \qquad \frac{P \equiv_\pi Q \rightarrow_\pi Q' \equiv_\pi P'}{P \rightarrow_\pi P'}$$

Barbs (observations) are as follows: x is observable in P, written $P \downarrow x$, when

$$\overline{u}\,\widetilde{x}.P \downarrow_\pi u \qquad u(\widetilde{x}).P \downarrow_\pi u \qquad !u(\widetilde{x}).P \downarrow_\pi u \qquad P|Q \downarrow_\pi u \text{ if } P \downarrow_\pi u \text{ or } Q \downarrow_\pi u$$
$$\nu x.P \downarrow_\pi u \text{ if } P \downarrow_\pi u \text{ and } u \neq x$$

Barbed bisimulation $\dot{\approx}_\pi$ is as in Table 2, using $\downarrow_\pi$ and $\rightarrow_\pi$. **Barbed congruence**, written $P \approx Q$, holds if for all contexts C then $C[P] \dot{\approx}_\pi C[Q]$.

Table 3: The pi calculus.

The following two programs are barbed congruent: $\nu x.\overline{x} \approx \mathbf{0}$. Essentially this is because in any context $C[\nu x.\overline{x}\,]$, when the fresh name x' is created, then x' will be fresh also with respect to the context (and hence unobservable by the context).

There is a subtlety here. The observation relation $\downarrow$ is a blunt tool which observes every free name, even if no context could ever observe it in practice. To defend against this, the pi calculus retains restriction and disallows observation of restricted names. This amounts to adding part of the contextual equivalence semantics into the operational semantics. The global pi calculus achieves the same end but without compromising the operational semantics – instead it parameterises the observation relation according to which names it can feasibly observe. Writing $\overset{\widetilde{v}}{\approx}$ for a bisimulation which only ever observes names in $\widetilde{v}$, then

$$\nu x.(\overline{x}\,|\overline{y}\,) \;\overset{y}{\approx}\; \overline{y} \qquad \text{and} \qquad \overline{x}\,'|\overline{y} \;\overset{y}{\approx}\; \overline{y} \qquad \text{and} \qquad P \overset{\emptyset}{\approx} Q$$

The pi calculus uses restrictions to ensure that, with respect to bisimulation of $C[P]$ and $C[Q]$, the observation relation never observes any names that were not originally free in $C[P]$ or $C[Q]$. The global pi calculus uses its parameterised

observation instead of restrictions for the same end. Hence the two calculi make the same judgement as to whether P and Q are behaviourally equivalent:

Theorem 1 *$P \approx_\pi Q$ if and only if $P \approx Q$.*

The proof of this theorem is substantial. It involves considering all the rich behaviour afforded to the pi calculus from its compositionality, restrictions and structural congruence, and deriving it all from scratch for the global pi calculus. The proof structure is: define the *fresh pi*, an intermediate calculus; establish its 'rich behaviour'; prove full abstraction first between pi and fresh pi; then also with 'observation up to $\widetilde{v}$'; then also with global pi. In what follows we merely omit the (mostly straightforward) proofs. It might be that the theory of FM equivariance [13] allows for simpler proofs.

We reason in an intermediate calculus, the *fresh* pi calculus. Terms in the fresh calculus have the form $(\widetilde{x})\,P$, where $\widetilde{x}$ are the names that have so far been created fresh. This fresh calculus has the same operational semantics as the global pi calculus, but its annotation $(\widetilde{x})$ allows compositionality to be derived indirectly. For instance, if executing P causes some names $\widetilde{z}$ to be created $(\,)\,P \Rightarrow_f (\widetilde{z})\,P'$, then we can prove that an alternate execution must also have been possible: $(\,)\,P \Rightarrow_f (\widetilde{z'})\,P'\{\widetilde{z'}/\widetilde{z}\}$. And that therefore an alternate execution must also have been possible in the presence of a parallel term R, if $\widetilde{z'} \notin \mathrm{fn}(R)$: $(\,)\,P|R \Rightarrow_f (\widetilde{z'})\,P'\{\widetilde{z'}/\widetilde{z}\}|R$. In the pi calculus these examples are all axioms. In the fresh pi calculus they are indirect derivations. In the global pi calculus the answers are all still valid ($P \Rightarrow P'$ and $P \Rightarrow P'\{\widetilde{z'}/\widetilde{z}\}$ and $P|R \Rightarrow P'\{\widetilde{z}/\widetilde{z}\}|R$) but they cannot be derived from each other. A form of $(\widetilde{x})$ was previously used by the author along with Gardner and Laneve [14, 15]; the current results simplify and extend that earlier work.

Definition 2 (Fresh pi) *Terms in the fresh pi calculus have the form $(\widetilde{x})\,P$ where all $\widetilde{x}$s are distinct and their order immaterial; P and contexts C are as in Table 2. Identify terms $(\widetilde{x})\,P \equiv_f (\widetilde{y})\,Q$ when $\widetilde{y}$ is a permutation of $\widetilde{x}$, and P and Q are identical up to commutativity and associativity of $|$ with $\mathbf{0}$ as identity. The* heating *rule $\rightharpoonup_f$ and the reaction rule $\rightarrow_f$ are*

$$(\widetilde{z})\;\; R \mid \nu x.P \;\rightharpoonup_f\; (\widetilde{z}x')\;\; R \mid P\{x'/x\} \quad \text{where } x' \notin \{\widetilde{z}\} \cup \mathrm{fn}(R|\nu x.P)$$

$$(\widetilde{z})\;\; R \mid \overline{u}\,\widetilde{x}.P \mid u(\widetilde{y}).Q \;\rightarrow_f\; (\widetilde{z})\;\; R \mid P \mid Q\{\widetilde{x}/\widetilde{y}\}$$

$$(\widetilde{z})\;\; R \mid \overline{u}\,\widetilde{x}.P \mid !u(\widetilde{y}).Q \;\rightarrow_f\; (\widetilde{z})\;\; R \mid P \mid Q\{\widetilde{x}/\widetilde{y}\} \mid !u(\widetilde{y}).Q$$

Observations *are $(\widetilde{z})\,P \downarrow_f u$ if $u \notin \widetilde{z}$ and $P \downarrow u$ as in Table 2.* ***Barbed bisimulation*** *is as follows. Write $\Rightarrow_f$ for $(\rightharpoonup_f^* \rightarrow_f^*)^*$ and $\Downarrow_f$ for $\Rightarrow_f \downarrow_f$. Then barbed bisimulation is the largest symmetric relation $\dot{\approx}_f$ such that whenever $P \,\dot{\approx}_f\, Q$ then (1) $P \downarrow_f u$ implies $Q \Downarrow_f u$ and (2) $P \rightarrow_f P'$ or $P \rightharpoonup_f P'$ implies $Q \Rightarrow_f \dot{\approx}_f\, P'$. Two terms are* ***barbed congruent****, written $(\widetilde{x})\,P \approx_f (\widetilde{y})\,Q$, if for all contexts C with $\widetilde{x}\widetilde{y} \cap \mathrm{fn}(C) = \emptyset$ then $(\widetilde{x})\,C[P] \;\dot{\approx}_f\; (\widetilde{y})\,C[Q]$.* ***Barbed bisimulation with respect to names*** $\overset{\widetilde{z}}{\approx}_f$ *is as in Table 2, where $(\widetilde{x})\,P \downarrow_f^{\widetilde{v}} u$*

means $(\!|\widetilde{x}|\!)\, P \downarrow_f u$ *with* $u \in \widetilde{v}$*, and* $\xrightarrow{\widetilde{v}}_f$ *means a* new *transition where no name in* $x' \in \widetilde{v}$ *is chosen as the fresh name.*

Note that every sequence of transitions has the form $(\!|\widetilde{x}|\!)\, P \Rightarrow (\!|\widetilde{z}\widetilde{x}|\!)\, P'$. This is because the transitions only ever add new names to the list $(\!|\widetilde{x}|\!)$. Note also that $\mathrm{fn}(\nu\widetilde{z}.P') \subseteq \mathrm{fn}(P)$, and that if all names in $\widetilde{x}$ are pairwise distinct then so are all names in $\widetilde{z}\widetilde{x}$. The remainder of the proof uses the following lemmas, in order.

Lemmas. Alternatives to compositionality and structural congruence in transitions. If $(\!|\widetilde{x}|\!)\, P \Rightarrow_f (\!|\widetilde{x}\widetilde{z}|\!)\, P'$ then

1. (Old-intr) $(\!|\widetilde{x}\widetilde{y}|\!)\, P \Rightarrow_f (\!|\widetilde{x}\widetilde{y}\widetilde{z}|\!)\, P'$ where $\widetilde{y} \cap \{\widetilde{x}, \widetilde{z}\} = \emptyset$
2. (Old-elim) $(\!|\,|\!)\, P \Rightarrow_f (\!|\widetilde{z}|\!)\, P'$
3. (Old-alpha) $(\!|\widetilde{x}'|\!)\, P \Rightarrow_f (\!|\widetilde{x}'\widetilde{z}|\!)\, P'$ where $\widetilde{x}' \cap \widetilde{z} = \emptyset$
4. (Subst) $(\!|\widetilde{x}|\!)\, P\{u/v\} \Rightarrow_f (\!|\widetilde{x}\widetilde{z}|\!)\, P'\{u/v\}$ where $u \notin \widetilde{z}$
5. (New-alpha) $(\!|\widetilde{x}|\!)\, P \Rightarrow_f (\!|\widetilde{z}'\widetilde{x}|\!)\, P'\{\widetilde{z}'/\widetilde{z}\}$ where $\widetilde{z}'$ are pairwise distinct and $\widetilde{z}' \cap (\{\widetilde{x}\} \cup \mathrm{fn}\, P) = \emptyset$
6. (Par-intr) $(\!|\widetilde{x}|\!)\ P|Q \Rightarrow_f (\!|\widetilde{x}\widetilde{z}|\!)\ P'|Q$ where $\mathrm{fn}\, Q \cap \widetilde{z} = \emptyset$

Alternatives to compositionality and structural congruence up to congruence:

1. (Old-alpha) $(\!|\widetilde{x}|\!)\, P \approx_f (\!|\widetilde{x}'|\!)\, P\{\widetilde{x}'/\widetilde{x}\}$ where $\widetilde{x}' \notin \mathrm{fn}(\nu\widetilde{x}.P)$ and $\widetilde{x}'$ are pairwise distinct. (Prove this and the following with the Context Lemma).
2. (Old-for-new) $(\!|\,|\!)\, \nu x.P \approx_f (\!|x'|\!)\, P\{x'/x\}$ where $x \notin \mathrm{fn}(R)$.
3. (Old-intr) If $(\!|\widetilde{x}|\!)\, P \approx_f (\!|\widetilde{y}|\!)\, Q$ then $(\!|\widetilde{x}\widetilde{z}|\!)\, P \approx_f (\!|\widetilde{y}\widetilde{z}|\!)\, Q$ for $\widetilde{z} \cap \widetilde{x}\widetilde{y} = \emptyset$.
4. (Old-elim) If $y \notin \mathrm{fn}\, P$ then $(\!|\widetilde{x}y|\!)\, P \approx_f (\!|\widetilde{x}|\!)\, P$. The proof is tricky; use $\mathcal{S}= \{((\!|\widetilde{x}y|\!)\, P, B) : B \mathrel{\dot{\approx}_f} (\!|\widetilde{x}|\!)\, P$ and $y \notin \widetilde{x} \cup \mathrm{fn}(P)\}$ and prove it is a $\dot{\approx}_f$.
5. (Structural) If $P \equiv_\pi Q$ then $(\!|\,|\!)\, P \approx_f (\!|\,|\!)\, Q$.

Bisimulation and full abstraction:

1. *(Between pi and fresh pi)* $\nu\widetilde{x}.P \mathrel{\dot{\approx}_\pi} \nu\widetilde{y}.Q$ if and only if $(\!|\widetilde{x}|\!)\, P \mathrel{\dot{\approx}_f} (\!|\widetilde{y}|\!)\, Q$.
2. *(Full abstraction)* $\nu\widetilde{x}.P \approx_\pi \nu\widetilde{y}.Q$ if and only if $(\!|\widetilde{x}|\!)\, P \approx_f (\!|\widetilde{y}|\!)\, Q$.
3. *(Between fresh pi, and fresh pi observed up to* $\widetilde{v}$*)* $(\!|\widetilde{x}|\!)\, P \Rightarrow_f (\!|\widetilde{x}\widetilde{z}|\!)\, P'$ implies for every $\widetilde{v}$ then $(\!|\widetilde{x}|\!)\, P \stackrel{\widetilde{v}}{\Rightarrow}_f \mathrel{\dot{\approx}_f} (\!|\widetilde{x}\widetilde{z}|\!)\, P'$.
4. *(Bisimulation)* If $(\!|\widetilde{x}|\!)\, P \mathrel{\dot{\approx}_f} (\!|\widetilde{y}|\!)\, Q$ then $(\!|\widetilde{x}|\!)\, P \stackrel{\widetilde{v}}{\approx}_f (\!|\widetilde{y}|\!)\, Q$ for all $\widetilde{v}$. And if $(\!|\widetilde{x}|\!)\, P \stackrel{\widetilde{v}}{\approx}_f (\!|\widetilde{y}|\!)\, Q$ for some $\widetilde{v} \supseteq \mathrm{fn}(\nu\widetilde{x}.P|\nu\widetilde{y}.Q)$, then $(\!|\widetilde{x}|\!)\, P \mathrel{\dot{\approx}_f} (\!|\widetilde{y}|\!)\, Q$.
5. *(Full abstraction)* If $(\!|\widetilde{x}|\!)\, P \approx_f (\!|\widetilde{y}|\!)\, Q$ then for all contexts C and all $\widetilde{v}$ then $(\!|\widetilde{x}|\!)\, C[P] \stackrel{\widetilde{v}}{\approx}_f (\!|\widetilde{y}|\!)\, C[Q]$. And if for all contexts C there exists some $\widetilde{v} \supseteq \mathrm{fn}(\nu\widetilde{x}.C[P]|\nu\widetilde{y}.C[Q])$ such that $(\!|\widetilde{x}|\!)\, C[P] \stackrel{\widetilde{v}}{\approx}_f (\!|\widetilde{y}|\!)\, C[Q]$, then $(\!|\widetilde{x}|\!)\, P \approx_f (\!|\widetilde{y}|\!)\, Q$.
6. *(Between fresh and global pi)* $P \stackrel{\widetilde{v}}{\Rightarrow} P'$ implies for all $\widetilde{x}$ there exists $\widetilde{z}$ such that $(\!|\widetilde{x}|\!)\, P \stackrel{\widetilde{v}}{\Rightarrow}_f \stackrel{\widetilde{v}}{\approx}_f (\!|\widetilde{x}\widetilde{z}|\!)\, P'$ and $\widetilde{z} \cap \widetilde{v} = \emptyset$. Proof by induction on the derivation of $\stackrel{\widetilde{v}}{\Rightarrow}$. In the *new* case, either the freshly created name z' is in $\widetilde{x}$ or not. If it is, then react first to make z'', then use *(old-elim)* to remove z' from $\widetilde{x}$, then do $\{z'/z''\}$.
7. *(Bisimulation)* $P \stackrel{\widetilde{v}}{\approx} Q$ if and only if $(\!|\,|\!)\, P \stackrel{\widetilde{v}}{\approx}_f (\!|\,|\!)\, Q$
8. *(Full abstraction)* $P \approx Q$ if and only if $(\!|\,|\!)\, P \approx_f (\!|\,|\!)\, Q$

3 Protocol for synchronous rendezvous

As a practical application of the global semantics, we describe an implementation of the pi calculus over a broadcast network, such as wireless or ethernet. The pi calculus has previously been implemented for single-processor machines [5, 4, 18]; also for wide-area web services [6]. A broadcast network fits between the two, but uses different techniques. Notably, in a broadcast network the requirement of *locality* can be relaxed. (Locality is when all the receivers for a given channel are at the same location). Also, in single-processor and wide-area implementations, *nu* allocates a single central space for the new channel; in a broadcast network it need merely generate a fresh name.

For modelling the broadcast network we use Prasad's Calculus of Broadcasting Systems [19]. We also assume no failures. This is clearly not realistic; in the sequel [23] we add failures, develop a new protocol for reliable rendezvous, and prove it correct using the techniques developed here.

Assume a broadcast network with several locations. Each location starts with some pieces of pi calculus bytecode P. Certain bytecode, when executed, results in a message being broadcast over the network. In addition to bytecode, a location can also contain *atoms* A. An atom is a blocked program which is waiting to rendezvous, and also contains extra state information related to the protocol. When a message is heard over the network, this state information may be updated. We abstract away from locations, and consider the entire system merely as a collection of pi programs P and atoms A. The protocol is as follows.

1. A 'send' command $\overline{u}\,x.P$ broadcasts an *offer* to send on channel u. It generates a globally unique identifier to identify this particular offer. The program then blocks until someone accepts the offer.
2. A 'receive' command $u\,(y).Q$ can hear the offer. It broadcasts its acceptance, mentioning the offer's unique identifier. This acceptance signifies that the rendezvous will take place between the two parties. If there were multiple receivers competing for the offer, then necessarily only one of them can have accepted first; this one is deemed the winner.
3. Symmetrically, receivers can initiate offers, to be accepted by senders.
4. While a program is blocked after making an offer, it can still accept other offers that it hears.

Although the protocol seems simple, we at first made several mistakes. If a party makes an offer n, then decides to accept an offer m but loses the competition for it, we forgot that it had to remain open to n. Or if it won the competition for m, we forgot that it had to rescind its former offer on n. It is reassuring that we have provided a formalisation and proof for the final version of the protocol.

Our motive for studying a broadcast network – rather than the more usual point-to-point – was that it seemed easier. We thought there would be broadcast-specific optimisations that would make reliability easier in the presence of failure, along the lines of 'winning the competition' in Step 2 above. This hope turned out false: in the presence of failures, point-to-point it basically like broadcast

where each message makes it through to just one recipient; hence one cannot rely on 'competition' optimisations. Failures are discussed in the sequel [23].

We now formalise the protocol. Assume an infinite set of *channel-names* ranged over by $u, x, \ldots$, and an infinite set of *offer identifiers* ranged over by $m, n, \ldots$. Write $\widetilde{x}$ to stand for the sequence $x_1, \ldots, x_n$, and α for $\overline{u}\,\widetilde{x}$ or $u(\widetilde{x})$, and $\widetilde{m.\alpha}$ for $m_1\alpha_1, \ldots, m_n\alpha_n$.

Definition 3 (Syntax) *Programs P, atoms A, machines M are*

$$\begin{array}{lcl} P & ::= & \overline{u}\,\widetilde{x}.P \;\mid\; u(\widetilde{x}).P \;\mid\; !u(\widetilde{x}).P \;\mid\; \nu x.P \;\mid\; P|P \;\mid\; \mathbf{0} \\ A & ::= & n; \widetilde{m.\alpha}; \alpha.P \\ M & ::= & P \;\mid\; A \;\mid\; M|M \end{array}$$

Identify terms up to commutativity and associativity of $|$ with $\mathbf{0}$ as identity, and up to reordering of the multiset $\widetilde{m.\alpha}$.

Programs P are standard from the pi calculus. The *atom* $n; \widetilde{m.\alpha}; \alpha.P$ means that the program $\alpha.P$ has already made an offer n to react; it has also heard several offers m_i which it wishes to accept, each m_i having offered the corresponding action α_i. The $\widetilde{m.\alpha}$ are not binding in $\alpha.P$. Through abuse of notation we use the same symbol for parallel composition $M|M$ of machine-fragments, as for a single piece of bytecode $P|P$. The difference is unimportant since the intended meaning of the bytecode is that, upon execution, it will produce two parallel machine fragments.

Note that, up to commutativity and associativity of $|$, a machine M is simply a multiset consisting of atoms A and the pi terms $\overline{u}\,\widetilde{x}.P$, $u(\widetilde{x}).P$, $!u(\widetilde{x}).P$, νx.

For the operational semantics we follow Prasad's broadcast semantics [19]: only one message can be sent at a time; when it is sent, all other parties hear it. Write $M \xrightarrow{!\mu} M'$ for when the machine M broadcasts a message μ (to be defined later) and consequently evolves into state M'. Write $M \xrightarrow{?\mu} M'$ for when a machine can evolve through hearing the message μ. The broadcast and hearing of a message are combined thus:

$$\frac{M_! \xrightarrow{!\mu} M'_1 \quad M_2 \xrightarrow{?\mu} M'_2 \;\ldots\; M_n \xrightarrow{?\mu} M'_n}{M_1|\ldots|M_n \rightarrow_m M'_1|\ldots|M'_n}$$

Note that every term must be *input-enabled*: it must be able to hear any message at any time, even if its response is to do nothing. Each machine will hear many messages, including many in which it has no interest. The semantics must give an explicit 'discard' rule for deducing a vacuous transition in this case. But in an implementation, any message without an explicit handler will be discarded automatically: the discard rules need not be implemented.

The following messages μ may be broadcast:

$$\begin{array}{lcll} \mu & ::= & n{=}\mathsf{off}\,\overline{u}\,\widetilde{x} \;\mid\; n{=}\mathsf{off}\,u(\widetilde{x}) & \textit{Offers} \\ & \mid & \mathsf{acc}\,n.u(\widetilde{x})\text{-}m \;\mid\; \mathsf{acc}\,n.\overline{u}\,\widetilde{x}\text{-}m & \textit{Acceptances} \\ & \mid & \nu x & \textit{New channel} \end{array}$$

The two *offers* indicate that a program is willing to react on channel u, and if it does react it will either send the arguments $\widetilde{x}$ or receive some formal arguments $(\widetilde{x})$, and that any subsequent acceptances should refer to the offer by its globally unique identifier n. The *acceptances* indicate that a term accepts the offer m to react, that its contribution will be to either receive formal arguments $(\widetilde{x})$ or send arguments $\widetilde{x}$, and that simultaneously it is rescinding its own offer on m. The final message νx indicates that a new channel x has been created; this is purely a notational convenience, so that new-channel-creation can be treated uniformly as a broadcast step. The sense of these labels can be a little confusing. We offer the following mnemonic: $n; m.\alpha; \beta.P \xrightarrow{!\,\mathsf{acc}\, m.\beta\text{-}n}$ means 'I am the term $\beta.P$, I have received an offer m which would give me the action α, and I accept the offer m by giving in return the action β; simultaneously I rescind my own offer n.'

Definition 4 (Operation) *The following rules give broadcast* $M \xrightarrow{!\mu} M'$ *and listening* $M \xrightarrow{?\mu} M'$ *transitions.*

$$\overline{u}\,\widetilde{x}.P \xrightarrow{!n=\mathsf{off}\,\overline{u}\,\widetilde{x}} n;;\overline{u}\,\widetilde{x}.P, \quad n \text{ fresh} \qquad \text{(make offer)}$$

$$u(\widetilde{x}).P \xrightarrow{!n=\mathsf{off}\,u(\widetilde{x})} n;;u(\widetilde{x}).P, \quad n \text{ fresh} \qquad \text{(make offer)}$$

$$!u(\widetilde{x}).P \xrightarrow{!n=\mathsf{off}\,u(\widetilde{x})} n;;!u(\widetilde{x}).P, \quad n \text{ fresh} \qquad \text{(make offer)}$$

$$n; \widetilde{m.\alpha}; \overline{u}\,\widetilde{x}.P \xrightarrow{?m'=\mathsf{off}\,u(\widetilde{y})} n; \widetilde{m.\alpha}, m'.u(\widetilde{y}); \overline{u}\,\widetilde{x}.P \qquad \text{(hear offer)}$$

$$n; \widetilde{m.\alpha}; u(\widetilde{x}).P \xrightarrow{?m'=\mathsf{off}\,\overline{u}\,\widetilde{y}} n; \widetilde{m.\alpha}, m'.\overline{u}\,\widetilde{y}; u(\widetilde{x}).P \qquad \text{(hear offer)}$$

$$n; \widetilde{m.\alpha}; !u(\widetilde{x}).P \xrightarrow{?m'=\mathsf{off}\,\overline{u}\,\widetilde{y}} n; \widetilde{m.\alpha}, m'.\overline{u}\,\widetilde{y}; !u(\widetilde{x}).P \qquad \text{(hear offer)}$$

$$n; \widetilde{m.\alpha}, m'.u(\widetilde{y}); \overline{u}\,\widetilde{x}.P \xrightarrow{!\,\mathsf{acc}\, m'.\overline{u}\,\widetilde{x}\text{-}n} P \qquad \text{(accept offer)}$$

$$n; \widetilde{m.\alpha}, m'.\overline{u}\,\widetilde{y}; u(\widetilde{x}).P \xrightarrow{!\,\mathsf{acc}\, m'.u(\widetilde{x})\text{-}n} P\{\widetilde{y}/\widetilde{x}\} \qquad \text{(accept offer)}$$

$$n; \widetilde{m.\alpha}, m'.\overline{u}\,\widetilde{y}; !u(\widetilde{x}).P \xrightarrow{!\,\mathsf{acc}\, m'.u(\widetilde{x})\text{-}n} P\{\widetilde{y}/\widetilde{x}\} \mid !u(\widetilde{x}).P \qquad \text{(accept offer)}$$

$$n; \widetilde{m.\alpha}, m'.\alpha'; \beta.P \xrightarrow{?\,\mathsf{acc}\, m'.\gamma\text{-}n'} n; \widetilde{m.\alpha}; \beta.P \qquad \text{(lose competition)}$$

$$n; \widetilde{m.\alpha}, n'.\alpha'; \beta.P \xrightarrow{?\,\mathsf{acc}\, m'.\gamma\text{-}n'} n; \widetilde{m.\alpha}; \beta.P \qquad \text{(lose opportunity)}$$

$$n; \widetilde{m.\alpha}; \overline{u}\,\widetilde{x}.P \xrightarrow{?\,\mathsf{acc}\, n.u(\widetilde{y})\text{-}n'} P \qquad \text{(hear acceptance)}$$

$$n; \widetilde{m.\alpha}; u(\widetilde{x}).P \xrightarrow{?\,\mathsf{acc}\, n.\overline{u}\,\widetilde{y}\text{-}n'} P\{\widetilde{y}/\widetilde{x}\} \qquad \text{(hear acceptance)}$$

$$n; \widetilde{m.\alpha}; !u(\widetilde{x}).P \xrightarrow{?\,\mathsf{acc}\, n.\overline{u}\,\widetilde{y}\text{-}n'} P\{\widetilde{y}/\widetilde{x}\} \mid !u(\widetilde{x}).P \qquad \text{(hear acceptance)}$$

$$\nu x.P \xrightarrow{!\nu x'} P\{x'/x\}, \quad x' \text{ fresh} \qquad \text{(create new channel)}$$

The following are 'discard' transitions. (Equivalently, one could simply say that $M \xrightarrow{?\mu} M$ *for* $M \neq M_1|M_2$ *and not matched by any of the above rules.)*

$$P \xrightarrow{?\mu} P$$

$$A \xrightarrow{?\nu x} A$$

$$n; \widetilde{m.\alpha}; \beta.P \xrightarrow{?\,\mathsf{acc}\, m'.\gamma;\text{-}n'} n; \widetilde{m.\alpha}; \beta.P \qquad \text{if } n' \neq n \text{ and } m' \notin \widetilde{m}$$

$$n; \widetilde{m.\alpha}; \overline{u}\,\widetilde{x}.P \xrightarrow{?m' = \mathsf{off}\, v(\widetilde{x})} n; \widetilde{m.\alpha}; \overline{u}\,\widetilde{x}.P \qquad \text{if } u \neq v$$

$$n; \widetilde{m.\alpha}; \overline{u}\,\widetilde{x}.P \xrightarrow{?m' = \mathsf{off}\, \overline{v}\,\widetilde{y}} n; \widetilde{m.\alpha}; \overline{u}\,\widetilde{x}.P \qquad \text{if } u = v \text{ or } u \neq v$$

$$n; \widetilde{m.\alpha}; u(\widetilde{x}).P \xrightarrow{?m' = \mathsf{off}\, \overline{v}\,\widetilde{x}} n; \widetilde{m.\alpha}; u(\widetilde{x}).P \qquad \text{if } u \neq v$$

$$n; \widetilde{m.\alpha}; u(\widetilde{x}).P \xrightarrow{?m' = \mathsf{off}\, v(\widetilde{x})} n; \widetilde{m.\alpha}; u(\widetilde{x}).P \qquad \text{if } u = v \text{ or } u \neq v$$

$$n; \widetilde{m.\alpha}; !u(\widetilde{x}).P \xrightarrow{?m' = \mathsf{off}\, \overline{v}\,\widetilde{x}} n; \widetilde{m.\alpha}; !u(\widetilde{x}).P \qquad \text{if } u \neq v$$

$$n; \widetilde{m.\alpha}; !u(\widetilde{x}).P \xrightarrow{?m' = \mathsf{off}\, v(\widetilde{x})} n; \widetilde{m.\alpha}; !u(\widetilde{x}).P \qquad \text{if } u = v \text{ or } u \neq v$$

The following final rule is characteristic of a broadcast network. Any fresh names or identifiers generated by $M_1 \xrightarrow{!\mu} M_1'$ *are assumed fresh also with respect to* $M_2 \ldots M_n$*.*

$$\frac{M_! \xrightarrow{!\mu} M_1' \quad M_2 \xrightarrow{?\mu} M_2' \;\ldots\; M_n \xrightarrow{?\mu} M_n'}{M_1|\ldots|M_n \rightarrow_m M_1'|\ldots|M_n'} \textit{ broadcast}$$

The following **well-formedness** invariants are preserved by transitions:

1. For each offer identifier n, no more than a single atom of multiset M has the form $n; \widetilde{m.\alpha}; \beta.P$. *(No offer is made twice).*
2. For all atoms $n; \widetilde{m.\alpha}; \beta.P$ then $m_i = m_j$ implies $i = j$. *(No offer is heard twice).*
3. For any atom $n; \widetilde{m.\alpha}; \beta.P$ then either β is an output $\overline{u}\,\widetilde{x}$ and all α are inputs $u(\widetilde{x})$ or $!u(\widetilde{x})$ on the same name u, or vice versa. *(Only complimentary offers are heard).*
4. If the machine contains an atom $n; \widetilde{m.\alpha}; \beta.P$, then for every $m_i.\alpha_i \in \widetilde{m.\alpha}$ the machine must also contain some $m_i; \widetilde{m'.\alpha'}; \alpha_i.Q$. *(Every heard offer must previously have been made).*
5. If the machine contains atoms $n; \widetilde{m.\alpha}; \overline{u}\,\widetilde{x}.P$ and $n'; \widetilde{m'.\alpha'}; u(\widetilde{y}).Q$ then either $n \in \widetilde{m}'$ or $n' \in \widetilde{m}$. Similarly for $!u(\widetilde{y}).Q$. *(Of any two complimentary offers on a channel, one came first).*

Note that the well-formedness invariants are trivially satisfied by a machine P that contains no atoms. Henceforth we assume all machines to be well-formed. Note also that, because of complementarity, given an atom A and a message μ, it is impossible that $A \xrightarrow{?\mu}$ admits both the *lose competition* and the *lose opportunity* transitions.

Definition 5 (Bisimulation) *A machine* M *is observable at* u *with respect to* $\widetilde{v}$*, written* $M \downarrow_m^{\widetilde{v}} u$*, if* $u \in \widetilde{v}$ *and* $M \downarrow_m u$ *as follows:*

$$n; \widetilde{m.\alpha}; \overline{u}\,\widetilde{x}.P \downarrow_m u \qquad n; \widetilde{m.\alpha}; u(\widetilde{x}).P \downarrow_m u \qquad n; \widetilde{m.\alpha}; !u(\widetilde{x}).P \downarrow_m u$$

$$M|N \downarrow_m u \text{ if } M \downarrow_m u \text{ or } N \downarrow_m u$$

Write $M \xrightarrow{\widetilde{v}}_m M'$ *when the message was not* νv *for any* $v \in \widetilde{v}$*. Write* $\stackrel{\widetilde{v}}{\Rightarrow}_m$ *for* $\xrightarrow{\widetilde{v}}^*$ *and* $\Downarrow_m^{\widetilde{v}}$ *for* $\stackrel{\widetilde{v}}{\Rightarrow}_m \downarrow_m^{\widetilde{v}}$*.* ***Barbed bisimulation*** *with respect to* $\widetilde{v}$ *is the largest symmetric relation* $\stackrel{\widetilde{v}}{\approx}_m$ *such that whenever* $M \stackrel{\widetilde{v}}{\approx}_m N$ *then (1)* $M \downarrow_m^{\widetilde{v}} u$ *implies* $N \Downarrow_m u$ *and (2)* $M \stackrel{\widetilde{v}}{\Rightarrow}_m M'$ *implies* $N \stackrel{\widetilde{v}}{\Rightarrow}_m \stackrel{\widetilde{v}}{\approx}_m M'$*. Define* ***machine contexts*** $C_m ::= C \mid C_m|M \mid M|C_m$ *where* C *is as in Table 2. Then two programs are* ***barbed congruent*** *according to the machine semantics, written* $P \approx_m Q$*, if for all machine contexts* C_m *then* $C_m[P] \stackrel{\widetilde{v}}{\approx}_m C_m[Q]$ *for* $\widetilde{v} = \mathrm{fn}(C[P]|C[Q])$*.*

We remark that observations $P \downarrow u$ are normally understood to mean that an observer can interact with P over channel u. Exactly the same sense is used for machine observations $M \downarrow_m u$. For example, an observer can interact with the atom $n; \widetilde{m.\alpha}; \overline{u}\,\widetilde{x}.P$ by broadcasting the offer $\xrightarrow{!m' = \mathsf{off}\, u(\tilde{y})}$ and waiting for the atom to respond with $\xrightarrow{!\,\mathsf{acc}\, m'.\overline{u}\,\widetilde{x}\text{-}n}$.

We now demonstrate the implementation's correctness with respect to the global pi calculus (and hence, via Theorem 1, with the normal pi calculus). Use the following translation $[\![\cdot]\!]$ from machines to terms in the pi calculus:

$$[\![n; \widetilde{m.\alpha}; \beta.P]\!] = \beta.P \qquad [\![M|N]\!] = [\![M]\!] \mid [\![N]\!] \qquad [\![P]\!] = P$$

From the multiset perspective, the translation turns a machine multiset into the same program multiset but where each atom $n; \widetilde{m.\alpha}; \beta.P$ becomes just $\beta.P$. The translation strips away an atom's information about previously broadcast offers. In general this can affect meaning: for instance $M = n;;\overline{u}\,x \mid m;;u\,(y)$ admits no further interactions, but $[\![M]\!]$ does. However, this M violates the *one-came-first* well-formedness property. It turns out that translation preserves meaning for well-formed machines: (we omit the proof for lack of space).

Proposition 6 (Protocol correctnes) (Bisimulation) $M \stackrel{\widetilde{v}}{\approx}_m N$ *if and only if* $[\![M]\!] \stackrel{\widetilde{v}}{\approx}_m [\![N]\!]$*, and also* (Full abstraction) $P \approx_m Q$ *if and only if* $P \approx Q$*.*

4 Discussion

In the pi calculus, a restricted name's scope can be extruded or 'intruded'. This demarcates the set of programs which might know about the name – anything outside the scope does not know about it. We have eliminated scope extrusion entirely. But Greg Meredith has suggested in discussion that scope knowledge might prove useful for distributed error recovery: if the machine hosting channel x should crash, and a replacement channel x' is brought online, then only those names within the scope of x need be told about the replacement channel.

With respect to the idea creating fresh names (globally unique), Robin Milner has suggested in discussion that such freshness is not possible in principle. For instance, if the Internet on our 'globe' should be merged with that of Mars, then our already-created names might clash with those of the martians. This is a light-hearted example, but it illustrates the striking power of scope extrusion.

References

1. D. Berry, R. Milner, and D. N. Turner. A semantics for ML concurrency primitives. In *ACM SIGPLAN/SIGACT*, pages 119–129. ACM Press, 1992.
2. G. Berry and G. Boudol. The chemical abstract machine. *Theoretical Computer Science*, 96(1):217–248, 1992.
3. R. Bruni, F. Honsell, M. Lenisa, and M. Miculan. Modeling fresh names in pi-calculus using abstractions. Technical Report TR-21-02, Dipartimento di Matematica e Informatica, Università di Udine, Italy, 2002.
4. L. Cardelli. An implementation model of rendezvous communication. In *Seminar on Concurrency*, LNCS 197:449–457, 1984.
5. L. Cardelli and R. Pike. Squeak: A language for communicating with mice. In *Proceedings of SIGGRAPH '85*, volume 19, pages 199–204. ACM Press.
6. M. Corporation. Biztalk server. http:// www.microsoft.com/ biztalk/.
7. R. de Nicola, G. Ferrari, and R. Pugliese. KLAIM: A kernel language for agents interaction and mobility. *IEEE Trans. Software Engineering*, 24(5):315–330, 1998.
8. J. Engelfriet. A multiset semantics for the pi-calculus with replication. *Theoretical Computer Science*, 153(1–2):65–94, 1996.
9. C. Fournet and G. Gonthier. The reflexive chemical abstract machine and the join-calculus. In *POPL 1996*, pages 372–385. ACM Press.
10. C. Fournet, G. Gonthier, J.-J. Lévy, L. Maranget, and D. Rémy. A calculus of mobile agents. In *CONCUR 1996*, LNCS 1119:406–421.
11. M. J. Gabbay. FM for process calculi that generate fresh names, 2003. Submitted for publication.
12. M. J. Gabbay. The pi-calculus in Fraenkel-Mostowski, 2003. Submitted for publication.
13. M. J. Gabbay and A. M. Pitts. A new approach to abstract syntax with variable binding. *Formal Aspects of Computing*, 13(0):1–23, 2001.
14. P. Gardner, C. Laneve, and L. Wischik. The fusion machine (extended abstract). In *CONCUR 2002*, LNCS 2421:418–433.
15. P. Gardner, C. Laneve, and L. Wischik. Linear forwarders. In *CONCUR 2003*, LNCS 2761:415–430.
16. A. Giacalone, P. Mishra, and S. Prasad. FACILE: A symmetric integration of concurrent and functional programming. *International Journal of Parallel Programming*, 18(2):121–160, 1989.
17. U. Montanari and M. Pistore. Pi-calculus, structured coalgebras, and minimal HD-automata. In *MFCS 2000*, LNCS 1893:569–578.
18. B. C. Pierce and D. N. Turner. Pict: A programming language based on the pi-calculus. In *Proof, Language and Interaction: Essays in Honour of Robin Milner*, Foundations of Computing, pages 455–494. MIT Press, 2000.
19. K. V. S. Prasad. A calculus of broadcasting systems. In *TAPSOFT 1991*, LNCS 493:338–358.
20. J. H. Reppy. CML: A higher-order concurrent language. *ACM SIGPLAN*, 26(6):293–305, 1991.
21. P. Sewell. On implementations and semantics of a concurrent programming language. In *CONCUR 1997*, LNCS 1243:391–405.
22. D. N. Turner. *The Polymorphic Pi-Calculus: Theory and Implementation.* PhD thesis, Department of Computer Science, University of Edinburgh, UK, 1996.
23. L. Wischik. Pi implementation (2): A reliable protocol for rendezvous, 2003. Awaiting submission.

Appendix: Proofs omitted from main paper.

Note to referees: Many of the proofs are straightforward. Even if I had more space I would include only sketches for half of them. But I have attached the full proofs here because (1) I like to keep the detailed proofs somewhere, (2) as indicated in the paper the results really are non-trivial, and (3) an earlier version of the paper was rejected elsewhere for faulty proofs – should the same referees be asked to referee this version as well, I'd like them to know that the faults have been fixed.

Lemma 7 *If* $(\!|\widetilde{x}|\!)\, P \Rightarrow_f (\!|\widetilde{x}\widetilde{z}|\!)\, P'$ *then*

1. *(Old-intr)* $(\!|\widetilde{x}\widetilde{y}|\!)\, P \Rightarrow_f (\!|\widetilde{x}\widetilde{y}\widetilde{z}|\!)\, P'$ *where* $\widetilde{y} \cap \{\widetilde{x}, \widetilde{z}\} = \emptyset$
2. *(Old-elim)* $(\!|\,|\!)\, P \Rightarrow_f (\!|\widetilde{z}|\!)\, P'$
3. *(Old-alpha)* $(\!|\widetilde{x}'|\!)\, P \Rightarrow_f (\!|\widetilde{x}'\widetilde{z}|\!)\, P'$ *where* $\widetilde{x}' \cap \widetilde{z} = \emptyset$
4. *(Subst)* $(\!|\widetilde{x}|\!)\, P\{u/v\} \Rightarrow_f (\!|\widetilde{x}\widetilde{z}|\!)\, P'\{u/v\}$ *where* $u \notin \widetilde{z}$
5. *(New-alpha)* $(\!|\widetilde{x}|\!)\, P \Rightarrow_f (\!|\widetilde{z}'\widetilde{x}|\!)\, P'\{\widetilde{z}'/\widetilde{z}\}$ *where* $\widetilde{z}'$ *are pairwise distinct and* $\widetilde{z}' \cap (\{\widetilde{x}\} \cup \mathrm{fn}\, P) = \emptyset$
6. *(Par-intr)* $(\!|\widetilde{x}|\!)\ P|Q \Rightarrow_f (\!|\widetilde{x}\widetilde{z}|\!)\ P'|Q$ *where* $\mathrm{fn}\, Q \cap \widetilde{z} = \emptyset$

Proof. **Part 1** is a simple induction on the derivation of $\Rightarrow_f$. So is **Part 2**. In its induction step, we assume the derivation

$$\frac{(\!|\widetilde{x}|\!)\, P \Rightarrow_f (\!|\widetilde{x}\widetilde{z}_1|\!)\, P' \qquad (\!|\widetilde{x}\widetilde{z}_1|\!)\, P' \Rightarrow_f (\!|\widetilde{x}\widetilde{z}_1\widetilde{z}_2|\!)\, P''}{(\!|\widetilde{x}|\!)\, P \Rightarrow_f (\!|\widetilde{x}\widetilde{z}_1\widetilde{z}_2|\!)\, P''}$$

Note that $\widetilde{z}_1 \cap \widetilde{z}_2 = \emptyset$, since reduction always results in pairwise-distinct names. Then

$$\dfrac{(\!|\,|\!)\, P \Rightarrow_f (\!|\widetilde{z}_1|\!)\, P' \ \textit{(i.h.)} \qquad \dfrac{(\!|\,|\!)\, P' \Rightarrow_f (\!|\widetilde{z}_2|\!)\, P'' \ \textit{(i.h.)} \qquad \widetilde{z}_1 \cap \widetilde{z}_2 = \emptyset}{(\!|\widetilde{z}_1|\!)\, P' \Rightarrow_f (\!|\widetilde{z}_1\widetilde{z}_2|\!)\, P''}\text{old-intr}}{(\!|\,|\!)\, P \Rightarrow_f (\!|\widetilde{z}_1\widetilde{z}_2|\!)\, P''}\text{trans.}$$

completing the proof.

Part 3 follows from applying *old-elim* to remove $\widetilde{x}$ and then *old-intr* to add $\widetilde{x}'$.

Part 4 is another induction on the derivation of $\Rightarrow_f$. For the *new* base case, we assume $(\!|\widetilde{x}|\!)\, R|\nu x.P \rightharpoonup_f (\!|\widetilde{x}|\!)\, R|P\{z/x\}$ for $z \notin \{\widetilde{x}\} \cup \mathrm{fn}(R|\nu x.P)$. There are the usual capture-avoiding issues; let us ignore them and assume neither u nor v is x. Then we must deduce

$$(\!|\widetilde{x}|\!)\, R\{u/v\} \mid \nu x.P\{u/v\} \qquad \rightharpoonup_f \qquad (\!|z\widetilde{x}|\!)\, R\{u/v\} \mid P\{u/v\}$$

This holds whenever $z \notin \{\widetilde{x}\} \cup \mathrm{fn}(R\{u/v\}|\nu x.P\{u/v\}$. We already know that $z \notin \widetilde{x}$. By the side condition on the statement of Part 4, $z \neq u$. And the free names of $R\{u/v\}$ are a subset of $\{u\} \cup \mathrm{fn}(R)$. Hence the desired heating step holds.

Part 5 is again an induction on the derivation of $\Rightarrow_f$. Consider the induction step. Assume the derivation

$$\frac{(\widetilde{x})\,P \Rightarrow_f (\widetilde{x}\widetilde{z}_1)\,P' \qquad (\widetilde{x}\widetilde{z}_1)\,P' \Rightarrow_f (\widetilde{x}\widetilde{z}_1\widetilde{z}_2)\,P''}{(\widetilde{x})\,P \Rightarrow_f (\widetilde{x}\widetilde{z}_1\widetilde{z}_2)\,P''}$$

and assume $\widetilde{z}'_1\widetilde{z}'_2$ are pairwise distinct and $\widetilde{z}'_1\widetilde{z}'_2 \cap (\widetilde{x} \cup \mathrm{fn}\,P) = \emptyset$. The first induction hypothesis is that $(\widetilde{x})\,P \Rightarrow_f (\widetilde{x}\widetilde{z}'_1)\,P'\{\widetilde{z}'_1/\widetilde{z}_1\}$ since $\widetilde{z}'_1 \cap (\widetilde{x} \cup \mathrm{fn}\,P) = \emptyset$. For the second induction hypothesis, with the assumption $\widetilde{z}'_2 \cap (\widetilde{x}\widetilde{z}_1 \cup \mathrm{fn}\,P) = \emptyset$, we make the following inferences:

$(\widetilde{x}\widetilde{z}_1)\,P' \Rightarrow_f (\widetilde{x}\widetilde{z}_1\widetilde{z}'_2)\,P''\{\widetilde{z}'_2/\widetilde{z}_2\}$	(i.h.), since $\widetilde{z}'_2 \cap (\widetilde{x}\widetilde{z}_1 \cap \mathrm{fn}\,P) = \emptyset$
$()\,P' \Rightarrow_f (\widetilde{z}'_2)\,P''\{\widetilde{z}'_2/\widetilde{z}_2\}$	(old-elim), since $\widetilde{z}'_1 \cap \widetilde{z}'_2 = \emptyset$
$()\,P'\{\widetilde{z}'_1/\widetilde{z}_1\} \Rightarrow_f (\widetilde{z}'_2)\,P''\{\widetilde{z}'_2/\widetilde{z}_2\}\{\widetilde{z}'_1/\widetilde{z}_1\}$	(subst), since $\widetilde{z}'_1 \cap \widetilde{z}'_2 = \emptyset$
$()\,P'\{\widetilde{z}'_1/\widetilde{z}_1\} \Rightarrow_f (\widetilde{z}'_2)\,P''\{\widetilde{z}'_1\widetilde{z}'_2/\widetilde{z}_1\widetilde{z}_2\}$	(*)
$(\widetilde{x}\widetilde{z}'_1)\,P'\{\widetilde{z}'_1/\widetilde{z}_1\} \Rightarrow_f (\widetilde{x}\widetilde{z}'_1\widetilde{z}'_2)\,P''\{\widetilde{z}'_1\widetilde{z}'_2/\widetilde{z}_1\widetilde{z}_2\}$	(old-intr), since $\widetilde{x}_1\widetilde{z}'_1 \cap \widetilde{z}'_2 = \emptyset$

The result follows from transitivity of the first induction hypothesis, and this final inference. In the inference marked (*) we assume that the substitutions $\{\widetilde{z}'_2/\widetilde{z}_2\}\{\widetilde{z}'_1/\widetilde{z}_1\}$ can be re-ordered. To achieve this, first prove the special case of Part 5 where all names $\widetilde{z}'$ are completely fresh, and so (*) is satisfied. Then deduce the general Part 5 by applying this special case once to rename to the completely fresh $\widetilde{z}'$, and a second time to $\widetilde{z}''$ which are then allowed to clash with the original $\widetilde{z}$.

Part 6 is a straightforward induction on the derivation of $\Rightarrow_f$. □

Lemma 8

1. *(Old-alpha)* $(\widetilde{x})\,P \approx_f (\widetilde{x}')\,P\{\widetilde{x}'/\widetilde{x}\}$ *where* $\widetilde{x}' \notin \mathrm{fn}(\nu\widetilde{x}.P)$ *and* $\widetilde{x}'$ *are pairwise distinct.*
2. *(Old-for-new)* $()\,\nu x.P \approx_f (x')\,P\{x'/x\}$ *where* $x \notin \mathrm{fn}(R)$.
3. *(Old-intr) If* $(\widetilde{x})\,P \approx_f (\widetilde{y})\,Q$ *then* $(\widetilde{x}\widetilde{z})\,P \approx_f (\widetilde{y}\widetilde{z})\,Q$ *for* $\widetilde{z} \cap \widetilde{x}\widetilde{y} = \emptyset$.
4. *(Old-elim) If* $y \notin \mathrm{fn}\,P$ *then* $(\widetilde{x}y)\,P \approx_f (\widetilde{x})\,P$.
5. *(Structural) If* $P \equiv_\pi Q$ *then* $()\,P \approx_f ()\,Q$.

Proof. For **Part 1**, by the Context Lemma it is enough to prove $(\widetilde{x})\,P\sigma|R \mathrel{\dot{\approx}_f} (\widetilde{x}')\,P\{\widetilde{x}'/\widetilde{x}\}\sigma|R$ where $\{\widetilde{x}\widetilde{x}'\} \cap (\mathrm{dom}(\sigma) \cup \mathrm{fn}(R)) = \emptyset$. This degenerates to proving just $(\widetilde{x})\,P \mathrel{\dot{\approx}_f} (\widetilde{x}')\,P\{\widetilde{x}'/\widetilde{x}\}$. Construct $\mathcal{S} = \{((\widetilde{x})\,P, (\widetilde{x}')\,P\{\widetilde{x}'/\widetilde{x}\})\}$ for all $\widetilde{x}, \widetilde{x}', P$ such that $\widetilde{x}' \notin \mathrm{fn}(\nu\widetilde{x}.P)$ and $\widetilde{x}$ is pairwise distinct. If the left hand side makes a reaction $(\widetilde{x})\,P \rightarrow_f (\widetilde{x})\,P'$ then it is matched exactly by $(\widetilde{x}')\,P\{\widetilde{x}'/\widetilde{x}\} \rightarrow_f (\widetilde{x}')\,P'\{\widetilde{x}/\widetilde{x}\}$. If the left hand side makes a heating step

$$(\nu\widetilde{x})\,\nu z.P|R \;\rightharpoonup_f\; (\widetilde{x}z')\,P\{z'/z\}|R \tag{1}$$

for any $z' \notin \widetilde{x} \cup \mathrm{fn}(\nu z.P|R)$. This is matched on the right hand side by

$$(\nu\widetilde{x}')\,\nu z.P\{\widetilde{x}'/\widetilde{x}\}|R \;\rightharpoonup_f\; (\widetilde{x}'z'')\,P\{\widetilde{x}'/\widetilde{x}\}\{z''/z\}|R \tag{2}$$

for all $z'' \notin \widetilde{x}' \cup \mathrm{fn}(\nu z.P|R)$. But notice that the right hand side of Equation 2 is obtained by the substitution $\{^{z''}/_{z}\}$ on that of Equation 1, and hence they are related in $\mathcal{S}$.

As for reactions on the right hand side of $\mathcal{S}$ being matched by those on the left, the left is obtained by substituting $\{^{\widetilde{x}}/_{\widetilde{x}'}\}$ on the right and so the same argument applies.

For **Part 2**, via the Context Lemma, it is enough to prove over contexts $\_\sigma|R$. Without loss of generality, suppose this σ did not involve x or x'. Construct $\mathcal{S} = \{((\!|)\, \nu x.P\sigma|R, (\!|x'|\!)\, P\sigma\{^{x'}/_{x}\}|R)\} \cup \dot{\approx}_f$ for all $x' \notin \mathrm{fn}\, R$. It is clearly a $\dot{\approx}_f$: any transition the right can make, the left can too after first doing

$$(\!|)\, \nu x.P\sigma|R \rightharpoonup_f (\!|x'|\!)\, P\sigma\{^{x'}/_{x}\}|R.$$

Any transition the left can make is either from R (in which case the results are matched directly); or else it is a heating transition

$$(\!|)\, \nu x.P\sigma|R \rightharpoonup_f (\!|x''|\!)\, P\sigma\{^{x''}/_{x}\}|R.$$

And this version with x'' is bisimilar to the version with x', by Part 1.

For **Part 3**, by the Context Lemma, it is enough to prove $(\!|\widetilde{x}\widetilde{z}|\!)\, P\sigma|R \;\dot{\approx}_f\; (\!|\widetilde{y}\widetilde{z}|\!)\, Q\sigma|R$. Without loss of generality assume $\widetilde{z}\widetilde{x}\widetilde{y} \cap \mathrm{fn}\, R = \emptyset$, and $|\widetilde{z}| > 0$. Take $\sigma = \{^{\widetilde{u}}/_{\widetilde{v}}\}$. By assumption, $(\!|\widetilde{x}|\!)\, P \;\dot{\approx}_f\; (\!|\widetilde{y}|\!)\, Q$ in all contexts C. In particular, put $C = \nu\widetilde{z}.(\overline{z}_1\widetilde{u}|z_1(\widetilde{v}).\_|R)$. By Part 2,

$$(\!|\widetilde{x}|\!)\, \nu\widetilde{z}.(\overline{z}_1\widetilde{u}|z_1(\widetilde{v}).P|R) \;\dot{\approx}_f\; (\!|\widetilde{x}\widetilde{z}|\!)\, \overline{z}_1\widetilde{u}|z_1(\widetilde{v}).P|R$$

and likewise for Q. Since $z_1 \notin \mathrm{fn}\, R$ this term is bisimilar to its reacted form:

$$(\!|\widetilde{x}\widetilde{z}|\!)\, \overline{z}_1\widetilde{u}|z_1(\widetilde{v}).P|R \;\dot{\approx}_f\; (\!|\widetilde{x}\widetilde{z}|\!)\, P\{^{\widetilde{u}}/_{\widetilde{v}}\}|R$$

and likewise for P. Because $(\!|\widetilde{x}|\!)\, P \;\dot{\approx}_f\; (\!|\widetilde{y}|\!)\, Q$ in the initial context C, and following the chain of bisimulations for P and Q, we get

$$(\!|\widetilde{x}\widetilde{z}|\!)\, P\sigma|R \;\dot{\approx}_f\; (\!|\widetilde{y}\widetilde{z}|\!)\, Q\sigma|R$$

as desired.

For **Part 4**, by the Context Lemma, it is enough to prove $(\!|\widetilde{x}y|\!)\, P\sigma|R \;\dot{\approx}_f\; (\!|\widetilde{x}|\!)\, P\sigma|R$. This degenerates to proving just $(\!|\widetilde{x}y|\!)\, P \;\dot{\approx}_f\; (\!|\widetilde{x}|\!)\, P$. Construct

$$\mathcal{S} = \{((\!|\widetilde{x}y|\!)\, P, B) : B \;\dot{\approx}_f\; (\!|\widetilde{x}|\!)\, P \text{ and } y \notin \widetilde{x} \cup \mathrm{fn}(P)\}.$$

Suppose the left side makes a transition $(\!|\widetilde{x}y|\!)\, P \Rightarrow_f (\!|\widetilde{x}y\widetilde{z}|\!)\, P'$. By Lemma 7.2 *(old-elim)*, $(\!|\widetilde{x}|\!)\, P \Rightarrow_f (\!|\widetilde{x}\widetilde{z}|\!)\, P'$. Because $B \;\dot{\approx}_f\; (\!|\widetilde{x}|\!)\, P$ then $B \Rightarrow_f B' \;\dot{\approx}_f\; (\!|\widetilde{x}\widetilde{z}|\!)\, P'$. Hence $(\!|\widetilde{x}y\widetilde{z}|\!)\, P'\; \mathcal{S}\; B'$ as required.

Suppose the left side makes an observation $(\!|\widetilde{x}y|\!)\, P \downarrow_f u$. This comes from $P \downarrow u$ with $u \notin \widetilde{x}y$. Hence also $(\!|\widetilde{x}|\!)\, P \downarrow_u$ as required.

The reverse direction is more difficult. That is because a transition $(\!|\widetilde{x}|\!)\, P \Rightarrow_f (\!|\widetilde{x}\widetilde{z}|\!)\, P'$ cannot necessarily be matched by $(\!|\widetilde{x}y|\!)\, P$ (in the case that $y \in \widetilde{z}$). Instead we work up to bisimulation. Suppose the right side makes a transition $B \Rightarrow_f B'$. The following commutative diagrams applies:

$$\begin{array}{lll}
B & \Rightarrow_f & B' \\
\dot{\approx}_f & & \dot{\approx}_f \quad \textit{by construction } B \dot{\approx}_f (\!|\widetilde{x}|\!)\, P \\
(\!|\widetilde{x}|\!)\, P & \Rightarrow_f & (\!|\widetilde{x}\widetilde{z}|\!)\, P' \\
= & & \dot{\approx}_f \quad \textit{Lemma 7.5 (new-alpha) and Part 4 (old-alpha)} \\
(\!|\widetilde{x}|\!)\, P & \Rightarrow_f & (\!|\widetilde{x}\widetilde{z}'|\!)\, P'\{\widetilde{z}'/\widetilde{z}\} \quad \textit{for some } \widetilde{z}' : \widetilde{z} \cap (\widetilde{x}y \cup \operatorname{fn} P) = \emptyset \\
 & & \quad \textit{Lemma 7.1 (old-intr) since } y \notin \widetilde{x}\widetilde{z}' \\
(\!|\widetilde{x}y|\!)\, P & \Rightarrow_f & (\!|\widetilde{x}y\widetilde{z}'|\!)\, P'\{\widetilde{z}'/\widetilde{z}\}
\end{array}$$

And because $B' \dot{\approx}_f (\!|\widetilde{x}\widetilde{z}'|\!)\, P'\{\widetilde{z}'/\widetilde{z}\}$ then $(\!|\widetilde{x}y\widetilde{z}'|\!)\, P'\{\widetilde{z}'/\widetilde{z}\}\ \mathcal{S}\ B'$ as required.

Suppose the right side makes an observation $B \downarrow_f u$. By construction $B \dot{\approx}_f (\!|\widetilde{x}|\!)\, P$ then also $(\!|\widetilde{x}|\!)\, P \Downarrow_f u$, or in other words

$$(\!|\widetilde{x}|\!)\, P \Rightarrow_f (\!|\widetilde{x}\widetilde{z}|\!)\, P' \downarrow_f u$$

with $P' \downarrow u$ and $u \notin \widetilde{x}\widetilde{z}$. By Lemma 7.5 (new-alpha),

$$(\!|\widetilde{x}|\!)\, P \Rightarrow_f (\!|\widetilde{x}\widetilde{z}'|\!)\, P'\{\widetilde{z}'/\widetilde{z}\}$$

for some $\widetilde{z}'$ such that $\widetilde{z}' \cap (\widetilde{x}y \cup \operatorname{fn} P) = \emptyset$. By Lemma 7.1 (old-intr),

$$(\!|\widetilde{x}y|\!)\, P \Rightarrow_f (\!|\widetilde{x}y\widetilde{z}'|\!)\, P'\{\widetilde{z}'/\widetilde{z}\}.$$

From above, $P' \downarrow u$ and $u \notin \widetilde{z}$. Hence also $P'\{\widetilde{z}'/\widetilde{z}\} \downarrow u$. Now prove that $u \neq y$ as follows. Given the reduction, then $\operatorname{fn}(\nu\widetilde{z}'.P'\{\widetilde{z}'/\widetilde{z}\}) \subseteq \operatorname{fn}(P)$. By assumption, $y \not\in \operatorname{fn}(P)$. Hence either $y \in \widetilde{z}'$ or $y \notin \operatorname{fn}(P'\{\widetilde{z}'/\widetilde{z}\})$. From above, the first does not hold; therefore the second does; therefore $y \neq u$.

Thus it is established that $u \notin \widetilde{x}y\widetilde{z}'$ and $P'\{\widetilde{z}'/\widetilde{z}\} \downarrow u$. Hence $(\!|\widetilde{x}y\widetilde{z}'|\!)\, P'\{\widetilde{z}'/\widetilde{z}\} \downarrow u$ as required.

For **Part 5**, prove by induction on the derivation of $P \equiv_\pi Q$.

1. For the commutativity and associativity of $|$ with $\mathbf{0}$ as identity, these rules are present also in fresh pi.
2. For $\nu x.\nu y.P \equiv_\pi \nu y.\nu x.P$, use Part 2, and recall that the fresh names $(\!|xy|\!)$ constitute a set and so can be reordered:
$$(\!|\,|\!)\, \nu x.\nu y.P \approx_f (\!|xy|\!)\, P \approx_f (\!|yx|\!)\, P \approx_f (\!|\,|\!)\, \nu y.\nu x.P$$
3. For $\nu x.(P|Q) \equiv_\pi P|\nu x.Q$ with $x \notin \operatorname{fn} P$, Part 2 says
$$(\!|\,|\!)\, \nu x.Q \approx_f (\!|x|\!)\, Q.$$

This is a congruence in all contexts whose free names do not clash with x, so apply the context $P|\_$:

$$(\!|\,|\!)\, P|\nu x.Q \approx_f (\!|x|\!)\, P|Q.$$

Finally, Part 2 on the right hand side gives $\approx_f \nu x.(P|Q)$.

4. For $\nu y.P \equiv_\pi \nu x.P\{x/y\}$ with $x \notin \mathrm{fn}\, P$, Part 1 gives $(\!|y|\!)\, P \approx_f (\!|x|\!)\, P\{x/y\}$. Applying Part 2 to both sides yields the result.
5. For $\nu x.\mathbf{0} \equiv_\pi \mathbf{0}$, this follows from Part 4.
6. Symmetry and transitivity of $\equiv_\pi$ are matched by symmetry and transitivity of $\approx_f$. Closure under contexts $\nu x.\_$ and $\_|Q$ is matched by closure under contexts of $\approx_f$.
7. For $!P \equiv_\pi P|!P$, by the Context Lemma and without loss of generality it is enough to prove

$$(\!|\,|\!)\, !u(\widetilde{x}).P|R \mathrel{\dot\approx_f} (\!|\,|\!)\, u(\widetilde{x}).P|!u(\widetilde{x}).P|R.$$

Therefore construct the smallest $\mathcal{S}$ which contains these two terms and also $\mathrel{\dot\approx_f}$, and prove it is a $\mathrel{\dot\approx_f}$. Any transitions made by R alone are trivially matched. Consider the other possibilities.
On the left, the only other possible reaction is when $R = \overline{u}\,\widetilde{y}.R_1|R_2$, giving

$$(\!|\,|\!)\, !u(\widetilde{x}).P \mid \overline{u}\,\widetilde{y}.R_1 \mid R_2 \;\rightarrow_f\; (\!|\,|\!)\, P\{\widetilde{y}/\widetilde{x}\} \mid !u(\widetilde{x}).P \mid R_1 \mid R_2. \tag{3}$$

On the right, this can be matched by reacting with the 'already-unwrapped' copy of the replicated input:

$$(\!|\,|\!)\, u(\widetilde{x}).P \mid !u(\widetilde{x}).P \mid \overline{u}\,\widetilde{y}.R_1 \mid R_2 \;\rightarrow_f\; (\!|\,|\!)\, P\{\widetilde{y}/\widetilde{x}\} \mid !u(\widetilde{x}).P \mid R_1 \mid R_2.$$

The results are identical, and hence in $\mathrel{\dot\approx_f}$.
On the right, reaction again requires $R = \overline{u}\,\widetilde{y}.R_1|R_2$, and there are two possible reactions that the right can make:

$$(\!|\,|\!)\, u(\widetilde{x}).P \mid !u(\widetilde{x}).P \mid \overline{u}\,\widetilde{y}.R_1 \mid R_2 \;\rightarrow_f\; (\!|\,|\!)\, P\{\widetilde{y}/\widetilde{x}\} \mid !u(\widetilde{x}).P \mid R_1 \mid R_2$$
$$(\!|\,|\!)\, u(\widetilde{x}).P \mid !u(\widetilde{x}).P \mid \overline{u}\,\widetilde{y}.R_1 \mid R_2 \;\rightarrow_f\; (\!|\,|\!)\, P\{\widetilde{y}/\widetilde{x}\} \mid u(\widetilde{x}).P \mid !u(\widetilde{x}).P \mid R_1 \mid R_2.$$

The left makes the reaction given in Equation 3. The results are either related directly by $\mathrel{\dot\approx_f}$ or at least by $\mathcal{S}$.

This completes the proof for Part 4. □

Proposition 9 (Between fresh pi, and the pi calculus)

1. *$P \Rightarrow_\pi P'$ implies $(\!|\,|\!)\, P \Rightarrow_f \approx_f (\!|\,|\!)\, P'$*
2. *$(\!|\widetilde{x}|\!)\, P \Rightarrow_f (\!|\widetilde{x}\widetilde{z}|\!)\, P'$ implies $\nu\widetilde{x}.P \Rightarrow_\pi \nu\widetilde{x}\widetilde{z}.P'$*
3. *$P \downarrow_\pi u$ implies $(\!|\,|\!)\, P \Downarrow_f u$*
4. (Bisimulation) *$\nu\widetilde{x}.P \mathrel{\dot\approx_\pi} \nu\widetilde{y}.Q$ if and only if $(\!|\widetilde{x}|\!)\, P \mathrel{\dot\approx_f} (\!|\widetilde{y}|\!)\, Q$.*

5. (Full abstraction) $\nu\widetilde{x}.P \approx_\pi \nu\widetilde{y}.Q$ *if and only if* $(\!|\widetilde{x}|\!)\, P \approx_f (\!|\widetilde{y}|\!)\, Q$.

Proof. **Part 1** is by induction on the derivation of $\Rightarrow_\pi$:

1. The base case is $\overline{u}\,\widetilde{y}.P \mid u(\widetilde{x}).Q \rightarrow_\pi P \mid Q\{\widetilde{y}/\widetilde{x}\}$. This is directly matched in the fresh pi calculus.
2. For restriction, the derivation was

$$\frac{P \rightarrow_\pi P'}{\nu x.P \rightarrow_\pi \nu x.P'}$$

   The induction hypothesis gives

$$(\!|\,|\!)\, P \Rightarrow_f (\!|\widetilde{z}|\!)\, P'' \approx_f (\!|\,|\!)\, P'.$$

   By Lemma 7.4 *(new-alpha)* assume $x \notin \widetilde{z}$. Then by Lemma 7.1 *(old-intr)* and Lemma 8.3 *(old-contexts)* and Lemma 8.2 *(old-for-new)*,

$$(\!|\,|\!)\, \nu x.P \rightharpoonup_f (\!|x|\!)\, P \Rightarrow_f (\!|x\widetilde{z}|\!)\, P'' \approx_f (\!|x|\!)\, P \approx_f (\!|\,|\!)\, \nu x.P.$$

3. For parallel, the derivation was

$$\frac{P \rightarrow_\pi P'}{P|Q \rightarrow_\pi P'|Q}$$

   The induction hypothesis is again as in Equation 2. In the following use Lemma 7.4 *(new-alpha)* to assume $\widetilde{z}\,\mathrm{fn}(Q) = \emptyset$, and the context-closure of $\approx_f$:

$$(\!|\,|\!)\, P|Q \Rightarrow_f (\!|\widetilde{z}|\!)\, P''|Q \approx_f (\!|\,|\!)\, P'|Q.$$

4. For structural congruence, the derivation was

$$\frac{P \equiv_\pi Q \rightarrow_\pi Q' \equiv_\pi P'}{P \rightarrow_\pi P'}$$

   The induction hypothesis is again as in Equation 2. By Lemma 8.4,

$$(\!|\,|\!)\, P \approx_f (\!|\,|\!)\, Q \Rightarrow_f (\!|\widetilde{z}|\!)\, Q'' \approx_f (\!|\,|\!)\, Q' \approx_f (\!|\,|\!)\, P'.$$

   Hence $(\!|\,|\!)\, P \Rightarrow_f\!\approx_f (\!|\,|\!)\, P'$.
5. For transitive closure, the derivation was

$$\frac{P \Rightarrow_\pi P' \quad P' \Rightarrow_\pi P''}{P \Rightarrow_\pi P''}$$

   The induction hypotheses give

$$(\!|\,|\!)\, P \Rightarrow_f\!\approx_f (\!|\,|\!)\, P' \Rightarrow_f\!\approx_f (\!|\,|\!)\, P''.$$

   Hence $(\!|\,|\!)\, P \Rightarrow_f\!\approx_f (\!|\,|\!)\, P''$ as desired. This concludes the proof of Part 1.

Part 2 is by induction on the derivation of $(\widetilde{x})\, P \Rightarrow_f (\widetilde{x}\widetilde{z})\, P'$. There are four cases:

1. The fresh pi heating step

$$(\widetilde{x})\, R|\nu z.P \rightharpoonup_f (\widetilde{x}z')\, R|P\{z'/z\}$$

with $z' \notin \{\widetilde{z}\} \cup \mathrm{fn}(R|\nu x.P)$; this is matched in the pi calculus by

$$\nu\widetilde{x}.(R|\nu z.P) \equiv_\pi \nu\widetilde{x}.\nu z'.(R|P\{z'/z\}).$$

2. The fresh pi reaction step

$$(\widetilde{x})\, \overline{u}\,\widetilde{z}.P|u(\widetilde{y}).Q|R \rightarrow_f (\widetilde{x})\, P|Q\{\widetilde{z}/\widetilde{y}\}|R$$

is matched in the pi calculus by

$$\nu\widetilde{x}.(\overline{u}\,\widetilde{z}.P|u(\widetilde{y}).Q|R) \rightarrow_\pi \nu\widetilde{x}.(P|Q\{\widetilde{z}/\widetilde{y}\}|R).$$

3. Replicated reaction is similar.
4. Transitivity in fresh pi reduction is derived from

$$\frac{(\widetilde{x})\, P \Rightarrow_f (\widetilde{x}\widetilde{z}_1)\, P' \qquad (\widetilde{x}\widetilde{z}_1)\, P' \Rightarrow_f (\widetilde{x}\widetilde{z}_2)\, P''}{(\widetilde{x})\, P \Rightarrow_f (\widetilde{x}\widetilde{z}_1\widetilde{z}_2)\, P''}$$

By the induction hypotheses,

$$\nu\widetilde{x}.P \Rightarrow_\pi \nu\widetilde{x}\widetilde{z}_1.P' \Rightarrow_\pi \nu\widetilde{x}\widetilde{z}_1\widetilde{z}_2.P''$$

and hence $\nu\widetilde{x}.P \Rightarrow_\pi \nu\widetilde{x}\widetilde{z}_1\widetilde{z}_2 P''$ as desired. This completes the proof of Part 2.

Part 3 is by induction on the derivation of $P \downarrow_\pi u$. The only non-trivial case is *new*. In this case the pi derivation is

$$\frac{P \downarrow_\pi u \quad u \neq x}{\nu x.P \downarrow_\pi u}$$

The induction hypothesis is $(\,)\, P \Downarrow_f u$, or equivalently $(\,)\, P \Rightarrow_f (\widetilde{z})\, P' \downarrow u$ with $P' \downarrow u$ and $u \notin \widetilde{z}$. Assuming x' not to clash, and by Lemma 7.1 *(old-intr)* and Lemma 7.4 *(subst)*,

$$(\,)\, \nu x.P \rightharpoonup_f (x')\, P\{x'/x\} \Rightarrow_f (x\widetilde{z})\, P'\{x'/x\}.$$

Since $P' \downarrow u$ and $u \neq x$ then $P'\{x'/x\} \downarrow u$. This yields $(\,)\, \nu x.P \Rightarrow_f \downarrow u$as desired.

For Part 4, in the forwards direction, construct $\mathcal{S} = \{(\nu\widetilde{x}.P, \nu\widetilde{y}.Q) : (\widetilde{x})\, P \mathrel{\dot{\approx}}_f (\widetilde{y})\, Q\}$ and prove that it is a $\mathrel{\dot{\approx}}_\pi$:

(1) Suppose $\nu\widetilde{x}.P \rightarrow_\pi P'$. Hence

$(\!\mid) \nu\widetilde{x}.P \Rightarrow_f \dot{\approx}_f (\!\mid) P'$	*Part 1*
$(\widetilde{x}) P \Rightarrow_f \dot{\approx}_f (\!\mid) P'$	*Lemma 7.7, (old-for-new)*
$(\widetilde{y}) Q \Rightarrow_f (\widetilde{y}\widetilde{z}) Q' \dot{\approx}_f (\!\mid) P'$	*by construction of* $\mathcal{S}$
$\nu\widetilde{y}.Q \Rightarrow_\pi \nu\widetilde{y}\widetilde{z}.Q'$	*Part 2*

From $(\widetilde{y}\widetilde{z}) Q' \dot{\approx}_f (\!\mid) P'$, and by construction of $\mathcal{S}$, then $P' \; \mathcal{S} \; \nu\widetilde{y}\widetilde{z}.Q'$ as required.

(2) Suppose $\nu\widetilde{x}.P \downarrow_\pi u$. Hence

$(\!\mid) \nu\widetilde{x}.P \Downarrow_f u$	*Part 3*
$(\widetilde{x}) P \Downarrow_f u$	*Lemma 7.7, (old-for-new)*
$(\widetilde{y}) Q \Downarrow_f u$	*by construction of* $\mathcal{S}$

This last must be deduced from $(\widetilde{y}) Q \Rightarrow_f (\widetilde{y}\widetilde{z}) Q' \downarrow u$ with $Q' \downarrow u$ and $u \notin \widetilde{y}\widetilde{z}$. By Part 2 also $\nu\widetilde{y}.Q \Rightarrow_\pi \nu\widetilde{y}\widetilde{z}.Q' \downarrow u$ as required.

In the reverse direction of Part 4, we must construct a more complicated relation $\mathcal{S}$. That is because some structural congruences in the pi calculus, such as $!P \equiv P|!P$, are matched only by $\dot{\approx}_f$ in the fresh calculus. Hence we incorporate $\dot{\approx}_f$ into $\mathcal{S}$:

$$\mathcal{S} \;=\; \left\{ (A,B) : \; A \dot{\approx}_f (\nu\widetilde{x}).P, \quad B \dot{\approx}_f (\nu\widetilde{y}).Q, \quad \nu\widetilde{x}.P \dot{\approx}_\pi \nu\widetilde{y}.Q \right\}$$

We now prove that $\mathcal{S}$ is a $\dot{\approx}_f$. Assume $(A,B) \in \mathcal{S}$ with $A \dot{\approx}_f (\widetilde{x}) P$ and $B \dot{\approx}_f (\widetilde{y}) Q$ and $\nu\widetilde{x}.P \dot{\approx}_\pi \nu\widetilde{y}.Q$:

(1) Suppose $A \rightarrow_f A'$ or $A \rightharpoonup_f A'$. Then

$(\widetilde{x}) P \Rightarrow_f (\widetilde{x}\widetilde{z}) P' \dot{\approx}_f A'$	*since* $A \dot{\approx}_f (\widetilde{x}) P$
$\nu\widetilde{x}.P \Rightarrow_\pi \nu\widetilde{x}\widetilde{z}.P'$	*Part 2*
$\nu\widetilde{y}.Q \Rightarrow_\pi Q' \dot{\approx}_\pi \nu\widetilde{x}\widetilde{z}.P'$	*by assumption* $\nu\widetilde{x}.P \dot{\approx}_\pi \nu\widetilde{y}.Q$
$(\!\mid) \nu\widetilde{y}.Q \Rightarrow_f \dot{\approx}_f (\!\mid) Q'$	*Part 1*
$(\widetilde{y}) Q \Rightarrow_f B' \dot{\approx}_f (\!\mid) Q'$	*Lemma 7.7, (old-for-new)*

Given $A' \dot{\approx}_f (\widetilde{x}\widetilde{z}) P'$ and $B' \dot{\approx}_f (\!\mid) Q'$ and $\nu\widetilde{x}\widetilde{z}.P' \dot{\approx}_\pi Q'$, then $A' \; \mathcal{S} \; B'$ as required.

(2) Suppose $A \downarrow u$. Then

$(\widetilde{x}) P \Rightarrow_f (\widetilde{x}\widetilde{z}) P' \downarrow u$	*since* $A \dot{\approx}_f (\widetilde{x}) P$
$\nu\widetilde{x}.P \Rightarrow_\pi \nu\widetilde{x}\widetilde{z}.P' \downarrow u$	*Part 2*
$\nu\widetilde{y}.Q \Rightarrow_\pi Q' \downarrow u$	*since* $\nu\widetilde{x}.P \dot{\approx}_\pi \nu\widetilde{y}.Q$
$(\!\mid) \nu\widetilde{y}.Q \Rightarrow_f B' \dot{\approx}_f (\!\mid) Q' \downarrow u$	*Part 1*
$(\widetilde{y}) Q \Rightarrow_f \downarrow u$	*Lemma 7.7, (old-for-new)*

This concludes the proof of Part 4.

For Part 5 reverse direction we prove that $(\!|)\, P \not\approx_f (\!|)\, Q$ implies $P \not\approx_\pi Q$. In particular, assume there exists a C such that $(\widetilde{x})\, C[P] \,\dot{\not\approx}_f\, (\widetilde{y})\, C[Q]$. By Part 4 then $\nu\widetilde{x}.C[P] \,\dot{\not\approx}_\pi\, \nu\widetilde{y}.C[Q]$ as required.

For Part 5 forwards direction we prove that $P \not\approx_\pi Q$ implies $(\!|)\, P \not\approx_f (\!|)\, Q$. In particular, assume there exists a C such that $C[P] \,\dot{\not\approx}_\pi\, C[Q]$. By the Context Lemma, there exists R, σ such that $(\nu\widetilde{x}.P)\sigma|R \,\dot{\not\approx}_\pi\, (\nu\widetilde{y}.Q)\sigma|R$. Without loss of generality, assume that $(\widetilde{x}, \widetilde{y})$ does not clash with the domain of σ or the free names of R. Hence

$$\nu\widetilde{x}.P\sigma \mid R \,\dot{\not\approx}_\pi\, \nu\widetilde{y}.Q\sigma \mid R.$$

By Part 4,

$$(\!|)\, \nu\widetilde{x}.P\sigma \mid R \,\dot{\not\approx}_f\, (\!|)\, \nu\widetilde{y}.Q\sigma \mid R.$$

By Lemma 7.7 *(old-for-new)*, the 'new' names $\widetilde{x}$ and $\widetilde{y}$ can be turned into 'old' names $(\widetilde{x})$ and $(\widetilde{y})$ up to $\dot{\approx}_f$. Hence

$$(\widetilde{x})\, P\sigma \mid R \,\dot{\not\approx}_f\, (\widetilde{y})\, Q\sigma \mid R$$

as required. □

Lemma 10 (Between fresh pi, and fresh pi up to $\widetilde{v}$)

1. $(\widetilde{x})\, P \stackrel{\widetilde{v}}{\Rightarrow}_f (\widetilde{x}\widetilde{z})\, P'$ *implies* $(\widetilde{x})\, P \Rightarrow_f (\widetilde{x}\widetilde{z})\, P'$
2. $(\widetilde{x})\, P \Rightarrow_f (\widetilde{x}\widetilde{z})\, P'$ *implies for every* $\widetilde{v}$ *then* $(\widetilde{x})\, P \stackrel{\widetilde{v}}{\Rightarrow}_f \dot{\approx}_f (\widetilde{x}\widetilde{z})\, P'$
3. $(\widetilde{x})\, P \Downarrow_f^{\widetilde{v}} u$ *implies* $(\widetilde{x})\, P \Downarrow_f u$
4. $(\widetilde{x})\, P \Downarrow_f u$ *and* $u \in \widetilde{v}$ *implies* $(\widetilde{x})\, P \Downarrow_f^{\widetilde{v}} u$
5. *(Bisimulation) If* $(\widetilde{x})\, P \;\dot{\approx}_f\; (\widetilde{y})\, Q$ *then* $(\widetilde{x})\, P \;\stackrel{\widetilde{v}}{\approx}_f\; (\widetilde{y})\, Q$ *for all* $\widetilde{v}$. *And if* $(\widetilde{x})\, P \stackrel{\widetilde{v}}{\approx}_f (\widetilde{y})\, Q$ *for some* $\widetilde{v} \supseteq \mathrm{fn}(\nu\widetilde{x}.P|\nu\widetilde{y}.Q)$, *then* $(\widetilde{x})\, P \,\dot{\approx}_f\, (\widetilde{y})\, Q$.
6. *(Full abstraction) If* $(\widetilde{x})\, P \approx_f (\widetilde{y})\, Q$ *then for all contexts* C *and all* $\widetilde{v}$ *then* $(\widetilde{x})\, C[P] \;\stackrel{\widetilde{v}}{\approx}_f\; (\widetilde{y})\, C[Q]$. *And if for all contexts* C *there exists some* $\widetilde{v} \supseteq \mathrm{fn}(\nu\widetilde{x}.C[P]|\nu\widetilde{y}.C[Q])$ *such that* $(\widetilde{x})\, C[P] \stackrel{\widetilde{v}}{\approx}_f (\widetilde{y})\, C[Q]$, *then* $(\widetilde{x})\, P \approx_f (\widetilde{y})\, Q$.

Proof. **Parts 1 and 3** are trivial, since the definitions of $\stackrel{\widetilde{v}}{\Rightarrow}_f$ and $\Downarrow_f^{\widetilde{v}}$ are a subset of those of $\Rightarrow_f$ and $\Downarrow_f$.

For Part 2, suppose $(\widetilde{x})\, P \Rightarrow_f (\widetilde{x}\widetilde{z})\, P'$. By Lemma 7.5 *(new-alpha)*, and picking $\widetilde{z}'$ which does not clash with $\widetilde{x}\widetilde{v}$ or $\mathrm{fn}(P)$,

$$(\widetilde{x})\, P \Rightarrow_f (\widetilde{x}\widetilde{z}')\, P'\{\widetilde{z}'/\widetilde{z}\}. \tag{4}$$

The new names created in this transition do not clash with $\widetilde{z}$, and hence it is also a $\stackrel{\widetilde{v}}{\Rightarrow}_f$ transition. Finally, by Lemma 8.1 *(old-alpha)*, $(\widetilde{x}\widetilde{z})\, P' \;\dot{\approx}_f\; (\widetilde{x}\widetilde{z}')\, P'\{\widetilde{z}'/\widetilde{z}\}$ as required.

For Part 4, suppose $P \Downarrow_f u$. This means $(\!|\widetilde{x}|\!)\, P \Rightarrow_f (\!|\widetilde{x}\widetilde{z}|\!)\, P' \downarrow_f u$, with $P' \downarrow u$ and $u \notin \widetilde{x}\widetilde{z}$. From assumption $u \in \widetilde{v}$ then also $P' \downarrow^{\widetilde{v}} u$. Now derive Equation 4 and $\widetilde{z}'$ as before. Since $u \notin \widetilde{z}$ then $P'\{\widetilde{z}'/\widetilde{z}\} \downarrow^{\widetilde{v}} u$. Given $u \in \widetilde{v}$ and $\widetilde{z}' \cap \widetilde{v} = \emptyset$ then $u \notin \widetilde{z}'$. Also $u \notin \widetilde{x}$ as above. Hence $(\!|\widetilde{x}\widetilde{z}'|\!)\, P'\{\widetilde{z}'/\widetilde{z}\} \downarrow_f^{\widetilde{v}} u$ as required.

For Part 5 forwards direction, construct $\mathcal{S}$ as follows and prove it is a $\overset{\widetilde{v}}{\dot{\approx}}_f$ for every $\widetilde{v}$.

$$\mathcal{S} = \{((\!|\widetilde{x}|\!)\, P, (\!|\widetilde{y}|\!)\, Q) : (\!|\widetilde{x}|\!)\, P \dot{\approx}_f (\!|\widetilde{y}|\!)\, Q\}.$$

The following commutative diagram proves that transitions are matched by left and right hand sides:

$(\!\|\widetilde{x}\|\!)\, P$	$\overset{\widetilde{v}}{\Rightarrow}_f$	A'	
$=$		$=$	*Part 1*
$(\!\|\widetilde{x}\|\!)\, P$	$\Rightarrow_f$	A'	
$\dot{\approx}_f$		$\dot{\approx}_f$	*by construction*
$(\!\|\widetilde{y}\|\!)\, Q$	$\Rightarrow_f$	B'	
$=$		$\dot{\approx}_f$	*Part 2*
$(\!\|\widetilde{y}\|\!)\, Q$	$\overset{\widetilde{v}}{\Rightarrow}_f$	B''	

And $(A', B'') \in \mathcal{S}$ since $A' \dot{\approx}_f B''$.

As for observations, suppose $(\!|\widetilde{x}|\!)\, P \downarrow_f^{\widetilde{v}} u$. Hence $u \in \widetilde{v}$. By Part 3, $(\!|\widetilde{x}|\!)\, P \Downarrow_f u$. By construction, $(\!|\widetilde{y}|\!)\, Q \Downarrow_f u$. By Part 4 and because $u \in \widetilde{v}$, $(\!|\widetilde{y}|\!)\, Q \Downarrow_f^{\widetilde{v}} u$ as required.

For Part 5 reverse direction, it is a little more complicated for two reasons. First, transitions $\Rightarrow_f$ are only matched by $\overset{\widetilde{v}}{\Rightarrow}_f$ up to $\dot{\approx}_f$ (Part 2). We therefore construct $\mathcal{S}_{\widetilde{v}}$ only up to $\dot{\approx}_f$. Second, there may be names $v \notin \widetilde{v}$ which distinguish two terms up to $\dot{\approx}_f$, even though the terms are indistinguishable up to $\overset{\widetilde{v}}{\dot{\approx}}_f$. We therefore limit $\mathcal{S}_{\widetilde{v}}$ to just those terms that have free names in $\widetilde{v}$, and prove it is a $\dot{\approx}$:

$$\mathcal{S}_{\widetilde{v}} = \{(A, B) : A \dot{\approx}_f (\!|\widetilde{x}|\!)\, P \overset{\widetilde{v}}{\dot{\approx}}_f (\!|\widetilde{y}|\!)\, Q \dot{\approx}_f B \text{ such that } \mathrm{fn}(\nu\widetilde{x}.P|\nu\widetilde{y}.Q) \subseteq \widetilde{v}\}.$$

Note that when $(\!|\widetilde{x}|\!)\, P \overset{\widetilde{v}}{\Rightarrow}_f (\!|\widetilde{x}\widetilde{z}|\!)\, P'$ then $\mathrm{fn}(\nu\widetilde{x}\widetilde{z}.P') \subseteq \mathrm{fn}(\nu\widetilde{x}.P)$. This justifies constraint on the construction of $\mathcal{S}_{\widetilde{v}}$.

The following commutative diagram proves that transitions are matched by left and right sides of $\mathcal{S}_{\widetilde{v}}$:

A	$\Rightarrow_f$	A'	
$\dot{\approx}_f$		$\dot{\approx}_f$	*by construction* $A \dot{\approx}_f (\!\|\widetilde{x}\|\!)\, P$
$(\!\|\widetilde{x}\|\!)\, P$	$\Rightarrow_f$	A''	
$=$		$\dot{\approx}_f$	*by Part 2*

$$
\begin{array}{lll}
(\!|\widetilde{x}|\!)\,P & \stackrel{\widetilde{v}}{\Rightarrow}_f & A''' \qquad \textit{note: } \mathrm{fn}(A''') \subseteq \mathrm{fn}(\nu\widetilde{x}.P) \\
\stackrel{\widetilde{v}}{\approx}_f & & \stackrel{\widetilde{v}}{\approx}_f \qquad \textit{by construction } (\!|\widetilde{x}|\!)\,P \stackrel{\widetilde{v}}{\approx}_f (\!|\widetilde{y}|\!)\,Q \\
(\!|\widetilde{y}|\!)\,Q & \stackrel{\widetilde{v}}{\Rightarrow}_f & B'' \qquad \textit{note: } \mathrm{fn}(B'') \subseteq \mathrm{fn}(\nu\widetilde{y}.Q) \\
= & & = \qquad \textit{by Part 1} \\
(\!|\widetilde{y}|\!) & \Rightarrow_f & B'' \\
\dot{\approx}_f & & \dot{\approx}_f \qquad \textit{by construction } (\!|\widetilde{y}|\!)\,Q \,\dot{\approx}_f\, B \\
B & \Rightarrow_f & B'
\end{array}
$$

And by construction, $(A', B') \in \mathcal{S}_{\widetilde{v}}$.

As for observations, suppose $A \downarrow_f u$. Therefore $(\!|\widetilde{x}|\!)\,P \Downarrow_f u$. This must be from $u \in \mathrm{fn}(\nu\widetilde{x}.P)$, and hence $u \in \widetilde{v}$. With this and Part 4, $(\!|\widetilde{x}|\!)\,P \Downarrow_f^{\widetilde{v}} u$. By construction, $(\!|\widetilde{y}|\!)\,Q \Downarrow_f^{\widetilde{v}} u$. By Part 3, $(\!|\widetilde{y}|\!)\,Q \Downarrow_f u$. By construction, also $B \Downarrow_f u$ as required.

Part 6 is a trivial consequence of Part 5. □

Proposition 11 (Between global pi, and fresh pi)

1. $(\!|\widetilde{x}|\!)\,P \stackrel{\widetilde{v}}{\Rightarrow}_f (\!|\widetilde{x}\widetilde{z}|\!)\,P'$ *implies* $P \stackrel{\widetilde{v}}{\Rightarrow} P'$
2. $P \stackrel{\widetilde{v}}{\Rightarrow} P'$ *implies for all* $\widetilde{x}$ *there exists* $\widetilde{z}$ *such that* $(\!|\widetilde{x}|\!)\,P \stackrel{\widetilde{v}}{\Rightarrow}_f \stackrel{\widetilde{v}}{\approx}_f (\!|\widetilde{x}\widetilde{z}|\!)\,P'$ *and* $\widetilde{z} \cap \widetilde{v} = \emptyset$
3. $(\!|\widetilde{x}|\!)\,P \Downarrow_f^{\widetilde{v}} u$ *implies* $P \Downarrow^{\widetilde{v}} u$
4. $P \Downarrow^{\widetilde{v}} u$ *implies for all* $\widetilde{x}$ *such that* $\widetilde{x} \cap \widetilde{v} = \emptyset$ *then* $(\!|\widetilde{x}|\!)\,P \Downarrow_f^{\widetilde{v}} u$
5. (Bisimulation) $P \stackrel{\widetilde{v}}{\approx} Q$ *if and only if* $(\!|\,|\!)\,P \stackrel{\widetilde{v}}{\approx}_f (\!|\,|\!)\,Q$
6. (Full abstraction) $P \approx Q$ *if and only if* $(\!|\,|\!)\,P \approx_f (\!|\,|\!)\,Q$

Proof. **Part 1** is a trivial induction on the derivation of $\stackrel{\widetilde{v}}{\Rightarrow}_f$. **Part 3** follows trivially from Part 1.

Part 2 is by induction on the derivation of *wredvv*. The *react* and *replicated-react* base cases are trivial. The *new* base case is as follows. Assume $\nu z.P|R \stackrel{\widetilde{v}}{\rightarrow} P\{^{z'}/_z\}|R$ with $z' \notin \widetilde{v} \cup \mathrm{fn}(\nu z.P|R)$. The task is to match this with a transition from $(\!|\widetilde{x}|\!)\,\nu z.P|R$ for any $\widetilde{x}$. In the case where $z' \notin \widetilde{x}$ it is trivial:

$$(\!|\widetilde{x}|\!)\,\nu z.P|R \stackrel{\widetilde{v}}{\rightarrow}_f (\!|\widetilde{x}z'|\!)\,P\{^{z'}/_z\}|R.$$

In the case where $z' \in \widetilde{x}$, work as follows. Pick a z'' such that $z'' \notin z'\widetilde{x}\widetilde{v} \cup \mathrm{fn}(\nu z.P|R)$. Then

$$(\!|\widetilde{x}|\!)\,\nu z.P|R \stackrel{\widetilde{v}}{\rightarrow}_f (\!|\widetilde{x}z''|\!)\,P\{^{z''}/_z\}|R.$$

We have that $z' \notin \mathrm{fn}(\nu z.P|R)$ and $z' \in \widetilde{x}$. Hence apply Lemma 8.4 *(old-elim)* and Lemma 10.6 to remove z' from $\widetilde{x}$: writing $\widetilde{x}\backslash z'$ for the result of this removal,

$$(\!|\widetilde{x}z''|\!)\,P\{^{z''}/_z\}|R \stackrel{\widetilde{v}}{\approx}_f (\!|\widetilde{x}\backslash z', z''|\!)\,P\{^{z''}/_z\}|R.$$

Now use Lemma 8.1 *(old-alpha)* and Lemma 10.6 to substitute $\{z'/z''\}$:

$$(\widetilde{x}\backslash z', z'')\, P\{z''/z\}|R \overset{\widetilde{v}}{\approx}_f (\widetilde{x})\, P\{z'/z\}|R.$$

The *transitivity* case for Part 2 is as follows. Assume the derivation

$$\frac{P \overset{\widetilde{v}}{\Rightarrow} P' \qquad P' \overset{\widetilde{v}}{\rightarrow} P''}{P \overset{\widetilde{v}}{\rightarrow} P''}$$

The induction hypotheses for $P \overset{\widetilde{v}}{\Rightarrow} P'$ gives $(\widetilde{x})\, P \overset{\widetilde{v}}{\rightarrow}_f \overset{\widetilde{v}}{\approx}_f (\widetilde{x}\widetilde{z}_1)\, P'$, and for $P' \overset{\widetilde{v}}{\Rightarrow} P''$ it gives $(\widetilde{x}\widetilde{z}_1)\, P' \overset{\widetilde{v}}{\rightarrow}_f \overset{\widetilde{v}}{\approx}_f (\widetilde{x}\widetilde{z}_1\widetilde{z}_2)\, P''$, both with $\widetilde{v} \cap \widetilde{z}_1\widetilde{z}_2 = \emptyset$. Then the following diagram commutes:

$$\begin{array}{lllll} (\widetilde{x})\, P & \overset{\widetilde{v}}{\Rightarrow}_f & A' & \overset{\widetilde{v}}{\Rightarrow}_f & C' \\ & & \overset{\widetilde{v}}{\approx}_f & & \overset{\widetilde{v}}{\approx}_f \\ & & (\widetilde{x}\widetilde{z}_1)\, P' & \overset{\widetilde{v}}{\Rightarrow}_f & B' \\ & & & & \overset{\widetilde{v}}{\approx}_f \\ & & & & (\widetilde{x}\widetilde{z}_1\widetilde{z}_2)\, P'' \end{array}$$

Hence $(\widetilde{x})\, P \overset{\widetilde{v}}{\Rightarrow}_f C' \overset{\widetilde{v}}{\approx}_f (\widetilde{x}\widetilde{z}_1\widetilde{z}_2)\, P''$ as required. This completes the proof for Part 2.

For Part 4 assume $P \Downarrow^{\widetilde{v}} u$, which means $P \overset{\widetilde{v}}{\Rightarrow} P' \downarrow^{v} u$ and $u \in \widetilde{v}$. Given any $\widetilde{x}$ with $\widetilde{x} \cap \widetilde{v} = \emptyset$, Part 2 gives $(\widetilde{x})\, P \overset{\widetilde{v}}{\Rightarrow}_f A' \overset{\widetilde{v}}{\approx}_f (\widetilde{x}\widetilde{z})\, P'$ with $\widetilde{z} \cap \widetilde{v} = \emptyset$. Because $u \in \widetilde{v}$ and $\widetilde{v} \cap \widetilde{x}\widetilde{z} = \emptyset$, then $(\widetilde{x}\widetilde{z})\, P' \Downarrow^{\widetilde{v}}_f u$. Hence $(\widetilde{x})\, P \Downarrow^{\widetilde{v}}_f u$ as required.

For Part 5 forwards direction, we are given $\widetilde{v}$ and construct $\mathcal{S}_{\widetilde{v}}$ predicated upon it, and prove it is a $\overset{\widetilde{v}}{\approx}$:

$$\mathcal{S}_{\widetilde{v}} = \{(P, Q) : \exists \widetilde{x}, \widetilde{y} : (\widetilde{x})\, P \overset{\widetilde{v}}{\approx}_f (\widetilde{y})\, Q \text{ and } \widetilde{v} \cap \widetilde{x}\widetilde{y} = \emptyset\}.$$

If the left side of $\mathcal{S}_{\widetilde{v}}$ makes a transition $P \overset{\widetilde{v}}{\Rightarrow} P'$ then (Part 2) also $(\widetilde{x})\, P \overset{\widetilde{v}}{\Rightarrow}_f \overset{\widetilde{v}}{\approx}_f (\widetilde{x}\widetilde{z}_1)\, P'$ with $\widetilde{z}_1 \cap \widetilde{v} = \emptyset$. By construction, $(\widetilde{y})\, Q \overset{\widetilde{v}}{\Rightarrow}_f (\widetilde{y}\widetilde{z}_2)\, Q'$ with $\widetilde{z}_2 \cap \widetilde{v} = \emptyset$ and $(\widetilde{x}\widetilde{z}_1)\, P' \overset{\widetilde{v}}{\approx}_f (\widetilde{y}\widetilde{z}_2)\, Q'$. By Part 1, $Q \overset{\widetilde{v}}{\Rightarrow} Q'$; and by construction $P'\ \mathcal{S}_{\widetilde{v}}\ Q'$ as required.

If the left side makes an observation $P \downarrow^{\widetilde{v}} u$ then $u \in \widetilde{v}$. By Part 4, $(\widetilde{x})\, P \Downarrow^{\widetilde{v}}_f u$. By construction, $(\widetilde{y})\, Q \Downarrow^{\widetilde{v}}_f u$. By Part 3, $Q \Downarrow^{\widetilde{v}} u$ as required. This completes the forwards direction of Part 5.

For Part 5 reverse direction, the construction is more awkward. That is because a transition $P \overset{\widetilde{v}}{\Rightarrow} P'$ might have created a fresh name $z \in \widetilde{x}$, so that $(\widetilde{x})\, P \overset{\widetilde{y}}{\not\Rightarrow} (\widetilde{x}z)\, P'$. Instead we construct $\mathcal{S}_{\widetilde{v}}$ up to $\overset{\widetilde{v}}{\approx}_f$ and prove it is a $\overset{\widetilde{v}}{\approx}_f$:

$$\mathcal{S}_{\widetilde{v}} = \{(A, B) : \exists \widetilde{x}, \widetilde{y} : A \overset{\widetilde{v}}{\approx}_f (\widetilde{x})\, P,\ P \overset{\widetilde{v}}{\approx} Q,\ (\widetilde{y})\, Q \overset{\widetilde{v}}{\approx}_f B \text{ and } \widetilde{x}\widetilde{y} \cap \widetilde{v} = \emptyset\}.$$

Suppose the left side makes a transition $A \stackrel{\widetilde{v}}{\Rightarrow}_f A'$. The following commutative diagram applies.

$$
\begin{array}{lll}
A & \stackrel{\widetilde{v}}{\Rightarrow}_f & A' \\
\stackrel{\widetilde{v}}{\approx}_f & & \stackrel{\widetilde{v}}{\approx}_f \\
(\!|\widetilde{x}|\!)\, P & \stackrel{\widetilde{v}}{\Rightarrow}_f & (\!|\widetilde{x}\widetilde{z}_1|\!)\, P' \qquad \widetilde{z}_1 \cap \widetilde{v} = \emptyset
\end{array}
$$

by construction $A \stackrel{\widetilde{v}}{\approx}_f (\!|\widetilde{x}|\!)\, P$

Part 1

$$
\begin{array}{lll}
P & \stackrel{\widetilde{v}}{\Rightarrow} & P' \\
\stackrel{\widetilde{v}}{\approx} & & \stackrel{\widetilde{v}}{\approx} \\
Q & \stackrel{\widetilde{v}}{\Rightarrow} & Q'
\end{array}
$$

by construction $P \stackrel{\widetilde{v}}{\approx} Q$

Part 2

$$
\begin{array}{lll}
(\!|\widetilde{y}|\!)\, Q & \stackrel{\widetilde{v}}{\Rightarrow}_f & B'' \stackrel{\widetilde{v}}{\approx}_f (\!|\widetilde{y}\widetilde{z}_2|\!)\, Q' \qquad \widetilde{z}_2 \cap \widetilde{v} = \emptyset \\
\stackrel{\widetilde{v}}{\approx}_f & & \stackrel{\widetilde{v}}{\approx}_f \\
B & \stackrel{\widetilde{v}}{\Rightarrow}_f & B'
\end{array}
$$

by construction $(\!|\widetilde{y}|\!)\, Q \stackrel{\widetilde{v}}{\approx}_f B$

Thus, given $A \stackrel{\widetilde{v}}{\Rightarrow}_f A'$ we have deduced $B \stackrel{\widetilde{v}}{\Rightarrow}_f B'$ with $A' \stackrel{\widetilde{v}}{\approx}_f (\!|\widetilde{x}\widetilde{z}_1|\!)\, P'$, $P' \stackrel{\widetilde{v}}{\approx} Q'$ and $(\!|\widetilde{y}\widetilde{z}_2|\!)\, Q' \stackrel{\widetilde{v}}{\approx}_f B'$. Hence $A' \mathcal{S} B'$ as required.

Suppose the left side makes an observation $A \downarrow_f^{\widetilde{v}} u$. By construction, $(\!|\widetilde{x}|\!)\, P \Downarrow_f^{\widetilde{v}} u$. By Part 3, $P \Downarrow^{\widetilde{v}} u$. By construction, $Q \Downarrow^{\widetilde{v}} u$. By Part 4, $(\!|\widetilde{y}|\!)\, Q \Downarrow_f^{\widetilde{v}} u$. By construction $B \Downarrow_f^{\widetilde{v}} u$ as required. This completes the reverse direction of Part 5.

Part 6 is a trivial consequence of Part 5. □

Proposition 12 (Protocol correctness)

1. $M \stackrel{\widetilde{v}}{\Rightarrow} M'$ *implies* $[\![M]\!] \stackrel{\widetilde{v}}{\Rightarrow} [\![M']\!]$
2. $[\![M]\!] \stackrel{\widetilde{v}}{\Rightarrow} P'$ *implies there exists* M' *such that* $P' = [\![M']\!]$ *and* $M \stackrel{\widetilde{v}}{\Rightarrow} M'$
3. $M \Downarrow_m^{\widetilde{v}} u$ *implies* $[\![M]\!] \Downarrow^{\widetilde{v}} u$
4. $[\![M]\!] \Downarrow^{\widetilde{v}} u$ *implies* $M \Downarrow_m^{\widetilde{v}} u$
5. (Bisimulation) $M \stackrel{\widetilde{v}}{\approx}_m N$ *if and only if* $[\![M]\!] \stackrel{\widetilde{v}}{\approx}_m [\![N]\!]$
6. (Full abstraction) $P \approx_m Q$ *if and only if* $P \approx Q$

Proof. **For Part 1**, consider the case of a single transition $M \stackrel{\widetilde{v}}{\rightarrow}_m M'$. This must be derived from the broadcast rule

$$
\frac{M_1 \stackrel{!\mu}{\longrightarrow}_m M_1' \quad M_2 \stackrel{?\mu}{\longrightarrow}_m M_2' \quad \ldots \quad M_n \stackrel{?\mu}{\longrightarrow}_m M_n'}{M_1 | \ldots | M_n' \stackrel{!\mu}{\longrightarrow} M_1' | \ldots | M_n'}
$$

where $\mu \neq \nu v$ for any $v \in \widetilde{v}$. There are three possibilities for μ: either $\nu x'$ or off or acc. In the first case, the derivation was actually

$$\frac{\nu x.P \xrightarrow{!\nu x'}_m P\{x'/x\} \quad M_2 \xrightarrow{?\mu} M_2 \quad \ldots \quad M_n \xrightarrow{?\mu} M_n}{\nu x.P|M_2|\ldots|M_n \xrightarrow{!\nu x'}_m P\{x'/x\}|M_2|\ldots|M_n}$$

with x' fresh. Translating this bottom line with $[\![\cdot]\!]$ gives

$$\nu x.P|[\![M_2]\!]|\ldots|[\![M_n]\!] \xrightarrow{\widetilde{v}} P\{x'/x\}|[\![M_2]\!]|\ldots|[\![M_n]\!]$$

which, since x' is fresh and not in $\widetilde{v}$, is an axiom of the global pi calculus.

The second case is when μ is a off message. All transitions which send $M_1 \xrightarrow{!\,\mathsf{off}} M_1'$ or receive $M_i \xrightarrow{?\,\mathsf{off}} M_i'$ an off message have $[\![M_i]\!] = [\![M_i']\!]$. This yields the result trivially.

The third case is when μ is an acc message, for instance $\mu = \mathsf{acc}\, m'.\overline{u}\,\widetilde{x}\text{-}n$. This must have been generated by $M_1 = n; \widetilde{m.\alpha}, m'.u(\widetilde{y}); \overline{u}\,\widetilde{x}.P \xrightarrow{!\,\mathsf{acc}\, m'.\overline{u}\,\widetilde{x}\text{-}n} P$. By well-formedness *(heard offer previously made)*, then M must also contain an atom $M_2 = m'; \widetilde{m.\alpha}; u(\widetilde{y}).Q$, and this admits $M_2 \xrightarrow{?\mu} Q\{\widetilde{x}/\widetilde{y}\}$. By well-formedness *(no offer made twice)*, then every other $M_i \xrightarrow{?\mu}$ transition is either lose-competition, lose-opportunity or discard, all giving $[\![M_i]\!] = [\![M_i']\!]$. In all the transition $M \xrightarrow{\widetilde{v}}_m M'$ was

$$n; \widetilde{m.\alpha}; m'.u(\widetilde{y}); \overline{u}\,\widetilde{x}.P \mid m'; \widetilde{m'.\alpha'}; u(\widetilde{y}).Q \mid M_3|\ldots|M_n \xrightarrow{\widetilde{v}}_m P \mid Q\{\widetilde{x}/\widetilde{y}\} \mid M_3'|\ldots|M_n'.$$

Translating this transition with $[\![\cdot]\!]$ gives

$$\overline{u}\,\widetilde{x}.P \mid u(\widetilde{y}).Q \mid [\![M_3]\!]|\ldots|[\![M_n]\!] \xrightarrow{\widetilde{v}} P \mid Q\{\widetilde{x}/\widetilde{y}\} \mid [\![M_3']\!]|\ldots|[\![M_n']\!]$$

which, given $[\![M_i]\!] = [\![M_i']\!]$ for $i >= 3$, is an axiom of the global pi calculus. The other two cases for acc (when M_1 is an input or replicated input atom) are similar. This concludes the proof for Part 1.

For Part 2, the transition $[\![M]\!] \xrightarrow{\widetilde{v}} P'$ is derived from one of the three rules in the global pi calculus. If it was the *new* rule, the transition was

$$\nu x.P \mid R \xrightarrow{\widetilde{v}} P\{x'/x\} \mid R$$

with x' fresh and not in $\widetilde{v}$. Since the translation $[\![\cdot]\!]$ just translates a machine multiset element by element, then $M = \nu x.P \mid M_1|\ldots|M_n$ such that $[\![M_1]\!] \mid \ldots \mid [\![M_n]\!] = R$. Because x' is fresh and not in $\widetilde{v}$ then also M admits the matching transition:

$$\frac{\nu x.P \xrightarrow{!\nu x'}_m P'\{x'/x\} \quad M_1 \xrightarrow{?\nu x'}_m M_1 \quad \ldots \quad M_n \xrightarrow{?\nu x'}_m M_n}{\nu x.P|M_1|\ldots|M_n \xrightarrow{!\nu x'}_m P\{x'/x\}|M_1|\ldots|M_n}$$

If the transition $[\![M]\!] \xrightarrow{\widetilde{v}} P'$ was derived from the *react* rule, the transition was

$$\overline{u}\,\widetilde{x}.P \mid u(\widetilde{y}).Q \mid R \xrightarrow{\widetilde{v}} P \mid Q\{\widetilde{x}/\widetilde{y}\} \mid R.$$

We consider the form of M such that its translation $[\![M]\!]$ should have the form indicated. It must have had $M_1 = \overline{u}\,\widetilde{x}.P$ or $M_1 = n; \widetilde{m.\alpha}; \overline{u}\,\widetilde{x}.P$. It must have had $M_2 = u(\widetilde{y}).Q$ or $n'; \widetilde{m'.\alpha'}; u(\widetilde{y}).Q$. And it must have had $M_3|\ldots|M_n$ such that $[\![M_3]\!]|\ldots|[\![M_n]\!] = R$. If M_1 was just $\overline{u}\,\widetilde{x}.P$ then it can make the offer $\overline{u}\,\widetilde{x}.P \xrightarrow{!n=\mathsf{off}\,\overline{u}\,\widetilde{x}}_m n; \widetilde{m.\alpha}; \overline{u}\,\widetilde{x}.P$. Note that every other $M_i \xrightarrow{?n=\mathsf{off}\,\overline{u}\,\widetilde{x}}_m M_i'$ satisfies $[\![M_i]\!] = [\![M_i']\!]$. Similarly if M_2 was just $u(\widetilde{y}).Q$ then it too can make an offer. So we assume without loss of generality that M_1 and M_2 are both offers. By well-formedness *(one-came-first)* then either $n \in \widetilde{m}'$ or $n' \in \widetilde{m}$. The two cases are similar; we consider the first. By well-formedness *(offer-previously-made)* then the machine is

$$n; \widetilde{m.\alpha}; \overline{u}\,\widetilde{x}.P \mid n'; \widetilde{m'.\alpha'}, n.\overline{u}\,\widetilde{x}; u(\widetilde{y}).Q \mid M_2|\ldots|M_n.$$

This admits the transition

$$\xrightarrow{!\,\mathsf{acc}\,n.u(\widetilde{y})\text{-}n'}_m P \mid Q\{\widetilde{x}/\widetilde{y}\} \mid M_1'|\ldots|M_n'.$$

For each $i >= 3$ the transition $M_i \xrightarrow{!\mu}_m M_i'$ must have been lose-competition or lose-opportunity or discard, so $[\![M_i]\!] = [\![M_i']\!]$. Hence, translating Equation 6 with $[\![\cdot]\!]$ yields Equation 6 as required.

The case where $[\![M]\!] \xrightarrow{\widetilde{v}} P'$ was derived from the *replicated react* rule is similar. This completes the proof for Part 2.

For Part 3 it is given that $M \Downarrow_m^{\widetilde{v}} u$, or equivalently $M \stackrel{\widetilde{v}}{\Rightarrow}_m M' \downarrow_m^{\widetilde{v}} u$. Hence $u \in \widetilde{v}$ and M' contains an atom $n; \widetilde{m.\alpha}; \overline{u}\,\widetilde{x}.P$ or $\ldots; u(\widetilde{x}).P$ or $\ldots; !u(\widetilde{x}).P$. By Part 1, $[\![M]\!] \stackrel{\widetilde{v}}{\Rightarrow} [\![M']\!]$. Since M' contains one of the three specified atom, then the translation $[\![M']\!]$ contains one of $\overline{u}\,\widetilde{x}.P$ or $u(\widetilde{x}).P$ or $!u(\widetilde{x}).P$. Hence $[\![M']\!] \downarrow^{\widetilde{v}} u$ as required.

For Part 4 it is given that $[\![M]\!] \Downarrow_m^{\widetilde{v}} u$, or equivalently $[\![M]\!] \stackrel{\widetilde{v}}{\Rightarrow} P' \downarrow^{\widetilde{v}} u$. Hence $u \in \widetilde{v}$ and P' contains $\overline{u}\,\widetilde{x}.P$ or $u(\widetilde{x}).P$ or $!u(\widetilde{x}).P$. We consider the case $\overline{u}\,\widetilde{x}.P$; the others are similar. By Part 2, $M \stackrel{\widetilde{v}}{\Rightarrow}_m M'$ with $P' = [\![M']\!]$. Since P' contains $\overline{u}\,\widetilde{x}.P$ then M' must have contained either $\overline{u}\,\widetilde{x}.P$ or $n; \widetilde{m.\alpha}; \overline{u}\,\widetilde{x}.P$. If the latter, then $M' \downarrow_m^{\widetilde{v}} u$ as required. If the former, then M' admits the transition

$$\overline{u}\,\widetilde{x}.P \mid M_2|\ldots|M_n \xrightarrow{!n=\mathsf{off}\,\overline{u}\,\widetilde{x}}_m n;;\overline{u}\,\widetilde{x}.P \mid M_2'|\ldots|M_n'.$$

This result then admits $\downarrow_m^{\widetilde{v}} u$ as required. This completes the proof for Part 4.

For Part 5 forwards direction, construct

$$\mathcal{S}_{\widetilde{v}} = \{([\![M]\!], [\![N]\!]) : M \stackrel{\widetilde{v}}{\approx}_m N\}$$

and prove it is a $\stackrel{\widetilde{v}}{\approx}$. First show that the transition $[\![M]\!] \stackrel{\widetilde{v}}{\Rightarrow} P'$ is matched by $[\![N]\!]$. By Part 2, then $M \stackrel{\widetilde{v}}{\Rightarrow}_m M'$. By construction, $N \stackrel{\widetilde{v}}{\Rightarrow}_m N' \stackrel{\widetilde{v}}{\approx}_m M'$. By Part 1, $[\![N]\!] \stackrel{\widetilde{v}}{\Rightarrow} [\![N']\!]$. Hence by construction $P' = [\![M']\!] \mathcal{S} [\![N']\!]$ as required. Second show that the barb $[\![M]\!] \Downarrow^{\widetilde{v}} u$ is matched by $[\![N]\!]$. By Part 4, $M \Downarrow^{\widetilde{v}}_m u$. By construction, $N \Downarrow^{\widetilde{v}}_m u$. By Part 3, $[\![M]\!] \Downarrow^{\widetilde{v}}_m u$ as required.

For Part 5 reverse direction, construct

$$\mathcal{S}_{\widetilde{v}} = \{(M, N) : [\![M]\!] \stackrel{\widetilde{v}}{\approx} [\![N]\!]\}$$

and prove it is a $\stackrel{\widetilde{v}}{\approx}_m$. First show that the transition $M \stackrel{\widetilde{v}}{\Rightarrow}_m M'$ is matched by N. By Part 1, $[\![M]\!] \stackrel{\widetilde{v}}{\Rightarrow} [\![M']\!]$. By construction, $[\![N]\!] \stackrel{\widetilde{v}}{\Rightarrow} Q' \stackrel{\widetilde{v}}{\approx} [\![M']\!]$. By Part 2, $N \stackrel{\widetilde{v}}{\Rightarrow}_m N'$ with $[\![N']\!] = Q'$. Hence by construction $[\![M']\!] \stackrel{\widetilde{v}}{\approx} Q' = [\![N']\!]$ as required. Second show that the barb $M \Downarrow^{\widetilde{v}} u$ is matched by N. By Part 3, $[\![M]\!] \Downarrow^{\widetilde{v}}_m u$. By construction, $[\![N]\!] \Downarrow^{\widetilde{v}} u$. By Part 4, $N \Downarrow^{\widetilde{v}} u$ as required. This concludes the proof of Part 5.

Part 6 is a trivial consequence of Part 5. □