REPLICATION IN CONCURRENT COMBINATORS

KOHEI HONDA
`kohei@mt.cs.keio.ac.jp`
NOBUKO YOSHIDA
`yoshida@mt.cs.keio.ac.jp`

Department of Computer Science, Keio University
3-14-1 Hiyoshi, Kohoku-ku, Yokohama, 223, Japan

Abstract. We establish the behavioural representability of *input prefix* and *replication*, which are two basic operators of the asynchronous ν-calculus [5, 6] (cf. [2]), an offspring of π-calculus [14, 11], in a system of concurrent combinators which is an extension of **cc** [7]. Systems of concurrent combinators are simple formal systems based on a finite number of atoms with fixed interaction rules and two connectives for term formation, namely, *concurrent composition* and *name hiding*. Since such a system does not own inductively defined syntactic constructs for prefix or replication, their representability using a fixed number of atoms is theoretically interesting, reminiscent of the similar results obtained by Schönfinkel and Curry in the context of the classical combinators.

1. Introduction

The aim of the present paper is to establish the behavioural representability of *input prefix* and *replication*, which are two basic operators of the asynchronous ν-calculus [5, 2, 6], an offspring of π-calculus [14, 11], in a system of concurrent combinators which is an extension of **cc** [7]. Systems of concurrent combinators are simple formal systems based on a finite number of atoms with fixed interaction rules and two connectives for term formation, namely, *concurrent composition* and *name hiding.* Since such a system does not own inductively defined syntactic constructs for prefix or replication, their representability using a fixed number of atoms is theoretically interesting. Specifically the result is reminiscent of the decomposition of functional abstraction in the classical combinators, though the basic computational scheme is quite different.

Regarding the input prefix, written "$ax.P$" in ν-calculus, [7] shows that one can "analyze away" the construct without losing its functionality, yielding, in the course of analysis, seven atoms which embody basic process behaviour. The resulting system, though without a corresponding syntactic construct, is capable of representing

---

Appeared in: TACS'94, LNCS 789, pp.786–805, Springer-Verlag, 1994.

an agent denoted by $a^*x.P$, which has, essentially speaking, a behaviour:

$$a^*x.P, \mathcal{M}(ab) \longrightarrow P\{b/x\}$$

where "$\mathcal{M}(ab)$" is a message carrying b to a, "," denotes concurrent composition, and $\longrightarrow$ is one-step reduction. Thus "$a^*x.P$" receives a message at the name a, and instantiates the received value in P. We note that the representability automatically results in the representability of other kinds of prefixes, such as the prefix of Milner's polyadic π-calculus [12]. As a sequel to [7], the present paper shows that the remaining operation, *replication*, can also be analyzed away, and that the resulting system is capable of behaviourally representing, in addition to the input prefix, a *replicator* $!^*_a P$, which has a behaviour:

$$!^*_a P,\ \mathcal{M}(a) \longrightarrow\ !^*_a P,\ P\ .$$

Thus $!^*_a P$ generates a copy of P when it is activated at the port a, where $\mathcal{M}(a)$ is an activation message without considering a value. The representability accompanies a handful of additional atoms, which are systematically derived from the seven basic ones, and endows the system with general computing power. We also show that this simple replication is powerful enough to realize other kinds of replication, notably the lazy version by Milner, Parrow and Walker [14] and the eager one by Milner [11]. Other consequences of the representability will also be discussed briefly.

Before embarking on the main study, we may spend some words on the necessity of having *replication* in addition to *communication* in the name passing paradigm, since this is quite different from the situation in the pure λ-calculus, where the single operation called β-reduction suffices to have general computing power. The difference comes from the fact that the name passing operation only decreases the length of terms as well as the number of port connections (cf. 3.11), which precludes representability of any computational structure with unbounded behaviour, e.g. μ-recursion in the Kleene's scheme of partial recursive function or continuous behaviour in some concurrent systems such as operating systems and banking systems. Thus the replication becomes an indispensable element in order to have general computing power in the name passing framework. We should note that the same effect may be realized by communication of *terms* as found in CHOCS [15]. However, unlike in the function paradigm, the decomposition of the term passing operation into atoms seems accompanied by many nontrivial difficulties [4]. Thus the decomposition of the basic notion of concurrent computation into communication and replication turns out to be effective in the present setting.[1]

Apart from the main result itself, one may find one of the essential aspects of concurrent computing, namely the notion of interference and its control [8], in our construction which is used to establish the representability. The scheme can be interesting in the context of the execution theories for concurrent programming languages where interference and non-determinism are regarded as a norm. Some algebraic laws presented along the way, where the summation does not play the central role, may also be of interest from both theoretical and pragmatic viewpoints.

---

[1] The idea of such decomposition may originate in Girard's Linear Logic [3] (cf. [11]).

We now outline the structure of the paper. Section 2 introduces the family of formal systems, called *systems of (generalized) concurrent combinators* and related notions, followed by the statement of the main result of the paper. Verification of the statement starts with Section 3, which introduces an extension of **cc** [7] with additional atoms embodying a simple notion of replication. On its basis, Section 4 defines the system $\mathbf{cc}_\varphi$, with which we establish the two closure properties. Section 4 proves the prefix closure for $\mathbf{cc}_\varphi$. Section 5 actually constructs a simple replicator in $\mathbf{cc}_\varphi$ and verifies its behaviour, establishing the main result. Section 6 lists some open issues.

Details of the proofs are often omitted, for which the reader may refer to [4].

2. PRELIMINARIES AND MAIN THEOREM

2.1. Generalized Concurrent Combinators. An instance of the family of systems we shall define in the following, was initially introduced as a "combinatory" decomposition of the finite part of the ν-calculus in [7]. The computational scheme based on dyadic interaction is, in some aspects, reminiscent of Lafont's Interaction Net [9]. The framework of the formal system easily generalizes, as given below.

Let us assume, as the base sets, $\mathbf{N}$, a countable set of *names*, ranged over by $a, b, c, ..$, and $\mathbf{V}$, a countable set of *name variables*, disjoint from $\mathbf{N}$, ranged over by $x, y, z, ...$ $u, v, w, ..$ range over their union, the set of *identifiers*. Then a *system of (generalized) concurrent combinators* is defined by a triple $\langle \mathcal{P}_{\mathbb{A}},\ \equiv,\ \longrightarrow \rangle$, where the first element is a set of terms of the system derived from $\mathbb{A}$, which is the set of atoms equipped with polarities and arities, the second element is the structural equality over the terms, and the final element is the reduction relation, again over the terms. We begin with introducing the notion of atoms and terms.

2.2. Atoms and Terms. Let $\mathbb{A}$ be a finite set of *atoms*, ranged over by $C, C', ...$, with two associating functions $ar : \mathbb{A} \to \mathbb{N}^{+}$ and $pol : \mathbb{A} \to \{+, -\}$. $ar(C)$ and $pol(C)$ are called the *arity* and the *polarity* of C, respectively. We write $C^{(3)}$, C^{+}, or $C^{(3)+}$ to denote the arity and/or polarity of the atom. $\theta, \theta', ..$ range over the set $\{+, -\}$. Then $\mathcal{P}_{\mathbb{A}}$, the set of terms with a base $\mathbb{A}$, ranged over by $P, Q, R, ...$, is given by the following grammar:

$$P \quad ::= \quad C(\tilde{u}) \ \mid \ (P, Q) \ \mid \ a \blacktriangleright P \ \mid \ \Lambda$$

where $\tilde{u}$ is a sequence from $\mathbf{N} \cup \mathbf{V}$, and $ar(C) = len(\tilde{u})$ with $len(\tilde{u})$ a function which denotes the length of $\tilde{u}$.

In the above, $C(\tilde{u})$ is an *atomic agent.* In $C(u\tilde{v})$, u is called its *subject.* (P, Q) is *concurrent composition* of P and Q, $a \blacktriangleright P$ is *scope restriction* of a in P, while Λ is *inaction* or *nil.* We will often call terms as *agents.* We assume that "$\blacktriangleright$" associates stronger than "," and "," associates to the left. "$ab \blacktriangleright P$" denotes "$a \blacktriangleright (b \blacktriangleright P)$".

In "$a \blacktriangleright P$," "$a \blacktriangleright$" functions as a binder so that free occurrences of a in P becomes bound by it. $\mathcal{FN}(P)$ and $\mathcal{BN}(P)$ denote the set of free names and the set of bound names in P, respectively (variables always occur free). $\equiv_\alpha$ denotes α-equality over $\mathcal{P}_{\mathbb{A}}$. We also assume the usual notion of (multiple) substitutions, written $\{\tilde{v}/\tilde{u}\}$

where $\tilde{u}$ and $\tilde{v}$ are strings of identifiers (names or variables) with the same length and all identifiers in $\tilde{u}$ are distinct. If not explicitly specified, a substitution always maps a variable to a name, a variable to a variable, or a name to a name, but not a name to a variable. σ, σ' etc., range over the set of substitutions. We say σ *does not use* v if v does not occur anywhere in σ. *Open terms* are those with variables, while *closed terms* are those without variables. A *closing substitution* denotes a substitution from variables to names which eliminates all occurrences of variables from a term.

2.3. Structural Equality. Fix a set of terms, say $\mathcal{P}_{\mathbb{A}}$. Over the set we stipulate the basic algebra, following Milner [11]. If we view terms as standing for some (hyper) graphs, the algebra is more than natural. Formally $\equiv$ (implicitly parameterized by $\mathcal{P}_{\mathbb{A}}$) is the smallest congruence relation over terms generated by:

(i) $P \equiv Q$ if $P \equiv_\alpha Q$

(ii) $P, Q \ \equiv\ Q, P \quad (P, Q), R \ \equiv\ P, (Q, R) \quad P, \Lambda \ \equiv\ P$

(iii) $aa \blacktriangleright P \ \equiv\ a \blacktriangleright P \quad ab \blacktriangleright P \ \equiv\ ba \blacktriangleright P \quad a \blacktriangleright \Lambda \ \equiv\ \Lambda \quad a \blacktriangleright P, Q \ \equiv\ a \blacktriangleright (P, Q)$ if $a \notin \mathcal{FN}(Q)$

Given $\equiv$, the following assumption is convenient while resulting in no loss of generality: *a term is always supposed to have pairwise distinct names for name occurrences in binders, which should also be disjoint from the free names in the term.*

2.4. Interaction Rules. The essential idea of computation in a system of concurrent combinators is, if a pair of atoms with different polarities meet together at a certain communication port, an interaction would occur; and, moreover, this is the only cause of computation within the system.

Given $\mathbb{A}$, we have a set of interaction rules, which is denoted by $\Re$. Each rule in $\Re$ should take a form, with C_1, C_2 from $\mathbb{A}$:

$$C_1^-(uv_1..v_n),\ C_2^+(uw_1..w_m) \ \longrightarrow\ P$$

where only $u, v_1, .., v_n$ and $w_1, .., w_m$ (all being pairwise distinct) may occur free in P, and, moreover, for each combination of a positive atom and a negative atom in $\mathbb{A}$, there should be one and only one such rule in $\Re$. A rule in $\Re$ should be read as a rule schema, where meta-variables over identifiers are universally quantified. Thus an *instance* of such a rule is obtained by instantiating meta-variables into names and variables. Now we can formulate the notion of reduction in our system.

2.5. Definition. (*reduction relation*)

(i) *One-step reduction derived from* $\Re$, is defined as the smallest relation closed under the following rules.

(COM) $C_1^-(uv_1..v_n),\ C_2^+(uw_1..w_m) \ \longrightarrow\ P$ if this is an instance from $\Re$

(PAR) $P \longrightarrow Q \ \Rightarrow\ P, R \longrightarrow Q, R.$

(RES) $P \longrightarrow Q \ \Rightarrow\ a \blacktriangleright P \longrightarrow a \blacktriangleright Q.$

(STR) $P \equiv P' \quad P' \longrightarrow Q' \quad Q \equiv Q' \ \Rightarrow\ P \longrightarrow Q.$

(ii) *Multi-step reduction relation*, $\twoheadrightarrow$, is defined by: $\twoheadrightarrow \overset{\text{def}}{=} \longrightarrow^{*} \cup \equiv$.

This, then, concludes the presentation of the basic construction for a system of concurrent combinators. We now proceed to several important derived notions for such a system. We call the family of formal systems S, which is ranged over by $\mathbb{S}$, $\mathbb{S}'$, …. We note that the system **cc** in [7] is clearly an instance of S.

2.6. Equality Over Agents. We now stipulate the notion of equality over agents, definable uniformly for each instance of S. The formulation is based on *reduction-closure* for equality [6] as well as *action predicate* as a refinement of *convergence predicate* in the line of e.g. [1, 13, 2], adapted for the scheme of dyadic interaction. The former says that the notion of equality should be consistent with the notion of state changes induced by reduction, while the second gives a further constraint to make the equality consistent with the basic idea of actions in the present computational framework.

Let us fix one instance of S, hence the set of terms, structural equality, and reduction relation. Then the action predicate is defined as: $P \Downarrow_{a^\theta} \overset{\text{def}}{\Leftrightarrow} P \twoheadrightarrow P' \wedge a \in \mathcal{AN}_\theta(P')$, with $\mathcal{AN}_\theta(P) \overset{\text{def}}{=} \{a \mid P \equiv \tilde{c} \blacktriangleright (C^\theta(a\tilde{v}), R) \wedge a \notin \{\tilde{c}\}\}$ (we call them *active name occurrences*). Then a congruence $\cong$ over closed terms is $\Downarrow_{a^\theta}$*-sound*, or simply *sound*, if

(i) $\cong \supset \equiv$.

(ii) $\cong$ is *reduction-closed*, i.e. $P \cong Q$ and $P \twoheadrightarrow P'$ implies, for some Q', $Q \twoheadrightarrow Q'$ and $P' \cong Q'$.

(iii) $\cong$ *respects action predicates*, i.e. $P \cong Q$ and $P \Downarrow_{a^\theta}$ implies $Q \Downarrow_{a^\theta}$.

Then, for any P and Q, $P \cong Q$ iff $P\sigma \cong Q\sigma$ for every closing σ. Then we have:

2.7. Proposition. *In any instance of S, there is the maximum $\Downarrow_{a^\theta}$-sound congruence.*

The proof is easy, see [4]. Such equality is immediately non-trivial if the system contains at least one atom. Henceforth we shall be regarding this congruence as denoting the basic notion of equality over agents in each system.[2] We write $P =_{\mathbb{S}} Q$ when P and Q are equated by the maximum sound equality (in the system $\mathbb{S}$).

2.8. Pointedness. The following notion called *pointedness* is important in that it formalizes the idea of *synchronization* which is one of the essential operational ideas in concurrency, coming from the functionality usually found in various prefixes in process calculi. We say that a closed term P is a^θ*-pointed*, iff P satisfies the following condition ($\bar{\theta}$ is the inverse of θ).

(i) $\mathcal{AN}_\theta(P) = \{a\}$, and $\mathcal{AN}_{\bar{\theta}}(P) = \emptyset$.

(ii) Above a is the unique active occurrence in P.

(iii) $P \not\longrightarrow$.

---

[2] We relegate the discussions on the semantic background underlying this equality notion to elsewhere.

Pointedness of a term tells us that there is only one interacting point in a given term (possibly unfolding further points after interaction). An open term is u^θ-pointed iff, for each closed substitution, the above condition is satisfied by a name substituted for u. We often write $P_{\langle u^\theta \rangle}$ etc. to denote a u^θ-pointed term P. Note that atomic agents are always pointed in this sense.

The following notion of β-equality, which was initially introduced in [7], provides a tractable subset of the canonical equality, again uniformly for any instance of S.

2.9. Definition. (*β-equality*) We define $\rightarrow_\beta$, one-step β-reduction as the smallest relation closed under:

$$(\mathrm{COM}_\beta) \quad c\blacktriangleright(P_{\langle c^- \rangle},\, Q_{\langle c^+ \rangle}) \ \rightarrow_\beta\ c\blacktriangleright R \quad \text{if} \quad P_{\langle c^- \rangle},\, Q_{\langle c^+ \rangle} \ \longrightarrow\ R$$

and (PAR), (RES), and (STR) of Definition 2.5, $\longrightarrow$ being replaced with $\rightarrow_\beta$. Then $\twoheadrightarrow_\beta \overset{\text{def}}{=} \rightarrow_\beta^* \cup \equiv$. β-convertibility, $=_\beta$, is the symmetric closure of $\twoheadrightarrow_\beta$.

The proof of the following was given in [4] in the context of a specific instance of S called **cc**: the proof easily generalizes and is therefore omitted.

2.10. Proposition. *For each $\mathbb{S} \in S$, $=_\beta$ is sound, hence is a subset of $=_\mathbb{S}$.*

Note $\rightarrow_\beta$ and $\twoheadrightarrow_\beta$ are subsets of $=_\beta$, hence of $=_\mathbb{S}$.

2.11. The Prefix Closure. Using the notion of pointedness, we are ready to state one of the essential facts regarding S, namely the representability of *prefix* in some instances of S, and its consequences. The representability tells us that, though instances of S lack the syntactic construct corresponding to the prefix in (name passing) process calculi, some of them are nevertheless capable of representing the notion behaviourally.

First, a definition: an instance of $\mathbb{S} \in S$ is *behaviourally closed under prefix*, when, with a fixed θ,[3] for any u and v, there is a term $\leftarrow u^* v$, with $(\leftarrow u^* v)\sigma =_\mathbb{S} \leftarrow u\sigma^* v\sigma$, and also, for any u, x and P, there is a term $u^* x.P$, with $(u^* x.P)\sigma =_\mathbb{S} u\sigma^* x.P\sigma$ if x is not used in σ and $u^* x.P =_\mathbb{S} u^* z.P\{z/x\}$ for z fresh, such that:

(1) $\leftarrow u^* v$ is u^θ-pointed and $u^* x.P$ is $u^{\overline{\theta}}$-pointed.

(2) $u^* x.P,\ \leftarrow u^* v \ \longrightarrow =_\mathbb{S}\ P\{v/x\}$.

The conditions for substitution such as $(\leftarrow u^* v)\sigma =_\mathbb{S} \leftarrow u\sigma^* v\sigma$ as well as the restriction that the reduction in (2) is one-step, are semantically important, as seen in e.g. the proof of Corollary 2.13.

The proof of the following proposition, together with the accompanying constructions, appear in [7], using a much stronger mediating equality for the clause (2) above.

---

[3] This θ can be arbitrary since we can easily show, if a system enjoys the property with, say, $\theta = -$, then it also enjoys the closure property with $\theta = +$. Similarly for subsequent formulations.

2.12. Proposition. *There exists an instance of S which is behaviourally closed under prefix.*

Two significant consequences of the closure property follow. They say that, once we have the prefix closure property, more complicated prefixes such as those of Polyadic π-calculus [12] (subsuming prefixes of CCS [10] and the monadic π-calculus [14]), or even those involving local branching operations, can be behaviourally represented.

2.13. Corollary (1). (multiple name passing) *Suppose $\mathbb{S} \in S$ is behaviourally closed under prefix. Then, with a fixed θ, for any $u, v_1, .., v_n$ and P, there is a term $\bar{u}^* : [v_1 v_2 .. v_n].P$, and, for any u, P and distinct $x_1, .., x_n$, there is a term $u^* : (x_1 .. x_n).P$, both of which conforming to the conditions for substitution as in 2.11, such that:*

(1) $\bar{u}^*\!:[v_1 v_2 .. v_n].P$ is u^θ-pointed and $u^*\!:(x_1 ... x_n).P$ is $u^{\overline{\theta}}$-pointed.
(2) $u^*\!:(x_1 x_2 .. x_n).P,\ \bar{u}^*\!:[v_1 v_2 .. v_n].Q \ \longrightarrow\ =_{\mathbb{S}}\ P\{v_1 v_2 .. v_n / x_1 x_2 .. x_n\},\ Q$.

Proof. Use the following construction, assuming z, c, and y are all fresh below:

$$\begin{aligned} \bar{u}^*\!:[v_1 .. v_n].P &\overset{\text{def}}{=} c \blacktriangleright (\leftarrow u^* c, c^* z.(\leftarrow z^* v_1, c^* z.(.., c^* z.(\leftarrow z^* v_n, P)..))) \\ u^*\!:(x_1 .. x_n).P &\overset{\text{def}}{=} u^* y. c \blacktriangleright (\leftarrow y^* c, c^* x_1.(c \blacktriangleright (\leftarrow y^* c, .. c \blacktriangleright (\leftarrow y^* c, c^* x_n.P)))) \ . \end{aligned}$$

Then pointedness is immediate. For reduction, relying on pointedness and decomposability of substitution, we have:

$$\bar{u}^*\!:[v_1 .. v_n].P, u^*\!:(x_1 .. x_n).Q \ \longrightarrow\ \rightarrow_\beta =_{\mathbb{S}} \rightarrow_\beta \ \ldots\ \rightarrow_\beta =_{\mathbb{S}} \ P\{v_1 .. v_n / x_1 .. x_n\}$$

then we use Proposition 2.10. □

2.14. Corollary (2). (branching and selection) *Suppose $\mathbb{S} \in S$ is behaviourally closed under prefix. Then, with a fixed θ, for any $u, \tilde{v} \overset{\text{def}}{=} v_1 ... v_k$ and P, there is a term $\bar{u}^* : \mathsf{in}^i_n[\tilde{v}].P$, and, for any $u, \tilde{x}_i \overset{\text{def}}{=} x_{i1} ... x_{ik}$ and P_i $(1 \leq i \leq n)$, there is a term $u^* : (\tilde{x}_1).P_1 \& (\tilde{x}_1).P_2 \& .. (\tilde{x}_n).P_n$ denoted $u^* : \&^n_{i=1}(x_{i1} .. x_{ik}).P_i$, both of which conforming to the conditions for substitution as in 2.11, such that:*

(1) $\bar{u}^* : \mathsf{in}^i_n[v_1 .. v_k].P$ is u^θ-pointed and $u^* : \&^n_{i=1}(x_{i1} .. x_{ik}).Q_i$ is $u^{\overline{\theta}}$-pointed.
(2) $\bar{u}^* : \mathsf{in}^i_n[v_1 .. v_k].P,\ u^* : \&^n_{i=1}(x_{i1} .. x_{ik}).Q_i \ \longrightarrow\ =_{\mathbb{S}}\ P,\ Q_i\{v_1 .. v_k / x_{i1} .. x_{ik}\}$.

Proof. Use the following construction with newly introduced names being all fresh and distinct, and reason as in 2.13 (we also need: $c \blacktriangleright P_{\langle c^\theta \rangle} =_{\mathbb{S}} \Lambda$).

$$\begin{cases} \bar{u}^* : \mathsf{in}_1^1[\tilde{v}].P & \overset{\text{def}}{=} \ \bar{u}^*:[\tilde{v}].P \\ \bar{u}^* : \mathsf{in}_{n+1}^i[\tilde{v}].P & \overset{\text{def}}{=} \ c \blacktriangleright (\mathcal{M}(uc),\ c^*:(z_1 z_2).\bar{z}_1{}^* : \mathsf{in}_n^i[\tilde{v}].P) \\ \bar{u}^* : \mathsf{in}_{n+1}^{n+1}[\tilde{v}].P & \overset{\text{def}}{=} \ c \blacktriangleright (\mathcal{M}(uc),\ c^*:(z_1 z_2).\bar{z}_2{}^*:[\tilde{v}].P) \end{cases}$$

$$\begin{cases} u^* : \&_{i=1}^1(\tilde{x}).P & \overset{\text{def}}{=} \ u^*:(\tilde{x}).P \\ u^* : \&_{i=1}^{n+1}(\tilde{x}_i).P_i & \overset{\text{def}}{=} \ u^* z.c_1 c_2 \blacktriangleright \bar{z}^*:[c_1 c_2].(c_1{}^* : \&_{i=1}^n(\tilde{x}_i).P_i,\ c_2^*:(\tilde{x}_{n+1}).P_{n+1}) \end{cases} \qquad \square$$

2.15. The Notion of Replication. As was discussed in Introduction, the representability of the operation of communication cannot be enough in the name passing paradigm, making it necessary to consider the notion of replication separately. Below we formulate the representability in systems of concurrent combinators concerning three variants of replication which include those already presented in [14, 11].

(i) A system of S *is behaviourally closed under simple replication*, if, with a fixed θ, for any u, there is a term $\mathbf{?}^* u$, and for any u and P there is a term $\mathbf{!}^*_u P$, with $(\mathbf{?}^* u)\sigma =_{\mathbb{S}} \mathbf{?}^* u\sigma$ and $(\mathbf{!}^*_u P)\sigma =_{\mathbb{S}} \mathbf{!}^*_{u\sigma} P\sigma$, such that:

   (1) $\mathbf{?}^* u$ is u^θ-pointed and $\mathbf{!}^*_u P$ is $u^{\overline{\theta}}$-pointed.

   (2) $\mathbf{!}^*_u P,\ \mathbf{?}^* u \ \longrightarrow\ =_{\mathbb{S}}\ \mathbf{!}^*_u P,\ P.$

(ii) A system of S *is behaviourally closed under lazy replication*, if, with a fixed θ, for any u and v, there is a term $?u^* v$, and for any u, x and P there is a term $!u^* x.P$, both of which conforming to the conditions for substitution as in 2.11, such that:

   (1) $?u^* v$ is u^θ-pointed and $!u^* x.P$ is $u^{\overline{\theta}}$-pointed.

   (2) $!u^* x.P,\ ?u^* v \ \longrightarrow\ =_{\mathbb{S}}\ !u^* x.P,\ P\{v/x\}.$

(iii) Finally, a system of S *is behaviourally closed under eager replication*, if, for any P, there is an agent $!^* P$, where $(!^* P)\sigma \equiv !^* P\sigma$, such that:

$$!^* P \ =_{\mathbb{S}}\ !^* P,\ P$$

(i) seems new. (ii) essentially comes from [14] while (iii) is Milner's replication law in [11]. Regarding (iii), note that even $(!P,\ Q)$ in the sense of [11] satisfies the equation so that (iii) should be a little too weak. Concerning the relationship between these three variants of replication, we know the following.

2.16. Proposition. *Let* $\mathbb{S}$ *be behaviourally closed under prefix. Then:*

(i) $\mathbb{S}$ *is behaviourally closed under simple replication if and only if* $\mathbb{S}$ *is behaviourally closed under lazy replication.*

(ii) *If $\mathbb{S}$ is behaviourally closed under simple replication, or, equivalently by (i), lazy replication, then $\mathbb{S}$ is also behaviourally closed under eager replication.*

PROOF. Below pointedness of replicators is essential to obtain β-reduction.

(i) ($\Longrightarrow$) Take $?u^* v \overset{\text{def}}{=} \leftarrow u^* v$ and $!u^* x.P \overset{\text{def}}{=} c \blacktriangleright (u^* x.(\mathbf{?}^* c, P), \mathbf{!}_c u^* x.(\mathbf{?}^* c, P))$. Then pointedness and decomposition of a substitution are easy. For reduction, we have:

$$!u^* x.P, ?u^* v \longrightarrow =_{\mathbb{S}} \rightarrow_\beta !u^* x.P, P\{v/x\}$$

then we use Proposition 2.10.
($\Longleftarrow$) Take $c \blacktriangleright ?u^* c$ for $\mathbf{?}^* u$ and $!u^* x.P$ with x not free in P for $\mathbf{!}^*_u P$. Then the required properties are immediate.

(ii) We assume $\mathbb{S}$ is closed under simple replication. Then take $!^* P \overset{\text{def}}{=} c \blacktriangleright (\mathbf{!}^*_c(P, \mathbf{?}^* c), \mathbf{?}^* c)$ with c fresh. For this we have:

$$!^* P \quad \rightarrow_\beta =_{\mathbb{S}} \quad !^* P, \quad P$$

and we use Proposition 2.10.[4] □

Thus we know that the simple replicator is strong enough to imply other two kinds of replicators. Our aim of the present paper is to establish the following theorem.

2.17. Theorem. (Main Theorem, Representability) *There is an instance of S which is behaviourally closed both under prefix and simple replication.*

By Proposition 2.16, this immediately implies the representability of other two kinds of replications. The proof of the theorem is done starting from Sections 3 and concluding in Section 5. It actually proves the closure properties for a specific element of S, which is an extension of **cc** originally introduced in [7].

2.18. Consequences of Theorem 2.17. The following significant consequences follow from Theorem 2.17, on which we can only discuss briefly here. See [4] for the formal account of these and other consequences of the theorem.

1. (*representability of computable functions*) If $\mathbb{S}$ satisfies the properties stated in Theorem 2.17, then, assuming a certain encoding of numerals, say $\lceil N \rceil(u)$, which is u^--pointed and $N \neq M$ implying $\lceil N \rceil(u) \neq_{\mathbb{S}} \lceil M \rceil(u)$, we can show that if F is a partial recursive function with arity n, then there exists an agent $\overline{F}(u_0 u_1 .. u_n)$ for which, given any vector of natural numbers of length n, say $N_1 .. N_n$, we have:

$$\begin{cases} c_1..c_n \blacktriangleright (\overline{F}(uc_1..c_n),\ \lceil N_1 \rceil(c_1), .., \lceil N_1 \rceil(c_n)) \ =_{\mathbb{S}}\ \lceil F(N_1..N_n) \rceil(u) & \text{if defined.} \\ c_1..c_n \blacktriangleright (\overline{F}(uc_1..c_n),\ \lceil N_1 \rceil(c_1), .., \lceil N_1 \rceil(c_n)) \ =_{\mathbb{S}}\ \Lambda & \text{otherwise.} \end{cases}$$

---

[4] The construction has other properties such as $!^* P =_{\mathbb{S}} !^* P, !^* P$ and $!^* (!^* P) =_{\mathbb{S}} !^* P$.

2. (*simulation of arbitrary instances of S*) The closure property implies the capability of the concerned system to simulate behaviour of terms of *any* instance of S regarding principal behavioural properties such as multi-step reduction, action predicate, and term formation rules, modulo the canonical equality (for this result we use 2.14 as well as replication). In other words, if a system is behaviourally closed under prefix and replication, then any instance of S can be "implemented" in the system in a behaviourally correct way. The property is closely related with a certain completeness property of such a system, whose formulation needs a more abstract setting than the equational construction here.
3. (*embedding of other calculi*) We can show that name passing process calculi such as ν-calculus and π-calculus [6, 11, 12], including their variants which allow ξ-rule, can be faithfully embedded in any system with the closure properties, in a way similar to the results regarding **cc** presented in [7], as far as we consider the fragments of these calculi without summation. Also, using the encodings of [11, 2], λ-calculi can be embedded in such a system, inducing certain λ-theories.

We regard, among others, the above 2 as one of the essential properties which testify the significance of the combination of the two closure properties in Theorem 2.17.

3. Atomic Replication

3.1. Atoms for Replicators. In the following, we introduce $\mathbf{cc}_!$, an extension of **cc** [7], where we add atoms called *atomic replicators* to those originally found in **cc**. The system will be further extended to $\mathbf{cc}_\varphi$ in the next section, for which we prove the two closure properties in Theorem 2.17. Several notions can be conveniently formulated in the context of $\mathbf{cc}_!$ rather than in its extension, however.

We first remember that **cc** has seven atoms, which are born in the course of analyzing away the input prefix of the ν-calculus. We call them *the basic atoms*, which are given by:

$$\mathbf{A} \overset{\text{def}}{=} \{\, \mathcal{M}^{(2)+},\ \ \mathcal{D}^{(3)-},\ \ \mathcal{FW}^{(2)-},\ \ \mathcal{K}^{(1)-},\ \ \mathcal{B}_l^{(2)-},\ \ \mathcal{B}_r^{(2)-},\ \ \mathcal{S}^{(3)-}\}$$

Some informal description of the basic atoms: $\mathcal{M}$ is a *message* carrying a name to a name. $\mathcal{D}$ is a *duplicator*, which duplicates a received message to send to two different locations. $\mathcal{FW}$ is a *forwarder* which just forwards a message to another location. $\mathcal{K}$ is called a *killer*, and kills a message. $\mathcal{B}_l$ is the left binder, and $\mathcal{B}_r$ is the right binder, both producing a forwarder using the value it receives. $\mathcal{S}$ is a *synchronizer* used for pure synchronization. The rules of interaction for them are given later.

In order to represent copying, we first introduce additional 6 atoms called *atomic replicators*. While later we shall see that they are not sufficient to represent the replication in full generality, they provide a building block for the full construction. The new set is given by:

$$\mathbf{A}_! \overset{\text{def}}{=} \{!C \mid C^- \in \mathbf{A}\}$$

where $ar(!C) = ar(C)$ and $pol(!C) = -$. Note ! is not a syntactic constructor, e.g. $!\mathcal{D}$ denotes just an atom. We note that we do not need replication of a message (which may have been written $!\mathcal{M}$) for our purpose. Essentially speaking, "$!C$" is a *persistent version* of the atom C; it has the same interactive behaviour as C, except that it remains as it originally was after interaction without any change.

The system $\mathbf{cc}_!$ is formed using $\mathbf{A} \cup \mathbf{A}_!$ as its atoms. Using the term formation rule in 2.2, we get the set of terms which will be denoted by $\mathcal{P}_{cc_!}$.

3.2. Interaction Rules. In the following, we specify interaction rules for atoms in $\mathbf{cc}_!$, from which reduction relation is formed. First we introduce the six rules for the basic atoms.

$$
(\mathrm{COM}) \quad \left\{ \begin{array}{l}
\mathcal{D}(uww'),\, \mathcal{M}(uv) \;\longrightarrow\; \mathcal{M}(wv),\, \mathcal{M}(w'v) \\
\mathcal{FW}(uw),\, \mathcal{M}(uv) \;\longrightarrow\; \mathcal{M}(wv) \\
\mathcal{K}(u),\, \mathcal{M}(uv) \;\longrightarrow\; \Lambda \\
\mathcal{B}_l(uw),\, \mathcal{M}(uv) \;\longrightarrow\; \mathcal{FW}(vw) \\
\mathcal{B}_r(uw),\, \mathcal{M}(uv) \;\longrightarrow\; \mathcal{FW}(wv) \\
\mathcal{S}(uww'),\, \mathcal{M}(uv) \;\longrightarrow\; \mathcal{FW}(ww')
\end{array} \right.
$$

One can compare these rules with informal description of each atom in 3.1. Rules for $\mathbf{A}_!$ is defined based on (COM). If $C(u\tilde{v}), \mathcal{M}(uw) \longrightarrow P$, then

$$
(\mathrm{REP}) \qquad !C(u\tilde{v}),\, \mathcal{M}(uw) \;\longrightarrow\; !C(u\tilde{v}),\, P
$$

A simple example is: $!\mathcal{D}(abc), \mathcal{M}(ae) \longrightarrow !\mathcal{D}(abc), \mathcal{M}(be), \mathcal{M}(ce)$. Reduction relation is formed from these rules in the way as specified in 2.5.

This finishes the definition of $\mathbf{cc}_!$. The set of terms, $\mathcal{P}_{cc_!}$, is equipped with the maximum sound equality in this context, which is written $=_{cc}$. For our subsequent study, some equality notions which are subsets of $=_{cc}$ become useful. They are *generalized β-equality* (accompanied with related reduction relation) as well as *simulation preorder* based on the equality. Since the notions are applicable to any instance of S, we formulate them in a general way.

3.3. Generalized β-reduction. For any instance of S, whose multi-step reduction is written $\longrightarrow\!\!\!\!\!\rightarrow$, we define the *generalized β-reduction*, denoted by $\Rightarrow_{\mathbb{B}}$, which is the largest relation such that:

(i) $\equiv \, \subseteq \, \Rightarrow_{\mathbb{B}} \, \subseteq \, \longrightarrow\!\!\!\!\!\rightarrow$.

(ii) $P \Rightarrow_{\mathbb{B}} Q_1$ and $P \longrightarrow\!\!\!\!\!\rightarrow Q_2$ then for some Q' $Q_1 \longrightarrow\!\!\!\!\!\rightarrow Q'$ and $Q_2 \Rightarrow_{\mathbb{B}} Q'$.

(iii) $P \Rightarrow_{\mathbb{B}} Q$ then $a \blacktriangleright P \Rightarrow_{\mathbb{B}} a \blacktriangleright Q$.
$P \Rightarrow_{\mathbb{B}} Q$ then $P, R \Rightarrow_{\mathbb{B}} Q, R$.
$P \Rightarrow_{\mathbb{B}} Q$ and $Q \Rightarrow_{\mathbb{B}} R$ then $P \Rightarrow_{\mathbb{B}} R$.
$P \Rightarrow_{\mathbb{B}} Q$ then $\mathcal{AN}_\theta(P) \subset \mathcal{AN}_\theta(Q)$ with $\theta = +, -$.

The generalized β-convertibility, $=_{\mathbb{B}}$, is the symmetric closure of $\Rightarrow_{\mathbb{B}}$.

We note that the well-definedness of the notion is easy to show by a property similar to 2.7. The formulation is motivated by the properties of the β-equality, which are used to establish its soundness in [7]. Then the reasoning as in [7] easily proves:

3.4. Proposition. *For any system* $\mathbb{S} \in S$, $\quad P =_{\mathbb{B}} Q$ *implies* $P =_{\mathbb{S}} Q$.

The notion becomes important in the verification of representability in Section 5.

3.5. Simulation Preorder. Below we introduce a rather strong behavioural preorder which further extends $=_{\mathbb{B}}$ but stays within $=_{\mathbb{S}}$. This is needed to reason about behaviour of prefixed and replicated terms, which cannot reach the target only by reduction. The definition follows. Below we write $P \downarrow_{\mathbb{B}} P'$ if $P \Rightarrow_{\mathbb{B}} P' \not\Rightarrow_{\mathbb{B}}$ (note such P' is unique up to $\equiv$).

(i) $P \overset{\bullet}{\succ}_0 C^{\theta}(u\tilde{v})$ iff $P \downarrow_{\mathbb{B}} P_{\langle u^{\theta}\rangle}$. Then $\succ_0$ is its precongruent closure.

(ii) (a) $P \overset{\bullet}{\succ}_{n+1} C^{\theta}(u\tilde{v})$ iff $P \downarrow_{\mathbb{B}} P'_{\langle u^{\theta}\rangle}$ where, for any $C^{\overline{\theta}}$, we have:

1. $P', C^{\overline{\theta}}(u\tilde{w}) \longrightarrow Q' \;\Rightarrow\; (C^{\theta}(u\tilde{v}), C^{\overline{\theta}}(u\tilde{w}) \longrightarrow Q \;\wedge\; Q' \succ_n Q)$,
2. $C^{\theta}(u\tilde{v}), C^{\overline{\theta}}(u\tilde{w}) \longrightarrow Q \;\Rightarrow\; (P', C^{\overline{\theta}}(u\tilde{w}) \longrightarrow Q' \;\wedge\; Q' \succ_n Q)$.

(b) $\succ_{n+1}$ is the precongruent closure of $\overset{\bullet}{\succ}_{n+1}$.

Then $P \succ Q$ if, for all $n \in \omega$, $P \succ_n Q$. Intuitively, $P \succ Q$ means any atomic agent in Q has its close "simulator" in P. We note that the relation is weaker (greater) than $\succ$ in [7]. For this we can verify the following by constructing the symmetric closure of the precongruence and showing that it is reduction-closed and respects action predicates.

3.6. Proposition. $\succ$ *is a subset of the maximum sound equality in any system.*

Now we go back to the system $\mathbf{cc}_!$. The following $\mathcal{G}_l(ab)$ and $\mathcal{G}_r(ab)$ are the key to the prefix map in 3.9 later. Here a "floating message" retains the "state" of the agent. The verification is mechanical and is therefore omitted.

3.7. Proposition. (persistent agents) *We define* $\mathcal{G}_l(ab)$ *and* $\mathcal{G}_r(ab)$ *with* $\tilde{c} \overset{\text{def}}{=} c_1 ... c_5$ *fresh and pairwise distinct as follows.*

$$\mathcal{G}_l(ab) \overset{\text{def}}{=} \tilde{c} \blacktriangleright (!\mathcal{D}(ac_1c_2),\ !\mathcal{B}_l(c_1c_3),\ !\mathcal{D}(c_3c_4c_5),\ !\mathcal{FW}(c_4b),\ !\mathcal{S}(c_5c_2a))$$
$$\mathcal{G}_r(ab) \overset{\text{def}}{=} \tilde{c} \blacktriangleright (!\mathcal{D}(ac_1c_2),\ !\mathcal{B}_r(c_1c_3),\ !\mathcal{D}(c_4c_3c_5),\ !\mathcal{FW}(bc_4),\ !\mathcal{S}(c_5c_2a))$$

Then we have: $c \blacktriangleright (\mathcal{M}(cb), \mathcal{G}_l(ce)) \succ !\mathcal{FW}(be)$ *and* $c \blacktriangleright (\mathcal{M}(cb), \mathcal{G}_r(ce)) \succ !\mathcal{FW}(eb)$. *Note* $\mathcal{G}_l(ab)$ *is* a^-*-pointed, but* $\mathcal{G}_r(ab)$ *is not* a^-*-pointed.*

Next we introduce important algebraic laws related with one of the essential operations in the name passing framework, i.e. *forwarding*, assuming the following annotations, which denote how each name is used in the rules of interaction:

$$\mathcal{M}(u^+v^{\pm}),\ \mathcal{D}(u^-v^+w^+),\ \mathcal{K}(u^-),\ \mathcal{FW}(u^-v^+),\ \mathcal{B}_l(u^-v^+),\ \mathcal{B}_r(u^-v^-),\ \mathcal{S}(u^-v^-w^+).$$
$$!\mathcal{D}(u^-v^+w^+),\ !\mathcal{K}(u^-),\ !\mathcal{FW}(u^-v^+),\ !\mathcal{B}_l(u^-v^+),\ !\mathcal{B}_r(u^-v^-),\ !\mathcal{S}(u^-v^-w^+).$$

These laws are valid for $\mathbf{cc}_!$ and its extension in the next section (and also in a wide range of systems who have atoms with similar behaviour). Verification is mechanical and is omitted again.

3.8. Proposition. (*algebraic laws* (1))

(i) $c \blacktriangleright (\mathcal{FW}(e^-c^+), C(c^-\tilde{w})) \succ C(e^-\tilde{w})$ with $c \notin \{e, \tilde{w}\}$ (if $C \in \mathbf{A}$).

(ii) $c \blacktriangleright (!\mathcal{FW}(e^-c^+), !C(c^-\tilde{w})) \succ !C(e^-\tilde{w})$ with $c \notin \{e, \tilde{w}\}$.

(iii) $c \blacktriangleright (\mathcal{FW}(c^-e^+), C(\tilde{v}c^+\tilde{w})) \succ C(\tilde{v}c^+\tilde{w})$ with $c \notin \{e, \tilde{v}, \tilde{w}\}$ (if $C \in \mathbf{A}$).

(iv) $c \blacktriangleright (!\mathcal{FW}(c^-e^+), !C(\tilde{v}c^+\tilde{w})) \succ !C(\tilde{v}c^+\tilde{w})$ with $c \notin \{e, \tilde{v}, \tilde{w}\}$.

3.9. The Prefix Map. In [7], it was shown that we can form, in **cc**, an agent written as $a^*x.P$ whose behaviour is essentially an agent with input guard as found in the ν-calculus. We now show that $\mathbf{cc}_!$ enjoys the same property. In the following rules, (I) to (XII) are the same as found in [7], while the remaining ones are newly introduced.
We define $u^*x.P$, called *u-pointed abstraction of x over P*, by inductively applying rules from (I) to (XIX) in this order.

$$
\begin{array}{lrcll}
(\textsc{i}) & u^*x.(P,\ Q) & \stackrel{\mathrm{def}}{=} & c_1c_2 \blacktriangleright (\mathcal{D}(uc_1c_2),\ c_1^*x.P,\ c_2^*x.Q) & \\
(\textsc{ii}) & u^*x.c' \blacktriangleright P & \stackrel{\mathrm{def}}{=} & c \blacktriangleright u^*x.P\{c/c'\} & \\
(\textsc{iii}) & u^*x.\Lambda & \stackrel{\mathrm{def}}{=} & \mathcal{K}(u) & \\
(\textsc{iv}) & u^*x.C(v^+\tilde{w}) & \stackrel{\mathrm{def}}{=} & c \blacktriangleright (\mathcal{S}(ucv),\ C(c^+\tilde{w})) & (\mathrm{a}) \\
(\textsc{v}) & u^*x.C(v^-\tilde{w}) & \stackrel{\mathrm{def}}{=} & c \blacktriangleright (\mathcal{S}(uvc),\ C(c^-\tilde{w})) & (\mathrm{a}) \\
(\textsc{vi}) & u^*x.\mathcal{M}(vx) & \stackrel{\mathrm{def}}{=} & \mathcal{FW}(uv) & (\mathrm{b}) \\
(\textsc{vii}) & u^*x.\mathcal{FW}(xv) & \stackrel{\mathrm{def}}{=} & \mathcal{B}_l(uv) & (\mathrm{b}) \\
(\textsc{viii}) & u^*x.\mathcal{FW}(vx) & \stackrel{\mathrm{def}}{=} & \mathcal{B}_r(uv) & (\mathrm{b}) \\
(\textsc{ix}) & u^*x.C(\tilde{v}x^+\tilde{w}) & \stackrel{\mathrm{def}}{=} & c \blacktriangleright u^*x.(\mathcal{FW}(cx),\ C(\tilde{v}c^+\tilde{w})) & (\mathrm{c}) \\
(\textsc{x}) & u^*x.C(x^-\tilde{v}) & \stackrel{\mathrm{def}}{=} & c \blacktriangleright u^*x.(\mathcal{FW}(xc),\ C(c^-\tilde{v})) & \\
(\textsc{xi}) & u^*x.\mathcal{B}_r(vx^-) & \stackrel{\mathrm{def}}{=} & c_1c_2c_3 \blacktriangleright u^*x.(\mathcal{D}(vc_1c_2),\ \mathcal{B}_r(c_1c_3),\ \mathcal{S}(c_2xc_3)) & (\mathrm{b}) \\
(\textsc{xii}) & u^*x.\mathcal{S}(vx^-w) & \stackrel{\mathrm{def}}{=} & c_1c_2 \blacktriangleright u^*x.(\mathcal{S}(vc_1c_2),\ \mathcal{M}(c_1x),\ \mathcal{B}_l(c_2w)) & (\mathrm{b}) \\
(\textsc{xiii}) & u^*x.!C(v^-\tilde{w}) & \stackrel{\mathrm{def}}{=} & c_1c_2 \blacktriangleright (u^*x.\mathcal{M}(c_1v),\ \mathcal{G}_l(c_1c_2),\ !C(c_2\tilde{w})) & (\mathrm{a}) \\
(\textsc{xiv}) & u^*x.!\mathcal{FW}(xv) & \stackrel{\mathrm{def}}{=} & c_1c_2c_3 \blacktriangleright (u^*x.(\mathcal{M}(c_1x),\ \mathcal{M}(c_2v)),\ \mathcal{G}_l(c_1c_3),\ \mathcal{G}_r(c_2c_3)) & (\mathrm{b}) \\
(\textsc{xv}) & u^*x.!\mathcal{FW}(vx) & \stackrel{\mathrm{def}}{=} & c_1c_2c_3 \blacktriangleright (u^*x.(\mathcal{M}(c_1v),\ \mathcal{M}(c_2x)),\ \mathcal{G}_l(c_1c_3),\ \mathcal{G}_r(c_2c_3)) & (\mathrm{b}) \\
(\textsc{xvi}) & u^*x.!C(\tilde{v}x^+\tilde{w}) & \stackrel{\mathrm{def}}{=} & c_1c_2 \blacktriangleright (u^*x.(\mathcal{M}(c_1x),\ !C(\tilde{v}c_2^+\tilde{w})),\ \mathcal{G}_r(c_1c_2)) & (\mathrm{c}) \\
(\textsc{xvii}) & u^*x.!C(x^-\tilde{w}) & \stackrel{\mathrm{def}}{=} & c_1c_2 \blacktriangleright (u^*x.(\mathcal{M}(c_1x),\ !C(c_2\tilde{w})),\ \mathcal{G}_l(c_1c_2)) & \\
(\textsc{xviii}) & u^*x.!\mathcal{B}_r(vx^-) & \stackrel{\mathrm{def}}{=} & c_1c_2c_3 \blacktriangleright u^*x.(!\mathcal{D}(vc_1c_2),\ !\mathcal{B}_r(c_1c_3),\ !\mathcal{S}(c_2xc_3)) & (\mathrm{b}) \\
(\textsc{xix}) & u^*x.!\mathcal{S}(vx^-w) & \stackrel{\mathrm{def}}{=} & c_1c_2c_3 \blacktriangleright u^*x.(!\mathcal{S}(vc_1c_2),\ \mathcal{M}(c_1x),\ !\mathcal{D}(c_2c_3c_1),\ !\mathcal{B}_l(c_3w)) & (\mathrm{b})
\end{array}
$$

In the above, (a) stands for $(x \notin \{v\tilde{w}\})$, (b) stands for $(x \neq v)$ and (c) stands for $(x \notin \{\tilde{v}\})$ with c, c_1, c_2, c_3 are fresh and pairwise distinct.

With the mapping, we can show that, as well as $(u^*x.P)\sigma \equiv_\alpha u\sigma^*x.P\sigma$ with σ not using x and $u^*x.P \equiv_\alpha u^*z.P\{z/x\}$ (z fresh),

(1) $u^*x.P$ is u^--pointed.

(2) $u^*x.P,\ \mathcal{M}(uv) \ \longrightarrow\!\!\succ\ P\{v/x\}$.

(this will be formally stated in the next section for the full system). On the basis of the mapping, we use the following construction, which already appeared in the proof of 2.10: with z, c, and y all fresh and distinct,

$$
\begin{aligned}
\bar{u}^*\!:[v_1..v_n].P &\stackrel{\text{def}}{=} c \blacktriangleright(\mathcal{M}(uc), c^*z.(\mathcal{M}(zv_1), c^*z.(.., c^*z.(\mathcal{M}(zv_n), P)..)))\\
u^*\!:(x_1..x_n).P &\stackrel{\text{def}}{=} u^*y.c \blacktriangleright(\mathcal{M}(yc), c^*x_1.(c \blacktriangleright(\mathcal{M}(yc), ..c \blacktriangleright(\mathcal{M}(yc), c^*x_n.P))))
\end{aligned}
$$

for which properties as stated in 2.10 easily follow (note $\mathcal{M}(uv)$ is u^+-pointed).

We end this section with an interesting property of $\mathbf{cc}_!$, which, in effect, tells us that $\mathbf{cc}_!$ can *never* get the power of representing unbounded replication of terms. This shall be shown with the following notion of *name distinction* (cf. [14]).

3.10. Definition. (*distinction*) First we define: $\langle\varepsilon\rangle \stackrel{\text{def}}{=} \varepsilon$ and, if $v \in \{\tilde{v}\}$ then $\langle\tilde{v}v\rangle \stackrel{\text{def}}{=} \langle\tilde{v}\rangle$, otherwise $\langle\tilde{v}v\rangle \stackrel{\text{def}}{=} \langle\tilde{v}\rangle\cdot v$. Then we introduce two mappings $f_N(\ ) : \mathcal{P}_{cc_!} \to (\mathbf{N}\cup\mathbf{V})^*$ and $f_R(\ ) : \mathcal{P}_{cc_!} \to \mathbf{N}^*$, inductively as follows. Remember, by our convention stipulated in 2.3, we only consider terms where all names in binding occurrences are distinct and disjoint from the free names.

$$
\begin{cases}
f_N(\Lambda) \stackrel{\text{def}}{=} \varepsilon\\
f_N(C(\tilde{v})) \stackrel{\text{def}}{=} \langle\tilde{v}\rangle\\
f_N(a \blacktriangleright P) \stackrel{\text{def}}{=} f_N(P)\\
f_N(P,Q) \stackrel{\text{def}}{=} \langle f_N(P)\cdot f_N(Q)\rangle
\end{cases}
\qquad
\begin{cases}
f_R(\Lambda) \stackrel{\text{def}}{=} \varepsilon\\
f_R(C(\tilde{v})) \stackrel{\text{def}}{=} \varepsilon\\
f_R(P,Q) \stackrel{\text{def}}{=} f_R(P)\cdot f_R(Q)\\
f_R(a \blacktriangleright P) \stackrel{\text{def}}{=} \begin{cases} \langle f_R(P)\rangle & \text{if } a \notin \{f_N(P)\}\\ \langle a\cdot f_R(P)\rangle & \text{otherwise}\end{cases}
\end{cases}
$$

We easily verify that, with $\equiv'$ formed as $\equiv$ without $\equiv_\alpha$, $P \equiv' Q$ implies $\{f_N(P)\} = \{f_N(Q)\}$ and $\{f_R(P)\} = \{f_R(Q)\}$. Note that $f_R(P)$ is the set of bound names in P which *do occur* in P (e.g. $f_R(a \blacktriangleright \Lambda) = \varepsilon$), and that $\{f_R(P)\} \subset \{f_N(P)\}$.

The set $\{f_N(P)\}$ (resp. $\{f_R(P)\}$) is called *distinction of identifires* (resp. *bound names*). Now we show that, in $\mathbf{cc}_!$, distinction can never increase during reduction.

3.11. Proposition. *Let $P \in \mathcal{P}_{cc_!}$, then, for any P', $P \twoheadrightarrow P'$ implies, for some $P'' \equiv_\alpha P'$, $\{f_N(P)\} \supset \{f_N(P'')\}$, and $\{f_R(P)\} \supset \{f_R(P'')\}$.*

The proposition intuitively shows that the replication such as $!^*_a c \blacktriangleright \mathcal{M}(bc)$ may not be able to be represented in $\mathbf{cc}_!$. This, in fact, can be formally proved, though the proof is established using not only the notion of distinction but also several equational appratus which we do not discuss here. Interested readers may consult [4].

4. THE FULL SYSTEM

4.1. The System $\mathbf{cc}_\varphi$. The full system for which we prove Theorem 2.17 is introduced in the following. Only atoms called a name generator and templates are added on the top of $\mathbf{cc}_!$. We begin with the atom called *name generator.*

$\#\mathcal{N}$ denotes a name generator, with $ar(\#\mathcal{N}) = 1$ and $pol(\#\mathcal{N}) = -$. The rule of interaction of $\#\mathcal{N}(u)$ is as follows.

$$(\mathrm{GEN}) \qquad \#\mathcal{N}(u), \mathcal{M}(uv) \longrightarrow c \blacktriangleright \mathcal{M}(vc),\ \#\mathcal{N}(u)$$

Thus it unboundedly generates a new name at each request, and sends it to v which it has just received. If one uses the prefix notation, it may be written as $!u^*x.c \blacktriangleright \mathcal{M}(xc)$. What we want to do next is to instantiate a new atomic agent by new and existing names. For the purpose we introduce the set of *template atoms*, or simply *templates*, as follows.

$$\mathbf{A}_\# \stackrel{\mathrm{def}}{=} \{\#C \mid C \in \mathbf{A} \cup \mathbf{A}_!\ \}$$

with $ar(\#C) = 1$ and $pol(\#C) = -$. Note $\#$ is not a syntactic connective: $\#\mathcal{D}$ is only another atom. Thus we have $\#\mathcal{D}$ as a template of $\mathcal{D}$, and $\#!\mathcal{D}$ as that of $!\mathcal{D}$. The interaction rule for each template is given:

$$(\mathrm{TEM}) \qquad \#C(u),\ \mathcal{M}(uv) \longrightarrow \#C(u),\ v^*{:}(\tilde{x}).C(\tilde{x})$$

where $v^*{:}\,(\tilde{x}).C(\tilde{x})$ with $ar(C) = len(\tilde{x})$ and variables in $\tilde{x}$ distinct and fresh, is given by the mapping in 3.9, so that a template creates a mass of basic atoms and atomic replicators, which as a whole is ready to receive an instantiation message. With these additional agents we can make atoms with names as we like, e.g., writing $\bar{a}^*{:}[\tilde{x}]$ for $\bar{a}^*{:}[\tilde{x}].\Lambda$:

$$\begin{array}{rcl} e \blacktriangleright (\#\mathcal{FW}(t),\ \mathcal{M}(te),\ c \blacktriangleright \bar{e}^*{:}[bc]) & \longrightarrow & e \blacktriangleright (e^*{:}(xy).\mathcal{FW}(xy), c \blacktriangleright \bar{e}^*{:}[bc]),\ \#\mathcal{FW}(t) \\ & \succ & c \blacktriangleright \mathcal{FW}(bc), \#\mathcal{FW}(t). \end{array}$$

Now the set of all atoms for the full system is given by the following union.

$$\mathbf{A}_\varphi \stackrel{\mathrm{def}}{=} \mathbf{A}\ \cup\ \mathbf{A}_!\ \cup\ \mathbf{A}_\#\ \cup\ \{\ \#\mathcal{N}\ \}$$

This finishes the introduction of the system called $\mathbf{cc}_\varphi$. Reduction relation is defined following the scheme in 2.5, and the equality notions are given likewise. The maximum sound equality is denoted by $=_{cc}$ again. $\mathcal{P}_{cc_\varphi}$ is the set of terms for the full systems. Hereafter symbols for relations generally defined for systems in S are considered to be their instances for $\mathbf{cc}_\varphi$: they include such symbols as $\rightarrow_\beta$, $=_\beta$, $\Rightarrow_{\mathbb{B}}$, $=_{\mathbb{B}}$, $\succ$ etc. We note that algebraic laws in 3.8 hold without any change in $\mathbf{cc}_\varphi$, with the extension of:

$$c \blacktriangleright (!\mathcal{FW}(ec),\ \#C(c^-))\ \succ\ \#C(e^-) \qquad \text{with } e \neq c$$

which is again mechanically verified.

4.2. Prefix Closure. For the newly introduced atoms, we formulate the mapping for prefixing. We only add Rules (XX) and (XXI) which correspond to Rules (XIII) and (XVII) in Definition 3.9, respectively. Throughout the paper, $\#\mathcal{N}$ is included in $\#C$.

$$
\begin{array}{llll}
(\text{XX}) & u^*x.\#C(v^-) & \overset{\text{def}}{=} & c_1c_2 \blacktriangleright (u^*x.\mathcal{M}(c_1v),\ \mathcal{G}_l(c_1c_2),\ \#C(c_2^-)) \quad (x \neq v) \\
(\text{XXI}) & u^*x.\#C(x^-) & \overset{\text{def}}{=} & c_1c_2 \blacktriangleright (\mathcal{FW}(uc_1),\ \mathcal{G}_l(c_1c_2),\ \#C(c_2^-))
\end{array}
$$

The next result, together with $(u^*x.P)\sigma \equiv_\alpha u\sigma^*x.P\sigma$ for σ not using x and $u^*x.P \equiv_\alpha u^*z.P\{z/x\}$ for z fresh, both of which are easy, shows the behavioural closure under prefix in $\mathbf{cc}_\varphi$.

4.3. Proposition. (*prefix closure*) *For $u^*x.P$, we have:*

(1) $u^*x.P$ is u^--pointed.
(2) $u^*x.P,\ \mathcal{M}(uv) \ \longrightarrow\!\succ\ P\{v/x\}$.

PROOF. By rule induction on the Rules (I)~(XXI). Pointedness is easy. For reduction, the reasoning for Rules (I)~(XII) are just similar to [7]. Regarding the remaining rules, the rules except (XVIII) and (XIX) can be directly proved using Proposition 3.7 and the the algebraic laws. Regarding Rules (XVIII) and (XIX), we show the property with a similar reasoning to Rules (XI) and (XII) as found in [7], except that we use $\Rightarrow_{\mathbb{B}}$ instead of $\twoheadrightarrow_\beta$. □

4.4. Algebraic Laws (2). For atoms which are *not* templates, one can construct their new instances using their templates and (possibly new) names as we like. Regarding the templates themselves, however, the situation is different, since we cannot introduce a template for a template, ending up with an unbounded number of atoms.

The following however shows a possibility of forming a template for a template within $\mathbf{cc}_\varphi$, which is finally realized in 5.5. Using the fact $c \blacktriangleright (!\mathcal{FW}(ac), \#C(c)) \succ \#C(a)$ (cf. 4.1), we introduce the precongruent closure of the following relation, denoted by $\succ_{\text{sh}}$ ($\prod_{i=1}^n P_n$ stands for $P_1, P_2, ..., P_n$ and if $n = 0$, it is defined as Λ):

$$c \blacktriangleright (\textstyle\prod_{i=1}^n !\mathcal{FW}(a_ic),\ C(c\tilde{v}),\ Q) \ \succ_{\text{sh}}\ c \blacktriangleright (\textstyle\prod_{i=1}^n C(a_i\tilde{v}),\ C(c\tilde{v}),\ Q)$$

where (1) $C \in \mathbf{A}_\varphi \backslash \mathbf{A}$ ("\" is the set difference) and (2) after any reduction sequence, free c can never occur negatively and also it will never occur in an atomic agent whose subject is free, where *positive/negative occurrence* is understood in regard of annotations of polarities as found in 3.9 and 4.2 (we note that the condition can be made statically checkable by a mild restriction). By showing the corresponding congruence is reduction-closed and respects $\Downarrow_{a^\theta}$, we know $\succ_{\text{sh}} \subset =_{cc}$. Let us define $\gtrsim$ as the precongruent closure of the union of $\succ$ and $\succ_{\text{sh}}$. Together with Propositions 3.6, we have:

4.5. Proposition. $P \gtrsim Q$ *implies* $P =_{cc} Q$.

We shall use this precongruence for the verification of our main result.

This finishes introduction of $\mathbf{cc}_\varphi$ and related notions as far as necessary for the subsequent development. We now go into the construction of replication for $\mathbf{cc}_\varphi$.

Remark. One may wonder whether adding only $\#\mathcal{N}(u)$ as a new atom is sufficient or not, considering the following scheme for the simple replicator, assuming c, c_1, c_2, y fresh and distinct.

$$!^*_a c \blacktriangleright P \overset{\text{def}}{=} c_1 c_2 \blacktriangleright (\#\mathcal{N}(c_1),\ !^*_a(\mathcal{M}(c_1 c_2), c_2^* y.P\{y/c\}))$$

But this does not work, since the map $c_2^* y.(P\{y/c\})$ itself introduces new names inside "$!_a$". Actually the rules for the name abstraction tell us that we *always* need new names to decompose prefixing, hence subterms become larger and larger, making the mapping non-well founded. The next natural question is whether all atomic replicators can be "defined" using combination of their templates and other atoms or not. We conjecture that this is not the case (though some can be indeed). Firstly, we must create at least $!\mathcal{FW}(ac)$ to share templates (cf. Definition 5.5). Then we need simple replicators to represent the name abstraction of $\#!\mathcal{FW}(ax)$ and $!\mathcal{FW}(ax)$ because $\mathcal{G}_l(ab)$ and $\mathcal{G}_r(ab)$ are used there. More formal account which would explicate the situation is called for.

5. ANALYSIS OF REPLICATION

5.1. The Basic Construction. The basic idea to form a simple replicator in $\mathbf{cc}_\varphi$, is to prepare a template for each atomic agent constituting the term to be replicated, and to instantiate these templates by appropriate names, which may include both new and existing names, new names being generated by $\#\mathcal{N}$. Those names should arrive in the specific order to each template: to realize this, however, we *cannot* use newly generated names for consecutive name passing as in the coding of 2.13, since this easily makes the mapping non-well-founded. But the replicator may be activated multiple times, and, if the activation should occur during its replication process, a template will receive two instantiation messages simultaneously, which may cause "confusion" of names of the kind given in footnote 5. To preclude this, while using a fixed number of private names, we let $!^*_u P$ receive the next activation from the outside only after the whole P is generated, informed by an internal acknowledgment message. We divide the construction into the following three parts, each being defined inductively. Suppose we want to form $!^*_u P$, where the cardinality of the distinction of P is n.

(1) Φ_H is the *interface* with the outside. Receiving an activation at u, it activates (each sub-part of) Φ_N, and waits for the message informing the accomplishment of replication from Φ_G. It is connected with n connections with Φ_N.

(2) Φ_N is the *emitter* of names. It sends the new and free names needed to make a copy of P to Φ_G. It consists of n sub-parts, each connected to and activated by Φ_H, emitting a new or an existing name to Φ_G. Thus Φ_N is connected to Φ_G with exactly n connections again.

(3) Φ_G does the actual replication. It has n input ports connected to Φ_N, through which it receives n distinct names. Names are then distributed to the templates of the atomic agents of P, reflecting the sharing structure of names among them. When all atomic agents are generated, it sends an acknowledgment message to Φ_H, to make Φ_H ready for the next request from the outside.

These three parts cooperate together to construct a required copy each time replication is requested. Now we form each part one by one. Throughout the construction, we need a simple mapping written $u^{\varphi}x.P$, which are derived from the definition of the prefix map in 3.9 and 4.2.

5.2. Definition. Let P be a term of **cc**. Then $u^{\varphi}x.P$ is defined as follows with c, c_1, c_2, c_3 fresh where rules are applied from (I) to (XII) in this order.

$$\begin{array}{rrcll}
(\textsc{i}) & u^{\varphi}x.(P,Q) & \overset{\text{def}}{=} & c_1c_2 \blacktriangleright(!\mathcal{D}(uc_1c_2),\ c_1{}^{\varphi}x.P,\ c_2{}^{\varphi}x.Q) & \\
(\textsc{ii}) & u^{\varphi}x.c' \blacktriangleright P & \overset{\text{def}}{=} & c \blacktriangleright u^{\varphi}x.P\{c/c'\} & \\
(\textsc{iii}) & u^{\varphi}x.\Lambda & \overset{\text{def}}{=} & !\mathcal{K}(u) & \\
(\textsc{iv}) & u^{\varphi}x.\mathcal{M}(vw) & \overset{\text{def}}{=} & c_1c_2 \blacktriangleright(!\mathcal{S}(uc_1c_2),\ !\mathcal{D}(c_2vc_1),\ \mathcal{M}(c_1w)) & (x \notin \{vw\}) \\
(\textsc{v}) & u^{\varphi}x.C(v^-\tilde{w}) & \overset{\text{def}}{=} & c \blacktriangleright(!\mathcal{S}(uvc),\ !C(c^-\tilde{w})) & (x \notin \{v\tilde{w}\}) \\
(\textsc{vi}) & u^{\varphi}x.\mathcal{M}(vx) & \overset{\text{def}}{=} & !\mathcal{FW}(uv) & (x \neq v) \\
(\textsc{vii}) & u^{\varphi}x.\mathcal{FW}(xv) & \overset{\text{def}}{=} & !\mathcal{B}_l(uv) & (x \neq v) \\
(\textsc{viii}) & u^{\varphi}x.\mathcal{FW}(vx) & \overset{\text{def}}{=} & !\mathcal{B}_r(uv) & (x \neq v) \\
(\textsc{ix}) & u^{\varphi}x.C(\tilde{v}_1x^+\tilde{v}_2) & \overset{\text{def}}{=} & c \blacktriangleright u^{\varphi}x.(\mathcal{FW}(cx),\ C(\tilde{v}_1c^+\tilde{v}_2)) & (x \notin \{\tilde{v}_1\}) \\
(\textsc{x}) & u^{\varphi}x.C(x^-\tilde{v}) & \overset{\text{def}}{=} & c \blacktriangleright u^{\varphi}x.(\mathcal{FW}(xc),\ C(c^-\tilde{v})) & \\
(\textsc{xi}) & u^{\varphi}x.\mathcal{B}_r(vx^-) & \overset{\text{def}}{=} & c_1c_2c_3 \blacktriangleright u^{\varphi}x.(\mathcal{D}(vc_1c_2),\ \mathcal{B}_r(c_1c_3),\ \mathcal{S}(c_2xc_3)) & (x \neq v) \\
(\textsc{xii}) & u^{\varphi}x.\mathcal{S}(vx^-w) & \overset{\text{def}}{=} & c_1c_2 \blacktriangleright u^{\varphi}x.(\mathcal{S}(vc_1c_2),\ \mathcal{M}(c_1x),\ \mathcal{B}_l(c_2w)) & (x \neq v)
\end{array}$$

Note that $u^{\varphi}x.P$ is u^--pointed: however it does *not* satisfy the reduction property of the lazy replication in 2.15.[5]

Now we define the mapping Φ_H, which generates activation messages, the number of which is the cardinality of the distinction of P. We use some notations: $u^{\varphi}.P$ stands for $u^{\varphi}x.P$ with x fresh, and $\mathcal{M}(u)$ stands for $c \blacktriangleright \mathcal{M}(uc)$ with c fresh.

---

[5] A conspicuous example is: $a^{\varphi}x.\mathcal{M}(xx) \overset{\text{def}}{=} cc_1c_2 \blacktriangleright(!\mathcal{D}(ac_1c_2),\ !\mathcal{B}_r(c_1c),\ !\mathcal{FW}(c_2c))$, for which we have, against our expectation: $u^{\varphi}x.\mathcal{M}(xx),\ \mathcal{M}(uv),\ \mathcal{M}(uw) \longrightarrow\!\!\!\!\rightarrow u^{\varphi}x.\mathcal{M}(xx),\ \mathcal{M}(wv),\ \mathcal{M}(vw)$. Thus private ports get confused between two actions, each of which should be semantically atomic. Note that this again suggests the necessity to generate new names each time replication occurs.

5.3. Definition. (*the mapping* Φ_H) Let $P \in \mathcal{P}_{cc_\varphi}$ and $len(f_N(P)) = n$. Then, given arbitrary u, h and $\tilde{b}$ with $\tilde{b} \stackrel{\text{def}}{=} b_1...b_n$, we define the mapping $\Phi_H\langle P\rangle(u, \tilde{b}, h)$ as follows.

$$\Phi_H\langle P\rangle(u, \tilde{b}, h) \stackrel{\text{def}}{=} \begin{cases} !\,\mathcal{D}_0(u) & \text{if } n = 0 \\ \mathcal{D}_n(ub_1...b_n),\ h^\varphi.\mathcal{D}_n(ub_1...b_n) & \text{otherwise} \end{cases}$$

where $\mathcal{D}_n(ub_1...b_n)$ is given by:

$$\begin{aligned} \mathcal{D}_0(u) &\stackrel{\text{def}}{=} \mathcal{K}(u) \\ \mathcal{D}_1(ub_1) &\stackrel{\text{def}}{=} \mathcal{FW}(ub_1) \\ \mathcal{D}_2(ub_1b_2) &\stackrel{\text{def}}{=} \mathcal{D}(ub_1b_2) \\ \mathcal{D}_n(ub_1..b_n) &\stackrel{\text{def}}{=} c \blacktriangleright(\mathcal{D}_{n-1}(ub_1...c),\ \mathcal{D}(cb_{n-1}b_n)) \quad \text{with } n \geq 3 \text{ and } c \text{ fresh.} \end{aligned}$$

Note that, for any $n \in \mathbb{N}$, we have: $\mathcal{D}_n(ub_1...b_n),\ \mathcal{M}(ue) \longrightarrow\!\!\twoheadrightarrow_\beta \prod_{i=1}^n \mathcal{M}(b_ie)$.

Next we define the mapping Φ_N, which consists of several n_F's and several n_B's as its subparts. n_F emits a free name, while n_B emits a newly created name.

5.4. Definition. (*the mapping* Φ_N) Let $f_N(P) \stackrel{\text{def}}{=} l_1...l_n$, $\tilde{b} \stackrel{\text{def}}{=} b_1...b_n$ and $\tilde{c} \stackrel{\text{def}}{=} c_1...c_n$. Then we define

$$\Phi_N\langle P\rangle(\tilde{b}, \tilde{c}) \stackrel{\text{def}}{=} \prod_{i=1}^n \psi_N\langle P\rangle(l_i, b_i, c_i)$$

where:

$$\psi_N\langle P\rangle(l_i, b_i, c_i) \stackrel{\text{def}}{=} \begin{cases} n_B(b_ic_i) & \text{if } l_i \in \{f_R(P)\} \\ n_F(l_ib_ic_i) & \text{otherwise} \end{cases}$$

with $n_B(b_ic_i) \stackrel{\text{def}}{=} c \blacktriangleright(b_i{}^\varphi.\mathcal{M}(cc_i),\ \#\mathcal{N}(c))$ with c fresh, and $n_F(l_ib_ic_i) \stackrel{\text{def}}{=} b_i{}^\varphi.\mathcal{M}(c_il_i)$. Specifically if $n = 0$ then we take $\Phi_N\langle P\rangle(\varepsilon, \varepsilon) \stackrel{\text{def}}{=} \Lambda$.

5.5. The Fine Structure of Φ_G. Φ_G performs the main work of replication. It consists of two sub-parts.

(1) The first part *distributes*, with necessary duplication, the names which it receives from Φ_N, to the second part which uses them to generate a new term. If the term to be replicated has n distinction and m atomic agents each of which uses j_i distinct names, with $1 \leq i \leq m$, then it has n connections with Φ_N and $\sum_{1 \leq i \leq m} j_i$ connections with the second part. This part is concerned with the realization of the structure of name sharing among atomic agents constituting a term to be replicated.

(2) The second part performs the real replication. It is made from m sub-parts if a term to be copied consists of m atomic agents. Each sub-part is connected to the first part with the number of (distinct) names it uses to create an agent, and, after receiving the messages, generates the atomic agent. Then each of them emits a termination message, which, as a whole, are collected to finally produce a message to inform Φ_H that Φ_G finishes its job.

Each part needs a certain construction to perform its specific role, as listed below.

(1) For the first part, we need the following structure. Suppose $f_N(P) = l_{11}l_{12}...l_{1n_1}$, $f_N(Q) = l_{21}l_{22}...l_{2n_2}$, $f_N(P,Q) = l_1 l_2 ... l_n$, $\tilde{c_1} = c_{11}c_{12}...c_{1n_1}$, $\tilde{c_2} = c_{21}c_{22}...c_{2n_2}$, and $\tilde{c} = c_1 c_2 ... c_n$. Then we define $\Psi_c(\ ,\ ,\ )$ as

$$\Psi_c(\tilde{c}, \langle P,\ \tilde{c_1}\rangle, \langle Q,\ \tilde{c_2}\rangle) \overset{\text{def}}{=} \textstyle\prod_{i=1}^{n} \psi_c(l_i, c_i, \tilde{c_1}, \tilde{c_2}, \tilde{l_1}, \tilde{l_2})$$

where, with $0 \leq j \leq n_1$, $0 \leq k \leq n_2$ and $0 \leq i \leq n$,

$$\psi_c(l_i, c_i, \tilde{c_1}, \tilde{c_2}, \tilde{l_1}, \tilde{l_2}) \overset{\text{def}}{=} \begin{cases} !\mathcal{D}(c_i c_{1j} c_{2k}) & \text{if } \ l_i = l_{1j} = l_{2k} \\ !\mathcal{FW}(c_i c_{1j}) & \text{else if } \ l_i = l_{1j} \\ !\mathcal{FW}(c_i c_{2k}) & \text{else if } \ l_i = l_{2k} \end{cases}$$

Essentially speaking, this provides a *web of multiple* $!\mathcal{D}$*'s*, with appropriate duplication, which distribute names received through each input port coming from Φ_N to the templates, where the structure of the web faithfully mimicks that of name sharing among atomic agents constituting the term to be replicated.

(2) For the second part, the following construction is needed. Let $C \in \mathbf{A}_\varphi$. We define the mapping $\Psi_g(\ ,\ )$ as follows.

$$\begin{array}{rcll} \Psi_g(C,\ t) & \overset{\text{def}}{=} & \#C(t) & \text{if } C \in \mathbf{A} \cup \mathbf{A}_! \\ \Psi_g(\#C,\ tt') & \overset{\text{def}}{=} & \#!\mathcal{FW}(t),\ \#C(t') & \end{array}$$

In the second line above, we try to simulate the generation of a template by generation of a forwarder to it, cf. 4.4.

The construction of Φ_G follows. We note that the "name distribution" part appears only in (iv) below.

5.6. Definition. (*the mapping* Φ_G) The inductive map Φ_G is given by:

(i) $\Phi_G\langle\Lambda\rangle(\varepsilon, h) \overset{\text{def}}{=} \Lambda$.

(ii) If $C \in \mathbf{A} \cup \mathbf{A}_!$, then:

$$\begin{array}{l} \Phi_G\langle C(\tilde{v})\rangle(\tilde{c}, h) \\ \quad \overset{\text{def}}{=} \ tc \blacktriangleright (c_1{}^{\varphi} x_1 . c_2^* x_2 .. c_n^* x_n . (\mathcal{M}(tc), \bar{c}^*{:}[\tilde{v}\{\tilde{x}/\langle\tilde{v}\rangle\}].\mathcal{M}(h)), \Psi_g(C,\ t)) \end{array}$$

where $\tilde{c} \overset{\text{def}}{=} c_1 ... c_n$ and $\tilde{x} \overset{\text{def}}{=} x_1 ... x_n$, all newly introduced identifiers are fresh and pairwise distinct, and $len(\langle\tilde{v}\rangle) = n$.

Otherwise, with t, t', c, c_1, x all fresh and pairwise distinct, we define

$$\Phi_G\langle \#C(v)\rangle(c_1, h) \overset{\text{def}}{=} tt'c \blacktriangleright (c_1{}^{\varphi} x . (\mathcal{M}(tc), \bar{c}^*{:}[xt'].\mathcal{M}(h)),\ \Psi_g(\#C,\ tt')).$$

(iii) $\Phi_G\langle a \blacktriangleright P\rangle(\tilde{c}, h) \overset{\text{def}}{=} \Phi_G\langle P\rangle(\tilde{c}, h)$.

(iv) $\Phi_G\langle P, Q\rangle(\tilde{c}, h)$
$\overset{\text{def}}{=} h_1 h_2 \tilde{c}_1 \tilde{c}_2 \blacktriangleright(\Phi_G\langle P\rangle(\tilde{c}_1, h_1), \Phi_G\langle Q\rangle(\tilde{c}_2, h_2), \Psi_c(\tilde{c}, \langle P,\ \tilde{c}_1\rangle, \langle Q,\ \tilde{c}_2\rangle), \psi_s(hh_1h_2))$
where names in $\tilde{c}, \tilde{c}_1, \tilde{c}_2$ and h, h_1, h_2 are fresh and pairwise distinct, and $\psi_s(hh_1h_2)$ is defined as:

$$\psi_s(hh_1h_2) \overset{\text{def}}{=} \begin{cases} !\mathcal{S}(h_1h_2h) & \text{if } \tilde{c}_1 \neq \varepsilon \text{ and } \tilde{c}_2 \neq \varepsilon \\ !\mathcal{FW}(h_2h) \ \ (\text{resp. } !\mathcal{FW}(h_1h)) & \text{else if } \tilde{c}_1 = \varepsilon \ \ (\text{resp. } \tilde{c}_2 = \varepsilon) \\ \Lambda & \text{otherwise} \end{cases}$$

With this, the preliminary construction is complete. We can now form the structure for simple replication in $\mathbf{cc}_\varphi$ using the parts we have prepared so far.

5.7. Proposition. (replication in $\mathbf{cc}_\varphi$) *Let us define:*

$$\begin{cases} \mathbf{?}^* u \overset{\text{def}}{=} e \blacktriangleright \mathcal{M}(ue) \\ \mathbf{!}^*_u P \overset{\text{def}}{=} \tilde{c}\tilde{b}h \blacktriangleright(\Phi_H\langle P\rangle(u, \tilde{b}, h),\ \Phi_N\langle P\rangle(\tilde{b}, \tilde{c}),\ \Phi_G\langle P\rangle(\tilde{c}, h)) \end{cases}$$

with $len(f_N(P)) = len(\tilde{b}) = len(\tilde{c})$ *and names in* $\tilde{c}$, $\tilde{b}$, *and* h, e *are all fresh and pairwise distinct. Then, for any* $P \in \mathcal{P}_{cc_\varphi}$ *and* u, *we have*, as well as $(\mathbf{?}^*u)\sigma \equiv_\alpha \mathbf{?}^*u\sigma$ *and* $(\mathbf{!}^*_u P)\sigma =_{\mathbb{S}} \mathbf{!}^*_{u\sigma} P\sigma$:

(1) $\mathbf{?}^*u$ is u^+-pointed and $\mathbf{!}^*_u P$ is u^--pointed.

(2) $\mathbf{!}^*_u P,\ \mathbf{?}^*u \ \longrightarrow \gtrsim\ P,\ \mathbf{!}^*_u P.$

Note that, by Proposition 4.5, this establishes the behavioural closure property stated in (i) of 2.15. Moreover, Proposition 4.3 has already shown that $\mathbf{cc}_\varphi$ is behaviourally closed under prefix. Thus, if the proposition holds, we have proved Theorem 2.17. The key reasoning to verify the correctness of the behaviour of the above $\mathbf{!}^*_u P$ follows.

PROOF. (outline) To verify the stated properties, we first show the following claims, one for each of the three parts of the $\mathbf{!}^*_u P$.

CLAIM I. With u, $\tilde{b} \overset{\text{def}}{=} b_1..b_n$, and h being pairwise distinct and fresh, if $n \neq 0$:

$$\Phi_H\langle P\rangle(u, \tilde{b}, h),\ \ \mathbf{?}^*u \ \ \longrightarrow\Rightarrow_{\mathbb{B}}\ e \blacktriangleright \textstyle\prod_{i=1}^{n} \mathcal{M}(b_ie),\ h^\varphi.\mathcal{D}_n(ub_1...b_n)$$

and if else: $\Phi_H\langle P\rangle(u, \tilde{b}, h),\ \ \mathbf{?}^*u \ \ \longrightarrow\ \ \Phi_H\langle P\rangle(u, \tilde{b}, h).$

CLAIM II. Assume $f_N(P) \overset{\text{def}}{=} l_1..l_n$ and the length of $f_R(P)$ is m, and $\tilde{b} \overset{\text{def}}{=} b_1..b_n$ and $\tilde{c} \overset{\text{def}}{=} c_1..c_n$ are pairwise distinct and fresh. We let $l_{j_1},..l_{j_m}$ be in $\{f_R(P)\}$, while $l_{k_1},..l_{k_{(n-m)}}$ be in $\{f_N(P)\}\backslash\{f_R(P)\}$. Then $\Phi_N\langle P\rangle(\tilde{b}, \tilde{c})$ contains no redex. Moreover:

$$\Phi_N\langle P\rangle(\tilde{b}, \tilde{c}),\ e \blacktriangleright \textstyle\prod_{i=1}^{n} \mathcal{M}(b_ie)\ \ (\longrightarrow\gtrsim)^n\ \ \prod_{i=1}^{n-m} \mathcal{M}(c_{k_i}l_{k_i}),\ g_1..g_m \blacktriangleright \prod_{i=1}^{m} \mathcal{M}(c_{j_i}g_i)$$

with $g_1..g_m$ being fresh and mutually distinct, and, moreover each reduction in the above only involves one of ports $b_1..b_n$, each of which is used only once.

Claim III. Assume $f_N(P) \overset{\text{def}}{=} w_1..w_n$, $f_R(P) \overset{\text{def}}{=} g_1..g_m$, and names in $\tilde{c} \overset{\text{def}}{=} c_1..c_n$ are pairwise distinct and fresh. Then we have: (1) $\Phi_G\langle P\rangle(\tilde{c}, h)$ does not contain any redex. (2) we have the following reduction:

$$g_1...g_m \blacktriangleright \prod_{i=1}^n \mathcal{M}(c_i w_i),\ \Phi_G\langle P\rangle(\tilde{c}, h)\ \ (\longrightarrow \gtrsim)^*\ \ P,\ \mathcal{M}(h),\ \ \Phi_G\langle P\rangle(\tilde{c}, h)$$

where each of the above reduction only involves one distinct port name other than those in $\{\tilde{w}\}$. And (3), if we have:

$$g_1..g_m \blacktriangleright \prod_{i=1}^n \mathcal{M}(c_i w_i),\ \Phi_G\langle P\rangle(\tilde{c}, h)\ \ \longrightarrow\!\!\!\!\rightarrow\ \ R_0\ \ \longrightarrow\ \ R_1$$

with $h \notin \mathcal{AN}_+(R_0)$ and $h \in \mathcal{AN}_+(R_1)$ and the above $\longrightarrow\!\!\!\!\rightarrow$ does *not* involve reduction through ports in $\{\tilde{w}\}$, then $R_1\ \ \gtrsim\ \ P,\ \mathcal{M}(h),\ \Phi_G\langle P\rangle(\tilde{c}, h)$ always.

Among the Claims, Claim I is easy by induction on the number of distinction of P. Then Claim II is shown by checking the behaviour of each n_F and n_B, formally using induction on P. Note that the case of $c \blacktriangleright P$ only differs from the case of P in a single replacement of n_F by n_B. The lack of redex is easy by inspection of the term structure. For Claim III, we should use the structural induction over P. (1) is easy. Regarding (2) and (3), the case of Λ is obvious. In the case of $C(\tilde{v})$, we need to check that the sequence of names $\tilde{v}$ is safely sent to its template using $\Rightarrow_{\mathbb{B}}$, then we use the algebraic law in 4.4. In the case of $c \blacktriangleright P$, by the inductive hypothesis of P, the result is obtained with $\Phi_G\langle c \blacktriangleright P\rangle(\tilde{c}, h) \overset{\text{def}}{=} \Phi_G\langle P\rangle(\tilde{c}, h)$. In the case of (P, Q), the only difficulty arises when there are some free names which occur both in P and Q. But they are distributed by $!\mathcal{D}$ in Ψ_c in 5.5 via $\Rightarrow_{\mathbb{B}}$ and, moreover, two final activations, $\mathcal{M}(h_1)$ and $\mathcal{M}(h_2)$, are collected by $!\mathcal{S}(h_1 h_2 h)$ in Definition 5.6 to generate $\mathcal{M}(h)$ by $\Rightarrow_{\mathbb{B}}$, hence the result.

Now, by inspecting that no redex exists between three parts constituting $!^*_u P$, we know that $!^*_u P$ is u^--pointed. Next, as to the reduction property, using the fact that each intermediate communication among Φ_H, Φ_N, and Φ_G can only involve distinct port name which is used only once (Claim III is crucial) and, moreover, is private within $!^*_u P$, we know that all such intermediate interactions belong to $\Rightarrow_{\mathbb{B}}$. Hence, noting that $\Rightarrow_{\mathbb{B}} \subset \gtrsim$, we have, for any P, $!^*_u P,\ ?^* u\ \longrightarrow \gtrsim\ !^*_u P,\ P$. Finally we can verify $(!^*_u P) =_{\mathbb{S}} !^*_{u\sigma} P\sigma$ by showing its congruent closure is sound by operational reasoning as was done for the reduction property, while $(?^* u)\sigma \equiv_\alpha ?^* u\sigma$ is obvious. Hence done. $\square$

This completes the establishment of Theorem 2.17.

6. Open Problems

Since the study of the representability of the kind we gave in the present paper is young, there are several elementary problems whose answers have not been known yet. Apart from such issues as more abstract treatment of the concerned computational scheme, some of immediately arising topics include:

(1) Our formulation of the behavioural closure property, both for prefix and replication, is given using the mediating equality $=_{\mathbb{S}}$, e.g. $!^*_u P,\ ?^* u \ \longrightarrow =_{\mathbb{S}}\ !^*_u P,\ P$ (2.15). This is quite different from the case of representability of the functional abstraction using SKI-combinators, where we can reach the required term by straightforward reduction, i.e. $(\lambda^* x.M)N \longrightarrow\!\!\!\!\rightarrow M\{N/x\}$. We note that, in our actual constructions in [7] and in the present paper, the necessary behavioural equality is a proper subset of the canonical equality, which can indeed be restricted to the one which is finitely axiomatizable. Thus the essential question in this regard is whether one can find an instance of S and the representation scheme such that the mediating equality is within, or close to, say, $\Rightarrow_{\mathbb{B}}$. We believe that to have $\Rightarrow_{\mathbb{B}}$ is very difficult.

(2) Another interest should be the minimality of the system which own the closure properties (or other ones for that matter). The present set of atoms comes from, as might be obvious, the inductive scheme we utilized for the verification of the properties itself. Can a system which uses a proper subset of those atoms (excluding those related with $\mathcal{FW}$ and $\mathcal{K}$ since we easily know they are definable from $\mathcal{D}$) have the similar closure and related properties? Can we prove its minimality? Other representation schemes than exposed in this paper would yield quite different combinations of atoms for the closure properties, though the present one seems to be one of the almost minimum constructions.

(3) There is one remaining operation which we have not treated so far, but whose semantic significance deserves analysis, namely some form of *summation* as found in CCS [10]. Since the full summation seems very subtle to treat at the initial step (in spite of the fact that a certain operational encoding does seem possible), we may begin with the following seemingly mild representability problem (the scheme of summation is from [12]: one may admit the relativisation of e.g. pointedness by $=_{\mathbb{S}}$): For some $\mathbb{S}$, given $P_{i\langle u_i{}^{\theta_i}\rangle}$ for $1 \leq i \leq n$, can we form an agent $\sum^*_{1\leq i\leq n} P_{i\langle u_i{}^{\theta_i}\rangle}$ with the following properties?

    (i) $\sum^*_{1\leq i\leq n} P_{i\langle u_i{}^{\theta_i}\rangle} \not\longrightarrow$.

    (ii) $P_{k\langle u_k{}^{\theta_k}\rangle}, C^{\overline{\theta_k}}(u_k\tilde{v}) \ \longrightarrow\ R \ \Rightarrow\ \sum^*_{1\leq i\leq n} P_{i\langle u_i{}^{\theta_i}\rangle},\ C^{\overline{\theta_k}}(u_k\tilde{v}) \ \longrightarrow =_{\mathbb{S}}\ R.$

    (iii) $\sum^*_{1\leq i\leq n} P_{i\langle u_i{}^{\theta_i}\rangle},\ C^{\overline{\theta}}(u\tilde{v}) \ \longrightarrow\ R \ \Rightarrow\ \exists k.\ P_{k\langle u^{\theta}\rangle}, C^{\overline{\theta}}(u\tilde{v}) \ \longrightarrow =_{\mathbb{S}}\ R.$

    Even this seems non-trivial, and some clear formal results in this context concerning this or other forms of summation may give us important insights on the semantic nature of the operation.

Acknowledgements The authors thank Makoto Kubo for his suggestions and advice. They also thank Professor Mario Tokoro for his support. Both authors are partially supported by JSPS Fellowships for Japanese Junior Scientists, for which they are grateful.

REFERENCES

1. Abramsky, S., The Lazy Lambda Calculus. D. Turner, editor, *Research topics in Functional Programming*, Addison Wesley, 1990.
2. Boudol, G., *Asynchrony and π-calculus*, INRIA Report 1702, INRIA, Sophia Antipolis, May, 1991.
3. Girard, J.-Y., Linear Logic, *Theoretical Computer Science*, Vol. 50, pp.1–102, 1987.
4. Honda, K., Kubo, M. and Yoshida, N., Combinatory Representation of Mobile Processes Part I/II, To appear as Keio CS Technical Report, 1994.
5. Honda, K. and Tokoro, M., An Object Calculus for Asynchronous Communication. *ECOOP'91*, LNCS 512, pp.21–52, Springer-Verlag 1991.
6. Honda, K. and Yoshida, N., On Reduction-Based Process Semantics, *Foundations of Software Technology and Theoretical Computer Science*, LNCS 761, pp.371–387, Springer-Verlag, 1993.
7. Honda, K. and Yoshida, N., Combinatory Representation of Mobile Processes, *Proc. 21th. Annual Symposium on Principles of Programming Languages*, pp.348–360, ACM press, 1994.
8. Jones, C.B., Constraining interference in an object-based design method, *TAPSOFT'93*, LNCS 668, pp.136–150, Springer Verlag, 1993.
9. Lafont, Y., Interaction Nets, *Proc. 17th. Annual Symposium on Principles of Programming Languages*, pp. 95–108, ACM press, 1990.
10. Milner, R., *A Calculus of Communicating Systems*, LNCS 76, Springer Verlag, 1980.
11. Milner, R., Functions as Processes. *Mathematical Structure in Computer Science*, 2(2), pp.119–146, 1992.
12. Milner, R., Polyadic π-Calculus. *Logic and Algebra of Specification*, Springer-Verlag, 1992.
13. Milner, R. and Sangiorgi, D., Barbed Bisimulation. *Proc. of ICALP'92*, LNCS 623, pp.685–695, Springer-Verlag, 1992.
14. Milner, R., Parrow, J.G. and Walker, D.J., A Calculus of Mobile Processes, *Information and Computation* 100(1), pp.1–77, 1992.
15. Thomsen, B., A calculus of higher order communicating systems. *Proc. 16th. Annual Symposium on Principles of Programming Languages*, pp. 143–145, 1989.