A fully-abstract semantics of $\lambda\mu$ in the π-calculus

Steffen van Bakel

Department of Computing, Imperial College London,
180 Queen's Gate, London SW7 2BZ, UK

s.vanbakel@imperial.ac.uk

Maria Grazia Vigliotti

Adelard LLP Exmouth House,
3-11 Pine Street London EC1R 0JH, UK

mgv@adelard.com

We study the $\lambda\mu$-calculus, extended with explicit substitution, and define a compositional output-based interpretation into a variant of the π-calculus with pairing that preserves single-step explicit head reduction with respect to weak bisimilarity. We define four notions of weak equivalence for $\lambda\mu$ – one based on weak reduction $\sim_{w\beta\mu}$, two modelling weak head-reduction and weak explicit head reduction, $\sim_{w\text{H}}$ and $\sim_{w\text{xH}}$ respectively (all considering terms without weak head-normal form equivalent as well), and one based on weak approximation $\sim_{\mathcal{A}}$ – and show they all coincide. We will then show full abstraction results for our interpretation for the weak equivalences with respect to weak bisimilarity on processes.

Introduction

The research presented in this paper is part of an ongoing investigation into the suitability of classical logic in the context of programming languages with control. Rather than looking at how to encode known control features into calculi like the λ-calculus [9, 7], Parigot's $\lambda\mu$-calculus [21], or $\Lambda\mu$ [13], as has been done in great detail by others, we focus on trying to understand what is exactly the notion of computation that is embedded in calculi like $\lambda\mu$; we approach that problem here by presenting a fully abstract interpretation for that calculus into the (perhaps better understood) π-calculus [20].

In the past, many researchers investigated interpretations into the π-calculus of various calculi that have their foundation in classical logic. From these papers it might seem that the interpretation of such 'classical' calculi comes at a great expense; for example, to encode *typed $\lambda\mu$*, [15] defines an extension of Milner's encoding and considers a strongly typed π-calculus; [3] shows preservation of reduction in $\mathcal{X}$ [4] only with respect to $\sqsubseteq_{\text{c}}$, the contextual ordering (so not with respect to $\sim_{\text{c}}$, contextual equivalence, nor with respect to weak bisimilarity); [10] defines a non-compositional interpretation of $\overline{\lambda}\mu\tilde{\mu}$ [11] that strongly depends on recursion, and does not regard the logical aspect.

In [6] we started our investigations by presenting an interpretation for de Groote's variant $\Lambda\mu$ into the π-calculus [20] and proved a soundness result; here we show that this interpretation is fully abstract, but have to limit the interpretation to $\lambda\mu$ terms. We study an output-based encoding of $\lambda\mu$ into the π-calculus that is an extension of the one we defined for the λ-calculus [5] and is a natural variant of that for $\Lambda\mu$ in [6]. In those papers, we have shown that our encoding respects *single-step explicit head reduction* (which only ever replaces the head variable of a term) modulo $\sim_{\text{c}}$.

We will here address the natural question that arises next: are two terms that are equal under the interpretation also operational equivalent, *i.e.*: is the interpretation *fully abstract*? We answer that question positively, using a new approach to showing full abstraction, for our interpretation of $\lambda\mu$-terms (rather than $\Lambda\mu$ as used in [6]) and thereby also for the standard λ-calculus. Following the approach of [6] we can show that our interpretation respects single-step explicit head reduction $\rightarrow_{\text{xH}}$ modulo *weak bisimularity* $\approx$ (rather than $\sim_{\text{c}}$ as used in [6]; we omit the details here). We extend this result to $\sim_{w\text{xH}}$, the equivalence relation generated by $\rightarrow_{\text{xH}}$ that equates also terms without (weak) normal form with respect

to $\rightarrow_{\mathbf{xH}}$. The main proof of the full abstraction result is then achieved through showing that $\sim_{w\mathbf{xH}}$ equates to $\sim_{w\beta\mu}$, the equivalence relation generated by standard reduction that also equates terms without weak head normal form.

This technique is considerably different from the one used by Sangiorgi, who has shown a full abstraction result [23, 24] for Milner's encoding $[\![M]\!]^{\mathrm{M}}a$ of the lazy λ-calculus [20]. To achieve full abstraction, Sangiorgi proves that $[\![M]\!]^{\mathrm{M}}a \approx [\![N]\!]^{\mathrm{M}}a$ if and only if $M \cong N$, where $\cong$ is the *applicative bisimularity* on λ-terms [2]. However, this result comes at a price, since applicative bisimulation equates terms that are not weakly bisimilar under $[\![\,\cdot\,]\!]^{\mathrm{M}}\cdot$: in order to achieve full abstraction, Sangiorgi had to extend Milner's encoding to Λ_c, a λ-calculus enriched with constants and by exploiting a more abstract encoding into the *Higher Order* π-calculus, a variant of the π-calculus with higher-order communications. Sangiorgi's result then essentially states that the interpretations of closed Λ_c-terms M and N are contextually equivalent if and only if M and N are applicatively bisimilar; in [23] he shows that the interpretation of terms in Λ_c in the standard π-calculus is weakly bisimilar if and only if they have the same Lévy-Longo tree.

We would like to stress that in order to achieve full abstraction for our interpretation, *we did not need to extend the interpreted calculus, and use a first order π-calculus*. In fact, the main contribution of this paper and novelty of our proof is the *structure of the proof* of the fact that our interpretation gives a fully abstract semantics. To wit, we define a choice of operational equivalences for the $\lambda\mu$-calculus, both with and without explicit substitution. We define the *weak explicit head equivalence* $\sim_{w\mathbf{xH}}$ and show that this is exactly the relation that is naturally representable in the π-calculus; we define *weak head equivalence* $\sim_{\mathbf{xH}}$ and show that for $\lambda\mu$-terms without explicit substitution, $\sim_{w\mathbf{xH}}$ corresponds to $\sim_{\mathbf{xH}}$. The relation $\sim_{w\mathbf{xH}}$ essentially equates terms that have the same Lévy-Longo tree, but of course defined for $\lambda\mu$, which gets shown through a notion of weak approximation. We then show that the relation $\sim_{\mathcal{A}_w}$, which expresses that terms have the same set of weak approximants, $\sim_{w\mathbf{xH}}$, and $\sim_{w\beta\mu}$ all correspond.

The combined results of [5, 6] and the full abstraction results we present here stress that the π-calculus constitutes a very powerful abstract machine indeed: although the notion of structural reduction in $\lambda\mu$ is very different from normal β-reduction, no special measures had to be taken in order to be able to express it through our interpretation. In fact, the distributive character of application in $\lambda\mu$, and of both term and context substitution is dealt with by congruence in π, and both naming and μ-binding are dealt with entirely statically by the interpretation.

Organisation of this paper: We start with revisiting the $\lambda\mu$-calculus in Section 1 and define a notion of *head-reduction* $\rightarrow_{\mathrm{H}}$. In Section 2 we revisit the π-calculus, enriched with *pairing*. In Section 3 we define $\lambda\mu\mathbf{x}$, a version of $\lambda\mu$ with *explicit substitution*, as well as a notion of *explicit head reduction* and in Section 4 define our *logical interpretation* of $\lambda\mu\mathbf{x}$ in to π.

Working towards our full abstraction result, in Section 5 we will define notions of weak reduction, in particular *weak head reduction* and *weak explicit head reduction*. We then define the two notions of equivalence these induce, also equating terms without weak head-normal form and show that these notions coincide on pure $\lambda\mu$ terms (*i.e.* without explicit substitutions). We also define the equivalence $\sim_{w\beta\mu}$ induced by $\rightarrow_{\beta\mu}$ on pure $\lambda\mu$ terms, that also equates terms without weak head-normal form. In Section 6, we define a notion of *weak approximation* for $\lambda\mu$, and show the semantics this induces, $\sim_{\mathcal{A}_w}$, is fully abstract with respect to both $\sim_{w\mathrm{H}}$ and $\sim_{w\beta\mu}$. We show that our logical interpretation is fully abstract with respect to weak bisimilarity $\approx$ on processes and $\sim_{w\mathbf{xH}}$, $\sim_{w\mathrm{H}}$, $\sim_{\mathcal{A}_w}$, and $\sim_{w\beta\mu}$ on pure $\lambda\mu$-terms.

Notation: We will use a vector notation $\vec{\cdot}$ as abbreviation for any sequence: for example, $\overline{x_i}$ stands for $x_1, \ldots, x_n$, for some n, or for $\{x_1, \ldots, x_n\}$, and $\langle \overrightarrow{\alpha_i := N_i \cdot \beta_i} \rangle$ for $\langle \alpha_1 := N_1 \cdot \beta_1 \rangle \cdots \langle \alpha_n := N_n \cdot \beta_n \rangle$, etc. When possible, we will drop the indices.

1 The $\lambda\mu$ calculus and explicit substitution

In this section, we will briefly discuss Parigot's $\lambda\mu$-calculus [21]; we assume the reader to be familiar with the λ-calculus and its notion of reduction $\rightarrow_\beta$ and equality $=_\beta$.

$\lambda\mu$ is a proof-term syntax for classical logic, expressed in Natural Deduction, defined as an extension of the Curry type assignment system for the λ-calculus by adding the concept of *named* terms, and adding the functionality of a *context switch*, allowing arguments to be fed to subterms.

Definition 1.1 (Syntax of $\lambda\mu$) *The $\lambda\mu$-terms we consider are defined over the set of* variables *(Roman characters) and* names, *or* context *variables (Greek characters), through:*

$$M, N \quad ::= \quad x \mid \lambda x.M \mid MN \mid \mu\alpha.[\beta]M$$

We will occasionally write C *for the pseudo-term* $[\alpha]M$.

As usual, $\lambda x.M$ binds x in M, and $\mu\alpha.$C binds α in C, and the notions of free variables $fv(M)$ and names $fn(M)$ are defined accordingly; the notion of α-conversion extends naturally to bound names and we assume Barendregt's convention in that we assume that free and bound variables and names are always distinct, using α-conversion when necessary. As usual, $M[N/x]$ stands for the substitution of all occurrences of x in M by N, and $M[N \cdot \gamma/\alpha]$, the *structural substitution*, for the term obtained from M when every (pseudo) sub-term of the form $[\alpha]M'$ is replaced by $[\gamma]M'N$. (We omit the formal definition here; see Def. 3.1 for the variant with *explicit* structural substitution.)

Definition 1.2 ($\lambda\mu$ reduction) *Reduction on $\lambda\mu$-terms is defined as the contextual closure of the rules:*

$$
\begin{array}{rrcl}
\textit{logical } (\beta): & (\lambda x.M)N & \rightarrow & M[N/x] \\
\textit{structural } (\mu): & (\mu\alpha.\mathsf{C})N & \rightarrow & \mu\gamma.(\mathsf{C}[N \cdot \gamma/\alpha]) \\
\textit{renaming}: & \mu\delta.[\beta](\mu\gamma.[\alpha]M) & \rightarrow & \mu\delta.[\alpha]M[\beta/\gamma] \\
\textit{erasing}: & \mu\alpha.[\alpha]M & \rightarrow & M \qquad\qquad (\alpha \notin fn(M))
\end{array}
$$

We use $\rightarrow^*_{\beta\mu}$ *for the pre-congruence based on these rules,* $=_{\beta\mu}$ *for the congruence, write* $M \rightarrow^{nf}_{\beta\mu} N$ *if* $M \rightarrow^*_{\beta\mu} N$ *and N is in normal form,* $M \rightarrow^{hnf}_{\beta\mu} N$ *if $M \rightarrow^*_{\beta\mu} N$ and N is in head-normal form,* $M \Downarrow$ *if there exists a finite reduction path starting from M, and $M \Uparrow$ if this is not the case; we will use these notations for other notions of reduction as well.*

That this notion of reduction is confluent was shown in [22]; so we have:

Proposition 1.3 *If $M =_{\beta\mu} N$ and $M \rightarrow^*_{\beta\mu} P$, then there exists Q such that $P \rightarrow^*_{\beta\mu} Q$ and $N \rightarrow^*_{\beta\mu} Q$.*

Definition 1.4 (Head reduction for $\lambda\mu$ (cf. [19])) *1. We define* head reduction $\rightarrow_{\mathrm{H}}$ *as the restriction of $\rightarrow_{\beta\mu}$ by removing the contextual rule:* $M \rightarrow N \;\Rightarrow\; LM \rightarrow LN$

*2. The $\lambda\mu$ head-normal forms (*HNF*) are defined through the grammar:*

$$
\begin{array}{rcll}
\boldsymbol{H} & ::= & \lambda x.\boldsymbol{H} & \\
& \mid & xM_1\cdots M_n & (n \geq 0) \\
& \mid & \mu\alpha.[\beta]\boldsymbol{H} & (\beta \neq \alpha \text{ or } \alpha \in \boldsymbol{H}, \text{ and } \boldsymbol{H} \neq \mu\gamma.[\delta]\boldsymbol{H}')
\end{array}
$$

The following is straightforward:

Proposition 1.5 ($\rightarrow_{\mathrm{H}}$ implements $\lambda\mu$'s head reduction) *If $M \rightarrow^*_{\beta\mu} N$ with N in* HNF *(so $M \rightarrow^{hnf}_{\beta\mu} N$), then there exists $\boldsymbol{H}$ such that $M \rightarrow^{nf}_{\mathrm{H}} \boldsymbol{H}$ (so $\boldsymbol{H}$ is in $\rightarrow_{\mathrm{H}}$-normal form) and $\boldsymbol{H} \rightarrow^*_{\beta\mu} N$ without using $\rightarrow_{\mathrm{H}}$.*

2   The synchronous π-calculus with pairing

The notion of π-calculus that we consider in this paper was already considered in [5] and is different from other systems studied in the literature [14] in that it adds *pairing* and uses a *let*-construct to deal with inputs of pairs of names that get distributed, similar to that defined in [1]; in contrast to [3, 5], we do not consider the asynchronous version of this calculus.

Definition 2.1 (Processes) Channel names *and* data *are defined by:*

$$a,b,c,d,x,y,z \quad \text{names} \qquad p \quad ::= \quad a \mid \langle a,b \rangle \quad \text{data}$$

Processes are defined by:

$$P,Q \quad ::= \quad 0 \mid P \mid Q \mid !P \mid (\nu a)\,P \mid a(x).P \mid \bar{a}\,p.P \mid \text{let } \langle x,y \rangle = p \text{ in } P$$

We see, as usual, ν as a binder, and call the name n bound in $(\nu n)\,P$, x bound in $a(x).P$ and x,y bound in let $\langle x,y \rangle = p$ in P; we write $bn(P)$ for the set of bound names in P; n is free *in P if it occurs in P but is not bound, and we write $fn(P)$ for the set of free names in P.*

Notice that data occurs only in two cases: $\bar{a}\,p$ and *let* $\langle x,y \rangle = p$ *in* P, and that then p is either a single name, or a pair of names; we therefore do not allow $a(\langle x,y \rangle).P$, nor $\bar{a}\,\langle\langle b,c \rangle,d \rangle.P$, nor $\overline{\langle b,c \rangle}\,p.P$, nor $(\nu\langle a,b \rangle)\,P$, nor *let* $\langle\langle a,b \rangle,y \rangle = p$ *in* P, etc.

We abbreviate $a(x).\text{let } \langle y,z \rangle = x \text{ in } P$ by $a(y,z).P$, $(\nu m)\,(\nu n)\,P$ by $(\nu mn)\,P$, and write $\bar{a}\,p$ for $\bar{a}\,p.0$. As in [24], we write $a{\to}b$ for the *forwarder* $a(x).\bar{b}\,x$.

Definition 2.2 (Structural Congruence) *The* structural congruence *is the smallest congruence generated by the rules:*

$$
\begin{array}{rclcrcll}
P \mid 0 & \equiv & P & \qquad & (P \mid Q) \mid R & \equiv & P \mid (Q \mid R) & \\
P \mid Q & \equiv & Q \mid P & \qquad & (\nu m)\,(\nu n)\,P & \equiv & (\nu n)\,(\nu m)\,P & \\
!P & \equiv & P \mid !P & \qquad & (\nu n)\,(P \mid Q) & \equiv & P \mid (\nu n)\,Q & \quad (n \notin fn(P)) \\
(\nu n)\,0 & \equiv & 0 & \qquad & \text{let } \langle x,y \rangle = \langle a,b \rangle \text{ in } P & \equiv & P[a/x,b/y] &
\end{array}
$$

As usual, we will consider processes modulo congruence and α-conversion: this implies that we will not deal explicitly with the process *let* $\langle x,y \rangle = \langle a,b \rangle$ *in* P, but rather with $P[a/x,b/y]$. Because of rule $(P \mid Q) \mid R \equiv P \mid (Q \mid R)$, we will not write brackets in a parallel composition of more than two processes. Computation in the π-calculus with pairing is expressed via the exchange of *data*.

Definition 2.3 (Reduction) *The* reduction relation *over the processes of the π-calculus is defined by the following (elementary) rules:*

$$
\begin{array}{rcl}
\bar{a}\,p.P \mid a(x).Q & \to_{\pi} & P \mid Q[p/x] \\
P \to_{\pi} P' & \Rightarrow & (\nu n)\,P \to_{\pi} (\nu n)\,P' \\
P \to_{\pi} P' & \Rightarrow & P \mid Q \to_{\pi} P' \mid Q \\
P \equiv Q \; \& \; Q \to_{\pi} Q' \; \& \; Q' \equiv P' & \Rightarrow & P \to_{\pi} P'
\end{array}
$$

Notice that the first rule is only allowed if $Q[p/x]$ is a well-defined process.

There are several notions of equivalence defined for the π-calculus: the one we consider here, and will show is related to our encoding, is that of weak-bisimilarity.

Definition 2.4 (Weak-bisimilarity)   *1. We write $P{\downarrow}\overline{n}$ and say that P outputs on n (or P exhibits an output barb on n) if $P \equiv (\nu\vec{b})\,(\overline{n}\,p.Q \mid R)$, where $n \notin \vec{b}$, and $P{\downarrow}n$ (P inputs on n) if $P \equiv (\nu\vec{b})\,(n(x).Q \mid R)$, where $n \notin \vec{b}$.*

2. *We write* $P \Downarrow \overline{n}$ *(P will output on n) if there exists Q such that* $P \rightarrow_\pi^* Q$ *and* $Q \downarrow \overline{n}$, *and* $P \Downarrow n$ *(P will input on n) if there exists Q such that* $P \rightarrow_\pi^* Q$ *and* $Q \downarrow n$.

3. *A* barbed bisimilarity $\stackrel{.}{\approx}$ *is the largest symmetric relation such that* $P \stackrel{.}{\approx} Q$ *satisfies:*

   - *for every name n: if* $P \downarrow \overline{n}$ *then* $Q \Downarrow \overline{n}$, *and if* $P \downarrow n$ *then* $Q \Downarrow n$;
   - *for all P′, if* $P \rightarrow_\pi^* P'$, *then there exists Q′ such that* $Q \rightarrow_\pi^* Q'$ *and* $P' \stackrel{.}{\approx} Q'$;

4. Weak-bisimilarity *is the largest relation* $\approx$ *defined by:* $P \approx Q$ *if and only if* $\mathsf{C}[P] \stackrel{.}{\approx} \mathsf{C}[Q]$ *for any context* $\mathsf{C}[\cdot]$.

3 $\lambda\mu\mathbf{x}$: $\lambda\mu$ **with explicit substitution**

One of the main achievements of [5] is that it establishes a strong link between reduction in the π-calculus and step-by-step *explicit substitution* [8] for the λ-calculus, by formulating a result with respect to explicit head reduction and the spine interpretation defined there.

In view of this, for the purpose of our interpretation it was natural to study a variant of $\Lambda\mu$ in [6] with explicit substitution as well; since here we work with $\lambda\mu$, we present here $\lambda\mu\mathbf{x}$, as a variant of $\Lambda\mu\mathbf{x}$ as presented in that paper. Explicit substitution treats substitution as a first-class operator, both for the logical and the structural substitution, and describes all the necessary steps to effectuate both.

Definition 3.1 ($\lambda\mu\mathbf{x}$)   *1. The syntax of the* explicit $\lambda\mu$ calculus, $\lambda\mu\mathbf{x}$, *is defined by:*

$$M, N \quad ::= \quad x \mid \lambda x.M \mid MN \mid M\langle x := N\rangle \mid \mu\alpha.[\beta]M \mid M\langle \alpha := N\cdot\gamma\rangle$$

We consider the occurrences of x in M bound in $M\langle x := N\rangle$, *and those of α in M in* $M\langle \alpha := N\cdot\gamma\rangle$; *by Barendregt's convention, x and α do not appear outside M.*

2. *The reduction relation* $\rightarrow_\mathbf{x}$ *on* $\lambda\mu\mathbf{x}$ *is defined as the contextual closure of the following rules:*

   (a) Main reduction rules:
   $$\begin{aligned}
   (\lambda x.M)N &\rightarrow & M\langle x := N\rangle & \\
   (\mu\alpha.\mathsf{C})N &\rightarrow & \mu\gamma.\mathsf{C}\langle \alpha := N\cdot\gamma\rangle & (\gamma \text{ fresh}) \\
   \mu\beta.[\beta]M &\rightarrow & M & (\beta \notin fn(M)) \\
   [\beta]\mu\gamma.\mathsf{C} &\rightarrow & \mathsf{C}[\beta/\gamma] &
   \end{aligned}$$

   (b) Term substitution rules:
   $$\begin{aligned}
   x\langle x := N\rangle &\rightarrow & N & \\
   M\langle x := N\rangle &\rightarrow & M & (x \notin fv(M)) \\
   (\lambda y.M)\langle x := N\rangle &\rightarrow & \lambda y.(M\langle x := N\rangle) & \\
   (PQ)\langle x := N\rangle &\rightarrow & (P\langle x := N\rangle)(Q\langle x := N\rangle) & \\
   (\mu\alpha.[\beta]M)\langle x := N\rangle &\rightarrow & \mu\alpha.[\beta](M\langle x := N\rangle) &
   \end{aligned}$$

   (c) Structural rules:
   $$\begin{aligned}
   (\mu\delta.\mathsf{C})\langle \alpha := N\cdot\gamma\rangle &\rightarrow & \mu\delta.(\mathsf{C}\langle \alpha := N\cdot\gamma\rangle) & \\
   ([\alpha]M)\langle \alpha := N\cdot\gamma\rangle &\rightarrow & [\gamma](M\langle \alpha := N\cdot\gamma\rangle)N & \\
   ([\beta]M)\langle \alpha := N\cdot\gamma\rangle &\rightarrow & [\beta](M\langle \alpha := N\cdot\gamma\rangle) & (\alpha \neq \beta) \\
   M\langle \alpha := N\cdot\gamma\rangle &\rightarrow & M & (\alpha \notin fn(M)) \\
   (\lambda x.M)\langle \alpha := N\cdot\gamma\rangle &\rightarrow & \lambda x.M\langle \alpha := N\cdot\gamma\rangle & \\
   (PQ)\langle \alpha := N\cdot\gamma\rangle &\rightarrow & (P\langle \alpha := N\cdot\gamma\rangle)(Q\langle \alpha := N\cdot\gamma\rangle) &
   \end{aligned}$$

3. *We use* $\rightarrow_{:=}$ *for the notion of reduction where only term substitution and structural rules are used (so not the main reduction rules).*

Notice that since reduction in $\lambda\mu\mathbf{x}$ is formulated via term rewriting rules [16], reduction is allowed to take place also inside the substitution term. The following is straightforward:

Proposition 3.2 ($\lambda\mu\mathbf{x}$ implements $\lambda\mu$-reduction)   *1. $M \to_{\beta\mu} N \Rightarrow M \to_{\mathbf{x}}^* N$.*

   2. $M \in \lambda\mu \ \& \ M \to_{\mathbf{x}} N \Rightarrow \exists L \in \lambda\mu \ [N \to_{:=}^* L]$.

In the context of head reduction and explicit substitution, we can economise further on how substitution is executed, and perform only those that are essential for the continuation of reduction. We will therefore limit substitution to allow it to *only replace* the head variable of a term. (This principle is also found in Krivine's machine [17].) The results of [5] show that this is exactly the kind of reduction that the π-calculus naturally encodes.

Definition 3.3 (Explicit head reduction) *The* head variable *of M, $hv(M)$, is defined as expected, adding $hv(M\langle x := N\rangle) = hv(M)$ if $hv(M) \neq x$, and the* head name *$hn(M)$ is defined by $hn(\mu\alpha.[\beta]\mathbf{H}) = \beta$, $hn(M\langle x := N\rangle) = hn(M)$, and $hn(M\langle\alpha := N\cdot\gamma\rangle) = hn(M)$ if $hn(M) \neq \alpha$.*
   We define explicit head reduction $\to_{\mathbf{xH}}$ *on $\lambda\mu\mathbf{x}$ as $\to_{\mathbf{x}}$, but change and add a few rules (we only give the changes):*

   1. We replace the term substitution rule for application and add side-conditions:

$$
\begin{array}{rcll}
(\lambda y.M)\langle x := N\rangle & \to & \lambda y.(M\langle x := N\rangle) & (x = hv(\lambda y.M)) \\
(PQ)\langle x := N\rangle & \to & (P\langle x := N\rangle\, Q)\langle x := N\rangle & (x = hv(PQ)) \\
(\mu\alpha.[\beta]M)\langle x := N\rangle & \to & \mu\alpha.[\beta](M\langle x := N\rangle) & (x = hv(\mu\alpha.[\beta]M))
\end{array}
$$

   2. There are only two structural rules:

$$
\begin{array}{rcll}
(\mu\beta.[\alpha]M)\langle\alpha := N\cdot\gamma\rangle & \to & \mu\beta.[\gamma](M\langle\alpha := N\cdot\gamma\rangle)N & \\
M\langle\alpha := N\cdot\gamma\rangle & \to & M & (\alpha \notin fn(M))
\end{array}
$$

   3. We remove the following contextual rules:

$$
M \to N \quad \Rightarrow \quad
\left\{
\begin{array}{rcl}
LM & \to & LN \\
L\langle x := M\rangle & \to & L\langle x := N\rangle \\
L\langle\alpha := M\cdot\gamma\rangle & \to & L\langle\alpha := N\cdot\gamma\rangle
\end{array}
\right.
$$

   4. We add four substitution rules:

$$
\begin{array}{rcll}
M\langle x := N\rangle\langle y := L\rangle & \to & M\langle y := L\rangle\langle x := N\rangle\langle y := L\rangle & (y = hv(M)) \\
M\langle\alpha := N\cdot\beta\rangle\langle y := L\rangle & \to & M\langle y := L\rangle\langle\alpha := N\cdot\beta\rangle\langle y := L\rangle & (y = hv(M)) \\
M\langle\alpha := N\cdot\gamma\rangle\langle\beta := L\cdot\delta\rangle & \to & M\langle\beta := L\cdot\delta\rangle\langle\alpha := N\cdot\gamma\rangle\langle\beta := L\cdot\delta\rangle & (\beta = hn(M)) \\
M\langle x := N\rangle\langle\beta := L\cdot\delta\rangle & \to & M\langle\beta := L\cdot\delta\rangle\langle x := N\rangle\langle\beta := L\cdot\delta\rangle & (\beta = hn(M))
\end{array}
$$

Notice that, for example, in case 1, the clause postpones the substitution $\langle x := N\rangle$ on Q until such time that an occurrence of the variable x in Q becomes the head-variable of the full term, and that we no longer allow reduction inside the substitution or inside the right-hand side of an application.
   The following proposition states the relation between explicit head reduction, head reduction, and explicit reduction.

Proposition 3.4   *1. If $M \to_{\mathrm{H}}^* N$, then there exists $L \in \lambda\mu\mathbf{x}$ such that $M \to_{\mathbf{xH}}^* L$ and $L \to_{:=}^* N$.*

   2. If $M \to_{\mathbf{xH}}^{nf} N$ with $M \in \lambda\mu$, then there exists $L \in \lambda\mu$ such that $N \to_{:=}^{nf} L$, and $M \to_{\mathrm{H}}^{nf} L$.

   3. $M \to_{\beta\mu}^{nf} N$ if and only if there exists $L \in \lambda\mu\mathbf{x}$ such that $M \to_{\mathbf{xH}}^{nf} L$ and $L \to_{\mathbf{x}}^{nf} N$.

This result gives that we can show our main results for $\lambda\mu\mathbf{x}$ for reductions that reduce to HNF.

4 A logical interpretation of $\lambda\mu$x-terms to π-processes

We will now define our logical,[1] output-based interpretation $\llbracket M \rrbracket a$ of the $\lambda\mu$x-calculus into the π-calculus (where M is a $\lambda\mu$-term, and a is the name given to its (anonymous) output), which is essentially the one presented in [6], but no longer considers $[\alpha]M$ to be a term. The reason for this change is the following: using the interpretation of [6],

$$\llbracket \mu\alpha.\lambda x.x \rrbracket a \;=\; (\nu s)\,((\nu xb)\,(x(u).!u{\rightarrow}b \mid \bar{s}\langle x,b\rangle))$$

is in normal form, and all inputs and outputs are restricted; thereby, it is weakly bisimilar to 0 and to $\llbracket(\lambda x.xx)(\lambda x.xx)\rrbracket a$. So using that interpretation, we cannot distinguish between *blocked* and *looping* computations, which clearly affects any full-abstraction result. When restricting our interpretation to $\lambda\mu$, this problem disappears: since naming has to follow μ-abstraction, $\mu\alpha.\lambda x.x$ is not a term in $\lambda\mu$. Since $\lambda\mu$ is a subcalculus of $\Lambda\mu$, this change clearly does not affect the results shown in [6] that all hold for the interpretation we consider here as well.

The main idea behind the interpretation, as in [5], is to give a name to the anonymous output of terms; it combines this with the inherent naming mechanism of $\lambda\mu$. As shown in [6], this encoding naturally represents explicit head reduction; we will need to consider weak reduction later for the full abstraction result, but not for soundness, completeness, or termination.

Definition 4.1 (Logical interpretation [6]) *The interpretation of $\lambda\mu$x terms into the π-calculus is defined by:*

$$
\begin{array}{rcll}
\llbracket x \rrbracket a & \triangleq & x(u).!u{\rightarrow}a & (u\;\textit{fresh}) \\
\llbracket \lambda x.M \rrbracket a & \triangleq & (\nu xb)\,(\llbracket M \rrbracket b \mid \bar{a}\langle x,b\rangle) & (b\;\textit{fresh}) \\
\llbracket MN \rrbracket a & \triangleq & (\nu c)\,(\llbracket M \rrbracket c \mid !c(v,d).(\llbracket v{:=}N \rrbracket \mid !d{\rightarrow}a)) & (c,v,d\;\textit{fresh}) \\
\llbracket M\langle x{:=}N\rangle \rrbracket a & \triangleq & (\nu x)\,(\llbracket M \rrbracket a \mid \llbracket x{:=}N \rrbracket) & \\
\llbracket x{:=}N \rrbracket & \triangleq & !\bar{x}(w).\llbracket N \rrbracket w & (w\;\textit{fresh}) \\
\llbracket \mu\gamma.\mathsf{C} \rrbracket a & \triangleq & (\nu s)\,\llbracket \mathsf{C} \rrbracket s[a/\gamma] & (s\;\textit{fresh}) \\
\llbracket [\beta]M \rrbracket a & \triangleq & \llbracket M \rrbracket \beta & \\
\llbracket M\langle \beta{:=}N{\cdot}\gamma\rangle \rrbracket a & \triangleq & (\nu\beta)\,(\llbracket M \rrbracket a \mid \llbracket \beta{:=}N{\cdot}\gamma \rrbracket) & \\
\llbracket \alpha{:=}M{\cdot}\gamma \rrbracket & \triangleq & !\alpha(v,d).(\llbracket v{:=}N \rrbracket \mid !d{\rightarrow}\gamma) & (v,d\;\textit{fresh})
\end{array}
$$

Notice that $\llbracket \mu\gamma.[\beta]M \rrbracket a \;\triangleq\; (\nu s)\,\llbracket [\beta]M \rrbracket s[a/\gamma] \;\triangleq\; (\nu s)\,\llbracket M \rrbracket \beta[a/\gamma] \;\equiv\; \llbracket M \rrbracket \beta[a/\gamma]$ which implies that we can add $\llbracket \mu\gamma.[\beta]M \rrbracket a \triangleq \llbracket M \rrbracket \beta[a/\gamma]$ to our encoding.

Observe the similarity between

$$
\begin{array}{rcll}
\llbracket MN \rrbracket a & \triangleq & (\nu c)\,(\llbracket M \rrbracket c \mid !c(v,d).(\llbracket v{:=}N \rrbracket \mid !d{\rightarrow}a)) & \text{and} \\
\llbracket M\langle c{:=}N{\cdot}\gamma\rangle \rrbracket a & \triangleq & (\nu c)\,(\llbracket M \rrbracket a \mid \llbracket c{:=}N{\cdot}\gamma \rrbracket) & \\
& \triangleq & (\nu c)\,(\llbracket M \rrbracket a \mid !c(v,d).(\llbracket v{:=}N \rrbracket \mid !d{\rightarrow}\gamma)) &
\end{array}
$$

The first communicates N via the output channel c of M (which might occur more than once inside $\llbracket M \rrbracket c$, so replication is needed), whereas the second communicates with all the sub-terms that have c as output name, and changes the output name of the process to γ. In other words, application is just a special case of explicit structural substitution; this allows us to write $(\nu c)\,(\llbracket M \rrbracket c \mid \llbracket c{:=}N{\cdot}a \rrbracket)$ for $\llbracket MN \rrbracket a$. This stresses that the π-calculus constitutes a very powerful abstract machine indeed: although the notion of structural reduction in $\lambda\mu$ is very different from normal β-reduction, no special measures had to be taken in order to be able to express it; the component of our interpretation that deals with pure λ-terms

---

[1] It is called *logical* because it has its foundation in the relation between natural deduction and Gentzen's sequent calculus.

is almost exactly that of [5] (ignoring for the moment that substitution is modelled using a guard, which affects also the interpretation of variables), but for the use of replication in the case for application.

We can now show a reduction-preservation result for explicit head reduction for $\lambda\mu\mathbf{x}$, by showing that $\llbracket\cdot\rrbracket\cdot$ preserves $\rightarrow_{\mathbf{xH}}$ up to weak bisimularity, stated using $\sim_{\mathrm{C}}$ in [6].

Theorem 4.2 (Operational Soundness [6])   *1. $M \rightarrow^*_{\mathbf{xH}} N \Rightarrow \llbracket M\rrbracket\, a \approx \llbracket N\rrbracket\, a$.*

  2. If $M \Uparrow_{\mathbf{xH}}$ then $\llbracket M\rrbracket\, a \Uparrow$.

The proof in [6] shows that β-reduction is implemented in π by at least one synchronisation.

We can also show that equality with explicit substitution, $=_{\mathbf{x}}$, is preserved under our encoding by weak bisimulation.

Theorem 4.3 *If $M =_{\mathbf{x}} N$, then $\llbracket M\rrbracket\, a \approx \llbracket N\rrbracket\, a$.*

Proof: By induction on the definition of $=_{\mathbf{x}}$. $\square$

Now the following is an immediate consequence:

Theorem 4.4 (Semantics) *If $M =_{\beta\mu} N$, then $\llbracket M\rrbracket\, a \approx \llbracket N\rrbracket\, a$.*

Proof: By induction on the definition of $=_{\beta\mu}$. The case $M \rightarrow^*_{\beta\mu} N$ follows from the fact that then, by Proposition 3.2, also $M \rightarrow^*_{\mathbf{x}} N$, so by Theorem 4.3, we have $\llbracket M\rrbracket\, a \approx \llbracket N\rrbracket\, a$. The steps to an equivalence relation follow directly from $\approx$. $\square$

Notice that it is clear that we cannot prove the exact reversal of this result, since terms without head-normal form are all interpreted by 0 (see also Lem. 5.6), but are not all related through $=_{\beta\mu}$. Using weak equivalence, we can deal with the reverse part, and will do so in the last sections of this paper.

5   Weak equivalences for $\lambda\mu$ and $\lambda\mu\mathbf{x}$

Since $\Delta\Delta$ and $\Omega\Omega$ (where $\Delta = \lambda x.xx$ and $\Omega = \lambda y.yyy$) are closed terms that do not interact with any context, they are contextually equivalent; any well-defined interpretation of these terms into the π-calculus, be it input based or output based, will therefore map those to processes that are weakly bisimilar to 0, and therefore to weakly bisimilar processes. Abstraction, on the other hand, enables interaction with a context, and therefore the interpretation of $\lambda z.\Delta\Delta$ will *not* be weakly bisimilar to 0. We therefore cannot hope to model standard $\beta\mu$-equality in the π-calculus in a fully-abstract way; rather, we need to consider a notion of reduction that considers *all* abstractions meaningful; therefore, the only kind of reduction on λ-calculi that can naturally be encoded into the π-calculus is *weak* reduction.

Definition 5.1 *We define the notion $\rightarrow_{w\beta\mu}$ of* weak $\beta\mu$-reduction *as in Def. 1.2, the notion $\rightarrow_{w\mathrm{H}}$ of* weak head reduction[2] *on $\lambda\mu$ as in Def. 1.4, and the notion $\rightarrow_{w\mathbf{xH}}$ of* weak explicit head reduction *on $\lambda\mu\mathbf{x}$ as in Def. 3.3, by (also) eliminating the rules:*

$$
\begin{aligned}
(\lambda y.M)\langle x := N\rangle &\rightarrow \lambda y.(M\langle x := N\rangle) \\
(\lambda x.M)\langle \alpha := N\cdot\gamma\rangle &\rightarrow \lambda x.(M\langle \alpha := N\cdot\gamma\rangle) \\
M \rightarrow N &\Rightarrow \lambda x.M \rightarrow \lambda x.N
\end{aligned}
$$

---

[2]This notion is also known as *lazy* reduction; for the sake of keeping our terminology consistent, we prefer to call it weak head reduction.

We define the notion of weak head-normal forms, the normal forms with respect to weak head-reduction:

Definition 5.2 (Weak head-normal forms for $\lambda\mu$)   *1. The $\lambda\mu$ weak head-normal forms (*WHNF*) are defined through the grammar:*

$$
\begin{aligned}
\boldsymbol{H}_w \quad ::= \quad & \lambda x.M \\
| \quad & xM_1\cdots M_n \quad (n \geq 0) \\
| \quad & \mu\alpha.[\beta]\boldsymbol{H}_w \quad (\alpha \neq \beta \text{ or } \alpha \in fn(\boldsymbol{H}_w), \boldsymbol{H}_w \neq \mu\gamma.[\delta]\boldsymbol{H}'_w)
\end{aligned}
$$

2. We say that M has a WHNF *if there exists $\boldsymbol{H}_w$ such that $M \to^*_{w\mathrm{H}} \boldsymbol{H}_w$.*

The main difference between HNFs and WHNFs is in the case of abstraction: where the definition of HNF only allows for the abstraction over a HNF, for WHNFs any term can be the body. Moreover, notice that both $\lambda z.\Delta\Delta$ and $\lambda z.\Omega\Omega$ are in WHNF.

Since $\to_{w\mathbf{x}\mathrm{H}} \subseteq \to_{\mathbf{x}\mathrm{H}}$, we can show the equivalent of Lem 1.5 and Thm. 4.2 also for *weak explicit head reduction*:

Theorem 5.3 (cf. [6])   *1. If $M \to^*_{w\mathbf{x}\mathrm{H}} N$, then $\llbracket M\rrbracket a \approx \llbracket N\rrbracket a$.*

2. If $M \to^*_{\beta\mu} N$ with N in WHNF, *then there exists $\boldsymbol{H}_{w\mathbf{x}}$ such that $M \to^{nf}_{w\mathbf{x}\mathrm{H}} \boldsymbol{H}_{w\mathbf{x}}$ and $\boldsymbol{H}_{w\mathbf{x}} \to^*_{\mathbf{x}} N$ without using $\to_{w\mathbf{x}\mathrm{H}}$.*

We also define weak explicit head-normal forms.

Definition 5.4 (Weak explicit head-normal forms)   *1. The $\lambda\mu\mathbf{x}$ weak explicit head-normal forms (*WEHNF*) are defined through:*

$$
\begin{aligned}
\boldsymbol{H}_{w\mathbf{x}} \quad ::= \quad & \lambda x.M \overrightarrow{\langle y := N\rangle}\, \overrightarrow{\langle \sigma := Q\cdot\tau\rangle} \\
| \quad & xM_1\cdots M_n \overrightarrow{\langle y := N\rangle}\, \overrightarrow{\langle \sigma := Q\cdot\tau\rangle} \qquad (n \geq 0,\ x \notin \vec{y}) \\
| \quad & \mu\alpha.[\beta]\boldsymbol{H}_{w\mathbf{x}} \overrightarrow{\langle y := N\rangle}\, \overrightarrow{\langle \sigma := Q\cdot\tau\rangle} \\
& (\beta \notin \vec{\sigma},\ \alpha \neq \beta \text{ or } \alpha \in fn(\boldsymbol{H}_{w\mathbf{x}}),\ and\ \boldsymbol{H}_{w\mathbf{x}} \neq \mu\gamma.[\delta]\boldsymbol{H}'_{w\mathbf{x}})
\end{aligned}
$$

2. We say that $M \in \lambda\mu\mathbf{x}$ has an WEHNF *if there exists $\boldsymbol{H}_{w\mathbf{x}}$ such that $M \to^*_{w\mathbf{x}\mathrm{H}} \boldsymbol{H}_{w\mathbf{x}}$.*

Remark 5.5 *In the context of reduction (normal and weak), when starting from pure terms, the substitution operation can be left inside terms in normal form, as in*

$$(\lambda x.yM)NL \quad \to_{\mathbf{x}\mathrm{H}} \quad yM\langle x := N\rangle\, L.$$

However, since, by Barendregt's convention, x does not appear free in L, the latter term is operationally equivalent to $yML\langle x := N\rangle$; in fact, these two are equivalent under $\sim_{w\mathrm{H}}$ (see Def. 5.10), and also congruent when interpreted as processes. Since in weak reduction the reduction $(\lambda x.M)\langle y := N\rangle$ for $\lambda x.(M\langle y := N\rangle)$ is not allowed, also this substitution can be considered to stay at the outside. Therefore, without loss of generality, for readability and ease of definition we will use a notation for terms that places all explicit substitutions on the outside.[3] So actual terms can have substitutions inside, but they are written as if they appear outside. To ease notation, we will use $\boldsymbol{S}$ for a set of substitutions of the shape $\langle x := N\rangle$ or $\langle \alpha := N\cdot\gamma\rangle$ when the exact contents of the substitutions is not relevant; we write $x \in \boldsymbol{S}$ if $\langle x := N\rangle \in \boldsymbol{S}$ and similarly for $\alpha \in \boldsymbol{S}$.

---

[3]This is exactly the approach of Krivine's machine, where explicit substitutions are called *closures* that form an environment in which a term is evaluated.

We can show that the interpretation of a term without WHNF is weakly bisimilar to *0*.

Lemma 5.6 *If M has no* WEHNF *(so M also has no* WHNF*), then $\llbracket M \rrbracket a \approx 0$.*

Proof: If M has no WEHNF, then M has no leading abstractions and all terms generated by reduction have a weak explicit head redex. If $M = \mu\alpha.[\beta]N$, then $\llbracket M \rrbracket a \triangleq \llbracket N \rrbracket \beta[a/\alpha] \approx 0$, so also $\llbracket N \rrbracket \beta \approx 0$; therefore we can assume M itself does not start with a context switch.

We reason by coinduction on the explicit weak head reduction sequence from M and analyse the cases of weak explicit head reduction. For example,

$$\llbracket (\lambda x.P_1)P_2\cdots P_n \langle \overrightarrow{y:=Q} \rangle \langle \overrightarrow{\alpha:=R\cdot\beta} \rangle \rrbracket a \triangleq$$
$$(\nu\vec{c})\left((\nu xb)\left(\llbracket P_1 \rrbracket b \mid \overline{c_1}\langle x,b\rangle\right) \mid \overrightarrow{\llbracket c_{i-1}:=P_i\cdot c_i \rrbracket} \mid \llbracket \overrightarrow{y:=Q} \rrbracket \mid \llbracket \overrightarrow{\alpha:=R\cdot\beta} \rrbracket\right)$$

where $c_{n-1} = a$. Since a synchronisation over c_1 is possible, the process is not in normal form. Observe that all outputs are over bound names or under guard, and since the result of the reduction has no head variable, no input is exposed. So $\llbracket M \rrbracket a \approx 0$.   □

We can show the following property.

Lemma 5.7 *Let M and N be pure $\lambda\mu$-terms; then $M \rightarrow^{nf}_{w\mathrm{H}} N$ if and only if there exists N', S such that $M \rightarrow^{nf}_{w\mathbf{x}\mathrm{H}} N'\, S$, and $N'\, S \rightarrow^{nf}_{\stackrel{.}{:=}} N$.*

We will now define equivalences $\sim_{w\beta\mu}$ and $\sim_{w\mathrm{H}}$ between terms of $\lambda\mu$, and $\sim_{w\mathbf{x}\mathrm{H}}$ between terms of $\lambda\mu\mathbf{x}$ (the last two are defined coinductively as bisimulations), that are based on weak reduction, and show that the last two equate the same pure $\lambda\mu$-terms. These notions all consider terms without WHNF equivalent. This is also the case for the approximation semantics we present in the next section.

First we define a weak equivalence generated by the reduction relation $\rightarrow_{w\beta\mu}$.

Definition 5.8 *We define $\sim_{w\beta\mu}$ as the smallest congruence that contains:*

$$
\begin{array}{rcll}
M, N \text{ have no } \textsc{whnf} & \Rightarrow & M \sim_{w\beta\mu} N & \\
(\lambda x.M)N & \sim_{w\beta\mu} & M[N/x] & \\
(\mu\alpha.\mathsf{C})N & \sim_{w\beta\mu} & \mu\gamma.\mathsf{C}[N\cdot\gamma/\alpha] & (\gamma \textit{ fresh}) \\
\mu\alpha.[\beta]\mu\gamma.[\delta]M & \sim_{w\beta\mu} & \mu\alpha.([\delta]M[\beta/\gamma]) & \\
\mu\alpha.[\alpha]M & \sim_{w\beta\mu} & M & (\alpha \notin M)
\end{array}
$$

Since reduction is confluent, the following is immediate.

Proposition 5.9 *If $M \sim_{w\beta\mu} N$ and $M \rightarrow^*_{w\beta\mu} H_w$, then there exists H'_w such that $H_w \sim_{w\beta\mu} H'_w$ and $N \rightarrow^*_{w\beta\mu} H'_w$.*

The other two equivalences we consider are generated by *weak head reduction* and *weak explicit head reduction*. We will show in Theorem 5.13 that these coincide for pure, substitution-free terms.

Definition 5.10 (Weak head equivalence) *The relation $\sim_{w\mathrm{H}}$ is defined co-inductively as the largest symmetric relation such that: $M \sim_{w\mathrm{H}} N$ if and only if either M and N have both no* WHNF*, or both $M \rightarrow^{nf}_{w\mathrm{H}} M'$ and $N \rightarrow^{nf}_{w\mathrm{H}} N'$, and either:*

- *if $M' = xM_1\cdots M_n$ $(n \geq 0)$, then $N' = xN_1\cdots N_n$ and $M_i \sim_{w\mathrm{H}} N_i$ for all $1 \leq i \leq n$; or*

- *if $M' = \lambda x.M''$, then $N' = \lambda x.N''$ and $M'' \sim_{w\mathrm{H}} N''$; or*

- *if $M' = \mu\alpha.[\beta]M''$, then $N' = \mu\alpha.[\beta]N''$ (so $\alpha \neq \beta$ or $\alpha \in fn(M'')$, $M'' \neq \mu\gamma.[\delta]R$, and similarly for N''), and $M'' \sim_{w\mathrm{H}} N''$.*

Notice that $\lambda z.\Delta\Delta \sim_{w\mathrm{H}} \lambda z.\Omega\Omega$ because $\Delta\Delta \sim_{w\mathrm{H}} \Omega\Omega$, since neither has a WHNF.

We will now define a notion of weak explicit head equivalence, that, in approach, corresponds the weak head equivalence but for the fact that now explicit substitutions are part of terms.

Definition 5.11 (Weak explicit head-equivalence) *The relation $\sim_{w\mathbf{x}\mathrm{H}}$ is defined co-inductively as the largest symmetric relation such that: $M \sim_{w\mathbf{x}\mathrm{H}} N$ if and only if either M and N have both no $\rightarrow_{w\mathbf{x}\mathrm{H}}$-normal form, or both $M \rightarrow^{nf}_{w\mathbf{x}\mathrm{H}} M'\,\mathbf{S}$ and $N \rightarrow^{nf}_{w\mathbf{x}\mathrm{H}} N'\,\mathbf{S}'$, and either:*

- *if $M' = xM_1\cdots M_n$ $(n \geq 0)$, then $N' = xN_1\cdots N_n$ (so $x \notin \mathbf{S}$, $x \notin \mathbf{S}'$) and $M_i\,\mathbf{S} \sim_{w\mathbf{x}\mathrm{H}} N_i\,\mathbf{S}'$ for all $1 \leq i \leq n$; or*

- *if $M' = \lambda x.M''$, then $N' = \lambda x.N''$ and $M''\,\mathbf{S} \sim_{w\mathbf{x}\mathrm{H}} N''\,\mathbf{S}'$; or*

- *if $M' = \mu\alpha.[\beta]M''$, then $N' = \mu\alpha.[\beta]N''$ (so $\alpha \neq \beta$ or $\alpha \in fn(M'')$, $M'' \neq \mu\gamma.[\delta]R$, so $\beta \notin \mathbf{S}$, $\beta \notin \mathbf{S}'$, and similarly for N'') and $M''\,\mathbf{S} \sim_{w\mathbf{x}\mathrm{H}} N''\,\mathbf{S}'$.*

Notice that $\mu\alpha.[\beta]\Delta\Delta \sim_{w\mathbf{x}\mathrm{H}} \Delta\Delta$.

The following results formulate the strong relation between $\sim_{w\mathrm{H}}$ and $\sim_{w\mathbf{x}\mathrm{H}}$, and therefore between $\rightarrow_{w\mathrm{H}}$ and $\rightarrow_{w\mathbf{x}\mathrm{H}}$. We first show that pure terms that are equivalent under $\sim_{w\mathbf{x}\mathrm{H}}$ are also so under $\sim_{w\mathrm{H}}$.

Lemma 5.12 *Let M and N be pure $\lambda\mu$-terms; then $M \sim_{w\mathrm{H}} N$ if and only if there are M', N' such that $M' \rightarrow^{nf}_{:=} M$ and $N' \rightarrow^{nf}_{:=} N$, and $M' \sim_{w\mathbf{x}\mathrm{H}} N'$.*

Proof: only if : By co-induction on the definition of $\sim_{w\mathrm{H}}$. If $M \sim_{w\mathrm{H}} N$, then either:

- *$M \rightarrow^{nf}_{w\mathrm{H}} xM_1\cdots M_n$ and $N \rightarrow^{nf}_{w\mathrm{H}} xN_1\cdots N_n$ and $M_i \sim_{w\mathrm{H}} N_i$, for all $1 \leq i \leq n$. Then, by Lem. 5.7, there are $\overrightarrow{M'_i}$ such that*

$$
\begin{array}{llll}
M & \rightarrow^{nf}_{w\mathbf{x}\mathrm{H}} & xM'_1\cdots M'_n\,\mathbf{S} & \rightarrow^{*}_{:=} & xM_1\cdots M_n \\
N & \rightarrow^{nf}_{w\mathbf{x}\mathrm{H}} & xN'_1\cdots N'_n\,\mathbf{S}' & \rightarrow^{*}_{:=} & xN_1\cdots N_n
\end{array}
$$

  But then $M'_i\,\mathbf{S} \rightarrow^{nf}_{:=} M_i$ and $N'_i\,\mathbf{S}' \rightarrow^{nf}_{:=} N_i$, for all $1 \leq i \leq n$; then by induction, $M'_i\,\mathbf{S} \sim_{w\mathbf{x}\mathrm{H}} N'_i\,\mathbf{S}'$ for all $1 \leq i \leq n$. But then $M \sim_{w\mathbf{x}\mathrm{H}} N$.

  The other cases are similar.

if : By co-induction on the definition of $\sim_{w\mathbf{x}\mathrm{H}}$. If there are M', N' such that $M' \rightarrow^{nf}_{:=} M$ and $N' \rightarrow^{nf}_{:=} N$, and $M' \sim_{w\mathbf{x}\mathrm{H}} N'$, then either:

- *$M' \rightarrow^{nf}_{w\mathbf{x}\mathrm{H}} xM'_1\cdots M'_n\,\mathbf{S}$, $N' \rightarrow^{nf}_{w\mathbf{x}\mathrm{H}} xN'_1\cdots N'_n\,\mathbf{S}'$ and $\overrightarrow{M'_i\,\mathbf{S} \sim_{w\mathbf{x}\mathrm{H}} N'_i\,\mathbf{S}'}$. Let, for all $1 \leq i \leq n$, $M'_i\,\mathbf{S} \rightarrow^{nf}_{:=} M_i$ and $N'_i\,\mathbf{S} \rightarrow^{nf}_{:=} N_i$ then by induction, $\overline{M_i \sim_{w\mathrm{H}} N_i}$. Notice that we have $M' \rightarrow^{nf}_{w\mathbf{x}\mathrm{H}} xM'_1\cdots M'_n\,\mathbf{S} \rightarrow^{nf}_{:=} xM_1\cdots M_n$. Let $M' = M''\,\mathbf{S}''$, so $M''\,\mathbf{S}'' \rightarrow^{nf}_{w\mathbf{x}\mathrm{H}} xM'_1\cdots M'_n\,\mathbf{S}'\,\mathbf{S}''$, where $\mathbf{S} = \mathbf{S}'\,\mathbf{S}''$. Let $M''\,\mathbf{S}'' \rightarrow^{nf}_{:=} M$, then by Lem. 5.7, we also have $M \rightarrow^{nf}_{w\mathbf{x}\mathrm{H}} xM''_1\cdots M''_n\,\mathbf{S}' \rightarrow^{nf}_{w\mathrm{H}} xM_1\cdots M_n$. Then, again by Lem. 5.7, $M \rightarrow^{nf}_{w\mathrm{H}} xM_1\cdots M_n$; likewise, we have $N \rightarrow^{nf}_{w\mathrm{H}} xN_1\cdots N_n$. But then $M \sim_{w\mathrm{H}} N$.*

  The other cases are similar. □

Notice that this lemma in fact shows:

Theorem 5.13 *Let $M, N \in \lambda\mu$, then $M \sim_{w\mathbf{x}\mathrm{H}} N \iff M \sim_{w\mathrm{H}} N$.*

6   Full abstraction for the logical interpretation

In this section we will show our main result, that the logical encoding is fully abstract with respect to weak equivalence between pure $\lambda\mu$-terms. To achieve this, we show in Thm. 6.9 that $\llbracket M \rrbracket a \approx \llbracket N \rrbracket a$ *iff* $M \sim_{w\mathbf{x}\mathrm{H}} N$. We are thus left with the obligation to show that $M \sim_{w\mathbf{x}\mathrm{H}} N$ *iff* $M \sim_{w\beta\mu} N$. In Thm. 5.13 we have shown that $M \sim_{w\mathbf{x}\mathrm{H}} N$ *iff* $M \sim_{w\mathrm{H}} N$, for pure terms; to achieve $M \sim_{w\mathrm{H}} N$ *iff* $M \sim_{w\beta\mu} N$, we go through a notion of *weak approximation*; based on Wadsworth's approach [26], we define $\sim_{\mathcal{A}_w}$ that expresses that terms have the same weak approximants and show that $M \sim_{w\mathrm{H}} N$ *iff* $M \sim_{\mathcal{A}_w} N$ *iff* $M \sim_{w\beta\mu} N$.

We can show that if the interpretation of M produces an output, then M reduces by head reduction to an abstraction; similarly, if the interpretation of M produces an input, then M reduces by head reduction to a term with a head variable.

Lemma 6.1   *1. If $\llbracket M \rrbracket a \Downarrow \overline{a}$, then there exist x, N and S such that $\llbracket M \rrbracket a \approx \llbracket \lambda x.N\,S \rrbracket a$, and $M \to^{nf}_{w\mathbf{x}\mathrm{H}} \lambda x.N\,S$.*

2. If $\llbracket M \rrbracket a \Downarrow \overline{c}$, with $a \neq c$, then there exist α, c, x, N and S such that $\llbracket M \rrbracket a \approx \llbracket \mu\alpha.[c]\lambda x.N\,S \rrbracket a$, and $M \to^{nf}_{w\mathbf{x}\mathrm{H}} \mu\alpha.[c]\lambda x.N\,S$.

3. If $\llbracket M \rrbracket a \Downarrow x$, then there exist $\vec{z_j}, x, \vec{N_i}, c$ and S with $x \notin \vec{z_j}$, $m \geq 0$, and $n \geq 0$ such that

- *$\llbracket M \rrbracket a \approx \llbracket \lambda z_1 \cdots z_m.x N_1 \cdots N_n\,S \rrbracket c$;*
- *$M \to^{nf}_{w\mathbf{x}\mathrm{H}} \lambda z_1 \cdots z_m.x N_1 \cdots N_n\,S$ if $a = c$;*
- *$M \to^{nf}_{w\mathbf{x}\mathrm{H}} \mu\alpha.[c]\lambda z_1 \cdots z_m.x N_1 \cdots N_n[a/\alpha]\,S$, if $a \neq c$.*

Proof: Straightforward. $\square$

As to the reverse, we can show:

Lemma 6.2   *1. If $M \to^{nf}_{w\mathbf{x}\mathrm{H}} \lambda x.N\,S$, then $\llbracket M \rrbracket a \Downarrow \overline{a}$.*

2. If $M \to^{nf}_{w\mathbf{x}\mathrm{H}} \mu\alpha.[\beta]\lambda x.N\,S$, then $\llbracket M \rrbracket a \Downarrow \overline{\beta}$.

3. $\llbracket M \rrbracket a \Downarrow x$ if $M \to^{nf}_{w\mathbf{x}\mathrm{H}} x N_1 \cdots N_n\,S$ or $M \to^{nf}_{w\mathbf{x}\mathrm{H}} \mu\alpha.[\beta]x N_1 \cdots N_n\,S$.

Proof: Straightforward. $\square$

Essentially following [26], we now define a *weak approximation semantics* for $\lambda\mu$. Approximation for $\lambda\mu$ has been studied by others as well [25, 12]; however, seen that we are mainly interested in *weak* reduction here, we will define *weak* approximants, which are normally not considered.

Definition 6.3 (Weak approximation for $\lambda\mu$)   *1. The set of $\lambda\mu$'s weak approximants $\mathcal{A}_w$ is defined through the grammar:*

$$
\begin{array}{rcll}
\boldsymbol{A}_w & ::= & \bot \mid \lambda x.\boldsymbol{A}_w \mid x\boldsymbol{A}_w^1 \cdots \boldsymbol{A}_w^n & (n \geq 0) \\
& \mid & \mu\alpha.[\beta]\boldsymbol{A}_w & (\alpha \neq \beta \text{ or } \alpha \in \boldsymbol{A}_w,\ \boldsymbol{A}_w \neq \mu\gamma.[\delta]\boldsymbol{A}_w',\ \boldsymbol{A}_w \neq \bot)
\end{array}
$$

2. The relation $\sqsubseteq\ \subseteq \mathcal{A}_w \times \lambda\mu$ is the smallest preorder that is the compatible extension of $\bot \sqsubseteq M$.

3. $\mathcal{A}_w(M) \triangleq \{\boldsymbol{A}_w \in \mathcal{A}_w \mid \exists N \in \lambda\mu\ [M \to^*_{\beta\mu} N\ \&\ \boldsymbol{A}_w \sqsubseteq N]\}$.

4. Weak approximation equivalence is defined through: $M \sim_{\mathcal{A}_w} N \triangleq \mathcal{A}_w(M) = \mathcal{A}_w(N)$.

Notice that, in part 3, the approximants are weak, not the reduction.

The relationship between the approximation relation and reduction is characterised by:

Lemma 6.4   *1. If $\boldsymbol{A}_w \sqsubseteq M$ and $M \to^*_{\beta\mu} N$, then $\boldsymbol{A}_w \sqsubseteq N$.*

2. *If $A_w \in \mathcal{A}_w(N)$ and $M \rightarrow^*_{\beta\mu} N$, then also $A_w \in \mathcal{A}_w(M)$.*

3. *If $A_w \in \mathcal{A}_w(M)$ and $M \rightarrow_{\beta\mu} N$, then there exists L such that $N \rightarrow^*_{\beta\mu} L$ and $A_w \sqsubseteq L$.*

4. *M is a WHNF if and only if there exists $A_w \neq \bot$ such that $A_w \sqsubseteq M$.*

As is standard in other settings, interpreting a $\lambda\mu$-term M through its set of weak approximants $\mathcal{A}_w(M)$ gives a semantics.

Theorem 6.5 (Weak approximation semantics) *If $M =_{\beta\mu} N$, then $M \sim_{\mathcal{A}_w} N$.*

Proof: Using Prop. 1.3 and Lem. 6.4.   □

The reverse implication of this result does not hold, since terms without WHNF (which have only $\bot$ as approximant) are not all related by reduction. But we can show the following full abstraction result:

Theorem 6.6 (Full abstraction of $\sim_{w\beta\mu}$ versus $\sim_{\mathcal{A}_w}$) *$M \sim_{w\beta\mu} N$ if and only if $M \sim_{\mathcal{A}_w} N$.*

Proof: *if*: By co-induction on the definition of the set of weak approximants.

only if: As the proof of Theorem 6.5, but using Proposition 5.9 rather than 1.3.   □

We can also show that weak head equivalence and weak approximation equivalence coincide:

Theorem 6.7   *$M \sim_{wH} N$ if and only if $M \sim_{\mathcal{A}_w} N$.*

Proof: Straightforward, by coinduction.   □

We can define $\llbracket M \rrbracket_{\mathcal{A}_w} = \sqcup \{ A_w \mid A_w \in \mathcal{A}_w(M) \}$, with $\sqcup$ the least-upper bound with respect to $\sqsubseteq$; then $\llbracket \cdot \rrbracket_{\mathcal{A}_w}$ corresponds to the ($\lambda\mu$ variant of) Lévy-Longo trees. Combined with the results shown in the previous section, we now also have the following result that states that all equivalences coincide:

Corollary 6.8   *Let $M, N \in \lambda\mu$, then $M \sim_{wxH} N \iff M \sim_{wH} N \iff M \sim_{\mathcal{A}_w} N \iff M \sim_{w\beta\mu} N$.*

We now come to the main result of this paper, where we show a full abstraction result for our logical interpretation. First we show the relation between weak explicit head equivalence and weak bisimilarity.

Theorem 6.9 (Full abstraction of $\approx$ versus $\sim_{wxH}$) *For any $M, N \in \lambda\mu x$: $\llbracket M \rrbracket a \approx \llbracket N \rrbracket a$ if and only if $M \sim_{wxH} N$.*

Proof: *if*: By co-induction on the definition of $\sim_{wxH}$. Let $M \sim_{wxH} N$, then either M and N have both no $\rightarrow_{wxH}$-normal form, so, by Lem. 5.6, their interpretations are both weakly bisimilar to the process 0; or both $M \rightarrow^{nf}_{wxH} M'S$ and $N \rightarrow^{nf}_{wxH} N'S'$ (let $S = \langle \overrightarrow{y:=P} \rangle \langle \overrightarrow{\alpha:=Q \cdot \beta} \rangle$, and $S' = \langle \overrightarrow{y:=P'} \rangle \langle \overrightarrow{\alpha:=Q' \cdot \beta} \rangle$), and either:

$M' = xM_1 \cdots M_n$ ($n \geq 0$), $N = xN_1 \cdots N_n$ and $M_iS \sim_{wxH} N_iS'$, for all $1 \leq i \leq n$:

We have $\llbracket M \rrbracket a \approx \llbracket xM_1 \cdots M_n S \rrbracket a$ and $\llbracket N \rrbracket a \approx \llbracket xN_1 \cdots N_n S' \rrbracket a$ by Corollary 5.3. Notice that

$$\llbracket xM_1 \cdots M_n S \rrbracket a \quad = \quad (\nu \overrightarrow{cy\alpha}) \, (x(u).!u{\rightarrow}c_1 \mid \overrightarrow{\llbracket c_i:=M_i \cdot c_{i+1} \rrbracket} \mid \llbracket S \rrbracket)$$

where $c_n = a$ and

$$\begin{aligned}
\llbracket S \rrbracket &= \overrightarrow{\llbracket y:=P \rrbracket} \mid \overrightarrow{\llbracket \alpha:=Q \cdot \beta \rrbracket} \\
\llbracket c_i:=M_i \cdot c_{i+1} \rrbracket &= !c_i(v,d).(!\overline{v}(w).\llbracket M_i \rrbracket w \mid !d{\rightarrow}c_{i+1}) \\
\llbracket y_j:=P_j \rrbracket &= !\overline{y_j}(w).\llbracket P_j \rrbracket w \\
\llbracket \alpha_k:=Q_k \cdot \beta_k \rrbracket &= !\alpha_k(v,d).(!\overline{v}(w).\llbracket Q_k \rrbracket w \mid !d{\rightarrow}\beta_k)
\end{aligned}$$

and similar for $\llbracket xN_1 \cdots N_n S' \rrbracket a$. By induction,

$$(\nu \overrightarrow{y\alpha}) \, (\llbracket M_i \rrbracket w \mid \llbracket S \rrbracket) \quad \triangleq \quad \llbracket M_i S \rrbracket w \quad \approx \quad \llbracket N_i S' \rrbracket w \quad \triangleq \quad (\nu \overrightarrow{y\alpha}) \, (\llbracket N_i \rrbracket w \mid \llbracket S' \rrbracket)$$

Since $\approx$ is a congruence, also

$$!c_i(v,d).(!\overline{v}(w).\llbracket M_i \rrbracket w \mid !d{\to}c_{i+1}) \mid \llbracket S \rrbracket \quad \approx \quad !c_i(v,d).(!\overline{v}(w).\llbracket N_i \rrbracket w \mid !d{\to}c_{i+1}) \mid \llbracket S' \rrbracket$$

for all $1 \leq i \leq n$, so also $\llbracket xM_1 \cdots M_n \, S \rrbracket a \approx \llbracket xN_1 \cdots N_n \, S' \rrbracket a$ but then also $\llbracket M \rrbracket a \approx \llbracket N \rrbracket a$.

$M' = \lambda x.M''$ *or* $M' = \mu\gamma.[\delta]M''$: Similar.

only if : We distinguish the following cases.

1. $\llbracket M \rrbracket a$ can never input nor output; then $\llbracket M \rrbracket a \approx 0 \approx \llbracket N \rrbracket a$. Assume M has a weak head-normal form, then by Lem. 6.2, $\llbracket M \rrbracket a$ is not weakly bisimilar to 0; therefore, M and N both have no weak head-normal form.

2. $\llbracket M \rrbracket a \Downarrow \overline{c}$, then by Lem. 6.1, $\llbracket M \rrbracket a \approx (\nu x b)(\llbracket M' \rrbracket b \mid \overline{c}\langle x,b \rangle \mid \llbracket S \rrbracket)$, and $M \to^*_{w\mathbf{xH}} \lambda x.M' \, S$. Since $\llbracket M \rrbracket a \approx \llbracket N \rrbracket a$, also $\llbracket N \rrbracket a \Downarrow \overline{c}$, so $\llbracket N \rrbracket a \approx (\nu x b)(\llbracket N' \rrbracket b \mid \overline{c}\langle x,b \rangle \mid \llbracket S' \rrbracket)$ and $N \to^*_{w\mathbf{xH}} \lambda x.N' \, S'$. Then also $\llbracket M' \rrbracket b \mid \llbracket S \rrbracket \approx \llbracket N' \rrbracket b \mid \llbracket S' \rrbracket$, so $\llbracket M' \, S \rrbracket a \approx \llbracket N' \, S' \rrbracket a$ and by induction, $M' \, S \sim_{w\mathbf{xH}} N' \, S'$; so also $M \sim_{w\mathbf{xH}} N$ by definition.

3. If $\llbracket M \rrbracket a \not\Downarrow \overline{c}$, but $\llbracket M \rrbracket a \Downarrow x$, then by Lem. 6.1, $\llbracket M \rrbracket a \approx \llbracket xM_1 \cdots M_n \, S \rrbracket a'$ and $M \to^*_{w\mathbf{xH}} xM_1 \cdots M_n \, S$. We have

$$\llbracket xM_1 \cdots M_n \, S \rrbracket a' \quad = \quad (\nu \overline{cy\vec{\alpha}})(x(u).!u{\to}c_1 \mid \overrightarrow{\llbracket c_i := M_i \cdot c_{i+1} \rrbracket} \mid \llbracket S \rrbracket)$$

with $\llbracket S \rrbracket$, $\llbracket c_i := M_i \cdot c_{i+1} \rrbracket$, $\llbracket y_j := P_j \rrbracket$, and $\llbracket \alpha_k := Q_k \cdot \beta_k \rrbracket$ are defined as above.

Since $\llbracket M \rrbracket a \approx \llbracket N \rrbracket a$, again by Lem. 6.1, $\llbracket N \rrbracket a \approx \llbracket xN_1 \cdots N_n \, S' \rrbracket a''$ and $N \to^*_{w\mathbf{xH}} xN_1 \cdots N_n \, S'$. Notice that

$$\llbracket xN_1 \cdots N_n \, S' \rrbracket a'' \quad = \quad (\nu \overline{cy\vec{\alpha}})(x(u).!u{\to}c_1 \mid \overrightarrow{\llbracket c_i := N_i \cdot c_{i+1} \rrbracket} \mid \llbracket S' \rrbracket)$$

with $\llbracket S' \rrbracket$, $\llbracket c_i := N_i \cdot c_{i+1} \rrbracket$, $\llbracket y_j := P'_j \rrbracket$, and $\llbracket \alpha_k := Q'_k \cdot \beta_k \rrbracket$ similar to above. Then we have

$$\llbracket xM_1 \cdots M_n \, S \rrbracket a' \quad \approx \quad \llbracket xN_1 \cdots N_n \, S' \rrbracket a'',$$

so $a' = a''$ and $\llbracket M'_i \, S \rrbracket w \approx \llbracket N'_i \, S' \rrbracket w$; then by induction, $M'_i \, S \sim_{w\mathbf{xH}} N'_i \, S'$, and $M \sim_{w\mathbf{xH}} N$.   □

We now obtain our main result:

Theorem 6.10 (Full abstraction) *Let* $M, N \in \lambda\mu$, *then* $\llbracket M \rrbracket a \approx \llbracket N \rrbracket a$ *if and only if* $M \sim_{w\beta\mu} N$.

Conclusions and future work

We have studied the output based, logic-inspired interpretation of untyped $\lambda\mu$ with explicit substitution into the π-calculus and shown that this interpretation is fully abstract with respect to weak equivalence between terms and weak bisimilarity between processes.

We have defined the weak equivalences $\sim_{w\beta\mu}$, $\sim_{w\mathbf{H}}$, $\sim_{w\mathbf{xH}}$, and $\sim_{\mathcal{A}_w}$ on $\lambda\mu$ terms, and shown that these all coincide. We then proved that $M \sim_{w\mathbf{xH}} N \Longleftrightarrow \llbracket M \rrbracket a \approx \llbracket N \rrbracket a$, which, combined with our other results, essentially shows that $\llbracket \cdot \rrbracket \cdot$ respects equality between Lévy-Longo trees for $\lambda\mu$.

We will investigate the relation between our interpretation and the CPS-translation of Lafont, Reus, and Streicher [18].

References

[1] M. Abadi & A. Gordon (1997): *A Calculus for Cryptographic Protocols: The Spi Calculus*. In: *4th CCS*, pp. 36–47, doi:10.1145/266420.266432.

[2] S. Abramsky & C.-H.L. Ong (1993): *Full Abstraction in the Lazy Lambda Calculus*. Information and Computation 105(2), pp. 159–267, doi:10.1006/inco.1998.2740.

[3] S. van Bakel, L. Cardelli & M.G. Vigliotti (2008): *From $\mathcal{X}$ to π; Representing the Classical Sequent Calculus in the π-calculus*. In: *CL&C'08*, arXiv:1109.4817.

[4] S. van Bakel & P. Lescanne (2008): *Computation with Classical Sequents. Mathematical Structures in Computer Science* 18, pp. 555–609, doi:10.1017/S0960129508006762.

[5] S. van Bakel & M.G. Vigliotti (2009): *A logical interpretation of the λ-calculus into the π-calculus, preserving spine reduction and types*. In *CONCUR'09*, 5710, Springer, pp. 84 – 98, doi:10.1007/978-3-642-04081-8_7.

[6] S. van Bakel & M.G. Vigliotti (2012): *An Output-Based Semantics of $\lambda\mu$ with Explicit Substitution in the π-calculus - Extended Abstract*. In *IFIP-TCS 2012*, *LNCS* 7604, Springer, pp. 372–387, doi:10.1007/978-3-642-33475-7_26.

[7] H. Barendregt (1984): *The Lambda Calculus: its Syntax and Semantics*, revised edition. North-Holland.

[8] R. Bloo & K.H. Rose (1995): *Preservation of Strong Normalisation in Named Lambda Calculi with Explicit Substitution and Garbage Collection*. In: *CSN'95*, pp. 62–72, doi:10.1.1.51.5026.

[9] A. Church (1936): *A Note on the Entscheidungsproblem*. *JSL* 1(1), pp. 40–41, doi:10.2307/2269326.

[10] M. Cimini, C. Sacerdoti Coen & D. Sangiorgi (2010): *Functions as Processes: Termination and the $\overline{\lambda}\mu\tilde{\mu}$-Calculus*. In *TGC'10*, *LNCS* 6084, Springer, pp. 73–86, doi:10.1007/978-3-642-15640-3_5.

[11] P.-L. Curien & H. Herbelin (2000): *The Duality of Computation*. In: *ICFP'00*, *ACM Sigplan Notices* 35.9, ACM, pp. 233–243, doi:10.1145/351240.351262.

[12] U. de'Liguoro (2014): *The Approximation Theorem for the $\Lambda\mu$-Calculus*. To appear in MSCS.

[13] Ph. de Groote (1994): *On the Relation between the $\lambda\mu$-Calculus and the Syntactic Theory of Sequential Control*. In: *LPAR'94*, *LNCS* 822, Springer, pp. 31–43, doi:10.1007/3-540-58216-9_27.

[14] K. Honda & M. Tokoro (1991): *An Object Calculus for Asynchronous Communication*. In *ECOOP'91*, *LNCS* 512, Springer, pp. 133–147, doi:10.1007/BFb0057019.

[15] K. Honda, N. Yoshida & M. Berger (2004): *Control in the π-Calculus*. In: *Proceedings of Fourth ACM-SIGPLAN Continuation Workshop* (CW'04).

[16] J.W. Klop (1992): *Term Rewriting Systems*. In *Handbook of Logic in Computer Science*, chapter 1, 2, Clarendon Press, pp. 1–116.

[17] J-L. Krivine (2007): *A call-by-name lambda-calculus machine*. *Higher Order and Symbolic Computation* 20(3), pp. 199–207, doi:10.1007/s10990-007-9018-9.

[18] Y. Lafont, B. Reus & Th. Streicher (1993): *Continuation Semantics or Expressing Implication by Negation*. Report 9321, Ludwig-Maximilians-Universität, München.

[19] S.B. Lassen (2006): *Head Normal Form Bisimulation for Pairs and the $\lambda\mu$-Calculus*. In: *LICS'06*, pp. 297–306. Available at http://doi.ieeecomputersociety.org/10.1109/LICS.2006.29.

[20] R. Milner (1992): *Functions as Processes*. *Mathematical Structures in Computer Science* 2(2), pp. 269–310, doi:10.1017/S0960129500001407.

[21] M. Parigot (1992): *An algorithmic interpretation of classical natural deduction*. In: *LPAR'92*, *LNCS* 624, Springer, pp. 190–201, doi:10.1007/BFb0013061.

[22] W. Py (1998): *Confluence en $\lambda\mu$-calcul*. Phd thesis, Université de Savoie.

[23] D. Sangiorgi (1994): *The Lazy Lambda Calculus in a Concurrency Scenario*. *I&C* 111(1), pp. 120–153, doi:10.1006/inco.1994.1042.

[24] D. Sangiorgi & D. Walker (2001): *The Pi-Calculus*. Cambridge University Press.

[25] A. Saurin (2010): *Standardization and Böhm Trees for $\lambda\mu$-calculus*. In M. Blume, N. Kobayashi & G. Vidal, editors: *FLOPS'10*, *LNCS* 6009, Springer, pp. 134–149, doi:10.1007/978-3-642-12251-4_11.

[26] C.P. Wadsworth (1976): *The Relation Between Computational and Denotational Properties for Scott's D_{infty}-Models of the Lambda-Calculus*. *SIAM JoC* 5(3), pp. 488–521, doi:10.1137/0205036.