

Head Normal Form Bisimulation for Pairs and the $\lambda\mu$ -Calculus (Extended Abstract)

Soren B. Lassen
Google, Inc.
soren@google.com

Abstract

Böhm tree equivalence up to possibly infinite η expansion for the pure λ -calculus can be characterized as a bisimulation equivalence. We call this co-inductive syntactic theory extensional head normal form bisimilarity and in this paper we extend it to the λ FP-calculus (the λ -calculus with functional and surjective pairing) and to two untyped variants of Parigot’s $\lambda\mu$ -calculus. We relate the extensional head normal form bisimulation theories for the different calculi via Fujita’s extensional CPS transform into the λ FP-calculus. We prove that extensional hnf bisimilarity is fully abstract for the pure λ -calculus by a co-inductive reformulation of Barendregt’s proof for Böhm tree equivalence up to possibly infinite η expansion. The proof uses the so-called Böhm-out technique from Böhm’s proof of the Separation Property for the λ -calculus. Moreover, we extend the full abstraction result to extensional hnf bisimilarity for the λ FP-calculus. For the “standard” $\lambda\mu$ -calculus, the Separation Property fails, as shown by David and Py, and for the same reason extensional hnf bisimilarity is not fully abstract. However, an “extended” variant of the $\lambda\mu$ -calculus satisfies the Separation Property, as shown by Saurin, and we show that extensional hnf bisimilarity is fully abstract for this extended $\lambda\mu$ -calculus.

1 Introduction

The goal of the work presented in this paper is to understand the computational behaviour of the untyped λ -calculus with pairs and the untyped $\lambda\mu$ -calculus [16] and to find principles for reasoning about program behaviour and equivalence. In particular, we wish to learn the properties of two recent contributions to these calculi. One is Fujita’s untyped adaptation [7] of Hofmann and Streicher’s extensional CPS transform [9] of the λ -calculus and $\lambda\mu$ -calculus into the λ -calculus with pairs. The other is Saurin’s extension of the untyped $\lambda\mu$ -calculus in [20].

We use the programming language semantics tools of structural operational semantics (SOS), continuation passing style (CPS) transform, and bisimulation equivalences. We take head reduction to head normal form (hnf) as our operational semantics, defined as big-step SOS evaluation relations, we use Fujita’s extensional CPS transform, and we use the bisimulation characterization of Böhm tree equivalence up to possibly infinite η expansion from [12] as our bisimulation equivalence.

We give new definitions of solvability, head normal forms (hnfs), and head evaluation for the calculi and we present new co-inductive theories called extensional hnf bisimilarity. For the λ FP-calculus and the extended $\lambda\mu$ -calculus we prove that extensional hnf bisimilarity coincides with solvable equivalence. Furthermore, we relate extensional hnf bisimilarity for the λ FP-calculus to extensional hnf bisimilarity for the other calculi via extensional CPS transforms.

Böhm tree equivalence up to possibly infinite η -expansion (or Nakajima tree equivalence) is a powerful syntactic characterization of the theory of solvable equivalence for the pure “sensible” λ -calculus [2]. It can be formulated co-inductively as a syntactic bisimulation equivalence, without reference to Böhm trees, based on head reduction to head normal form (hnf) [12]. We call this characterization *extensional hnf bisimilarity*. It is the greatest symmetric term relation \approx such that, whenever $t \approx t'$, either t and t' have no hnfs or t and t' both have hnfs that η -expand to hnfs

$$\lambda x_1 \dots x_m. y t_1 \dots t_n, \quad \lambda x_1 \dots x_m. y t'_1 \dots t'_n,$$

respectively, where $m, n \geq 0$ and $t_1 \approx t'_1, \dots, t_n \approx t'_n$.

This bisimulation characterization of solvable equivalence is useful because it makes it possible to prove program equivalences by the powerful and convenient bisimulation proof method. See [12, 14, 13] for examples of normal form bisimulation proofs of program equivalences.

Outline Section 2 describes related work. Sections 3 and 4 present the pure λ -calculus and the λ FP-calculus and solvability, hnfs, head evaluation, extensional hnf bisimilarity,

and Fujita’s CPS transform. Section 5 outlines the congruence proof for extensional hnf bisimilarity for the λ FP-calculus. Section 6 outlines full abstraction proofs for the pure λ -calculus and the λ FP-calculus. The proofs present the “Böhm-out” technique in a novel co-inductive fashion. Sections 7 and 8 extend solvability, head normal forms, head evaluation, extensional hnf bisimilarity, and the CPS transform to the standard $\lambda\mu$ -calculus and the extended $\lambda\mu$ -calculus. We show that extensional hnf bisimilarity *is not* fully abstract for the standard $\lambda\mu$ -calculus, using an example by David and Py, and we extend the Böhm-out technique to prove that extensional hnf bisimilarity *is* fully abstract for the extended $\lambda\mu$ -calculus

2 Related work

Sangiorgi [19] first gave a bisimulation characterization of Lévy-Longo tree equivalence, which he called open applicative bisimulation. The hnf bisimulation characterization of Böhm tree equivalence (up to infinite η -expansion) was introduced by Lassen [12] and by David [5]. Eager normal form bisimulation [13] is a recent variant of hnf bisimulation for the call-by-value λ -calculus.

Most of the literature about the λ -calculus with surjective/explicit/functional pairs (see the survey in [21]) is about reduction, confluence, and conversion. Our definitions of head normal forms, head evaluation, solvability, solvable equivalence, and bisimilarity appear to be new.

Parigot introduced the standard $\lambda\mu$ -calculus as a term assignment system for classical logic. David and Py [6] defined solvability, head normal forms, head reduction, and solvable equivalence for the untyped standard $\lambda\mu$ -calculus. Saurin introduced the untyped extended $\lambda\mu$ -calculus and defined head reduction and head normal forms. (Saurin names the calculus $\Lambda\mu$ and he uses a slightly different syntax.) We introduce alternative definitions of head normal forms and head evaluation that are better suited for hnf bisimulation. Our “big-step” definition of head evaluation is reminiscent of Ong and Stewart’s evaluation semantics for a typed call-by-value $\lambda\mu$ -calculus in [15]. Our definition of solvability is new.

David and Py established a “computational equivalence” result [6] for the standard $\lambda\mu$ -calculus which amounts to an applicative bisimulation characterization of solvable equivalence [1].

CPS Translation Theorems relate the extensional hnf bisimulation theories for the λ FP-calculus and the other calculi via Fujita’s CPS transform. This is parallel to Boudol’s correspondence result between Lévy-Longo tree equivalence and Böhm tree equivalence via Plotkin’s call-by-name CPS transform [4] and Lassen’s correspondence result between eager normal form (call-by-value) bisimilarity and hnf bisimilarity via Plotkin’s call-by-value CPS transform

[13] (Plotkin defined the CPS transforms in [17]).

Our syntactic congruence proof for extensional hnf bisimilarity for the λ FP-calculus is novel. The Main Lemma only shows that bisimilarity is substitutive, not compatible. Compatibility is then derived from the definition of bisimilarity. Otherwise the congruence proof is similar to the congruence proofs for Lévy-Longo tree equivalence and Böhm tree equivalence in [12], which can be viewed as more intuitive (and symmetric) variant of Howe’s congruence proof method [10]. Congruence of extensional hnf bisimilarity for the $\lambda\mu$ -calculus is derived via the CPS transform, as in [13].

Our full abstraction proof for the pure λ -calculus is a co-inductive reformulation of the full abstraction proofs by Wadsworth, Hyland, and Barendregt [22, 11, 2]. It uses the Böhm-out technique from Böhm’s proof of the Separation Property [3], a.k.a. Böhm’s Theorem. Our extension of the proof to the λ FP-calculus is new. Our proof for the extended $\lambda\mu$ -calculus combines the new co-inductive Böhm-out formulation with elements from Saurin’s proof of the Separation Property [20].

3 The λ -calculus

Let x, y, z range over variables. The terms of the pure λ -calculus are given by the grammar:

TERMS $t ::= x \mid \lambda x. t \mid t_1 t_2$

Application associates to the left and λ -abstraction extends as far to the right as possible, so $\lambda x. t_1 t_2 t_3 = \lambda x. ((t_1 t_2) t_3)$.

Let $FV(t)$ denote the set of free variables in t . We identify terms up to renaming of bound variables. Let $t_1[t_2/x]$ denote the capture-free substitution of t_2 for all free occurrences of x in t_1 .

Even though we will not be concerned with reduction and conversion, we list the usual β and η equations for the λ -calculus:

$$(\lambda x. t_1) t_2 =_{\beta} t_1[t_2/x] \quad (\beta)$$

$$\lambda x. t x =_{\eta} t, \text{ if } x \notin FV(t) \quad (\eta)$$

The β equation will form the basis for the evaluation relation below and both equations will be satisfied by extensional hnf bisimilarity.

Rather than the usual definition of head reduction and the derived definition of head normal forms as the normal forms of head reduction, here we first define the head normal forms (hnfs) and then define a head evaluation relation between terms and hnfs.

HNFS $h ::= f \mid \lambda x. h$

λ -FREE HNFS $f ::= x \mid f t$

That is, hnfs are terms of the form $\lambda x_1 \dots x_m. y t_1 \dots t_n$.

The head evaluation relation between terms t and hnfs h , written $t \rightsquigarrow h$, is defined inductively in the style of big-step structural operational semantics, except that we define head evaluation also for open terms.

$$\begin{aligned}
& \text{(E1)} \frac{}{x \rightsquigarrow x} \\
& \text{(E2)} \frac{t \rightsquigarrow h}{\lambda x. t \rightsquigarrow \lambda x. h} \\
& \text{(E3)} \frac{t_1 \rightsquigarrow f}{t_1 t_2 \rightsquigarrow f t_2} \\
& \text{(E4)} \frac{t_1 \rightsquigarrow \lambda x. h_1 \quad h_1[t_2/x] \rightsquigarrow h}{t_1 t_2 \rightsquigarrow h}
\end{aligned}$$

This definition is convenient for our uses. The head evaluation relation can also be defined in terms of conventional head reduction to hnf.

We say that a term t has a hnf if $t \rightsquigarrow h$ for some h . Not all terms have a hnf. For instance, $\Omega = (\lambda x. x x) (\lambda x. x x)$ has no hnf.

For the pure λ -calculus, a closed term t is *solvable* iff $t \vec{t} =_{\beta} \lambda x. x$, for some closed arguments \vec{t} (zero or more arguments) [2].

Lemma 1 (Wadsworth [22]). *A closed term is solvable iff it has a hnf.*

A term relation R is *semi sensible* iff, whenever t and t' are closed terms and $t R t'$, t is solvable iff t' is solvable. A semi sensible relation is *sensible* iff it relates all unsolvable terms, that is, it satisfies the equation:

$$t =_{\Omega} \Omega, \text{ if } t \text{ is unsolvable} \quad (\Omega)$$

Definition 1. Let *solvable equivalence* be the greatest compatible semi sensible relation. (A term relation is compatible if it is closed under all term constructors.) It relates terms t and t' iff, for all term contexts C (term with any number of holes $[\]$) such that $C[t]$ and $C[t']$ are closed, $C[t]$ is solvable iff $C[t']$ is solvable.

Given a term relation R , let $M(R)$ be the relation between hnfs defined inductively by the rules:

$$\begin{aligned}
& \text{(M1)} \frac{}{x M(R) x} \\
& \text{(M2)} \frac{f M(R) f'}{f t M(R) f' t'} \text{ if } t R t' \\
& \text{(M3)} \frac{h M(R) h'}{\lambda x. h M(R) \lambda x. h'} \\
& \text{(M4)} \frac{f x M(R) h}{f M(R) \lambda x. h} \text{ if } x \notin \text{FV}(f) \\
& \text{(M5)} \frac{h M(R) f x}{\lambda x. h M(R) f} \text{ if } x \notin \text{FV}(f)
\end{aligned}$$

In other words, $h M(R) h'$ if $h = \lambda x_1 \dots x_m. y t_1 \dots t_n$ and $h' = \lambda x_1 \dots x_{m+k}. y t'_1 \dots t'_{n+k}$ and

$$t_1 R t'_1, \dots, t_n R t'_n, x_{m+1} R t_{n+1}, \dots, x_{m+k} R t_{n+k}$$

and similarly if h has more leading lambdas than h' .

Definition 2. Given a term relation R , let $B(R)$ be the term relation defined as follows:

$$\begin{aligned}
& t B(R) t' \text{ iff} \\
& \quad (i) \forall h. t \rightsquigarrow h \Rightarrow \exists h'. t' \rightsquigarrow h' \ \& \ h M(R) h' \text{ and} \\
& \quad (ii) \forall h'. t' \rightsquigarrow h' \Rightarrow \exists h. t \rightsquigarrow h \ \& \ h M(R) h'
\end{aligned}$$

A term relation R is a *hnf bisimulation up to η* [12] iff it is a post-fixpoint of B , that is, $R \subseteq B(R)$. Let *extensional hnf bisimilarity*, \approx , be the greatest fixpoint of B which, by the Tarski-Knaster fixpoint theorem, exists and is the greatest hnf bisimulation up to η .

For example, we can show co-inductively that extensional hnf bisimilarity satisfies the well-known infinite η -expansion equation:

$$\lambda x. x \approx J, \text{ where } J = \Theta \lambda x y z. y x z \quad (1)$$

(Θ denotes Turing's fixpoint combinator) by checking that $\{(\lambda x. x, J), (z, J z) \mid z \text{ is a variable}\}$ is a hnf bisimulation up to η .

It is immediate from the definition that extensional hnf bisimilarity is semi sensible and it is easy to show that extensional hnf bisimilarity is an equivalence relation (reflexive, transitive, symmetric) and, using properties of head evaluation, that it includes the β , η , and Ω equations and satisfies the following extensionality property (see [2]):

$$t x \approx t' x \Rightarrow t \approx t', \text{ if } x \notin \text{FV}(t) \cup \text{FV}(t') \quad (\text{ext})$$

It follows from the definitions that extensional hnf bisimilarity is the same as Böhm tree equivalence up to possibly infinite η expansion. Barendregt showed that this is a congruence and that it is fully abstract with respect to (coincides with) solvable equivalence. A direct co-inductive congruence proof for extensional hnf bisimilarity is given in [12]. Later in this paper we will give a new congruence proof and outline a new co-inductive formulation of the full abstraction proof.

4 The λ FP-calculus

We now consider the λ FP-calculus which is an extension of the λ -calculus with functional and surjective pairing:

$$\text{TERMS } t ::= x \mid \lambda x. t \mid t_1 t_2 \mid \langle t_1, t_2 \rangle \mid \pi_1 t \mid \pi_2 t$$

The β equation from the pure λ -calculus together with the projection equation:

$$\pi_i \langle t_1, t_2 \rangle =_\pi t_i, \text{ if } i \in \{1, 2\} \quad (\pi)$$

and two commutation (or “functional pairing”) equations [21]:

$$\pi_i (t_1 t_2) =_{\pi_\nu} (\pi_i t_1) t_2, \text{ if } i \in \{1, 2\} \quad (\pi_\nu)$$

$$\pi_i \lambda x. t =_{\pi_\lambda} \lambda x. \pi_i t, \text{ if } i \in \{1, 2\} \quad (\pi_\lambda)$$

form the basis for the evaluation relation below. Extensional hnf bisimilarity for the λ FP-calculus, which we define below, satisfies all these equations as well as the η equation from the pure λ -calculus and the surjective pairing equation:

$$\langle \pi_1 t, \pi_2 t \rangle =_{\text{sp}} t \quad (\text{sp})$$

One motivation for studying the λ -calculus with pairs is that it is the target of Fujita’s continuation passing style (CPS) transform of the pure λ -calculus [7]. Following Danvy and Hatcliff [8], we change Fujita’s transform slightly by adding an η redex to the variable case.

$$\begin{aligned} \text{cps}(x) &= \lambda a. x a \\ \text{cps}(\lambda x. t) &= \lambda a. (\lambda x. \text{cps}(t)) (\pi_1 a) (\pi_2 a) \\ \text{cps}(t_1 t_2) &= \lambda a. \text{cps}(t_1) \langle \text{cps}(t_2), a \rangle \end{aligned}$$

where a is a fresh variable that does not occur free in the source terms. Fujita [7] has investigated relationships between reductions and equations in the source and target calculi with respect to his transform. In Section 5 we report a new CPS Translation Theorem that relates extensional hnf bisimilarity in the λ -calculus and the λ FP-calculus.

We define the head normal forms as follows:

$$\text{HNFS } h ::= f \mid \lambda x. h$$

$$\lambda\text{-FREE HNFS } f ::= p[x] \mid f t$$

$$\text{PROJECTION CONTEXTS } p ::= [] \mid \pi_1 p \mid \pi_2 p$$

To differentiate λ -terms and hnfs in the pure λ -calculus from the new definitions for the λ FP-calculus, we will refer to the former as *pure terms* and *pure hnfs* in the sequel.

These operations are used in the definitions of head evaluation and $M(R)$ below.

We define head evaluation from terms to hnfs by the rules:

$$(E1) \frac{}{p[x] \rightsquigarrow p[x]}$$

$$(E2) \frac{p[t] \rightsquigarrow h}{p[\lambda x. t] \rightsquigarrow \lambda x. h}$$

$$(E3) \frac{p[t_1] \rightsquigarrow f}{p[t_1 t_2] \rightsquigarrow f t_2}$$

$$(E4) \frac{p[t_1] \rightsquigarrow \lambda x. h_1 \quad h_1[t_2/x] \rightsquigarrow h}{p[t_1 t_2] \rightsquigarrow h}$$

$$(E5) \frac{p[t_i] \rightsquigarrow h}{p[\pi_i \langle t_1, t_2 \rangle] \rightsquigarrow h} \text{ if } i \in \{1, 2\}$$

For the λ FP-calculus, we define a closed term t to be solvable iff $p[t] \vec{t} =_{\beta\pi\pi_\nu\pi_\lambda} \lambda x. x$, for some projection context p and closed arguments \vec{t} . For example, Ω and $\langle \Omega, \Omega \rangle$ are unsolvable, but $t = \langle \lambda x. x, \Omega \rangle$ is solvable, even though t does not have a hnf, because $\pi_1 t =_\pi \lambda x. x$.

Lemma 2. *A closed term t is solvable iff there is a projection context p such that $p[t]$ has a hnf.*

As for the pure λ -calculus, we say that a relation R on λ FP-terms is semi sensible iff, whenever R relates two closed terms t and t' , t is solvable iff t' is solvable. Solvable equivalence is again the greatest compatible semi-sensible relation.

For the λ FP-calculus we extend the definition of $M(R)$ by generalizing (M1) as follows:

$$(M1) \frac{}{p[x] M(R) p[x]}$$

In the definition of $B(R)$ we will also use the notation $t \rightsquigarrow_p h$, defined by

$$t \rightsquigarrow_p \lambda \vec{x}. p'[y] \vec{t} \text{ iff } p[t] \rightsquigarrow \lambda \vec{x}. p[p'[y]] \vec{t}.$$

For example, $\langle \pi_1 h, \pi_2 h \rangle \rightsquigarrow_{\pi_1[]} h$.

Definition 3. Given a term relation R , let $B(R)$ be the term relation defined as follows:

$t B(R) t'$ iff

- (i) $\forall p, h. p[t] \rightsquigarrow h \Rightarrow \exists p', h'. p[t'] \rightsquigarrow_{p'} h' \ \& \ h M(R) h'$ and
- (ii) $\forall p, h'. p[t'] \rightsquigarrow h' \Rightarrow \exists p', h. p[t] \rightsquigarrow_{p'} h \ \& \ h M(R) h'$

Again a relation R is a hnf bisimulation up to η iff $R \subseteq B(R)$ and extensional hnf bisimilarity, \approx , is the greatest fixpoint of B .

Extensional hnf bisimilarity relates terms up to infinite sp-expansion, as illustrated by

$$\lambda x. x \approx L, \text{ where } L = \Theta \lambda x. y \langle \pi_1 y, x (\pi_2 y) \rangle \quad (2)$$

which follows immediately from the fact that \approx is a fixpoint of B . This example motivates the subtleties in the definition of $B(R)$ in Definition 3.

Just like for the pure λ -calculus, extensional hnf bisimilarity is a semi sensible equivalence relation and includes the Ω equation (for the new definition of solvability) and satisfies the extensionality property *ext*. Moreover, it includes extensional hnf bisimilarity for the pure λ -calculus and includes the $\beta, \eta, \pi, \pi_\nu, \pi_\lambda$, and sp equations.

Remark 1. The definitions of hnfs, head evaluation, solvability, $M(-)$, $B(-)$, and extensional hnf bisimilarity all cleanly extend those for the pure λ -calculus. In particular, two pure λ -terms are extensional hnf bisimilar in the pure λ -calculus (Definition 2) iff they are extensional hnf bisimilar in the λ FP-calculus (Definition 3). This is a general property of normal form bisimulation theories. The same is true for extensional hnf bisimilarity for the standard $\lambda\mu$ -calculus in Section 7 and again for the extended $\lambda\mu$ -calculus in Section 8. The whnf bisimulation equivalence for a non-deterministic λ -calculus with ambiguous choice in [14] exhibits the same property relative to whnf bisimilarity for the pure deterministic λ -calculus.

5 Congruence and CPS Translation Theorem

We now want to show that extensional hnf bisimilarity for the λ FP-calculus is a congruence. To this end we define, for any given term relation R , its substitutive hnf-closure $S(R)$ to be the term relation defined inductively by the rules:

$$(S1) \frac{}{t S(R) t'} \text{ if } t R t'$$

$$(S2) \frac{h M(S(R)) h'}{h S(R) h'}$$

$$(S3) \frac{t_1 S(R) t'_1 \quad t_2 S(R) t'_2}{t_1 [t_2/x] S(R) t'_1 [t'_2/x]}$$

Lemma 3 (Main Lemma). *If R is a hnf bisimulation up to η , so is $S(R)$.*

Proof outline. We instrument the head evaluation relation $t \rightsquigarrow h$ with a natural number n , written $t \overset{n}{\rightsquigarrow} h$, where n counts the number of β -reductions (rule (E5)) used in the derivation of $t \rightsquigarrow h$.

Now, under the assumption that $R \subseteq B(R)$, we prove

$$t S(R) t' \ \& \ p[t] \overset{n}{\rightsquigarrow} h \Rightarrow \\ \exists p', h'. p[t'] \rightsquigarrow_{p'} h' \ \& \ h M(S(R)) h'$$

by induction (1) on n and (2) on the derivation of $t S(R) t'$, ordered lexicographically. The argument resembles the congruence proofs in [12, 14]. This establishes part (i) of the inclusion $S(R) \subseteq B(S(R))$, cf. Definition 3. Part (ii) is symmetrical. \square

Theorem 4. *Extensional hnf bisimilarity is a congruence for the λ FP-calculus.*

Proof. We know that extensional hnf bisimilarity is an equivalence relation (reflexive, transitive, and symmetric), so we just need to show that it is compatible, namely:

$$t \approx t' \Rightarrow \lambda x. t \approx \lambda x. t' \quad (C1)$$

$$t_1 \approx t'_1 \ \& \ t_2 \approx t'_2 \Rightarrow t_1 t_2 \approx t'_1 t'_2 \quad (C2)$$

$$t_1 \approx t'_1 \ \& \ t_2 \approx t'_2 \Rightarrow \langle t_1, t_2 \rangle \approx \langle t'_1, t'_2 \rangle \quad (C3)$$

$$t \approx t' \Rightarrow \pi_i t \approx \pi_i t', \text{ if } i \in \{1, 2\} \quad (C4)$$

For (C1), $t \approx t'$ implies $\lambda x. t B(\approx) \lambda x. t'$ directly, without the Main Lemma, as follows. Suppose $p[\lambda x. t] \rightsquigarrow h$. This must be derived from (E2), so $h = \lambda x. h_0$ for some h_0 such that $p[t] \rightsquigarrow h_0$. If $t \approx t'$, $p[t'] \rightsquigarrow_{p'} h'_0$ for some p' and hnf h'_0 such that $h_0 M(\approx) h'_0$. By (E2), $p[\lambda x. t'] \rightsquigarrow_{p'} \lambda x. h'_0$ and, by (M3), $h = \lambda x. h_0 M(\approx) \lambda x. h'_0$. This establishes part (i) of $\lambda x. t B(\approx) \lambda x. t'$. Part (ii) is symmetrical.

Since extensional hnf bisimilarity is a hnf bisimulation up to η , the Main Lemma entails that its substitutive hnf closure is a hnf bisimulation up to η and thus contained in extensional hnf bisimilarity, by co-induction. We conclude that extensional hnf bisimilarity is substitutive:

$$t_1 \approx t'_1 \ \& \ t_2 \approx t'_2 \Rightarrow t_1 [t_2/x] \approx t'_1 [t'_2/x]$$

The clauses (C2), (C3), and (C4) follow from substitutivity and reflexivity. \square

Theorem 5 (CPS Translation). *$t \approx t'$ in the pure λ -calculus iff $\text{cps}(t) \approx \text{cps}(t')$ in the λ FP-calculus.*

The proof is analogous to the proof of the CPS Translation Theorem for Plotkin's CPS transform, eager normal form bisimilarity up to η , and extensional hnf bisimilarity in [13]: The proof relies on a CPS Simulation Theorem that relates head evaluation of terms t in the pure λ -calculus and head evaluation of $\text{cps}(t)$ in the λ FP-calculus, then both the 'if' and 'only if' directions of the CPS Translation Theorem are established by exhibiting hnf bisimulations up to η .

Since $\text{cps}(-)$ is compositional and extensional hnf bisimilarity is a congruence for the λ FP-calculus, we conclude:

Corollary 6. *Extensional hnf bisimilarity is a congruence for the pure λ -calculus.*

6 Böhm-out and full abstraction

In this section, we show that extensional hnf bisimilarity is fully abstract with respect to solvable equivalence. Our proof is based on the classical proofs [22, 11, 2] but we use a co-inductive formulation of the Böhm-out technique that elucidates the argument. We first prove full abstraction for the pure λ -calculus and then describe how the proof extends to the λ FP-calculus. In Section 8 we adapt the proof to the $\lambda\mu$ -calculus.

Let Λ be the set of all pure λ -terms. Then $\Lambda \times \Lambda$ is the universal relation on pure λ -terms. Two closed pure hnfs h and h' are *equivalent*, written $h \sim h'$, iff $h M(\Lambda \times \Lambda) h'$. For arbitrary closed pure λ -terms t and t' we define $t \sim t'$ iff $t B(\Lambda \times \Lambda) t'$, that is either both t and t' are unsolvable or both are solvable and have equivalent hnfs.

Lemma 7. $t \sim t'$, for all closed pure solvably equivalent λ -terms t and t' .

The *permutator* \mathbf{P}_n and *projection* \mathbf{U}_i^n , for $0 \leq i \leq n$, are the terms:

$$\begin{aligned} \mathbf{P}_n &= \lambda x_1 \dots x_n x_{n+1} \cdot x_{n+1} x_1 \dots x_n \\ \mathbf{U}_i^n &= \lambda x_1 \dots x_n \cdot x_i \end{aligned}$$

Collectively, we call permutators and projections *discriminator terms*. Let δ range over these and let Δ range over applicative contexts with discriminator arguments:

$$\text{DISCRIMINATORS } \delta ::= \mathbf{P}_n \mid \mathbf{U}_i^n$$

$$\text{DISCRIMINATION CONTEXTS } \Delta ::= [] \mid \Delta \delta$$

Now we define a ‘‘Böhm-out’’ relation \approx . For closed pure λ -terms t and t' ,

$$t \approx t' \text{ iff } \forall \Delta. \Delta[t] \sim \Delta[t']$$

On open terms we define \approx as the limit of an increasing sequence of relations, $\approx = \bigcup_{n < \omega} \approx_n$, where $t \approx_n t'$ iff

$$\begin{aligned} \forall \{x_1, \dots, x_m\} \supseteq \text{FV}(t) \cup \text{FV}(t'). \forall n_1, \dots, n_m \geq n. \\ t[\mathbf{P}_{n_1/x_1}] \dots [\mathbf{P}_{n_m/x_m}] \approx t'[\mathbf{P}_{n_1/x_1}] \dots [\mathbf{P}_{n_m/x_m}]. \end{aligned}$$

Lemma 8 (Böhm-out). \approx is a hnf bisimulation up to η .

Proof outline. We must show that $\approx \subseteq B(\approx)$, that is, $t \approx_n t'$ implies $t B(\approx) t'$, for all $n < \omega$ and terms t, t' . We show that $t \rightsquigarrow h$ implies there exist h' and n' such that $t' \rightsquigarrow h'$ and $h M(\approx_{n'}) h'$. The choice of n' depends on n and the structure of h . By symmetry we conclude that $t B(\approx) t'$, as required. \square

Theorem 9 (Full abstraction [2]). *Extensional hnf bisimilarity coincides with solvable equivalence for the pure λ -calculus.*

Proof. Extensional hnf bisimilarity is included in solvable equivalence because it is semi sensible and congruent. Conversely, since solvable equivalence is included in \sim and is congruent, it is easy to see that solvable equivalence is included in \approx which, by the Böhm-out Lemma and co-induction, is included in extensional hnf bisimilarity. \square

The definitions and lemmas extend to the λ FP-calculus, if we let terms and hnfs range over λ FP-terms and hnfs and we use $M(\Lambda\text{FP} \times \Lambda\text{FP})$ and $B(\Lambda\text{FP} \times \Lambda\text{FP})$ in the definition of \sim , where ΛFP is the set of all λ FP-terms and M and B are the extended relational operators from Section 4.

We extend the definition of discriminators:

$$\text{DISCRIMINATORS } \delta ::= \mathbf{P}_n \mid \bar{p}[\mathbf{U}_i^n]$$

where \bar{p} is the right-inverse of the projection context p , satisfying $p[\bar{p}[t]] =_\pi t$, defined as:

$$\begin{aligned} \bar{[]} &= [] \\ \bar{p[\pi_1 []]} &= \langle \bar{p}, \Omega \rangle \\ \bar{p[\pi_2 []]} &= \langle \Omega, \bar{p} \rangle \end{aligned}$$

The definitions of discrimination contexts, Δ , and the Böhm-out relation, \approx , are unchanged and the structure of the proof of the Böhm-out Lemma is the same as for the pure λ -calculus.

Theorem 10 (Full abstraction). *Extensional hnf bisimilarity coincides with solvable equivalence for the λ FP-calculus.*

Theorems 5, 9, and 10 entail that Fujita’s CPS transform of the λ -calculus is fully abstract.

7 The standard $\lambda\mu$ -calculus

We now consider the untyped $\lambda\mu$ -calculus [16]. Let a and b range over ‘‘names’’.

$$\text{TERMS } t ::= x \mid \lambda x. t \mid t_1 t_2 \mid \mu a. [b]t$$

Naming and μ -abstraction extend as far to the right as possible, so $\mu a. [b]t_1 t_2 = \mu a. [b](t_1 t_2)$.

We refer to this restricted syntax, where μ -abstraction and naming are coupled in a single syntactic construction $\mu a. [b]t$, as *standard*. It is some times useful to consider named terms as a separate syntactic category so that we can regard μ -abstractions as terms of the form $\mu a. \tau$ where τ is a *named term*:

$$\text{NAMED TERMS } \tau ::= [a]t$$

If ϕ is a syntactic phrase, notation $\phi[a \leftarrow t]$ denotes the capture-free substitution of $[a](t'[a \leftarrow t])$ for every occurrence of a named subterm of the form $[a]t'$ in ϕ . Notation

$\phi[a/b]$ denotes the capture-free substitution of a for all free occurrences of b in ϕ . We write $\text{FN}(\phi)$ for the set of free names in ϕ and we say ϕ is *closed* when $\text{FN}(\phi) = \text{FV}(\phi) = \emptyset$ and ϕ is *variable-closed* if $\text{FV}(\phi) = \emptyset$.

Apart from the β and η equations from the pure λ -calculus, the standard $\lambda\mu$ -calculus has the following additional equations:

$$\begin{aligned} (\mu a. \tau) t &=_{\mu} \mu a. \tau[a \leftarrow t], \text{ if } a \notin \text{FN}(t) & (\mu) \\ \mu a. [b] \mu a'. \tau &=_{\rho} \mu a. \tau[b/a'] & (\rho) \\ \mu a. [a] t &=_{\theta} t, \text{ if } a \notin \text{FN}(t) & (\theta) \end{aligned}$$

Fujita extended his extensional CPS transform to the standard $\lambda\mu$ -calculus by the following additional clauses:

$$\begin{aligned} \text{cps}(\mu a. \tau) &= \lambda a. \text{cps}(\tau) \\ \text{cps}([a]t) &= \text{cps}(t) a \end{aligned}$$

(For convenience, we include the names from the $\lambda\mu$ -calculus in the set of variables in the λFP -calculus.)

We define hnfs for the standard $\lambda\mu$ -calculus as follows:

$$\begin{aligned} \text{HNFS } h &::= g \mid \mu a. [b]g \\ \mu\text{-FREE HNFS } g &::= f \mid \lambda x. h \\ \lambda\text{-FREE HNFS } f &::= x \mid f t \end{aligned}$$

Furthermore, we let γ range over *named hnfs*:

$$\text{NAMED HNFS } \gamma ::= [a]g$$

Definition 4. The syntactic operation $[a]h$ maps a hnf h to a named hnf:

$$\begin{aligned} [a]g &= [a]g \\ [a]\mu b. \gamma &= \gamma[a/b] \end{aligned}$$

We define head evaluation to hnf by rules (E1) through (E4) from Section 3 plus the following additional rules:

$$\begin{aligned} (\text{E5}) \quad & \frac{t_1 \rightsquigarrow \mu a. \gamma \quad \mu a. \gamma[a \leftarrow t_2] \rightsquigarrow h}{t_1 t_2 \rightsquigarrow h} \text{ if } a \notin \text{FN}(t_2) \\ (\text{E6}) \quad & \frac{t \rightsquigarrow h}{\mu a. [b]t \rightsquigarrow \mu a. [b]h} \end{aligned}$$

Let us say that a closed term in the standard $\lambda\mu$ -calculus is solvable iff $t \vec{t} =_{\beta\mu\rho\theta} \lambda x. x$, for some closed arguments \vec{t} .

Lemma 11. *A closed standard $\lambda\mu$ -term is solvable iff it has a hnf.*

Given this definition of solvability, the definition of a semi sensible relation is the same as before and solvable equivalence is the greatest compatible semi sensible relation.

Definition 5. Define the syntactic operations $h[a \leftarrow x]$ on hnfs h as follows:

$$\begin{aligned} f[a \leftarrow x] &= f[a \leftarrow x] \\ (\lambda y. h)[a \leftarrow x] &= \lambda y. h[a \leftarrow x], \text{ if } x \neq y \\ (\mu b. [a']g)[a \leftarrow x] &= \mu b. [a'](g[a \leftarrow x]), \text{ if } a' \neq a \neq b \\ (\mu b. [a]f)[a \leftarrow x] &= \mu b. [a](f[a \leftarrow x] x), \text{ if } a \neq b \\ (\mu b. [a]\lambda y. h)[a \leftarrow x] &= \mu b. [a](h[x/y][a \leftarrow x]), \text{ if } a \neq b \end{aligned}$$

We extend the definition of $M(R)$ from the pure λ -calculus with three additional rules (M6) through (M8) and also define an auxiliary relation $N(R)$ on named hnfs:

$$\begin{aligned} (\text{M6}) \quad & \frac{\gamma N(R) \gamma'}{\mu a. \gamma M(R) \mu a. \gamma'} \\ (\text{M7}) \quad & \frac{\gamma N(R) [a]g}{\mu a. \gamma M(R) g} \text{ if } a \notin \text{FN}(g) \\ (\text{M8}) \quad & \frac{[a]g N(R) \gamma}{g M(R) \mu a. \gamma} \text{ if } a \notin \text{FN}(g) \\ (\text{N1}) \quad & \frac{f M(R) f'}{[a]f N(R) [a]f'} \\ (\text{N2}) \quad & \frac{[a](h[a \leftarrow x]) N(R) [a](h'[a \leftarrow x])}{[a]\lambda x. h N(R) [a]\lambda x. h'} \\ (\text{N3}) \quad & \frac{[a](h[a \leftarrow x]) N(R) [a](f[a \leftarrow x] x)}{[a]\lambda x. h N(R) [a]f} \\ (\text{N4}) \quad & \frac{[a](f[a \leftarrow x] x) N(R) [a](h[a \leftarrow x])}{[a]f N(R) [a]\lambda x. h} \end{aligned}$$

For illustration, observe how $x M(\{(y, y)\}) h$, where $h = \mu a. [a]\lambda y. \mu b. [a]x$, and compare to:

$$x =_{\rho\theta} \mu a. [a]\mu b. [a]x =_{\eta} \mu a. [a]\lambda y. (\mu b. [a]x) y =_{\mu} h.$$

Given the definitions of hnfs, evaluation, and $M(R)$ for the standard $\lambda\mu$ -calculus, the definition of $B(R)$, hnf bisimulation up to η , and extensional hnf bisimilarity are unchanged.

Extensional hnf bisimilarity for the standard $\lambda\mu$ -calculus is a semi sensible equivalence relation that includes the Ω equation (for the new definition of solvability) and satisfies the extensionality property *ext*. It includes extensional hnf bisimilarity for the pure λ -calculus and the β , η , μ , ρ , and θ equations.

The CPS Translation Theorem from the pure λ -calculus can be extended to the standard $\lambda\mu$ -calculus.

Theorem 12 (CPS Translation). *$t \approx t'$ in the standard $\lambda\mu$ -calculus iff $\text{cps}(t) \approx \text{cps}(t')$ in the λFP -calculus.*

Corollary 13. *Extensional hnf bisimilarity is a congruence for the standard $\lambda\mu$ -calculus.*

Full abstraction fails for extensional hnf bisimilarity and for Fujita’s CPS transform of the standard $\lambda\mu$ -calculus, as witnessed by David and Py’s counter-example to the Separation Property:

$$\begin{aligned} 0 &= \lambda x. \lambda y. y \\ 1 &= \lambda x. \lambda y. x \\ U_0 &= \mu b. [a]0 \\ W &= \lambda x. \mu a. [a]x (\mu b. [a]x U_0 y) U_0 \end{aligned}$$

David and Py [6] showed that $W[0/y]$ is solvably equivalent to $W[1/y]$, via a context lemma. However, $\mu b. [a]x U_0 0$ is not extensional hnf bisimilar (nor solvably equivalent) to $\mu b. [a]x U_0 1$ and therefore $W[0/y]$ is not extensional hnf bisimilar to $W[1/y]$.

Saurin [20] recovered the Separation Property for the $\lambda\mu$ -calculus by considering an extended syntax. This suggests that full abstraction is attainable for the extended $\lambda\mu$ -calculus. In the next section we extend the definitions of head normal forms, head evaluation, solvable equivalence, and extensional hnf bisimulation to the extended $\lambda\mu$ -calculus and prove full abstraction by adapting the Böhm-out technique to the extended $\lambda\mu$ -calculus.

8 The extended $\lambda\mu$ -calculus

Saurin extended the standard $\lambda\mu$ -calculus by relaxing the syntactic coupling between μ -abstraction to naming:

$$\text{TERMS } t ::= x \mid \lambda x. t \mid t_1 t_2 \mid \mu a. t \mid [a]t$$

and relaxed the μ and ρ equations correspondingly:

$$\begin{aligned} (\mu a. t_1) t_2 &=_{\mu} \mu a. t_1 [a \leftarrow t_2], \text{ if } a \notin \text{FN}(t_2) & (\mu) \\ [a] \mu b. t &=_{\rho} t [a/b] & (\rho) \end{aligned}$$

The β , η , and θ equations remain the same as in the standard $\lambda\mu$ -calculus.

In the sequel “the $\lambda\mu$ -calculus” and “ $\lambda\mu$ -terms” refer to the extended syntax.

The evaluation relation below is based on the β , μ , and ρ equations plus the equations:

$$\begin{aligned} [a_1][a_2]\lambda x. t &=_{\kappa_1} [a_2]\lambda x. \mu b. [a_1][b]t, & (\kappa_1) \\ &\text{if } a_1 \neq b \notin \text{FN}(t) \\ ([a]\lambda x. t_1) t_2 &=_{\kappa_2} [a]\lambda x. \mu b. ([b]t_1) t_2, & (\kappa_2) \\ &\text{if } x \notin \text{FV}(t_2) \text{ and } b \notin \text{FN}(t_1) \cup \text{FN}(t_2) \end{aligned}$$

which can be derived from the β , η , μ , and ρ equations.

We extend Fujita’s CPS transform as follows:

$$\begin{aligned} \text{cps}(\mu a. t) &= \lambda a. \text{cps}(t) \\ \text{cps}([b]t) &= \lambda a. \text{cps}(t) b a \end{aligned}$$

where a is a fresh variable that does not occur free in the source terms. (The added η redex in the transform of named terms is analogous to the η redex in the variable case.)

We define the hnfs for the $\lambda\mu$ -calculus as follows:

$$\begin{aligned} \text{HNFS } h &::= g \mid \mu a. h \\ \mu\text{-FREE HNFS } g &::= f \mid \lambda x. h \mid [a]\lambda x. h \\ \lambda\text{-FREE HNFS } f &::= x \mid f t \mid [a]f \end{aligned}$$

Definition 6. Now the syntactic operations $[a]h$ and $h[a \leftarrow x]$ map hnfs h to hnfs as follows:

$$\begin{aligned} [a]f &= [a]f \\ [a]\lambda x. h &= [a]\lambda x. h \\ [a][a_2]\lambda x. h &= [a_2]\lambda x. \mu b. [a]([b]h), \text{ if } a \neq b \notin \text{FN}(h) \\ [a]\mu b. h &= h[a/b] \\ f[a \leftarrow x] &= f[a \leftarrow x] \\ (\lambda y. h)[a \leftarrow x] &= \lambda y. h[a \leftarrow x], \text{ if } x \neq y \\ ([b]\lambda y. h)[a \leftarrow x] &= [b]\lambda y. h[a \leftarrow x], \text{ if } a \neq b, x \neq y \\ ([a]\lambda y. h)[a \leftarrow x] &= [a](h[x/y][a \leftarrow x]) \\ (\mu b. h)[a \leftarrow x] &= \mu b. h[a \leftarrow x], \text{ if } a \neq b \end{aligned}$$

We extend head evaluation to the extended $\lambda\mu$ -calculus by replacing the rules (E5) and (E6) from the standard $\lambda\mu$ -calculus by the following rules:

$$\begin{aligned} \text{(E5)} \quad &\frac{t_1 \rightsquigarrow \mu a. h_1 \quad h_1[a \leftarrow t_2] \rightsquigarrow h}{t_1 t_2 \rightsquigarrow \mu a. h} \text{ if } a \notin \text{FN}(t_2) \\ \text{(E6)} \quad &\frac{t_1 \rightsquigarrow [a]\lambda x. h_1 \quad ([b]h_1) t_2 \rightsquigarrow h}{t_1 t_2 \rightsquigarrow [a]\lambda x. \mu b. h} \\ &\text{if } x \notin \text{FV}(t_2), b \notin \text{FN}(t_1) \cup \text{FN}(t_2) \\ \text{(E7)} \quad &\frac{t \rightsquigarrow h}{\mu a. t \rightsquigarrow \mu a. h} \\ \text{(E8)} \quad &\frac{t \rightsquigarrow h}{[a]t \rightsquigarrow [a]h} \end{aligned}$$

Let a closed $\lambda\mu$ -term t be solvable iff

$$([a] \dots ([a]t \vec{t}_0) \vec{t}_1 \dots) \vec{t}_m =_{\beta\mu\rho} \lambda x. x$$

for some name a and closed arguments $\vec{t}_0, \vec{t}_1, \dots, \vec{t}_m$.

Lemma 14. *A closed $\lambda\mu$ -term is solvable iff it has a hnf.*

As before, a relation R on $\lambda\mu$ -terms is semi sensible iff, whenever R relates two closed terms t and t' , t is solvable iff t' is solvable. Solvable equivalence is the greatest semi sensible relation.

If we revisit David and Py's example from Section 7, we observe that $W^{[0/y]}$ and $W^{[1/y]}$ are not solvably equivalent in the $\lambda\mu$ -calculus, because taking

$$C = \mu a. ([a][\] \mu b. \lambda z. [b]z) 1 0$$

we get $C[W] \rightsquigarrow \mu a. [a]y$ and therefore $C[W^{[0/y]}] \Omega$ has hnf $\mu a. [a]\lambda y. y$ whereas $C[W^{[1/y]}] \Omega$ has no hnf.

We extend the definition of $M(R)$ from the pure λ -calculus with the additional rules:

$$(M6) \frac{h M(R) h'}{\mu a. h M(R) \mu a. h'}$$

$$(M7) \frac{h M(R) [a]g}{\mu a. h M(R) g} \text{ if } a \notin \text{FN}(g)$$

$$(M8) \frac{[a]g M(R) h}{g M(R) \mu a. h} \text{ if } a \notin \text{FN}(g)$$

$$(M9) \frac{f M(R) f'}{[a]f M(R) [a]f'}$$

$$(M10) \frac{[a](h[a \leftarrow x]) M(R) g[a \leftarrow x]}{[a]\lambda x. h M(R) g} \text{ if } x \notin \text{FV}(g)$$

$$(M11) \frac{g[a \leftarrow x] M(R) [a](h[a \leftarrow x])}{g M(R) [a]\lambda x. h} \text{ if } x \notin \text{FV}(g)$$

Given the definitions of hnfs, evaluation, and $M(R)$ for the $\lambda\mu$ -calculus, the definition of $B(R)$, hnf bisimulation up to η , and extensional hnf bisimilarity are the same as for the pure λ -calculus.

Extensional hnf bisimilarity for the extended $\lambda\mu$ -calculus includes extensional hnf bisimilarity for the standard $\lambda\mu$ -calculus (which in turn includes extensional hnf bisimilarity for the pure λ -calculus). Moreover, it is a semi sensible equivalence relation and includes the Ω equation (for the new definition of solvability) and satisfies the extensionality property *ext*.

The CPS Translation Theorem can be extended to the extended $\lambda\mu$ -calculus:

Theorem 15 (CPS Translation). $t \approx t'$ in the extended $\lambda\mu$ -calculus iff $\text{cps}(t) \approx \text{cps}(t')$ in the λFP -calculus.

Corollary 16. *Extensional hnf bisimilarity is a congruence for the extended $\lambda\mu$ -calculus.*

We now show that extensional hnf bisimilarity is fully abstract with respect to solvable equivalence for the $\lambda\mu$ -calculus. The proof extends our co-inductive proof from Section 6 with a $\lambda\mu$ -variant of the Böhm-out technique.

We use the “sub-term selectors” from Saurin's proof of the Separation Property for the $\lambda\mu$ -calculus but we eschew the “parametric pairs” and “fst-transform”.

Let $\Lambda\mu$ be the set of all $\lambda\mu$ -terms. Two variable-closed hnfs h and h' are equivalent, $h \sim h'$, iff $h M(\Lambda\mu \times \Lambda\mu) h'$. For arbitrary variable-closed $\lambda\mu$ -terms t and t' , $t \sim t'$ iff $t B(\Lambda\mu \times \Lambda\mu) t'$.

Lemma 17. $t \sim t'$, for all variable-closed solvably equivalent $\lambda\mu$ -terms t and t' .

We define the μ -permutator \mathbf{Q}_m , for $m \geq 0$, to be the $\lambda\mu$ -term:

$$\mathbf{Q}_m = \mu a_1. \dots \mu a_m. \lambda x. [a_m] \dots [a_1]x$$

We define the *sub-term selector* $\Phi_{i,j}^{m,n}$, for $i \geq 1, j \geq 1, m \geq 0, n \geq 0$, and $(m+1, n) \geq (i, j)$ lexicographically (that is, $m \geq i$ or $m+1 = i$ and $j \geq n$), to be the $\lambda\mu$ -term:

$$\Phi_{i,j}^{m,n} = \mu a_1 \dots \mu a_{i-1}. \lambda x_1 \dots \lambda x_j. \\ \mu a_i \dots \mu a_m. \lambda y_1 \dots \lambda y_n. x_j, \text{ if } m \geq i$$

$$\Phi_{m+1,j}^{m,n} = \mu a_1 \dots \mu a_m. \lambda y_1 \dots \lambda y_n. y_j, \text{ if } n \geq j$$

We call μ -permutators and sub-term selectors discriminator terms and we define discrimination contexts as follows:

$$\text{DISCRIMINATORS } \delta ::= \mathbf{Q}_m \mid \Phi_{i,j}^{m,n}$$

DISCRIMINATION CONTEXTS

$$\Delta ::= [\] \mid \Delta \delta \mid [a]\Delta \mid [a](\mu a. \Delta) \delta$$

We define the Böhm-out relation \approx for variable-closed $\lambda\mu$ -terms t and t' as follows:

$$t \approx t' \text{ iff } \forall \Delta. \Delta[t] \sim \Delta[t']$$

On open $\lambda\mu$ -terms $\approx = \bigcup_{n < \omega} \approx_n$, where $t \approx_n t'$ iff

$$\forall \{x_1, \dots, x_m\} \supseteq \text{FV}(t) \cup \text{FV}(t'). \forall n_1, \dots, n_m \geq n. \\ t[\mathbf{Q}_{n_1/x_1}] \dots [\mathbf{Q}_{n_m/x_m}] \approx t'[\mathbf{Q}_{n_1/x_1}] \dots [\mathbf{Q}_{n_m/x_m}].$$

Lemma 18 (Böhm-out). \approx is a hnf bisimulation up to η .

The structure of the proof is the same as in the proof of the Böhm-out Lemma for the pure λ -calculus

Theorem 19 (Full abstraction). *Extensional hnf bisimilarity coincides with solvable equivalence for the extended $\lambda\mu$ -calculus.*

Theorems 15, 19, and 10 entail that the CPS transform of the extended $\lambda\mu$ -calculus is fully abstract.

Acknowledgements Kristian Støvring pointed out errors in drafts of this paper and helped me find literature on the λ -calculus with surjective pairing. I thank the referees for helpful comments and Alexis Saurin and Paul Blain Levy for discussions about this work.

References

- [1] S. Abramsky. The lazy lambda calculus. In D. Turner, editor, *Research Topics in Functional Programming*, pages 65–116. Addison-Wesley, 1990.
- [2] H. P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics*. Number 103 in Studies in Logic and the Foundations of Mathematics. North-Holland, revised edition, 1984.
- [3] C. Böhm. Alcune proprietà delle forme $\beta\eta$ -normali nel λK -calculus. Pubblicazioni 696, Istituto per le Applicazioni del Calcolo, Roma, 1968.
- [4] G. Boudol. On the semantics of the call-by-name CPS transform. *Theoretical Computer Science*, 234:309–321, 2000.
- [5] R. David. Computing with Böhm trees. *Fundamenta Informaticae*, 45:53–77, 2001.
- [6] R. David and W. Py. $\lambda\mu$ -calculus and Böhm’s theorem. *Journal of Symbolic Logic*, 66(1):407–413, 2001.
- [7] K. Fujita. A sound and complete CPS-translation for $\lambda\mu$ -calculus. In *TLCA*, volume 2701 of *Lecture Notes in Computer Science*, pages 120–134. Springer-Verlag, 2003.
- [8] J. Hatcliff and O. Danvy. Thunks and the λ -calculus. *Journal of Functional Programming*, 7(3):303–319, 1997.
- [9] M. Hofmann and T. Streicher. Continuation models are universal for $\lambda\mu$ -calculus. In *Proc. 12th Annual IEEE Symposium on Logic in Computer Science*, pages 387–395, 1997.
- [10] D. J. Howe. Proving congruence of bisimulation in functional programming languages. *Information and Computation*, 124(2):103–112, 1996.
- [11] J. M. E. Hyland. A syntactic characterisation of the equality in some models for the lambda calculus. *Journal of the London Mathematical Society*, 12(3):361–370, 1976.
- [12] S. B. Lassen. Bisimulation in untyped lambda calculus: Böhm trees and bisimulation up to context. In *MFPS XV*, volume 20 of *Electronic Notes in Theoretical Computer Science*, pages 346–374. Elsevier, 1999.
- [13] S. B. Lassen. Eager normal form bisimulation. In *Proc. 20th Annual IEEE Symposium on Logic in Computer Science*, pages 345–354, 2005.
- [14] S. B. Lassen. Normal form simulation for McCarthy’s amb. In *MFPS XXI*, volume 155 of *Electronic Notes in Theoretical Computer Science*, pages 445–465. Elsevier, 2005.
- [15] C.-H. L. Ong and C. A. Stewart. A curry-howard foundation for functional computation with control. In *Proc. 24th ACM Symposium on Principles of Programming Languages*, 1997.
- [16] M. Parigot. $\lambda\mu$ -calculus: An algorithmic interpretation of classical natural deduction. In A. Voronkov, editor, *Proceedings Intl. Conf. on Logic Programming and Automated Reasoning, LPAR’92, St Petersburg*, volume 624 of *Lecture Notes in Computer Science*, pages 190–201. Springer-Verlag, 1992.
- [17] G. D. Plotkin. Call-by-name, call-by-value and the λ -calculus. *Theoretical Computer Science*, 1:125–159, 1975.
- [18] G. E. Révész. Categorical combinators with explicit products. *Fundamenta Informaticae*, 22:153–166, 1995.
- [19] D. Sangiorgi. The lazy lambda calculus in a concurrency scenario. *Information and Computation*, 111(1):120–153, 1994.
- [20] A. Saurin. Separation with streams in the $\lambda\mu$ -calculus. In *Proc. 20th Annual IEEE Symposium on Logic in Computer Science*, pages 356–365, 2005.
- [21] K. Støvring. Extending the extensional lambda calculus with surjective pairing is conservative. *Logical Methods in Computer Science*, 2(2:1):1–14, 2006.
- [22] C. P. Wadsworth. The relation between computational and denotational properties for Scott’s D_∞ -models of the lambda-calculus. *SIAM Journal on Computing*, 5(3):488–521, 1976.