Process Calculus Based Upon Evaluation to Committed Form

Andrew M. Pitts and Joshua R. X. Ross

Cambridge University Computer Laboratory, Cambridge CB2 3QG, UK

An approach to the semantics of CCS-like communicating processes is proposed that is based upon evaluation of processes to input- or output-committed form, with no explicit mention of silent actions. This leads to a co-inductively defined notion of *evaluation bisimilarity*—a form of weak branching-time equivalence which is shown to be a congruence, even in the presence of summation. The relationship between this evaluation-based approach and the more traditional, labelled transition semantics is investigated. In particular, with some restriction on sums, CCS observation equivalence is characterised purely in terms of evaluation to committed form, and evaluation bisimilarity is characterised as a weak delay equivalence. These results are extended to the higher order case, where evaluation bisimilarity coincides with Sangiorgi's weak context bisimilarity. An evaluation-based approach to π-calculus and the relationship with Milner and Sangiorgi's reduction-based notion of barbed bisimulation are also examined.

1 Introduction

Beginning with Milner's CCS [14], it has become commonplace to specify the operational semantics of languages for concurrent, communicating processes by means of an action-labelled transition relation between process expressions; and ideally, by one that is inductively defined by rules following the structure of expressions [23]. In particular this provides the means for defining notions of process equivalence in terms of various kinds of bisimulation relation derived from the labelled transition system, with associated co-induction proof techniques. This approach to process calculi has been very fruitful. So before proposing an alternative approach, as we do in this paper, it is necessary to examine the weak points of the status quo. We identify two which influenced the worked presented here.

First, the construction of *weak, branching-time congruences* is not as simple

as one might wish. The gap between CCS observation equivalence and observation congruence in the presence of summation is the best known example of the difficulties we have in mind; but see also [6]. The use of a transition system in which externally unobservable behaviour is represented *explicitly* (by τ-transitions) does not always fit well with defining congruences (i.e. equivalences respecting the language constructs) which abstract from such behaviour (i.e. are 'weak'), but which do not identify processes with different 'may' and 'must' behaviour with respect to external actions.

Secondly and perhaps more significantly, for languages that have higher-order features [17,28], or which combine concurrent communication with higher order functions [5,24], it has proved difficult to devise labelled transition semantics that are both simple and give rise to weak bisimilarities with expected properties. For example, witness the difficulties caused by the combination of (higher order) value-passing actions with static restriction discussed by Sangiorgi in [27].

Milner and Sangiorgi were partly addressing this second kind of problem when they introduced the notion of barbed bisimulation [18], defined in terms of a reduction relation and a convergence predicate. This approach is both simple (especially when combined with the use of 'chemistry' [2], i.e. a structural congruence relation) and uniform—in the sense that one can easily apply it to some quite different-looking calculi. It has certainly been applied successfully: see [16,25,4]. Yet there remain difficulties of the first kind mentioned above, to do with factoring out reductions (i.e. τ-transitions) in weak equivalences; and the 'barbed' approach usually involves quite heavy use of closure under contexts in order to obtain a congruence relation.

For sequential languages, the use of a reduction relation to specify operational semantics usually comes along with some fixed strategy for reducing configurations, including a notion of which configurations are in final, or *canonical*, form. Therefore, for many purposes one can abstract away from the single steps of reduction and just consider an *evaluation relation* between configurations and the canonical forms to which they give rise (if any). As for one-step reduction relations, so for 'big-step' evaluation relations, the ideal situation is where evaluation to canonical form is inductively defined by rules that follow the syntactical structure of the language. For programming languages, the best known example of a large scale operational semantics in this style is the definition of Standard ML [19]. In a somewhat purer vein, evaluation to canonical form is a key part of Martin-Löf's type-theoretic foundation for constructive mathematics [12].

This paper attempts to demonstrate that process calculi can be based upon evaluation to canonical form and that some of the problems mentioned above are solved thereby; in particular, in this approach there is no mention of τ-

transitions *a priori*. At first it might seem unlikely that the *interactive* nature of process communication can be adequately captured by an evaluation relation. But note that canonical forms may well contain unevaluated subexpressions that get 'activated' in bisimulation equivalences based upon evaluation. The paradigmatic example is Abramsky's 'lazy' lambda calculus [1], in which evaluation does not take place 'under the lambda'—canonical forms are lambda abstractions, $\lambda x.E$, with E unevaluated. Abramsky's *applicative bisimulation* is the greatest symmetric relation $\mathcal{R}$ between closed lambda terms such that if $M_1 \ \mathcal{R} \ M_2$ and $M_1 \Downarrow \lambda x.E_1$, then $M_2 \Downarrow \lambda x.E_2$ holds for some E_2 with $E_1[N/x] \ \mathcal{R} \ E_2[N/x]$ for all closed N. Here $\Downarrow$ denotes the (call-by-name) evaluation relation. The 'interaction' embodied in this definition is one of evaluating to a lambda abstraction versus supplying an argument for the parameter in the body of the abstraction.

To develop a similar style of semantics for processes, the crucial question is of course: "what are the canonical forms?" For CCS-like calculi, a natural answer is to take processes like $a(x).P(x)$ and $\bar{a}v.Q$ which are committed to input and output actions respectively. (We consider other answers in Section 4.2.) We develop this 'evaluation to committed form' approach in Section 2 (for the non value-passing case, for simplicity). As is the case for the reduction-based approach leading to barbed bisimilarity, we work modulo a structural congruence relation. In fact this seems to be necessary for the evaluation-based approach to yield a sufficiently rich theory (see Remark 4). We define an associated notion of *evaluation bisimilarity* and adapt Howe's work [10] on congruence properties of applicative bisimilarity to show that it is a congruence. (Although we put restrictions on summation in Section 2, the congruence property holds without them: see Section 4.4.) Besides being a congruence, evaluation bisimilarity seems a reasonable 'weak, branching-time' process equivalence whose definition is completely τ-free. In Section 3 we investigate its relationship to existing, transition-based equivalences.

To do that we first have to examine the relationship between our notion of evaluation to committed form, $P \Downarrow \ell.P'$, and the usual labelled transition relation. Roughly speaking, $P \Downarrow \ell.P'$ means that P can do some number of τ-transitions followed by an ℓ-transition to become a process strongly equivalent to P': see Lemma 19 and Theorem 21. These results permit one to characterise CCS observation equivalence purely in terms of evaluation to committed form (at least in the case that summation is restricted to action-guarded summands). Moreover, they lead to a characterisation of evaluation bisimilarity as *delay bisimulation equivalence* [13,29]—which is like CCS observation equivalence except that $\stackrel{\tau^*\ell}{\longrightarrow}$ is used in place of $\stackrel{\tau^*\ell\tau^*}{\longrightarrow}$: see Theorem 24. Delay bisimulation equivalence is finer than CCS observation equivalence, but coarser than Van Glabbeek and Weijland's *branching bisimulation equivalence*: see [6]. Similar delay equivalences have occurred recently in work on higher order process calculi [27] and on integrations of functions and processes [3]. Pleasingly, the

evaluation-based approach extends smoothly to higher order processes and we obtain a coincidence between evaluation bisimilarity and Sangiorgi's *weak context bisimilarity* (Theorem 27). This is described briefly in Section 4 along with a number of other topics: a treatment of asynchronous-output π-calculus in terms of evaluation to input-committed form, the relationship between our evaluation-based approach and the 'barbed' approach, and the relation between evaluation and transition in the presence of unrestricted summation.

2 Evaluation Bisimilarity

As a first illustration of the use of an evaluation relation to specify the behaviour of communicating processes, we consider a subset of CCS [14] which we call *normal* CCS, or NCCS for short. It has operators for composition, restriction, recursion, and synchronous input and output, but has summation restricted to *normal* processes—which by definition are (finite) sums of processes committed to input or output actions. Thus for example, the CCS process $x.\mathbf{0} + \bar{y}.\mathbf{0}$ is in NCCS, but $x.\mathbf{0} + (\bar{y}.\mathbf{0}|z.\mathbf{0})$ and $x.\mathbf{0} + \tau.\bar{y}.\mathbf{0}$ are not. [1]

Why use this restricted form of CCS to introduce the evaluation-based semantics of processes? The answer lies in the fact that with the restriction to input/output-guarded summation, there is a close correspondence between the evaluation- and the labelled transition-based semantics of CCS (see Section 3); whereas in the presence of unguarded summation, the situation is more complicated (see Section 4.4). Since the notions of *evaluation to committed form* and *evaluation bisimilarity* we are going to introduce seem rather natural ones, this 'misbehaviour' of unguarded summation is perhaps an indication of its semantically problematic nature. [2] It is worth remarking that unguarded *recursion* causes no problems for the tie-up between evaluation and transition, and so is included in NCCS.

NCCS process expressions are given by the grammar

$$\begin{array}{rl} \text{processes} & E ::= X \mid N \mid E|E \mid (\nu x)E \mid \mathit{fix}(X{=}E) \\ \text{normal processes} & N ::= \mathbf{0} \mid K \mid N + N \\ \text{committed processes} & K ::= x.E \mid \bar{x}.E \end{array}$$

where X ranges over a countably infinite set of process *variables* and x ranges over a countably infinite set of channel *names*. Name restriction is written

[1] Indeed, τ-prefixing is only included implicitly in NCCS—see Definition 8.

[2] Of course the fact that unguarded summation does not respect CCS observation equivalence is a better known indicator of its problematic nature.

$(\nu x)E$, rather than $E \setminus x$ as in CCS, and we prefer to make it a binding operation: free occurrences in E of the name x become bound in $(\nu x)E$. The other binding operation is for recursively defined processes: free occurrences in E of the process variable X become bound in $\mathit{fix}(X{=}E)$.

Note *Throughout the paper* we identify expressions up to α-conversion of bound names and variables, *and write* $E =_\alpha E'$ *to indicate that* E *and* E' *are syntactically identical modulo* α*-conversion.*

We use $fv(E)$ and $fn(E)$ to indicate respectively the finite set of free variables and free names of E. An NCCS process expression E is *closed* if $fv(E)$ is empty and *open* otherwise. Most of the time we will refer to closed process expressions simply as *processes*, and use letters like $P, Q, R, \ldots$ to denote them. For simplicity we have omitted any relabelling operator from NCCS. Instead we make do with name substitution as an operation on syntax: $E[x'/x]$ denotes the result (well-defined up to α-conversion) of substituting the name x' for all free occurrences of the name x in E. Similarly $E[E'/X]$ denotes the result of substituting the process expression E' for all free occurrences of the variable X in E. Following usual CCS practice, we write a typical committed process as $\ell.P$ where ℓ ranges over *labels*, which are either names (x) or *co-names* ($\bar{x}$):

$$\ell ::= x \mid \bar{x}.$$

As usual, $\bar{\ell} = \bar{x}$ if $\ell = x$ is a name, and $\bar{\ell} = x$ if $\ell = \bar{x}$ is a co-name.

Before defining an evaluation semantics for NCCS processes, we have to give a notion of structural congruence that turns out to be an essential ingredient of the definition.

Definition 1 *An NCCS* congruence relation, $\mathcal{E}$, *is an equivalence relation between NCCS process expressions which is closed under the following rules.*

$$\frac{E_1 \;\mathcal{E}\; E_2 \quad E_1' \;\mathcal{E}\; E_2'}{E_1|E_1' \;\mathcal{E}\; E_2|E_2'} \tag{cr1}$$

$$\frac{E_1 \;\mathcal{E}\; E_2}{(\nu x)E_1 \;\mathcal{E}\; (\nu x)E_2} \tag{cr2}$$

$$\frac{E_1 \;\mathcal{E}\; E_2}{\mathit{fix}(X{=}E_1) \;\mathcal{E}\; \mathit{fix}(X{=}E_2)} \tag{cr3}$$

$$\frac{E_1 \;\mathcal{E}\; E_2}{\ell.E_1 \;\mathcal{E}\; \ell.E_2} \tag{cr4}$$

$$\frac{N_1 \;\mathcal{E}\; N_2 \quad N_1' \;\mathcal{E}\; N_2'}{N_1 + N_1' \;\mathcal{E}\; N_2 + N_2'} \tag{cr5}$$

Structural congruence, $\equiv$, *is the smallest such relation containing the following*

pairs of processes:

$$
\begin{aligned}
P_1|(P_2|P_3) &\equiv (P_1|P_2)|P_3 \\
P_1|P_2 &\equiv P_2|P_1 \\
P|\mathbf{0} &\equiv P \\
(\nu x)(P_1|P_2) &\equiv ((\nu x)P_1)|P_2 \quad \textit{if } x \notin \mathit{fn}(P_2) \\
(\nu x_1)(\nu x_2)P &\equiv (\nu x_2)(\nu x_1)P \\
(\nu x)\mathbf{0} &\equiv \mathbf{0} \\
N_1 + (N_2 + N_3) &\equiv (N_1 + N_2) + N_3 \\
N_1 + N_2 &\equiv N_2 + N_1 \\
N + \mathbf{0} &\equiv N.
\end{aligned}
$$

Notions of structural congruence are an extremely useful way to simplify the specification of the operational semantics of reactive systems. They were first popularised by the 'chemical abstract machine' of Berry and Boudol [2]. The form we are using is like that used in Milner's presentation of reduction for π-calculus processes in [16]. In one sense the identifications made by such congruences just take us one step further up the path abstracting away from inessential choices in the concrete representation of syntax. Although there is some choice as to which identities should be 'structural' (for example, we have not included any identities for recursive processes), those relating to composition and restriction seem essential for evaluation to committed form to lead to a sufficiently rich theory of process evaluation and equivalence. (See Remark 4 below.)

Definition 2 (Evaluation to committed form) *The NCCS* evaluation relation *takes the form $P \Downarrow K$, where P and K are processes and K is in 'committed form', i.e. is of the form $\ell.P'$ for some name or co-name ℓ and some process P'. It is inductively generated by the following axiom and rules.*

$$\frac{P_1 \Downarrow \ell.P_1'}{P_2 \Downarrow \ell.P_2'} \quad \textit{if } P_1 \equiv P_2 \textit{ and } P_1' \equiv P_2' \qquad (\Downarrow 0)$$

$$(N + \ell.P)|Q \Downarrow \ell.(P|Q) \qquad (\Downarrow 1)$$

$$\frac{P_1 \Downarrow \ell.P_1' \quad P_2 \Downarrow \bar{\ell}.P_2' \quad P_1'|P_2' \Downarrow K}{P_1|P_2 \Downarrow K} \qquad (\Downarrow 2)$$

$$\frac{P \Downarrow \ell.P'}{(\nu x)P \Downarrow \ell.(\nu x)P'} \quad \textit{if } x \notin \{\ell, \bar{\ell}\} \qquad (\Downarrow 3)$$

$$\frac{E[\mathit{fix}(X{=}E)/X]|Q \Downarrow K}{\mathit{fix}(X{=}E)|Q \Downarrow K} \qquad (\Downarrow 4)$$

For readers familiar with the usual labelled transition semantics of CCS, the above rules should suggest that $P \Downarrow \ell.P'$ means that P can do some number of τ-transitions followed by an ℓ-transition to become P'. This intuition is roughly correct: we will make the relationship between evaluation and transition precise in Section 3 (see Corollary 22). Manifestly Definition 2 is a 'τ-free' description of how processes execute. Here is a simple example to illustrate a distinctive feature of the evaluation rule ($\Downarrow 2$) for synchronised communication—namely that the effects of such synchronisations (i.e. 'τ-transitions') are only observable if there is some externally observable (input or output) action that the process can offer.

Example 3 *Let $P = x.\mathbf{0}|\bar{x}.\mathbf{0}$. Then rule ($\Downarrow 2$) cannot be applied and $P \Downarrow K$ holds just for $K \equiv x.\bar{x}.\mathbf{0}$ and $K \equiv \bar{x}.x.\mathbf{0}$. For $P|y.\mathbf{0}$ however, in addition to evaluations committing to x and $\bar{x}$, the evaluation $P|y.\mathbf{0} \Downarrow y.\mathbf{0}$ can be deduced using rule ($\Downarrow 2$) together with rules ($\Downarrow 0$) and ($\Downarrow 1$).*

Remark 4 *Note that evaluation to committed form takes place modulo structural congruence—this is the force of rule ($\Downarrow 0$). Not only does this permit a simpler presentation of the rules, it appears to be* necessary *for the notion of evaluation bisimilarity given below to have the expected structural properties. For example without ($\Downarrow 0$), in Example 3 one could only deduce $(x.\mathbf{0}|\bar{x}.\mathbf{0})|y.\mathbf{0} \Downarrow y.P'$ for $P' = (x.\mathbf{0}|\bar{x}.\mathbf{0})|\mathbf{0}$, whereas $x.\mathbf{0}|(\bar{x}.\mathbf{0}|y.\mathbf{0}) \Downarrow y.(\mathbf{0}|(\mathbf{0}|\mathbf{0}))$ would still hold. Therefore, without structural congruence, the definition of evaluation bisimilarity given below would fail to make composition associative.*

Since one is working modulo structural congruence, in trying to construct the proof of an evaluation from the bottom up, one cannot deduce the last rule used in the proof merely from the syntactic structure of the process expression on the left hand side of $\Downarrow$. In this respect the situation is similar to that for reduction in the π-calculus as formulated in [16]. Note that rules ($\Downarrow 1$)–($\Downarrow 4$) explain how the various NCCS syntactic constructs evaluate, *but only in the context of some parallel process*, Q (which of course may be $\mathbf{0}$). Given that one is working modulo structural congruence anyway, the presence of such contexts is not much of a further complication to the business of constructing proofs of evaluation. Note that there is no need to use a context $[-]|Q$ in rules ($\Downarrow 2$) and ($\Downarrow 3$) since the apparently more general rules

$$\frac{P_1 \Downarrow \ell.P_1' \quad P_2 \Downarrow \bar{\ell}.P_2' \quad (P_1'|P_2')|Q \Downarrow K}{(P_1|P_2)|Q \Downarrow K}$$

$$\frac{P|Q \Downarrow \ell.P'}{((\nu x)P)|Q \Downarrow \ell.(\nu x)P'} \text{ if } x \notin \{\ell, \bar{\ell}\} \cup fn(Q)$$

are derivable. Here are some further derived properties of evaluation that we will need. They are easily established by induction on the proofs of evaluation.

Lemma 5*(i) If $P \Downarrow \ell.P'$, then $P|Q \Downarrow \ell.(P'|Q)$ for any Q.*
(ii) If $(\nu x)P \Downarrow \ell.P''$, then $P \Downarrow \ell.P'$ for some P' with $(\nu x)P' \equiv P''$.
(iii) Evaluation is name equivariant*, in the sense that for any permutation σ of the set of channel names, if $P \Downarrow K$ then $P[\sigma] \Downarrow K[\sigma]$. ($P[\sigma]$ indicates the substituted expression $P[\sigma(x)/x \mid x \in dom(\sigma)]$.)*
(iv) If $N \Downarrow K$, then $N + N' \Downarrow K$.

Definition 6 (Evaluation bisimilarity) *A binary relation $\mathcal{R}$ between NCCS processes is an* evaluation simulation *if $P_1 \mathcal{R} P_2$ implies for all Q that*

$$P_1|Q \Downarrow \ell.P_1' \Rightarrow \exists P_2' \, (P_2|Q \Downarrow \ell.P_2' \;\&\; P_1' \mathcal{R} P_2').$$

If the reciprocal relation $\mathcal{R}^{-1} \stackrel{\text{def}}{=} \{(P_1, P_2) \mid P_2 \mathcal{R} P_1\}$ is also an evaluation simulation, we say that $\mathcal{R}$ is an evaluation bisimulation*. Finally, two NCCS processes are* evaluation bisimilar*, written $P_1 \simeq_{\Downarrow} P_2$, if $P_1 \mathcal{R} P_2$ holds for some evaluation bisimulation $\mathcal{R}$.*

Here are some simple properties of $\simeq_{\Downarrow}$, proved using Lemma 5.

Lemma 7 *Evaluation bisimilarity is the greatest evaluation bisimulation. It is an equivalence relation and contains structural congruence. Moreover, if $P_1 \simeq_{\Downarrow} P_2$ then $P_1|Q \simeq_{\Downarrow} P_2|Q$, $(\nu x)P_1 \simeq_{\Downarrow} (\nu x)P_2$, and $P_1[\sigma] \simeq_{\Downarrow} P_2[\sigma]$ (for any process Q, name x, and permutation of names σ).*

Although the topic will be pursued in detail in the next section, we wish to give the reader some feel now for how evaluation bisimilarity compares with known equivalences on (N)CCS processes. To do so we need to introduce *τ-guarded* processes.

Definition 8 (τ-Prefixing) *Although we did not include an operation $\tau.P$ for prefixing by a silent action in the NCCS syntax, as one might expect it is present implicitly:*

$$\tau.P \quad \stackrel{\text{def}}{=} \quad (\nu x)(x.P|\bar{x}.\mathbf{0})$$

where x is not free in P. More generally, one can extend summation to include τ-guarded summands: given a normal process N and a process P define

$$N +_\tau P \quad \stackrel{\text{def}}{=} \quad (\nu x)((N + x.P)|\bar{x}.\mathbf{0})$$

where x is not free in N or P. (Clearly, one can also define sums with more than one τ-guarded summand.)

Note that $\tau.P$ is *not* a normal process (according to the grammar for NCCS expressions given at the beginning of this section). Thus '$N + \tau.P$' is not a well-formed NCCS expression; but as the notation $N +_\tau P$ is supposed to

indicate, this well-formed NCCS expression has the evaluation behaviour one might expect of the sum of N and $\tau.P$.

Example 9 *Evaluation bisimilarity satisfies the following* τ-laws *which illustrate that it is a 'weak' equivalence:*

$$P \simeq_{\Downarrow} \tau.P \tag{1}$$
$$N +_\tau N' \simeq_{\Downarrow} (N + N') +_\tau N'. \tag{2}$$

The validity of these laws will be established via the characterisation of $\simeq_{\Downarrow}$ in terms of delay bisimulation equivalence given in the next section (Theorem 24). Property (1) may seem surprisingly strong, given that $\simeq_{\Downarrow}$ is a congruence (Theorem 14) and that (1) fails for CCS observation *congruence*. But it does not imply that we can just erase τ in 'τ-prefixed' sums: for notwithstanding (1), in general $N +_\tau N'$ is *not* evaluation bisimilar to $N + N'$. The next example illustrates this (and is of course an inequivalence one might expect to hold of a weak, branching-time equivalence).

Example 10

$$x.\mathbf{0} +_\tau y.\mathbf{0} \not\simeq_{\Downarrow} x.\mathbf{0} + y.\mathbf{0} \qquad (x \neq y). \tag{3}$$

PROOF. We use the fact (Lemma 7) that $\simeq_{\Downarrow}$ is an evaluation bisimilarity. First note that from the definitions of $\equiv$ and $\Downarrow$ one has

$$(x.\mathbf{0} +_\tau y.\mathbf{0})|z.\mathbf{0} \stackrel{\text{def}}{=} (\nu x')((x.\mathbf{0} + x'.y.\mathbf{0})|\bar{x}'.\mathbf{0})|z.\mathbf{0} \Downarrow z.y.\mathbf{0}$$

whereas $(x.\mathbf{0} + y.\mathbf{0})|z.\mathbf{0} \Downarrow z.P$ holds only with $P \equiv x.\mathbf{0} + y.\mathbf{0}$ Hence if the two processes in (3) were evaluation bisimilar, then so would be $y.\mathbf{0}$ and $x.\mathbf{0} + y.\mathbf{0}$. But that is impossible because $x.\mathbf{0} + y.\mathbf{0} \Downarrow x.\mathbf{0}$ whereas $y.\mathbf{0} \Downarrow x.Q$ does not hold for any Q (since $y \neq x$).

To finish this series of examples, we give an example to show that $\simeq_{\Downarrow}$ does not coincide with the best known weak equivalence, CCS *observation equivalence* [14, 5.1]. (We will see in the next section that $\simeq_{\Downarrow}$ does coincide with another known equivalence—delay bisimulation equivalence—which is strictly finer than observation equivalence.)

Example 11

$$x.(y.\mathbf{0} +_\tau \mathbf{0}) \not\simeq_{\Downarrow} x.(y.\mathbf{0} +_\tau \mathbf{0}) + x.\mathbf{0} \qquad (x \neq y) \tag{4}$$

PROOF. Note that $x.(y.\mathbf{0} +_\tau \mathbf{0}) + x.\mathbf{0} \Downarrow x.\mathbf{0}$ whereas $x.(y.\mathbf{0} +_\tau \mathbf{0}) \Downarrow x.P$ holds only with $P \equiv y.\mathbf{0} +_\tau \mathbf{0}$. Hence if the two processes in (4) were evaluation

bisimilar, then so would be $\mathbf{0}$ and $y.\mathbf{0} +_\tau \mathbf{0}$—which is plainly false since $\mathbf{0} \not\Downarrow$ whereas $y.\mathbf{0} +_\tau \mathbf{0} \Downarrow y.(\nu x)\bar{x}.\mathbf{0}$. $\Box$

The definition of evaluation bisimilarity for NCCS is analogous to the notion of *applicative bisimilarity* for functional languages introduced by Abramsky [1] and studied by Howe [10] and others—so much so, that we can adapt Howe's method [11] for proving congruence properties of applicative bisimilarity in the presence of non-determinism to the case in point: see Theorem 14 below. However, there is one important complication compared with applicative bisimilarity—namely the quantification over contexts $[-]|Q$ which occurs in Definition 6. Here is an example to show that such contexts are necessary to obtain congruence properties of bisimilarity in this setting. The example uses $Q = y.\mathbf{0}$, with y a fresh name. We will see in the next section (Theorem 26) that this is in fact the *only* instance of Q one needs to consider.

Example 12 *Suppose that $\mathcal{R}$ satisfies*

$$\begin{array}{l} P_1 \mathcal{R} P_2 \;\&\; P_1 \Downarrow \ell.P_1' \Rightarrow \exists P_2' \, (P_2 \Downarrow \ell.P_2' \;\&\; P_1' \mathcal{R} P_2') \\ P_1 \mathcal{R} P_2 \;\&\; P_2 \Downarrow \ell.P_2' \Rightarrow \exists P_1' \, (P_1 \Downarrow \ell.P_1' \;\&\; P_1' \mathcal{R} P_2') \end{array} \tag{5}$$

Then it is not necessarily the case that $\mathcal{R} \subseteq \simeq_\Downarrow$, and hence in particular $\simeq_\Downarrow$ cannot be defined as the greatest relation satisfying (5).

PROOF. For example, define $\mathcal{R}$ by

$$\begin{array}{rcl} P_1 \mathcal{R} P_2 & \stackrel{\text{def}}{\Leftrightarrow} & (P_1 = x.\bar{x}.\mathbf{0} + \bar{x}.x.\mathbf{0} \;\&\; P_2 = x.\mathbf{0}|\bar{x}.\mathbf{0}) \\ & & \vee \, P_1 \equiv P_2 . \end{array}$$

If P is either $x.\bar{x}.\mathbf{0} + \bar{x}.x.\mathbf{0}$ or $x.\mathbf{0}|\bar{x}.\mathbf{0}$, then $P \Downarrow K$ holds just for $K = x.\bar{x}.\mathbf{0}$ and $K = \bar{x}.x.\mathbf{0}$. Therefore $\mathcal{R}$ certainly satisfies (5). However $x.\bar{x}.\mathbf{0} + \bar{x}.x.\mathbf{0}$ is not evaluation bisimilar to $x.\mathbf{0}|\bar{x}.\mathbf{0}$. For $(x.\mathbf{0}|\bar{x}.\mathbf{0})|y.\mathbf{0} \Downarrow y.\mathbf{0}$, whereas $(x.\bar{x}.\mathbf{0} + \bar{x}.x.\mathbf{0})|y.\mathbf{0} \Downarrow y.P$ only holds for $P = x.\bar{x}.\mathbf{0} + \bar{x}.x.\mathbf{0}$ and clearly $x.\bar{x}.\mathbf{0} + \bar{x}.x.\mathbf{0} \not\simeq_\Downarrow \mathbf{0}$. $\Box$

The greatest $\mathcal{R}$ satisfying (5) is indeed an equivalence relation, but not a congruence since this example shows that it relates $x.\bar{x}.\mathbf{0} + \bar{x}.x.\mathbf{0}$ to $x.\mathbf{0}|\bar{x}.\mathbf{0}$, but does not relate $(x.\bar{x}.\mathbf{0} + \bar{x}.x.\mathbf{0})|y.\mathbf{0}$ to $(x.\mathbf{0}|\bar{x}.\mathbf{0})|y.\mathbf{0}$. By contrast, we show now that $\simeq_\Downarrow$ is indeed a congruence for NCCS.

Definition 13 *Extend evaluation bisimilarity from closed to open process expressions by taking closed instantiations: we write*

$$E_1 \simeq_\Downarrow^\circ E_2$$

to mean that $E_1[\vec{P}/\vec{X}] \simeq_{\Downarrow} E_2[\vec{P}/\vec{X}]$ *holds for all substitutions of processes* $\vec{P}$ *for the free variables* $\vec{X}$ *of* E_1, E_2.

Theorem 14 $\simeq^{\circ}_{\Downarrow}$ *is an NCCS congruence relation (cf. Definition 1).*

That $\simeq^{\circ}_{\Downarrow}$ is an equivalence relation satisfying (cr1) and (cr2) follows from Lemma 7. To establish the other properties, the first proof strategy that comes to mind is to take the smallest relation containing $\simeq^{\circ}_{\Downarrow}$ and closed under (cr3)–(cr5), and show that its restriction to closed expressions is an evaluation bisimulation. It is hard to see how to do this directly, because in an evaluation $P \Downarrow \ell.P'$, P' may be structurally quite different from P. Instead we use an indirect approach adapted from [10,11] which makes use of the following 'congruence candidate' relation.

Definition 15 *Let* $\simeq^{*}_{\Downarrow}$ *be the binary relation between process expressions inductively defined by rules (cr1)–(cr5) together with*

$$\frac{E_1 \simeq^{*}_{\Downarrow} E_2}{E_1' \simeq^{*}_{\Downarrow} E_2'} \quad \text{if } E_1 \equiv E_1' \text{ and } E_2 \simeq^{\circ}_{\Downarrow} E_2' \qquad (\simeq^{*}_{\Downarrow}1)$$

$$X \simeq^{*}_{\Downarrow} X \qquad (\simeq^{*}_{\Downarrow}2)$$

$$\mathbf{0} \simeq^{*}_{\Downarrow} \mathbf{0} \qquad (\simeq^{*}_{\Downarrow}3)$$

To show that $\simeq^{\circ}_{\Downarrow}$ is closed under rules (cr3)–(cr5) (and hence complete the proof of Theorem 14), it suffices to prove that $\simeq^{\circ}_{\Downarrow}$ coincides with $\simeq^{*}_{\Downarrow}$, because $\simeq^{*}_{\Downarrow}$ is closed under those rules by definition. To do so, we need the following properties of $\simeq^{*}_{\Downarrow}$.

Lemma 16 *For all (open) NCCS process expressions* $E, E_1, E_2, \ldots$ *and all*

(closed) NCCS processes $P_1, P_2, \ldots$, the following properties hold:

$$E \simeq^*_{\Downarrow} E \tag{6}$$

$$E_1 \equiv E_2 \Rightarrow E_1 \simeq^*_{\Downarrow} E_2 \tag{7}$$

$$E_1 \simeq^\circ_{\Downarrow} E_2 \Rightarrow E_1 \simeq^*_{\Downarrow} E_2 \tag{8}$$

$$E_1 \simeq^*_{\Downarrow} E_2 \;\&\; E'_1 \simeq^*_{\Downarrow} E'_2 \Rightarrow E'_1[E_1/X] \simeq^*_{\Downarrow} E'_2[E_2/X] \tag{9}$$

$$E_1 \simeq^*_{\Downarrow} E_2 \Rightarrow E_1[\sigma] \simeq^*_{\Downarrow} E_2[\sigma] \tag{10}$$

$$P'_1|P''_1 \equiv P_1 \;\&\; P_1 \simeq^*_{\Downarrow} P_2 \Rightarrow \exists P'_2, P''_2\,(P'_1 \simeq^*_{\Downarrow} P'_2 \;\&\; P''_1 \simeq^*_{\Downarrow} P''_2 \;\&\; P'_2|P''_2 \simeq_{\Downarrow} P_2) \tag{11}$$

$$(\nu x)P'_1 \equiv P_1 \;\&\; P_1 \simeq^*_{\Downarrow} P_2 \Rightarrow \exists P'_2\,(P'_1 \simeq^*_{\Downarrow} P'_2 \;\&\; (\nu x)P'_2 \simeq_{\Downarrow} P_2) \tag{12}$$

$$\ell.P'_1 \equiv P_1 \;\&\; P_1 \simeq^*_{\Downarrow} P_2 \Rightarrow \exists P'_2\,(P'_1 \simeq^*_{\Downarrow} P'_2 \;\&\; \ell.P'_2 \simeq_{\Downarrow} P_2) \tag{13}$$

$$\mathit{fix}(X{=}E_1) \equiv P_1 \;\&\; P_1 \simeq^*_{\Downarrow} P_2 \Rightarrow \exists E_2\,(E_1 \simeq^*_{\Downarrow} E_2 \;\&\; \mathit{fix}(X{=}E_2) \simeq_{\Downarrow} P_2) \tag{14}$$

$$N'_1 + N''_1 \equiv P_1 \;\&\; P_1 \simeq^*_{\Downarrow} P_2 \Rightarrow \exists N'_2, N''_2\,(N'_1 \simeq^*_{\Downarrow} N'_2 \;\&\; N''_1 \simeq^*_{\Downarrow} N''_2 \;\&\; N'_2 + N''_2 \simeq_{\Downarrow} P_2) \tag{15}$$

$$E_1 \simeq^*_{\Downarrow} E_2 \Rightarrow E_2\ (\simeq^*_{\Downarrow})^{\mathrm{tc}}\ E_1 \tag{16}$$

where in (10) σ *is any permutation of the set of names, and in* (16) $(\simeq^*_{\Downarrow})^{\mathrm{tc}}$ *denotes the* transitive closure *of* $\simeq^*_{\Downarrow}$.

PROOF. Property (6) is easily proved by induction on the structure of the process expression E, and then properties (7) and (8) follow from this and rule ($\simeq^*_{\Downarrow}$1). Property (9) is proved by induction on the derivation of $E'_1 \simeq^*_{\Downarrow} E'_2$, using the fact that the same substitution property holds for $\simeq^\circ_{\Downarrow}$ (by definition) and for $\equiv$ (an easily verified fact). Property (10) is easily proved by induction on the derivation of $E_1 \simeq^*_{\Downarrow} E_2$. It is needed for the proofs of properties (11)–(15), to ensure that the processes asserted to exist on the right-hand sides of the implications can be chosen with their free names different from any given finite set of names not occurring free in the processes on the left-hand sides. Each of these properties is established by induction on the derivation of $P_1 \simeq^*_{\Downarrow} P_2$. Finally, property (16) follows by induction on the derivation of $E_1 \simeq^*_{\Downarrow} E_2$ using the fact that $\simeq_{\Downarrow}$ and $\equiv$ are symmetric relations, together with properties (6)–(8). □

The key property of $\simeq^*_{\Downarrow}$ is given by the following lemma. The presence of the structural congruence relation $\equiv$ introduces an extra complication compared with [10, Theorem 1] which is dealt with using properties (11)–(15).

Lemma 17

$$P_1 \Downarrow \ell.P'_1 \;\&\; P_1 \simeq^*_{\Downarrow} P_2 \Rightarrow \exists P'_2\,(P_2 \Downarrow \ell.P'_2 \;\&\; P'_1 \simeq^*_{\Downarrow} P'_2)$$

PROOF. We proceed by induction on the derivation of $P_1 \Downarrow \ell.P_1'$. More precisely, we show that

$$\mathcal{R} \stackrel{\text{def}}{=} \{(P_1, K) \mid \forall \ell, P_1', P_2\,(K = \ell.P_1' \;\&\; P_1 \simeq^*_{\Downarrow} P_2 \Rightarrow \exists P_2'\,(P_2 \Downarrow \ell.P_2' \;\&\; P_1' \simeq^*_{\Downarrow} P_2'))\}$$

is closed under the rules in Definition 2.

Case ($\Downarrow$0). Closure under this rule follows immediately from the fact that $\simeq^*_{\Downarrow}$ is closed under rule ($\simeq^*_{\Downarrow}$1), together with the fact that $\equiv$ is contained in $\simeq_{\Downarrow}$ (Lemma 7).

Case ($\Downarrow$1). We have to show

$$((N_1 + \ell.P_1')|P_1'' \,,\, \ell.(P_1'|P_1'')) \in \mathcal{R}. \tag{17}$$

But if $(N_1 + \ell.P_1')|P_1'' \simeq^*_{\Downarrow} P_2$, then by property (11), there are P_2', and P_2'' such that

$$(N_1 + \ell.P_1') \simeq^*_{\Downarrow} P_2' \tag{18}$$
$$P_1'' \simeq^*_{\Downarrow} P_2'' \tag{19}$$
$$P_2'|P_2'' \simeq_{\Downarrow} P_2. \tag{20}$$

By property (15) applied to (18), there are N_2' and N_2'' such that

$$N_1 \simeq^*_{\Downarrow} N_2' \tag{21}$$
$$\ell.P_1' \simeq^*_{\Downarrow} N_2'' \tag{22}$$
$$N_2' + N_2'' \simeq_{\Downarrow} P_2'. \tag{23}$$

By property (13) applied to (22), there is P_2''' such that

$$P_1' \simeq^*_{\Downarrow} P_2''' \qquad \ell.P_1''' \simeq_{\Downarrow} N_2''. \tag{24}$$

Since $\ell.P_2''' \Downarrow \ell.P_2'''$, it follows from this last equivalence that $N_2'' \Downarrow \ell.Q_2'''$ holds for some Q_2''' with

$$P_2''' \simeq_{\Downarrow} Q_2'''. \tag{25}$$

By parts (iv) and (i) of Lemma 5 applied to $N_2'' \Downarrow \ell.Q_2'''$ we get

$$(N_2' + N_2'')|P_2'' \Downarrow \ell.(Q_2'''|P_2''). \tag{26}$$

By Lemma 7 on (23) we get $(N_2' + N_2'')|P_2'' \simeq_{\Downarrow} P_2'|P_2''$ and hence by (20)

$$(N_2' + N_2'')|P_2'' \simeq_{\Downarrow} P_2.$$

Then by (26), $P_2 \Downarrow \ell.Q$ for some Q with $Q_2'''|P_2'' \simeq_{\Downarrow} Q$. Hence once again using Lemma 7, this time on (25), we get

$$P_2'''|P_2'' \simeq_{\Downarrow} Q_2'''|P_2'' \simeq_{\Downarrow} Q. \tag{27}$$

Applying the congruence property (cr1) that is part of the definition of $\simeq^*_{\Downarrow}$ to (24) and (19), we get $P_1'|P_1'' \simeq^*_{\Downarrow} P_2'''|P_2''$; and then $(\simeq^*_{\Downarrow}1)$ applied to this and (27) yields $P_1'|P_1'' \simeq^*_{\Downarrow} Q$, as required for (17).

Case ($\Downarrow$2). Suppose $(P_1', \ell.Q_1')$, $(P_1'', \bar{\ell}.Q_1'')$, and $(Q_1'|Q_1'', \ell'.Q_1)$ are all in $\mathcal{R}$. We have to prove that $(P_1'|P_1'', \ell'.Q_1) \in \mathcal{R}$, i.e. that if

$$P_1'|P_1'' \simeq^*_{\Downarrow} P_2 \tag{28}$$

then $P_2 \Downarrow \ell'.Q_2$ for some Q_2 satisfying $Q_1 \simeq^*_{\Downarrow} Q_2$.

Property (11) applied to (28) implies that there are P_2' and P_2'' so that

$$P_1' \simeq^*_{\Downarrow} P_2' \tag{29}$$
$$P_1'' \simeq^*_{\Downarrow} P_2'' \tag{30}$$
$$P_2'|P_2'' \simeq_{\Downarrow} P_2. \tag{31}$$

Since $(P_1', \ell.Q_1'), (P_1'', \bar{\ell}.Q_1'') \in \mathcal{R}$, from (29) and (30) we get

$$P_2' \Downarrow \ell.Q_2' \tag{32}$$
$$P_2'' \Downarrow \bar{\ell}.Q_2'' \tag{33}$$

for some Q_2' and Q_2'' satisfying $Q_1' \simeq^*_{\Downarrow} Q_2'$ and $Q_1'' \simeq^*_{\Downarrow} Q_2''$, and hence by (cr1) also satisfying $Q_1'|Q_1'' \simeq^*_{\Downarrow} Q_2'|Q_2''$. Then since $(Q_1'|Q_1'', \ell'.Q_1) \in \mathcal{R}$, there is some Q such that

$$Q_2'|Q_2'' \Downarrow \ell'.Q \tag{34}$$
$$Q_1 \simeq^*_{\Downarrow} Q. \tag{35}$$

Evaluation rule ($\Downarrow$2) on (32)–(34) yields $P_2'|P_2'' \Downarrow \ell'.Q$. Therefore from (31) we get that $P_2 \Downarrow \ell'.Q_2$ holds for some Q_2 satisfying $Q \simeq_{\Downarrow} Q_2$, and hence by $(\simeq^*_{\Downarrow}1)$ on (35), also satisfying $Q_1 \simeq^*_{\Downarrow} Q_2$, as required.

Case ($\Downarrow$3). The argument in this case is similar to the previous one and is omitted.

Case ($\Downarrow$4). Suppose $(E_1[\mathit{fix}(X{=}E_1)/X]|P_1'\,,\,\ell.Q_1) \in \mathcal{R}$. We have to show that $(\mathit{fix}(X{=}E_1)|P_1'\,,\,\ell.Q_1) \in \mathcal{R}$, i.e. that if

$$\mathit{fix}(X{=}E_1)|P_1' \simeq^*_{\Downarrow} P_2 \tag{36}$$

then $P_2 \Downarrow \ell.Q_2$ for some Q_2 satisfying $Q_1 \simeq^*_{\Downarrow} Q_2$.

Properties (11) and (14) plus Lemma 7 applied to (36) imply that there are E_2 and P_2' so that

$$E_1 \simeq^*_{\Downarrow} E_2 \tag{37}$$
$$P_1' \simeq^*_{\Downarrow} P_2' \tag{38}$$
$$\mathit{fix}(X{=}E_2)|P_2' \simeq_{\Downarrow} P_2. \tag{39}$$

Now by property (cr3) of $\simeq^*_{\Downarrow}$, from (37) we get $\mathit{fix}(X{=}E_1) \simeq^*_{\Downarrow} \mathit{fix}(X{=}E_2)$ and hence by (9) that $E_1[\mathit{fix}(X{=}E_1)/X] \simeq^*_{\Downarrow} E_2[\mathit{fix}(X{=}E_2)/X]$. Property (cr1) of $\simeq^*_{\Downarrow}$ applied to this and (38) yields

$$E_1[\mathit{fix}(X{=}E_1)/X]|P_1' \simeq^*_{\Downarrow} E_2[\mathit{fix}(X{=}E_2)/X]|P_2'.$$

So since $(E_1[\mathit{fix}(X{=}E_1)/X]|P_1'\,,\,\ell.Q_1) \in \mathcal{R}$, it follows that there is some Q_2' with

$$E_2[\mathit{fix}(X{=}E_2)/X]|P_2' \Downarrow \ell.Q_2' \tag{40}$$
$$Q_1 \simeq^*_{\Downarrow} Q_2'. \tag{41}$$

By ($\Downarrow$4) on (40), we get $\mathit{fix}(X{=}E_2)|P_2' \Downarrow \ell.Q_2'$. Therefore by (39) there is some Q_2 satisfying $P_2 \Downarrow \ell.Q_2$ and $Q_2' \simeq_{\Downarrow} Q_2$, and hence by ($\simeq^*_{\Downarrow}1$) on (41), also satisfying $Q_1 \simeq^*_{\Downarrow} Q_2$, as required. $\Box$

Proof of Theorem 14 We noted above that it suffices to prove that $\simeq^\circ_{\Downarrow}$ coincides with $\simeq^*_{\Downarrow}$. Property (8) of Lemma 16 gives the inclusion one way. For the reverse inclusion, it suffices to prove for *closed* process expressions that

$$P_1 \simeq^*_{\Downarrow} P_2 \Rightarrow P_1 \simeq_{\Downarrow} P_2 \tag{42}$$

since the general case for open expressions follows from (9) in Lemma 16 and the way $\simeq^\circ_{\Downarrow}$ is defined from $\simeq_{\Downarrow}$ (Definition 13). To prove (42), we exploit the fact that $\simeq_{\Downarrow}$ is the largest evaluation bisimulation. Since $\simeq^*_{\Downarrow}$ is by definition closed under the congruence rule (cr1), it follows from Lemma 17 that $\simeq^*_{\Downarrow}$ restricted to closed processes is an evaluation simulation (cf. Definition 6). Since the definition of $\simeq^*_{\Downarrow}$ is not symmetric (because of rule ($\simeq^*_{\Downarrow}1$)), one cannot immediately conclude that it is also an evaluation bisimulation. However, by property (16) of Lemma 16 its transitive closure, $(\simeq^*_{\Downarrow})^{tc}$, is a symmetric relation; and clearly Lemma 17 implies that

$$P_1 \Downarrow \ell.P_1' \;\&\; P_1\ (\simeq^*_{\Downarrow})^{\mathrm{tc}}\ P_2 \Rightarrow \exists P_2'\,(P_2 \Downarrow \ell.P_2' \;\&\; P_1'\ (\simeq^*_{\Downarrow})^{\mathrm{tc}}\ P_2').$$

Thus $(\simeq^*_{\Downarrow})^{\mathrm{tc}}$ is an evaluation bisimulation. Hence it is contained in $\simeq_{\Downarrow}$ and hence so is $\simeq^*_{\Downarrow}$, as required. $\Box$

Quite possibly there are other, more direct ways of proving this theorem for a calculus as simple as NCCS (for example, via the characterisation of evaluation bisimilarity given in by Theorem 24 in the next section). However, the above adaptation of 'Howe's method' [10,11] has the distinct advantage of *robustness*: our experience shows that the same method can be used for more complicated calculi, such as those considered in Sections 4.1 and 4.2.

We believe that evaluation to committed form and the associated notion of evaluation bisimilarity have a certain naturalness for the type of interaction embodied in CCS. The fact that $\simeq_{\Downarrow}$ yields a congruent notion of process equivalence for NCCS is at least some evidence in favour of this belief. But two interrelated questions immediately arise. What equational laws are validated by $\simeq_{\Downarrow}$, and what is its relationship to other, known process equivalences? We address both questions in the next section.

3 Evaluation versus Transition

The standard *labelled transition system* for CCS [14], adapted to the syntax of NCCS, takes the form $P \xrightarrow{\alpha} P'$, where P and P' are NCCS processes and the action α is either a name, a co-name, or the distinguished internal action τ. Labelled transitions are inductively generated by the axiom and rules in Figure 1. We write $\xrightarrow{\tau^*}$ for the reflexive-transitive closure of the relation $\xrightarrow{\tau}$, and write $P \xrightarrow{\tau^*\ell} P'$ (respectively $P \xrightarrow{\tau^*\ell\tau^*} P'$) to mean that $P \xrightarrow{\tau^*} P'' \xrightarrow{\ell} P'$ (respectively $P \xrightarrow{\tau^*\ell} P'' \xrightarrow{\tau^*} P'$) holds for some P''. Finally, recall from [14] that two processes are *strongly equivalent*, $P_1 \sim P_2$, if they are related by some symmetric binary relation $\mathcal{R}$ satisfying

$$\forall P_1, P_2, P_1', \alpha\,(P_1\ \mathcal{R}\ P_2\ \&\ P_1 \xrightarrow{\alpha} P_1' \Rightarrow \exists P_2'\,(P_2 \xrightarrow{\alpha} P_2'\ \&\ P_1\ \mathcal{R}\ P_2')).$$

We recall some facts about strong equivalence that we will need (see [14, Chapter 4]).

Lemma 18*(i) $\sim$ is a congruence relation (for the NCCS syntax) containing structural congruence and satisfying a 'back-and-forth' property with respect to actions of the form $\tau^*\ell$, i.e. if $P_1 \sim P_2$ and $P_1 \xrightarrow{\tau^*\ell} P_1'$, then $P_2 \xrightarrow{\tau^*\ell} P_2'$ for some P_2' with $P_1' \sim P_2'$.*

(ii) Recursive processes are strongly equivalent to their unfoldings: $\mathit{fix}(X{=}E) \sim E[\mathit{fix}(X{=}E)/X]$.

$$\ell.P \xrightarrow{\ell} P \qquad (\to 1)$$

$$\frac{P_1 \xrightarrow{\alpha} P_1'}{P_1|P_2 \xrightarrow{\alpha} P_1'|P_2} \qquad \frac{P_2 \xrightarrow{\alpha} P_2'}{P_1|P_2 \xrightarrow{\alpha} P_1|P_2'} \qquad (\to 2)$$

$$\frac{P_1 \xrightarrow{\ell} P_1' \quad P_2 \xrightarrow{\bar{\ell}} P_2'}{P_1|P_2 \xrightarrow{\tau} P_1'|P_2'} \qquad (\to 3)$$

$$\frac{P \xrightarrow{\alpha} P'}{(\nu x)P \xrightarrow{\alpha} (\nu x)P'} \quad \text{if } \alpha \notin \{x, \bar{x}\} \qquad (\to 4)$$

$$\frac{E[\mathit{fix}(X{=}E)/X] \xrightarrow{\alpha} P}{\mathit{fix}(X{=}E) \xrightarrow{\alpha} P} \qquad (\to 5)$$

$$\frac{N_1 \xrightarrow{\ell} P}{N_1 + N_2 \xrightarrow{\ell} P} \qquad \frac{N_2 \xrightarrow{\ell} P}{N_1 + N_2 \xrightarrow{\ell} P} \qquad (\to 6)$$

Fig. 1. Rules for NCCS labelled transitions

Lemma 19 *For all NCCS processes P, P', Q, committed processes K, and labels ℓ*

(i) If $P \xrightarrow{\ell} P'$ then $P \Downarrow \ell.P'$
(ii) If $P \xrightarrow{\tau} P'$ and $P'|Q \Downarrow K$, then $P|Q \Downarrow K$.
(iii) If $P \Downarrow \ell.P'$, then $P \xrightarrow{\tau^*\ell} P''$ for some P'' with $P'' \sim P'$.

PROOF. Properties (i) and (ii) are proved by induction on the derivation of labelled transitions from the rules $(\to 1)$–$(\to 6)$. Property (iii) is proved by induction on the derivation of the evaluation $P \Downarrow \ell.P'$ from the rules $(\Downarrow 0)$–$(\Downarrow 4)$ using the properties of $\sim$ mentioned in the preceding lemma. □

Remark 20 *In fact the proof of part (iii) of the lemma is valid with $\sim$ replaced by any relation satisfying properties (i) and (ii) of Lemma 18. Structural congruence itself possesses the first of these properties. However, it does not possess the second since we have not chosen to regard the unfolding of recursive process expressions as 'structural'—because of the use of substitution involved. (This is in contrast to the unfolding of* replicated *processes, $!P \equiv !P|P$, present in the π-calculus structural congruence [16].) Consequently, in part (iii) of the lemma we have to make do with the next best thing to $\equiv$, namely strong equivalence. For example, one has $\mathit{fix}(X{=}x.X)|y.\mathbf{0} \Downarrow y.P$ for $P = x.\mathit{fix}(X{=}x.X)$, but $\mathit{fix}(X{=}x.X)|y.\mathbf{0} \xrightarrow{\tau^*y} P'$ holds only with $P' = \mathit{fix}(X{=}x.X)|\mathbf{0}$ which is strongly equivalent, but not structurally congruent to P.*

Theorem 21 *For all NCCS processes P, P', labels ℓ, and names $x \notin fn(P)$*

(i) $\exists P''\,(P \xrightarrow{\tau^*\ell} P'' \;\&\; P'' \sim P') \Leftrightarrow \exists P''\,(P \Downarrow \ell.P'' \;\&\; P'' \sim P')$.
(ii) $\exists P''\,(P \xrightarrow{\tau^*} P'' \;\&\; P'' \sim P') \Leftrightarrow \exists P''\,(P|x.\mathbf{0} \Downarrow x.P'' \;\&\; P'' \sim P')$.
(iii) $\exists P''\,(P \xrightarrow{\tau^*\ell\tau^*} P'' \;\&\; P'' \sim P') \Leftrightarrow \exists P''\,(P|\bar{\ell}.x.\mathbf{0} \Downarrow x.P'' \;\&\; P'' \sim P')$.

PROOF. Combine Lemma 19 with the following simple properties of the labelled transition system:

$$P \xrightarrow{\ell} P' \Rightarrow \ell \in fn(P) \cup \overline{fn(P)}$$
$$P|x.\mathbf{0} \xrightarrow{\tau^*x} P' \Rightarrow \exists P''\,(P \xrightarrow{\tau^*} P'' \;\&\; P''|\mathbf{0} =_\alpha P')$$
$$P|\bar{\ell}.x.\mathbf{0} \xrightarrow{\tau^*x} P' \Rightarrow \exists P''\,(P \xrightarrow{\tau^*\ell\tau^*} P'' \;\&\; P''|\mathbf{0} =_\alpha P')$$

where $x \notin fn(P) \cup \{\ell\}$. $\square$

Note that modulo strong equivalence, part (i) of the theorem characterises evaluation to committed form in terms of transition, whereas part (ii) characterises NCCS *reduction*—i.e. zero or more τ-transitions—in terms of evaluation. The theorem also yields the following characterisation of the restriction to NCCS of CCS *observation equivalence*, $\approx$ (which coincides with observation *congruence*, because of the limited form of summation in NCCS). Recall from [14] that two processes are observation equivalent if they are related by some *weak bisimulation*—a relation $\mathcal{R}$ such that both $\mathcal{R}$ and $\mathcal{R}^{-1}$ satisfy: for all P_1, P_2 if $P_1\ \mathcal{R}\ P_2$ then

$$P_1 \xrightarrow{\tau} P_2 \Rightarrow \exists P_2'\,(P_2 \xrightarrow{\tau^*} P_2' \;\&\; P_1'\ \mathcal{R}\ P_2') \qquad \text{(wb1)}$$
$$\forall \ell\,(P_1 \xrightarrow{\ell} P_1' \Rightarrow \exists P_2'\,(P_2 \xrightarrow{\tau^*\ell\tau^*} P_2' \;\&\; P_1'\ \mathcal{R}\ P_2')) \qquad \text{(wb2)}$$

Corollary 22 *Observation equivalence is the largest symmetric binary relation $\mathcal{R}$ on NCCS processes satisfying that if $P_1\ \mathcal{R}\ P_2$ then*

$$P_1|x.\mathbf{0} \Downarrow x.P_1' \Rightarrow \exists P_2'\,(P_2|x.\mathbf{0} \Downarrow x.P_2' \;\&\; P_1'\ \mathcal{R}\ P_2')$$
$$P_1|\bar{\ell}.x.\mathbf{0} \Downarrow x.P_1' \Rightarrow \exists P_2'\,(P_2|\bar{\ell}.x.\mathbf{0} \Downarrow x.P_2' \;\&\; P_1'\ \mathcal{R}\ P_2')$$

hold for any label ℓ and any name $x \notin fn(P_1P_2)$ (or equivalently, for some such x, by the equivariance properties of evaluation with respect to permuting free names).

PROOF. Since strong equivalence is contained in $\approx$, it follows easily from Theorem 21 that $\approx$ is such an $\mathcal{R}$. Conversely, one can also use the theorem to show that for any such $\mathcal{R}$, the composition $\sim\mathcal{R}\sim$ is a weak bisimulation and hence $\mathcal{R} \subseteq \sim\mathcal{R}\sim \subseteq \approx$. $\square$

Part (i) of Theorem 21 immediately suggests a way to modify the notion of observation equivalence in order to obtain a transition-based bisimilarity coinciding with the notion of evaluation bisimilarity introduced in the previous section—namely change clause (wb2) to

$$\forall \ell \, (P_1 \stackrel{\ell}{\rightarrow} P_1' \Rightarrow \exists P_2' \, (P_2 \stackrel{\tau^* \ell}{\longrightarrow} P_2' \;\&\; P_1' \; \mathcal{R} \; P_2')) \qquad \text{(wb2}')$$

Definition 23 (Delay bisimulation equivalence) *A binary relation $\mathcal{R}$ between NCCS processes is a* delay simulation *if $P_1 \; \mathcal{R} \; P_2$ implies that both (wb1) and (wb2′) hold. If $\mathcal{R}^{-1}$ is also a delay simulation, we say $\mathcal{R}$ is a* delay bisimulation. *Two processes are* delay bisimilar*, written $P_1 \simeq_{\rm dl} P_2$, if $P_1 \; \mathcal{R} \; P_2$ holds for some delay bisimulation $\mathcal{R}$.*

This notion of process equivalence is studied by Weijland [29] who credits its formulation to Milner [13]; see also [6]. It is also the specialisation to first order processes of Sangiorgi's notion of *weak context bisimilarity* for higher order process calculi, studied in [27].

Theorem 24 *For NCCS processes, evaluation bisimilarity coincides with delay bisimilarity.*

PROOF. We will need the following facts about delay bisimulation equivalence which can easily be proved from the definition.

(a) $\simeq_{\rm dl}$ is the greatest delay bisimulation, is an equivalence relation, and contains strong equivalence.
(b) If $P_1 \simeq_{\rm dl} P_2$, then $P_1|Q \simeq_{\rm dl} P_2|Q$. (In fact delay bisimilarity is an NCCS congruence.)

These facts, together with part (i) of Theorem 21, imply that $\simeq_{\rm dl}$ is an evaluation bisimulation. Thus $P_1 \simeq_{\rm dl} P_2$ implies $P_1 \simeq_{\Downarrow} P_2$. For the converse implication it suffices to show that $\simeq_{\Downarrow}$ is a delay (bi)simulation. So suppose $P_1 \simeq_{\Downarrow} P_2$. There are two cases to consider.

Case $P_1 \stackrel{\tau}{\rightarrow} P_1'$. We have to show $P_2 \stackrel{\tau^*}{\rightarrow} P_2'$ for some P_2' with $P_1' \simeq_{\Downarrow} P_2'$. Picking any $x \notin fn(P_1 P_2)$, $P_1|x.\mathbf{0} \Downarrow x.P_1''$ holds for some $P_1'' \sim P_1'$ by Theorem 21(ii). Since $P_1 \simeq_{\Downarrow} P_2$ we also have $P_1|x.\mathbf{0} \simeq_{\Downarrow} P_2|x.\mathbf{0}$ (by Lemma 7), so $P_2|x.\mathbf{0} \Downarrow x.P_2''$ for some P_2'' with $P_1'' \simeq_{\Downarrow} P_2''$. By 21(ii) again, $P_2 \stackrel{\tau^*}{\rightarrow} P_2'$ for some $P_2' \sim P_2''$. By (a), since $P_i' \sim P_i''$ $(i = 1, 2)$, we also have that $P_i' \simeq_{\rm dl} P_i''$; and hence by the first part of the proof we have that $P_i' \simeq_{\Downarrow} P_i''$. Since $P_1'' \simeq_{\Downarrow} P_2''$, we do indeed have $P_1' \simeq_{\Downarrow} P_2'$, as required.

Case $P_1 \xrightarrow{\ell} P_1'$**.** We have to show $P_2 \xrightarrow{\tau^*\ell} P_2'$ for some P_2' with $P_1' \simeq_{\Downarrow} P_2'$. The proof is similar to the previous case, but using part (i) of Theorem 21. $\Box$

The theorem provides a simple way of establishing the τ-laws for evaluation bisimilarity mentioned in Example 9, since it is easy to see that they hold up to delay bisimulation equivalence. Indeed the theorem provides one route to establishing a complete axiomatisation of the equations between finite (i.e. *fix*-free), closed NCCS process expressions that are satisfied by evaluation bisimilarity—e.g. by reusing known axiomatisations for delay bisimilarity [29,6]. Since the primary concern of this paper is to introduce the notions of evaluation to committed form and evaluation bisimilarity for a range of calculi, we do not pursue the topic of axiomatisations any further here.

Remark 25 (Internal non-deterministic choice) *Just as one can code τ-guarded summation in NCCS (Definition 8),* internal non-deterministic choice*, $\oplus$, can be defined up to evaluation bisimilarity. To be more precise, consider extending the syntax of NCCS process expressions:*

$$E ::= \cdots \mid E \oplus E.$$

Extend the evaluation relation of Definition 2 with the rules

$$\frac{P_i|Q \Downarrow K}{(P_1 \oplus P_2)|Q \Downarrow K} \quad (i = 1, 2) \qquad (\Downarrow\oplus)$$

and extend the labelled transition relation with the usual axioms for internal choice (cf. [7])

$$P_1 \oplus P_2 \xrightarrow{\tau} P_i \quad (i = 1, 2).$$

Then Theorem 24 holds for this extended system. Moreover

$$P_1 \oplus P_2 \simeq_{\Downarrow} (\nu x)(x.P_1|\bar{x}.\mathbf{0}|x.P_2) \quad (x \notin fn(P_1P_2)) \qquad (43)$$

because it is simple enough to see that these two processes are delay bisimilar. Thus internal choice is already definable in NCCS up to evaluation bisimilarity. (The evaluation semantics of other forms of choice are considered in Section 4.4.)

We can also use Theorem 24 to resolve the question raised in the previous section about the extent to which quantification over contexts $[-]|Q$ in the definition of evaluation bisimulation can be avoided. As the following result shows, we need only consider a single context $[-]|x.\mathbf{0}$, with x fresh. This result is in the same spirit as Sangiorgi's characterisation of his weak context bisimilarity in terms of 'normal bisimulations': see [27, Theorem 7.4]. However, the

reduction in context quantification we are dealing with here is much less subtle than that involved in going from context bisimilarity to normal bisimilarity. We have more to say about a higher order version of evaluation bisimilarity in Section 4.1.

Theorem 26 *Evaluation bisimilarity is the largest symmetric binary relation $\mathcal{R}$ on NCCS processes satisfying that if $P_1\ \mathcal{R}\ P_2$, then for any name $x \notin fn(P_1P_2)$ (or equivalently, for some such x) and any P_1', ℓ*

$$P_1|x.\mathbf{0} \Downarrow \ell.P_1' \Rightarrow \exists P_2'\,(P_2|x.\mathbf{0} \Downarrow \ell.P_2' \;\&\; P_1'\ \mathcal{R}\ P_2'). \tag{44}$$

PROOF. It follows from the definition of $\simeq_{\Downarrow}$ that it is a symmetric relation satisfying (44). Conversely, given such an $\mathcal{R}$, we have to show $\mathcal{R} \subseteq \simeq_{\Downarrow}$. We use a form of 'bisimulation up to context' (and up to $\sim$) technique reminiscent of those considered in [26].

Let $\bar{\mathcal{R}}$ be the relation inductively defined by the following axiom and rule:

$$P_1\ \bar{\mathcal{R}}\ P_2 \quad \text{if } P_1 \sim Q_1\ \mathcal{R}\ Q_2 \sim P_2 \tag{45}$$

$$\frac{Q_1\ \bar{\mathcal{R}}\ Q_2}{P_1\ \bar{\mathcal{R}}\ P_2} \quad \begin{array}{l}\text{if } P_i|\bar{x}.\mathbf{0} \sim Q_i\ (i=1,2)\\ \text{and } x \notin fn(P_1P_2).\end{array} \tag{46}$$

Note that $\bar{\mathcal{R}}$ contains $\mathcal{R}$ and is symmetric, because $\mathcal{R}$ is. It suffices to show that $\bar{\mathcal{R}}$ is a delay simulation: for then it is also a delay bisimulation and so it, and hence also $\mathcal{R}$, is contained in the largest one, $\simeq_{\mathrm{dl}}$, which by Theorem 24 is equal to $\simeq_{\Downarrow}$. So one must prove that $P_1\ \bar{\mathcal{R}}\ P_2$ implies that (wb1) and (wb2′) hold of $\bar{\mathcal{R}}$. This can be done by induction on the derivation of $P_1\ \bar{\mathcal{R}}\ P_2$ from (45) and (46), using Theorem 21. □

4 Further Topics

In this section we outline briefly some further developments of the approach to process calculi based upon evaluation to committed form.

4.1 Evaluation bisimilarity for higher order calculi

Consider a higher order version of NCCS in which synchronised communication involves passing process expressions. Input-committed processes now take the form $x.F$ where $F = (X)E$ is an *abstraction* (and free occurrences of the

process variable X in E are bound in F); such a process is ready to receive a process P on channel x and then continue with $E[P/X]$. Output-committed processes take the form $\bar{x}.C$ where $C = (\nu\vec{x})\langle P_1\rangle P_2$ is a *concretion* (free occurrences of the names $\vec{x}$ in P_1 or P_2 are bound in C); such a process is ready to send P_1 on channel x and then continue with P_2, all within a scope in which the names $\vec{x}$ are restricted. See for example Sangiorgi [27] for further syntactic details and a labelled transition system formalising the intended input/output behaviour. Transitions now take the form $P \xrightarrow{\tau} P'$ and $P \xrightarrow{\ell} A$, where in the second case if ℓ is a name then A is an abstraction, and if ℓ is a co-name then A is a concretion. First order prefixing can be regarded as a special case of higher order prefixing if we define $x.P$ to mean $x.(X)P$ where $X \notin fv(P)$, and define $\bar{x}.P$ to mean $\bar{x}.\langle\mathbf{0}\rangle P$. (Sangiorgi also considers τ prefixing, but as we noted in Definition 8, this is definable in terms of label prefixing, composition and restriction.)

In *loc. cit.* Sangiorgi considers the problem of defining a suitable bisimilarity which, unlike previous attempts, identifies some pairs of processes (such as $\bar{y}.\langle\mathbf{0}\rangle\mathbf{0}$ and $(\nu x)\bar{y}.\langle x.\mathbf{0}\rangle\mathbf{0}$) which one can argue should be behaviourly equivalent in the presence of statically bound restrictions. He develops a congruent notion of bisimilarity, called (*weak*) *context bisimilarity*, and shows that it has the desired properties. Weak context bisimilarity is a generalisation to higher order of the notion of delay bisimilarity. Indeed the form of the definition is exactly as in Definition 23, except that in clause (wb2′) P_1' and P_2' are now abstractions or concretions (according to whether ℓ is a name or a co-name). So one has to extend the relation $\mathcal{R}$ from processes to these syntactic categories in order to assert in (wb2′) that P_1' and P_2' are related by $\mathcal{R}$. This is done by defining

$$\begin{array}{rcl} F_1\ \mathcal{R}\ F_2 & \stackrel{\text{def}}{\Leftrightarrow} & \forall C\,((F_1 \bullet C)\ \mathcal{R}\ (F_2 \bullet C)) \\ C_1\ \mathcal{R}\ C_2 & \stackrel{\text{def}}{\Leftrightarrow} & \forall F\,((C_1 \bullet F)\ \mathcal{R}\ (C_2 \bullet F)) \end{array} \tag{47}$$

where $F \bullet C \stackrel{\text{def}}{=} (\nu\vec{x})(E[P_1/X]|P_2)$ when $F = (X)E$, $C = (\nu\vec{x})\langle P_1\rangle P_2$, and $\vec{x} \cap fn(E) = \emptyset$; $C \bullet F$ is defined symmetrically.

Interestingly, it turns out that Theorem 24 easily extends to a coincidence of a higher order version of evaluation bisimilarity with Sangiorgi's weak context bisimilarity, as we now indicate. First, evaluation to committed form extends very naturally to the higher order case. We replace ($\Downarrow$2) by

$$\frac{P_1 \Downarrow \ell.A_1 \quad P_2 \Downarrow \bar{\ell}.A_2 \quad A_1 \bullet A_2 \Downarrow K}{P_1|P_2 \Downarrow K}\ .$$

The other evaluation rules remain essentially as in Definition 2, but one also has to suitably extend the notion of structural congruence to abstractions and concretions. Secondly, evaluation bisimilarity also extends naturally to

the higher order calculus: $\simeq_{\Downarrow}$ is the greatest symmetric relation $\mathcal{R}$ on higher order processes such that if $P_1 \ \mathcal{R} \ P_2$ then for all Q, if $P_1|Q \Downarrow \ell.A_1$ then $P_2|Q \Downarrow \ell.A_2$, for some A_2 with $A_1 \ \mathcal{R} \ A_2$ (where $\mathcal{R}$ is extended to abstractions and concretions as in (47)).

Theorem 27 *Higher order evaluation bisimilarity coincides with Sangiorgi's weak context bisimilarity [27, Definition 3.11].*

The proof is very much as for Theorem 24, once one has established the higher order analogue of Theorem 21. The latter uses Sangiorgi's *strong context bisimilarity* [27, Definition 3.1] where Theorem 21 uses strong equivalence, $\sim$. The proof of Theorem 26 also extends: *one can replace the quantification over Q with the use of a single process $Q = x.\mathbf{0}$ (x fresh) without affecting the relation of higher order evaluation bisimilarity.* (It may be that the quantification implicit in the use of (47) can also be reduced along the lines of [27, Section 7] using Sangiorgi's 'Factorisation Theorem' (*loc. cit.*, Theorem 4.7), but we have not checked this.)

4.2 Evaluation to input-committed form

If parallel composition in process calculus plays a role analogous to application in functional languages, then input-committed processes $x(X).E$ are somewhat like lambda abstractions $\lambda X.E$ 'located' at x. (The analogy can be made more precise, as in [15].) Experience with applicative bisimilarity for functional calculi [1,10] suggests considering an evaluation-based approach to process calculi in which the only canonical forms are input-committed processes. For variety, we illustrate how this looks for the π-calculus [16] with asynchronous output and no summation—the 'essence' of the language to judge by recent results [8,21,4]. The syntax of such processes is

$$P ::= x.(x)P \mid \bar{x}.\langle x\rangle \mid \mathbf{0} \mid P|P \mid (\nu x)P \mid !P$$

where x ranges over names. One works modulo a structural congruence relation, $\equiv$, generated by the relevant identities in Definition 1 together with an identity for unfolding replicated processes: $!P \equiv P|!P$. This identity means that we will not need an explicit evaluation rule for replicated processes. Similarly, by building restrictions into the other rules, we can do without an explicit rule for restriction (in other words ($\Downarrow$3) will become derivable). Altogether we arrive at the following remarkably compact evaluation semantics

for this variety of π-calculus.

$$\frac{P_1 \Downarrow y.(x)P_1'}{P_2 \Downarrow y.(x)P_2'} \text{ if } P_1 \equiv P_2 \text{ and } \forall x\,(P_1' \equiv P_2')$$

$$\frac{Q \Downarrow y_1.(x)Q' \quad (\nu\vec{x})Q'[y_2/x] \Downarrow K}{(\nu\vec{x})((\bar{y}_1.\langle y_2\rangle)|Q) \Downarrow K}$$

$$(\nu\vec{x})((y.(x)P)|Q) \Downarrow y.(x)(\nu\vec{x})(P|Q) \quad \text{if } y \notin \vec{x}$$

Then define *input-committed evaluation bisimilarity*, $\simeq_{\mathrm{ic}}$, for this calculus to be the largest symmetric binary relation $\mathcal{R}$ between processes such that if $P_1\ \mathcal{R}\ P_2$, then for all Q, y, and $(x)P_1'$

$$P_1|Q \Downarrow y.(x)P_1' \Rightarrow \exists P_2'\,(P_2|Q \Downarrow y.(x)P_2' \,\&\, \forall x\,(P_1'\ \mathcal{R}\ P_2')).$$

One can adapt the method used for the proof of Theorem 14 to show that $\simeq_{\mathrm{ic}}$ *is a congruence for this* π*-calculus.* We have not investigated the relationship between $\simeq_{\mathrm{ic}}$ and other notions of weak congruence that have been proposed in the literature. This is partly because the work of Honda and Yoshida [9], Fournet and Gonthier [4] and others, suggests that for this kind of asynchronous-output calculus one should observe outputs rather than inputs. It is possible to give a congruent notion of evaluation bisimilarity based on evaluation to output-committed form (which would be $\bar{x}.C$, with C a concretion of the form $(\nu\vec{x})(\langle y\rangle|P)$ in this case), but we do not give the details here.

4.3 Barbed bisimulation

Milner and Sangiorgi [18] introduced the notion of barbed bisimulations for process calculi, based upon a reduction-oriented approach to process semantics. It has proved useful for defining equivalences in the π-calculus and related systems (see [4], for example). The motivations for the evaluation-based approach we have introduced in this paper are quite similar to those expressed in [18]. Technically, the evaluation-to-committed-form approach seems more elegant: barbed bisimilarities are defined using a reduction relation and a convergence predicate and these usually have to be defined from a labelled transition system; whereas evaluation bisimilarity is defined using a single, inductively defined evaluation relation. On the other hand, the evaluation-to-committed-form approach is very much tied to defining equivalences that ignore internal actions (i.e. weak rather than strong equivalences); and it imposes a harder discipline than the reduction-based approach, since it may be easier to find reasonable notions of reduction and convergence for some 'new' process calculus which may arise.

Whatever the *pros* and *cons* of each approach, observe that for NCCS at least, the results of Section 3 mean that weak barbed bisimilarities can be defined starting just from the evaluation relation. For we saw in Theorem 21(ii) that reduction can be defined in terms of evaluation (modulo strong equivalence); and if we follow [18, Section 5.1] and define

$$P\Downarrow \quad \stackrel{\text{def}}{\Leftrightarrow} \quad \exists P', \ell\,(P \xrightarrow{\tau^*\ell\tau^*} P'),$$

then by Lemma 19 we have that

$$P\Downarrow \Leftrightarrow \exists K\,(P \Downarrow K)$$

(fortunately, from a notational point of view). Here is a characterisation of observation equivalence for NCCS as a barbed congruence whose definition is phrased in terms of evaluation. It seems unlikely that NCCS evaluation bisimilarity (i.e. delay bisimilarity) can be given a 'barbed' characterisation.

Theorem 28 *Observation equivalence, $\approx$, is the largest symmetric binary relation $\mathcal{R}$ on NCCS processes satisfying that if $P_1\ \mathcal{R}\ P_2$ then for any NCCS context $C[-]$, process P_1', and name $x \notin fn(C[P_1], C[P_2])$*

$$\begin{array}{c} C[P_1]|x.\mathbf{0} \Downarrow x.P_1' \Rightarrow \exists P_2'\,(C[P_2]|x.\mathbf{0} \Downarrow x.P_2' \ \&\ P_1'\ \mathcal{R}\ P_2') \\ C[P_1]\Downarrow \Rightarrow C[P_2]\Downarrow. \end{array} \tag{48}$$

PROOF. Let $\approx_b$ denote the largest such relation. Note that $\approx$ is an NCCS congruence, because of the restricted form of summation in the calculus. Using Theorem 21 it follows that $\approx$ is a relation satisfying the property stated in the theorem, and hence is contained in the largest one, i.e.

$$\approx \subseteq \approx_b. \tag{49}$$

To show the reverse containment, we verify that $\approx_b$ is a weak simulation, i.e. satisfies properties (wb1) and (wb2) mentioned in Section 3. In doing so, we will make use of the fact that $\approx_b$ is an NCCS congruence—this is clear from its definition.

$\approx_b$ has property (wb1). Suppose $P_1 \approx_b P_2$ and that $P_1 \xrightarrow{\tau} P_1'$. We have to find P_2' such that $P_2 \xrightarrow{\tau^*} P_2'$ and $P_1' \approx_b P_2'$.

Choosing any $x \notin fn(P_1P_2)$, by Lemma 19(ii) we have $P_1|x.\mathbf{0} \Downarrow x.P_1'$; hence by property (48) of $\approx_b$, $P_2|x.\mathbf{0} \Downarrow x.P_2''$ for some P_2'' with $P_1' \approx_b P_2''$. Then since $x \notin fn(P_2)$, by Theorem 21(ii), there is some P_2' with $P_2 \xrightarrow{\tau^*} P_2' \sim P_2''$. Since P_2' and P_2'' are strongly equivalent, they are certainly observation equivalent, and hence by (49) we have $P_1' \approx_b P_2'' \approx_b P_2'$, as required for (wb1).

$\approx_b$ has property (wb2). Suppose $P_1 \approx_b P_2$ and that $P_1 \xrightarrow{\ell} P_1'$. We have to find P_2' such that $P_2 \xrightarrow{\tau^*\ell\tau^*} P_2'$ and $P_1' \approx_b P_2'$.

Let $\vec{x}$ be all the names occurring free in P_1 or P_2 and choose names x, y, z distinct from $\vec{x}$ and from each other. Since $\approx_b$ is a congruence, we have

$$P_1|\bar{\ell}.x.\mathbf{0}|\bar{x}.y.\mathbf{0} \approx_b P_2|\bar{\ell}.x.\mathbf{0}|\bar{x}.y.\mathbf{0}.$$

By Lemma 19(ii) we have $(P_1|\bar{\ell}.x.\mathbf{0}|\bar{x}.y.\mathbf{0})|z.\mathbf{0} \Downarrow z.(P_1'|y.\mathbf{0})$; hence by property (48) of $\approx_b$ there is some P_2'' with $(P_2|\bar{\ell}.x.\mathbf{0}|\bar{x}.y.\mathbf{0})|z.\mathbf{0} \Downarrow z.P_2''$ and $P_1'|y.\mathbf{0} \approx_b P_2''$. It follows by Theorem 21(ii) that

$$P_2|\bar{\ell}.x.\mathbf{0}|\bar{x}.y.\mathbf{0} \xrightarrow{\tau^*} Q \tag{50}$$

holds for some Q with $Q \sim P_2''$. Since Q is strongly equivalent to P_2'' it is also observation equivalent to it and hence by (49) we have $Q \approx_b P_2''$. But $P_2'' \approx_b P_1'|y.\mathbf{0}$; so

$$P_1'|y.\mathbf{0} \approx_b Q \tag{51}$$

and hence by the congruence property of $\approx_b$ we have $(\nu\vec{x}y)Q \approx_b (\nu\vec{x}y)(P_1'|y.\mathbf{0})$. Now $fn((\nu\vec{x}y)(P_1'|y.\mathbf{0})) = \emptyset$, so by property (48) of $\approx_b$, we have

$$(\nu\vec{x}y)Q \not\Downarrow. \tag{52}$$

Now since $x, y \notin fn(P_2) \supseteq fn(\bar{\ell})$, (50) can only hold because either

(a) $P_2 \xrightarrow{\tau^*\ell\tau^*} P_2'$ holds for some P_2' such that $Q = P_2'|\mathbf{0}|y.\mathbf{0}$; or
(b) $P_2 \xrightarrow{\tau^*\ell\tau^*} P_2'$ holds for some P_2' such that $Q = P_2'|x.\mathbf{0}|\bar{x}.y.\mathbf{0}$; or
(c) $P_2 \xrightarrow{\tau^*} P_2'$ holds for some P_2' such that $Q = P_2'|\bar{\ell}.x.\mathbf{0}|\bar{x}.y.\mathbf{0}$.

In fact cases (b) and (c) are impossible. For in either case, $(\nu\vec{x}y)Q$ can do $\tau^*\bar{x}$ which contradicts (52)(by parts (i) and (ii) of Lemma 19). So case (a) holds. Since $P_1'|y.\mathbf{0} \approx_b Q \sim P_2'|y.\mathbf{0}$, it follows that

$$P_1' \approx P_1'|(\nu y)y.\mathbf{0} \approx (\nu y)(P_1'|y.\mathbf{0}) \approx_b (\nu y)(P_2'|y.\mathbf{0}) \approx P_2'|(\nu y)y.\mathbf{0} \approx P_2'.$$

Hence by (49) $P_1' \approx_b P_2'$, as required for (wb2). □

4.4 Unrestricted summation

Consider extending NCCS with an unrestricted binary summation operator:

$$E ::= \cdots \mid E + E.$$

What rules for evaluation to committed form should + satisfy? We have already seen in Remark 25 that one obvious rule ($\Downarrow\oplus$) leads to internal non-deterministic choice. Another possibility is just

$$\frac{P_i \Downarrow K}{P_1 + P_2 \Downarrow K}$$

for $i = 1, 2$. Note however that we chose to build the 'weakening' property of Lemma 5(i) into the rules ($\Downarrow 0$)–($\Downarrow 4$) rather than stating it as a separate rule. Accordingly, we should stabilise the above rule for + with respect to weakening—which leads to the rule

$$\frac{P_i \Downarrow \ell.P}{(P_1 + P_2)|Q \Downarrow \ell.(P|Q)} \quad (i = 1, 2). \qquad (\Downarrow 5)$$

Note that this conservatively extends the existing rules for evaluating normal summations: if P_1 and P_2 are normal processes, then the rule does not give any new evaluations for the NCCS process $P_1 + P_2$. Therefore when adding rule ($\Downarrow 5$) we may as well restrict axiom ($\Downarrow 1$) to

$$(\ell.P)|Q \Downarrow \ell.(P|Q) \qquad (\Downarrow 1')$$

The method for proving congruence given in the proof of Theorem 14 works just as well for this extended language equipped with rules ($\Downarrow 0$), ($\Downarrow 1'$), and ($\Downarrow 2$)–($\Downarrow 5$). So we obtain:

Theorem 29 *Evaluation bisimilarity (defined just as in Definition 6) is a congruence for NCCS extended with sums satisfying ($\Downarrow 1'$).*

Next we consider the problem of finding labelled transition rules for + which permit the results of Section 3 (Theorem 24 in particular) to go through. Two possibilities from the literature which come to mind are:

CCS summation (see [14]).

$$\frac{P_i \xrightarrow{\alpha} P}{P_1 + P_2 \xrightarrow{\alpha} P} \ (i = 1, 2).$$

External non-deterministic choice (see [7, Chapter 5])

$$\frac{P_1 \xrightarrow{\tau} P_1'}{P_1 + P_2 \xrightarrow{\tau} P_1' + P_2} \qquad \frac{P_2 \xrightarrow{\tau} P_2'}{P_1 + P_2 \xrightarrow{\tau} P_1' + P_2}$$

$$\frac{P_i \xrightarrow{\ell} P}{P_1 + P_2 \xrightarrow{\ell} P} \ (i = 1, 2).$$

(Recall that ℓ ranges over names and co-names, while α ranges over names, co-names and τ.) For either of these choices of transition rules, Theorem 24 fails, i.e. evaluation bisimilarity does not coincide with delay bisimulation equivalence. This is easy to see if $+$ is interpreted as CCS summation, because unlike $\simeq_{\Downarrow}$, delay bisimulation equivalence fails to be a congruence for $+$ for the same reason that observational equivalence fails to be a CCS congruence. For example, using the τ-prefixing operation of Definition 8, $x.\mathbf{0} + (\tau.y.\mathbf{0})$ is not delay bisimilar to $x.\mathbf{0} + y.\mathbf{0}$, but the two processes are evaluation bisimilar—using the fact that $\tau.y.\mathbf{0} \simeq_{\Downarrow} y.\mathbf{0}$ and the congruence property stated in the theorem above. (Contrast this with Example 10.)

If $+$ is interpreted as external non-deterministic choice, $\simeq_{\Downarrow}$ and $\simeq_{\mathrm{dl}}$ still fail to coincide. For example, consider

$$P_1 \stackrel{\mathrm{def}}{=} ((x_1.\mathbf{0} \oplus x_2.\mathbf{0}) + y.\mathbf{0})|z.\mathbf{0}$$
$$P_2 \stackrel{\mathrm{def}}{=} P_1 \oplus ((x_1.\mathbf{0} + y.\mathbf{0})|z.\mathbf{0})$$

where x, y, z are distinct and we take the internal choice operator $\oplus$ to be defined by (43). By calculating the possible labelled transitions of P_1 and P_2 using the rules for external choice, it is not hard to check that these two processes are delay bisimilar. However, they are not evaluation bisimilar: for $P_2 \Downarrow z.P_2'$ with $P_2' \simeq_{\Downarrow} x_1.\mathbf{0} + y.\mathbf{0}$; whereas if $P_1 \Downarrow z.P_1'$ then $P_1' \equiv (x_1.\mathbf{0} \oplus x_2.\mathbf{0}) + y.\mathbf{0}$, which clearly is not evaluation bisimilar to P_2'.

The following characterisation of evaluation bisimilarity for the calculus with unrestricted summation was suggested by Catuscia Palamidessi [20].

Theorem 30 *Consider NCCS extended with $+$. Let $\Downarrow$ be inductively generated by the rules ($\Downarrow$0), ($\Downarrow$1′), and ($\Downarrow$2)–($\Downarrow$5). Let $\stackrel{\alpha}{\rightarrow}$ be inductively generated by the rules ($\rightarrow$1)–($\rightarrow$5) together with the following countable collection of rules (one for each $n \geq 1$):*

$$\frac{P_i = Q_1 \stackrel{\tau}{\rightarrow} Q_2 \quad \ldots \quad Q_{n-1} \stackrel{\tau}{\rightarrow} Q_n \quad Q_n \stackrel{\ell}{\rightarrow} P}{P_1 + P_2 \stackrel{\ell}{\rightarrow} P} \quad (i = 1, 2). \qquad (\rightarrow 6_n)$$

Then evaluation bisimilarity (Definition 6) coincides with delay bisimulation equivalence (Definition 23).

PROOF. One first shows that Lemma 19 continues to hold for the extended system. The proof of part (ii) of the lemma is exactly as before, because ($\rightarrow 6_n$) does not introduce τ-transitions; in the proof of part (iii), closure of $\{(P, \ell.P') \mid \exists P''.\, P \stackrel{\tau^*\ell}{\longrightarrow} P'' \sim P'\}$ under ($\Downarrow$5) is straightforward using the rules ($\rightarrow 6_n$); and then part (i) can be proved, using part (ii) and ($\Downarrow$5) to get closure

under each rule ($\rightarrow 6_n$). Armed with this lemma, the proof of the theorem goes through just as for Theorem 24. $\Box$

5 Conclusions

We feel the results in this paper vindicate *evaluation to committed form* as a promising approach to the topic of 'weak' equivalence in process calculi, both in its own right and in the way it relates and sheds light on existing approaches. In conclusion, we mention two topics which may bear further investigation. First, evaluation seems at a slightly higher level of abstraction than labelled transition; moreover it places the emphasis upon composition and restriction as fundamental operations (indeed in some cases as purely structural ones—cf. Section 4.2). So maybe this approach can suggest new avenues in the rather under-developed subject of denotational semantics for communicating processes up to weak equivalence. Secondly, since evaluation relations are already a convenient way to specify the structural operational semantics of functional languages, our approach may aid in developing theories of equivalence for languages integrating functional and process-theoretic features. To that end, a comparison of an evaluation-based approach to CML [24] with the transition-based theory developed by Ferreira *et al* in [3] will appear in the second author's thesis.

Acknowledgement

A preliminary version of this paper appeared as [22]. The first author was partially supported by the EU HCM network 'EuroFoCS' and the second author by a UK EPSRC research studentship.

References

[1] S. Abramsky. The lazy λ-calculus. In D. A. Turner, editor, *Research Topics in Functional Programming*, chapter 4, pages 65–117. Addison Wesley, 1990.

[2] G. Berry and G. Boudol. The chemical abstract machine. *Theoretical Computer Science*, 96:217–248, 1992.

[3] W. Ferreira, M. Hennessy, and A. Jeffrey. A theory of weak bisimulation for core CML. Technical Report 05/95, Computer Science, University of Sussex, September 1995.

[4] C. Fournet and G. Gonthier. The reflexive CHAM and the join-calculus. In *23rd SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 372–385. ACM Press, January 1996.

[5] 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.

[6] R. J. van Glabbeek and W. P. Weijland. Branching time and abstraction in bisimulation semantics. *Journal of the ACM*, 43(3):555–600, May 1996.

[7] M. Hennessy. *Algebraic Theory of Processes.* MIT Press, 1988.

[8] K. Honda and M. Tokoro. On asynchronous communication semantics. In M. Tokoro and O. Nierstrasz, editors, *Proc. ECOOP'91 Workshop on Object-Based Concurrent Computing*, volume 612 of *Lecture Notes in Computer Science*, pages 21–51. Springer-Verlag, Berlin, 1991.

[9] K. Honda and N. Yoshida. On reduction-based process semantics. In *Proc. 13th Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 761 of *Lecture Notes in Computer Science*, pages 371–387. Springer-Verlag, Berlin, 1993.

[10] D. J. Howe. Equality in lazy computation systems. In *4th Annual Symposium on Logic in Computer Science*, pages 198–203. IEEE Computer Society Press, Washington, 1989.

[11] D. J. Howe. Proving congruence of bisimulation in functional programming languages. *Information and Computation*, 124(2):103–112, February 1996.

[12] P. Martin-Löf. Constructive mathematics and computer programming. In *Sixth International Congress for Logic, Methodology, and Philosophy of Science*, pages 153–175, Amsterdam, 1982. North-Holland.

[13] R. Milner. A modal characterisation of observable machine-behaviour. In *Proc. CAAP'81*, volume 112 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1981.

[14] R. Milner. *Communication and Concurrency.* Prentice Hall, 1989.

[15] R. Milner. Functions as processes. In *Proc. 17th Int. Coll. on Automata, Languages and Programming*, volume 443 of *Lecture Notes in Computer Science*, pages 167–180. Springer-Verlag, Berlin, 1990.

[16] R. Milner. The polyadic π-calculus: A tutorial. In F. L. Bauer *et al*, editor, *Logic and Algebra of Specification.* Springer-Verlag, Berlin, 1993.

[17] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes (parts I and II). *Information and Computation*, 100:1–77, 1992.

[18] R. Milner and D. Sangiorgi. Barbed bisimulation. In W. Kuich, editor, *Proc. 19th Int. Coll. on Automata, Languages and Programming*, volume 623 of *Lecture Notes in Computer Science*, pages 685–695. Springer-Verlag, Berlin, 1992.

[19] R. Milner, M. Tofte, and R. Harper. *The Definition of Standard ML.* MIT Press, 1990.

[20] C. Palamidessi. Private communication. September 1996.

[21] B. C. Pierce and D. N. Turner. Pict: A programming language based on the pi-calculus. to appear, 1996.

[22] A. M. Pitts and J. R. X. Ross. Process calculus based upon evaluation to committed form. In U. Montanari and V. Sassone, editors, *CONCUR'96: Concurrency Theory, Proc. 7th Int. Conf. Pisa, 1996.*, volume 1119 of *Lecture Notes in Computer Science*, pages 18–33. Springer-Verlag, Berlin, 1996.

[23] G. D. Plotkin. A structural approach to operational semantics. Technical Report DAIMI FN-19, Aarhus University, 1981.

[24] J. Reppy. CML: A higher-order concurrent language. In *Programming Language Design and Implementation*, pages 293–259. SIGPLAN, ACM, June 1991.

[25] D. Sangiorgi. *Expressing Mobility in Process Algebras: First-Order and Higher-Order Paradigms.* PhD thesis, Department of Computer Science, University of Edinburgh, 1992.

[26] D. Sangiorgi. On the bisimulation proof method. Technical Report ECS-LFCS-94-299, Department of Computer Science, Edinburgh University, August 1994.

[27] D. Sangiorgi. Bisimulation for higher-order process calculi. Technical Report RR-2508, INRIA, April 1995.

[28] B. Thomsen. Plain CHOCS. A second generation calculus for higher order processes. *Acta Informatica*, 30(1):1–59, 1993.

[29] W. P. Weijland. *Synchrony and Asynchrony in Process Algebra.* PhD thesis, Univ. Amsterdam, 1989.