Electronic Notes in Theoretical Computer Science 50 No. 2 (2001) – Proc. BOTH 2001
URL: http://www.elsevier.nl/locate/entcs/volume50.html 34 pages

From Böhm's Theorem to Observational Equivalences: an Informal Account [1]

Mariangiola Dezani-Ciancaglini and Elio Giovannetti

Dipartimento di Informatica, Università degli Studi di Torino
corso Svizzera 185, 10149 Torino, Italia,
e-mail: {dezani,elio}@di.unito.it

Abstract

There are essentially two ways of looking at the computational behaviours of λ-terms. One consists in putting the term within a context (possibly of λ-calculus extensions) and observing some properties (typically termination). The other consists in reducing the term until some meaningful information is obtained: this naturally leads to a tree representation of the information implicitly contained in the original term. The paper is an informal overview of the role played by Böhm's Theorem in these observations of terms.

Dedicated to Corrado Böhm
on the occasion of the EATCS Distinguished Service Award

1 Introduction: from böhming out to observing λ-terms

Böhm's Theorem, in its original formulation [5], states that if M and N are two distinct $\beta\eta$-normal forms, then there is a context $C[\;]$ such that $C[M] \to_\beta^* x$ and $C[N] \to_\beta^* y$, where x, y are arbitrary distinct variables (the opposite implication being obvious!). If $C[\;]$ is such a context, then a context like $(\lambda xy.C[\;])\mathbf{I}\Omega$, where $\mathbf{I} \equiv \lambda x.x$ and $\Omega \equiv (\lambda x.xx)(\lambda x.xx)$, is reducible to normal form when receiving M, and diverges when receiving N. The theorem can therefore be rephrased as stating that, given two distinct $\beta\eta$-normal forms, there is a context $C[\;]$ such that $C[M]$ has a normal form (i.e., converges to a *value*) while $C[N]$ is nonterminating.

In the same year as Böhm's Theorem (1968), Morris [24] for the first time defined a notion of *observational* or *contextual equivalence*, which was going to have such important developments in more recent years, particularly in the domain of interactive concurrent computing: two terms were defined equivalent if, whenever they are put in a same context, either they both make it

[1] Partially supported by MURST Cofin '00 AITCFA and MCTAAP II Projects.

reducible to a normal form (henceforth occasionally abbreviated to *nf*), or they both make it diverge, i.e.,

$$\forall C[\,].C[M] \text{ has a nf} \Longleftrightarrow C[N] \text{ has a nf.}$$

More generally, two terms (two programs, two processes, two computations, etc.) may be considered equivalent if, when observed from the outside, they exhibit the same behaviour, i.e., if whenever they are put in a same environment they give rise to the same observations. Of course, for nonterminating computations the equivalence can only be refuted, if at some point they behave differently from one another, but never verified.

One can define different contextual equivalences depending on the kind of context used and the kind of observation performed, and indeed many of them have been introduced over the years.

Morris' equivalence[2] so corresponds to the natural choice of the pure λ-calculus itself as an environment, and the context's ordinary convergence/divergence as an observation (of course, in this case the seemingly binary observation actually consists of an infinity of observing acts, and it is itself only semi-decidable). Böhm's Theorem can then be viewed as stating that such an observational equivalence coincides, for *normalizable* terms, with $\beta\eta$-convertibility.

The paramount historical importance of Böhm's Theorem lies however in the fact, already stressed by the author in the original paper and afterwards pointed out by various researchers, that its proof is constructive, since it consists of an algorithm that, given two distinct $\beta\eta$-normal forms, builds a discriminating context; an elegant implementation in CAML is given in [14].

As a matter of fact, a more specific formulation of the theorem is the following: two closed $\beta\eta$-normal forms M and N are distinct iff there exist closed terms L_1, L_2, ..., L_n, with $n \geq 0$, such that

$$ML_1 \ldots L_n xy \to_\beta^* x \text{ and } NL_1 \ldots L_n xy \to_\beta^* y.$$

The extension to open terms is obvious: two possibly open $\beta\eta$-normal forms M and N are distinct iff their closures are distinct, therefore iff there is a context $C[\,]$ of the form

$$(\lambda x_1 \ldots x_m.[\,])L_1 \ldots L_n xy$$

such that $C[M] \to_\beta^* x$ and $C[N] \to_\beta^* y$. To construct the separating context, Böhm introduced the so-called Böhm-out technique, based on an analysis of the term structure that Barendregt [3] later called, extending it to diverging terms, *Böhm tree.* The starting point is that β-normal forms satisfy an inductive property, which can be read as an inductive definition. Recall that a *head normal form* (henceforth occasionally abbreviated as *hnf*) is a term of the

[2] Actually, Morris defined four different equivalences, see [3].

form $\lambda x_1 \ldots x_n.xM_1 \ldots M_m$ with $n, m \geq 0$, where the *head variable* x is either free or identical to one of the x_i; β-normal forms may then be inductively defined as follows:

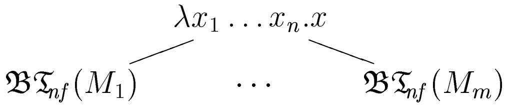
- $\lambda x_1 x_2 \ldots x_n.x$, where $n \geq 0$, is a (head normal form that also is) a β-normal form;
- a head normal form $\lambda x_1 x_2 \ldots x_n.xM_1 M_2 \ldots M_m$ where $n \geq 0, m \geq 1$, is a normal form if M_1, M_2, …, M_m are β-normal forms.

The Böhm tree of a β-normal form merely is the tree representation of this inductive structure of nested head normal forms.

Definition 1.1 Böhm trees of β-normal forms.

(i) $\mathfrak{B}\mathfrak{T}_{nf}(\lambda x_1 \ldots x_n.x) = \lambda x_1 \ldots x_n.x$ (for $n \geq 0$);

(ii) $\mathfrak{B}\mathfrak{T}_{nf}(\lambda x_1 \ldots x_n.xM_1 \ldots M_m) =$


$\lambda x_1 \ldots x_n.x$
$\mathfrak{B}\mathfrak{T}_{nf}(M_1)$
$\cdots$
$\mathfrak{B}\mathfrak{T}_{nf}(M_m)$

(for $n \geq 0, m \geq 1$)

The Böhm-out technique consists in building a sequence of suitable terms that, when fed as arguments to two different normal forms, brings to the top (possibly an instance of) a subterm by which these differ. This is achieved by successively binding the head variables to the appropriate selectors so as to go down the path from the root to the desired node. For example, let M and N be the two terms below, represented by the two trees in Fig. 1:

$$M = \lambda x.xt_1(\lambda yz.z(\underline{\lambda u.u})t_2)t_3 \qquad N = \lambda x.xt_1(\lambda yz.z(\underline{\lambda uv.v})t_2)t_3.$$

Then the basic discriminating context is $[\ ]U_2^3 L U_1^2$, which extracts the two underlined subterms. The U_i^ns are the selectors[3] of the i-th argument among n, so that starting from the root the second child among three is selected, and then the first among two; L is any term, only needed for saturating the abstraction on the non-head variable y. For a discrimination based on the convergence/divergence property, it is sufficient to append a diverging term Ω, which, when fed as an argument to the two non-matching subterms, makes one of them reduce to normal form and the other diverge; for a discrimination based on two distinct variables it is sufficient to complete the context as $[\ ]U_2^3 L U_1^2 (U_1^2 x)y$.

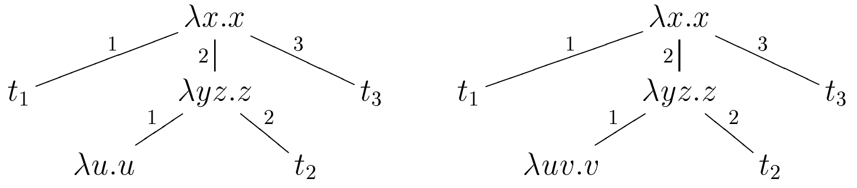


Fig. 1. Two Böhm trees for a simple Böhm-out

[3] by definition, U_i^n is the term $\lambda x_1 \ldots x_n.x_i$, see [5].

The problem is when along the path from the root to the discriminating node a same head variable occurs more than once, and different children are to be selected at different occurrences, like for example in two terms of the form $\lambda x.x(\lambda yz.xzt_1)$ and $\lambda x.x(\lambda yz.xzt_2)$, where t_1 and t_2 are the two non-matching subterms. The solution consists in first replacing different occurrences of the same variable with different bound variables (through the application to suitable combinators which add abstractions) and then, as in the simple case, replacing each variable with the needed selector.

Since the observation environment only consists of arguments to be passed to the observed term (apart from top-level abstractions, in case of open terms) and the Böhm-out technique recursively applies this principle to subterms, the method is obviously unable to discriminate between η-convertible forms: the terms M and $\lambda x.Mx$, when applied to an argument, behave the same; this is the reason why the theorem only regards $\beta\eta$-normal forms.

Böhm's theorem, through the use of the Böhm-out technique, thus also established the primigenious and simplest form of the *context lemma*, which allows quantification over all contexts to be replaced, in the definition of observational equivalences, by quantification restricted to *head contexts*, which are contexts with only one hole occurrence, situated in head position; i.e., contexts of the form $(\lambda x_1 \ldots x_n.[\,])M_1 \ldots M_m$, with $n, m \geq 0$.

Lemma 1.2 *Context Lemma for Normalizable Terms.*
If M and N are two normalizable terms, $\forall C[\,].C[M]$ has nf $\Longleftrightarrow C[N]$ has nf iff $\forall C_H[\,].C_H[M]$ has nf $\Longleftrightarrow C_H[N]$ has nf, where the $C_H[\,]$'s are head contexts.

The reason here is totally obvious: taking the lemma's contrapositive, if there is a generic context discriminating two normal forms, these cannot be identical, so there must also be a head context that performs the separation (in the other direction, a discriminating head context just is a context!).

If M and N are closed terms, the quantification over contexts may be further restricted – as previously observed – to *applicative contexts*, i.e., contexts of the form $[\,]M_1 \ldots M_m$.

This, in turn, allows observational equivalences to be defined in a coinductive style (see, for example, [16]) which, though not much meaningful in this case where the equivalence is between normal forms and the calculus is sequential, is however the one used in the study of concurrent and interactive systems. Let the notation $M =_{\mathcal{I}} N$ indicate the fact that M and N "in isolation behave the same", i.e., that either both M and N reduce to values (that is to nfs, or to hnfs, etc.) or both do not, and let the corresponding equivalence w.r.t. applicative contexts be denoted by the generic symbol $\simeq$:

$$M \simeq N \text{ iff, by definition, } \forall m \geq 0.\forall L_1 \ldots L_m.(ML_1 \ldots L_m =_{\mathcal{I}} NL_1 \ldots L_m).$$

Then the following holds:

$$M \simeq N \Longleftrightarrow M =_{\mathcal{I}} N \text{ and } \forall L.ML \simeq NL.$$

In fact, if $M \simeq N$, then if we take the empty context we have $M =_{\mathcal{I}} N$, and by trivially considering the application associativity we have $ML \simeq NL$ for all L; also the opposite direction obviously holds, hence the property stated above; it may be assumed as an alternative definition of observational equivalence, which is then usually called an *applicative bisimulation* or *bisimilarity*, since it was derived from adapting to λ-calculus [1] and functional programming the notions of bisimulation and bisimilarity originally introduced for concurrent processes [23,25].

More precisely, any equivalence $\simeq$ for which the left-to-right implication holds is called an applicative bisimulation; the greatest bisimulation, i.e., the one for which the reverse implication also holds, is called (applicative) bisimilarity [29].

The rest of paper is organized as follows. In Sect. 2 we examine the behaviours of terms within pure λ-calculus contexts, w.r.t. three different choices of what is to be assumed as the set of values. On the other hand, Sect. 3 discusses λ-calculus extensions allowing to discriminate terms exactly in the same manner as well-known tree representations of terms. We draw some conclusions in Sect. 4.

2 Observing pure λ-terms in pure λ-calculus

If we take two normalizable terms not in normal form, which could represent two programs still to be run, we might imagine to observe their behaviours by interactively creating, possibly with backtracking, a context that böhms out the subterms being computed. The proof of Böhm's theorem can thus be considered as the first prototypical example of a refutation procedure for observational equivalence.

The natural next step consisted in applying the same kind of technique to obtain a characterization of the observational equivalence for the class of all terms (i.e., also including those without normal forms), as in Morris' definition.

The crucial choice is that of the set of values: we will consider in the following three natural choices, i.e., the sets of normal forms, of head normal forms, and of weak head normal forms.

2.1 Normal forms as values

Since for normalizable terms the observational equivalence amounts to the coincidence of $\beta\eta$-normal forms, for generic terms one may expect that it should amount to the coincidence of some kind of generalized, possibly infinite, normal forms. Observe that the above reported inductive definition of β-normal form by means of the head normal form, if read coinductively, becomes the definition of a notion of possibly infinite β-normal form, with a possibly infinite Böhm tree representation.

The inductive definition exactly corresponds to the way the normal form is computed by the leftmost-outermost strategy: the term is first reduced to its head normal form, then the normal forms of its subterms are recursively computed. The coinductive definition corresponds to the way the possibly infinite normal form is gradually built by the same strategy in a possibly infinite approximating computation, like an irrational number is built by its successive rational approximations. It is therefore natural to define the notions of *approximate* or *partial* term [32], and correspondingly of approximate or partial Böhm tree, using the symbol $\perp$ for the subterms not yet in head normal form, i.e., the subterms yet to be computed. Like Böhm trees proper, the approximate trees were introduced by [3], with the name of Böhm-like trees.

Definition 2.1 Approximate Böhm trees.

(i) if $M = \lambda x_1 \ldots x_n.xM_1 \ldots M_m$ (with $n, m \geq 0$),

$$\text{then } \mathfrak{ABT}(M) = \begin{array}{ccc} & \lambda x_1 \ldots x_n.x & \\ \swarrow & & \searrow \\ \mathfrak{ABT}(M_1) & \cdots & \mathfrak{ABT}(M_m) \end{array}$$

(ii) otherwise (i.e., if M is not in head normal form) $\mathfrak{ABT}(M) = \perp$.

The obvious approximation partial order may be defined, and the infinite normal forms obtained by the coinductive reading of 1.1 are the limits of monotone (increasing) sequences of partial terms (approximate trees), which will then be called *approximants* of the limit.

The converse does not hold: coinductive infinite normal forms do not contain any occurrences of $\perp$, since this does not occur in the definition; on the contrary, owing to the phenomenon of unsolvability, i.e., the existence of terms without head normal forms, a computation may generate a sequence of partial terms (partial trees) where some $\perp$-labelled nodes do not expand any further, and cannot therefore be eliminated in the limit. A term that reduces to a term with unsolvable subterms, i.e., to a tree with $\perp$-stuck nodes, does not possess a normal form in the above sense, neither finite nor infinite; if we want to give it a meaning, we are naturally led to complete the space by considering as limits, i.e., as generalized normal forms, also (finite and infinite) terms (or trees) containing the constant $\perp$ as a representation of the unsolvable.

We will see in the following that finer notions of an infinite normal form may be introduced; remark, however, that one cannot take the extreme of assuming as definition of an infinite term (and thus of an infinite normal form) the mere coinductive reading of the ordinary definition of term, as this would lead – with any reasonable definition of limit – to the loss of the confluence property for β-reduction [18]. Take for example [4] the term HH, with $H \equiv \lambda x.\mathbf{I}(xx)$, which may be thought as resulting – through one step of β-reduction – from the application of the fixed point combinator $\mathbf{Y} \equiv \lambda f.(\lambda x.f(xx))(\lambda x.f(xx))$ to the identity: having the property that $HH \rightarrow_\beta \mathbf{I}(HH)$, but also that (in two steps) $HH \rightarrow_\beta^* \Omega$, it generates the two infinite reduction sequences

$$HH \to_\beta^* \Omega \to_\beta \Omega \to_\beta \cdots$$
$$HH \to_\beta \mathbf{I}(HH) \to_\beta \mathbf{I}(\mathbf{I}(HH)) \to_\beta \mathbf{I}(\mathbf{I}(\mathbf{I}(HH))) \to_\beta \cdots$$

which, though joinable at every finite step, converge to two different limits, namely Ω and $(\mathbf{I}(\mathbf{I}(\mathbf{I}\ldots)))$. With Def. 2.1, on the contrary, all the approximants of HH are $\perp$, so the one limit is trivially $\perp$; more generally, the uniqueness of limits is preserved, and with it the uniqueness of meaning (if we adhere to a "syntactic" view of semantics).

It is therefore natural to assume as definition of a kind of generalized β-normal form, existing for every term, the one emerging from the above considerations. To keep distinct the newly defined class of syntactic-semantic objects – possibly infinite and possibly containing the pseudo-finite $\perp$ – from the ordinary finite terms, the definition is formally given in terms of trees: the standard definition of Böhm trees. Also observe the subtly different role played by the symbol $\perp$ in Böhm trees, where it denotes the unsolvable, from the one played in the approximants where it denotes the unsolved, i.e., a redex still to be computed; unsolved that keep unsolved forever are unsolvable.

Definition 2.2 Böhm trees.

(i) if $M \to_\beta^* \lambda x_1 \ldots x_n.xM_1 \ldots M_m$ (with $n, m \geq 0$),

then $$\mathfrak{BT}(M) = \begin{array}{ccc} & \lambda x_1 \ldots x_n.x & \\ \swarrow & & \searrow \\ \mathfrak{BT}(M_1) & \cdots & \mathfrak{BT}(M_m) \end{array}$$

(ii) otherwise (i.e., if M does not have a head normal form) $\mathfrak{BT}(M) = \perp$.

The Böhm-out technique may be extended to Böhm trees proper, and Böhm's theorem, characterizing the observational equivalence restricted to normalizable terms, may thus be extended, as we anticipated, to characterize the equivalence in the class of all terms.

We observed that, since the technique is unable to discriminate with respect to applications of the η-rule, Böhm's equivalence equates terms possessing identical (finite) $\beta\eta$-normal forms. Analogously, the unrestricted observational equivalence is unable to discriminate "η-convertible" infinite β-normal forms, or Böhm trees; therefore, while two terms having the same Böhm tree are equivalent, the converse does not hold. Clearly, the equivalence corresponds to some notion of infinite $\beta\eta$-normal form, or Böhm η-tree.

To begin with, the nodes of a Böhm η-tree must be η-head normal forms, i.e., head normal forms that do not contain η-redexes in their top abstractions: a term of the form $\lambda x_1 \ldots x_{n-1}x_n.xM_1M_2 \ldots M_mx_n$, which may be more verbosely but more perspicuously written as $\lambda x_1 \ldots x_{n-1}\lambda x_n.(xM_1M_2 \ldots M_m)x_n$, η-reduces to $\lambda x_1 \ldots x_{n-1}.xM_1M_2 \ldots M_m$. Such reduction may in turn recursively give rise to another η-redex, if it reduces a rightmost child to the rightmost variable of its parent node's abstraction, as in the sequence:

$$\lambda x_1 \dots x_{n-1} x_n . x M_1 M_2 \dots M_m (\lambda y . x_n y) \rightarrow_\eta \lambda x_1 \dots x_{n-1} x_n . x M_1 M_2 \dots M_m x_n$$
$$\rightarrow_\eta \lambda x_1 \dots x_{n-1} . x M_1 M_2 \dots M_m .$$

When dealing with infinite forms, however, taking η-head normal forms is not sufficient [3]; thus, the observational equivalence in the class of all terms does not simply amount, as in the finite case, to $\beta\eta$-convertibility. There may be, in fact, infinite β-reduction sequences where at every finite step the term is not an η-redex, but it becomes an η-redex in the limit; the simplest example, following [3,2], is the term $\lambda y.(P_x\, y)y$, where P_x is a term (containing the free variable x) such that $P_x\, y \rightarrow_\beta x(P_x\, y)$ [4]; it generates the infinite reduction sequence

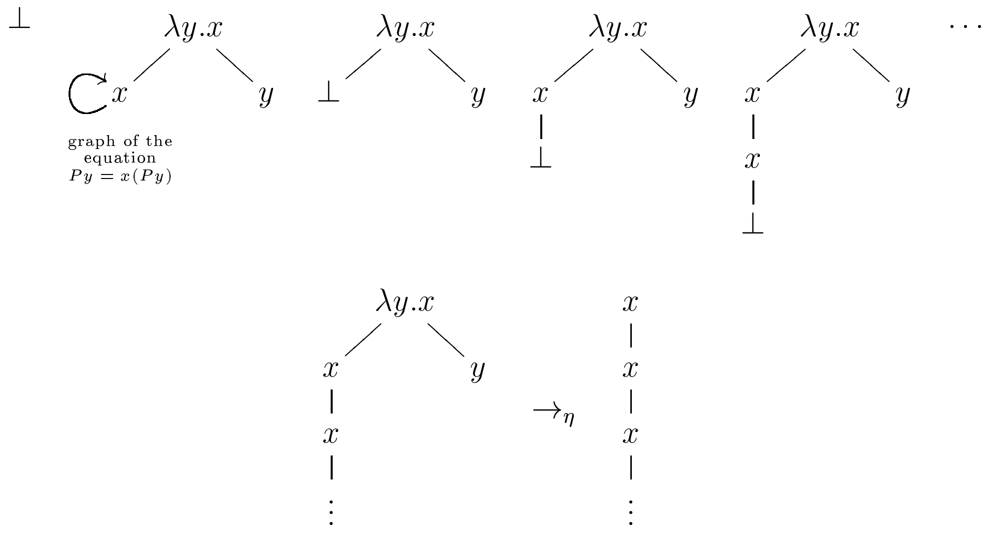


Fig. 2. The term $\lambda y.(P_x\, y)y$: approximants, Böhm tree, Böhm η-tree.

$$\lambda y.(P_x\, y)y \rightarrow_\beta \lambda y.x(P_x\, y)y \rightarrow_\beta \lambda y.x(x(P_x\, y))y \rightarrow_\beta \lambda y.x(x(x(P_x\, y)))y \rightarrow_\beta \dots$$

where at every finite step the term of the form $\lambda y.My$ is not an η-redex since M contains an occurrence of y, but it becomes an η-redex in the limit, where the y disappears behind an infinite number of x's. Correspondingly, its Böhm tree has an infinite branch generated by the graph of the recursive definition of $P_x\, y$, whose all finite approximants do not contain y, as shown in Fig. 2. The η-reduced tree is the representation of the infinite term $xxx\dots$, which coincides with the Böhm tree of any term Q_x satisfying the recursive relation

[4] P_x is easily obtained by translating its recursive definition with the use of the $\mathbf{Y}$ combinator, i.e., $\mathbf{Y}(\lambda py.x(py))$; from which one finally arrives through some reduction steps at $P_x \equiv A_x\, A_x$ where $A_x \equiv \lambda zy.x(zzy)$.

$Q_x \to_\beta xQ_x$: the simplest is $Q_x \equiv D_x D_x$ where $D_x \equiv \lambda z.x(zz)$, represented in Fig. 3. As a consequence, the two terms $\lambda y.(P_x y)y$ and Q_x, though not η- (nor β-) convertible, cannot be discriminated by observing their effects on the context: whatever the sequence of arguments given to their respective closures $\lambda xy.(P_x y)y$ and $\lambda x.Q_x$, i.e., whatever the context $C[\;] \equiv [\;]L_1 L_2 \ldots L_n$, one has:

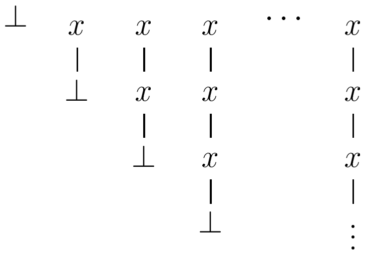


Fig. 3. The term $Q_x \equiv (\lambda z.x(zz))(\lambda z.x(zz))$: approximants and Böhm tree.

$$C[\lambda xy.(P_x y)y] \to_\beta P_{L_1} L_2 L_2 \ldots L_n \to_\beta L_1(P_{L_1} L_2)L_2 \ldots L_n$$

$$C[\lambda x.Q_x] \qquad \to_\beta Q_{L_1} L_2 \ldots L_n \quad \to_\beta L_1 Q_{L_1} L_2 \ldots L_n$$

where $P_{L_1} \equiv P_x[L_1/x], Q_{L_1} \equiv Q_x[L_1/x]$ and moreover

$$P_{L_1} L_2 \to_\beta L_1(P_{L_1} L_2), \quad Q_{L_1} \to_\beta L_1 Q_{L_1}.$$

The two resulting expressions have the same head L_1 which initially determines their behaviours, and the same tail $L_2 \ldots L_n$ of the argument sequence; they only differ in their first argument, which in both cases reproduces L_1 in head position and is therefore unable to differentiate the behaviours. For example, for $L_1 \equiv \mathbf{I}$ both contexts diverge, for $L_1 \equiv U_i^n$, with $1 < i \leq n$, both contexts behave like L_i.

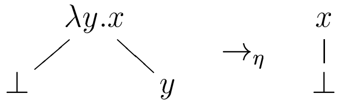


Fig. 4. The term $\lambda y.x(\Omega y)y$: its Böhm tree and Böhm η-tree.

Observe that the η-reduction cannot be performed on the approximants, since one does not know whether $\perp$ "contains" y or not. However, a term that is not η-reducible may happen to have a Böhm tree that is finite but η-reducible, if the subterm where the "forbidden" variable (i.e., the variable concerned by the η-rule) occurs is unsolvable: e.g., the term $\lambda y.x(\Omega y)y$ of Fig. 4 obviously does not η-reduce to $x(\Omega y)$ since this subterm contains y, but its Böhm tree, which in a term-like notation is $\lambda y.x\perp y$, η-reduces to $x\perp$, since the constant $\perp$ has "swallowed" the y in a single node (of course, the fact is that $\perp$ itself is the semantic counterpart of an infinite computation).

As the above considerations suggest, the notion of Böhm η-tree, representing a kind of infinite $\beta\eta$-normal form, may then be defined – again following [2] – as the η-normal form $\eta(T)$ of an ordinary Böhm tree T.

Definition 2.3 η-normal form of a Böhm tree.
The η-normal form $\eta(T)$ of a Böhm tree T is defined by cases as follows:

(i) $\eta(\perp) = \perp$

(ii) $\eta(\lambda x_1 \ldots x_n.x) = \lambda x_1 \ldots x_n.x$

(iii) $$\eta\left(\begin{array}{ccc} & \lambda x_1 \ldots x_n.x & \\ / & & \backslash \\ T_1 & \ldots & T_m \end{array}\right) = \begin{cases} \eta\left(\begin{array}{ccc} & \lambda x_1 \ldots x_{n-1}.x & \\ / & & \backslash \\ T_1 & \ldots & T_{m-1} \end{array}\right) & \begin{array}{l}\text{if } T_m \text{ is finite,} \\ \eta(T_m) = x_n \neq x \\ \text{and } x_n \notin FV(T_i) \\ \text{for } 1 \leq i \leq m{-}1. \end{array} \\ \begin{array}{ccc} & \lambda x_1 \ldots x_n.x & \\ / & & \backslash \\ \eta(T_1) & \ldots & \eta(T_m) \end{array} & \text{otherwise} \end{cases}$$

Definition 2.4 Böhm η-trees.
The Böhm η-tree $\mathfrak{BT}_\eta(M)$ of a term M is defined as:

$$\mathfrak{BT}_\eta(M) = \eta(\mathfrak{BT}(M)).$$

The observational equivalence of two terms in Böhm's and Morris' sense then exactly coincides with the equality of their Böhm η-trees, as was proved by Hyland [15] in 1975. We may express the result more formally by introducing an explicit notation for the equivalence, as follows.

Definition 2.5 Normal observational equivalence.
We say that two terms M and N are *observationally equivalent w.r.t. to normal convergence*, i.e., when the set of values is the set of normal forms, and we write $M \simeq_n N$, iff

$$\forall C[\,]\,.\,C[M] \text{ has nf } \iff C[N] \text{ has nf.}$$

As we will see, this is the finest (i.e., the most discriminating) observational equivalence that can be obtained with a pure λ-calculus context and respects Böhm tree equality.

Theorem 2.6 *(Hyland [15]).*
For any two terms M and N:

$$M \simeq_n N \quad \iff \quad \mathfrak{BT}_\eta(M) = \mathfrak{BT}_\eta(N)$$

i.e., expressed with the contrapositive:

$$\mathfrak{BT}_\eta(M) \neq \mathfrak{BT}_\eta(N) \iff$$
$$\exists C[\,]\,.\,C[M] \text{ has nf } \wedge\ C[N] \text{ has no nf}, \textit{ or the converse.}$$

2.2 Head normal forms as values

In the original Morris' paper [24], cited by [3], four different observational equivalences were defined, and three of them were proved to coincide. In particular, as can be expected, observing whether two closed terms, when put in a context, make it reduce to the *same* $\beta\eta$-normal form, does not add any discriminating power w.r.t. the normal equivalence defined in the previous section.

Another natural equivalence, introduced by Wadsworth [32] in 1976 along with the properties described below, consists in limiting oneself to observing whether the context reduces to a head normal form, thus possibly stopping its computation without waiting for a normal form to finally appear.

Definition 2.7 Head observational equivalence.
We say that two terms M and N are *observationally equivalent w.r.t. to head convergence*, i.e., when the set of values is the set of head normal forms, and we write $M \simeq_h N$, iff

$$\forall C[\,]\,.\,C[M] \text{ has hnf } \iff C[N] \text{ has hnf.}$$

Observe that, owing to the double implication contained in both the equivalence definitions, the fact that one of them implies the other does not trivially follows by boolean logics. It is however easy to prove that the head equivalence cannot discriminate more than the normal equivalence.

This relies upon two basic and well-known facts of λ-calculus. On the one hand, head divergence is stronger than ordinary divergence: a term (and thus a filled context) not possessing a hnf, a fortiori does not possess a nf; moreover, it also diverges when applied to any sequence of arguments, for at every successive reduction step it reduces to a term of the form $\lambda x_1 \ldots x_n.(\lambda z.M)NN_1 \ldots N_m$, with $n, m \geq 0$, where the head position is always occupied by a β-redex and never becomes a head variable. But binding a head variable to an argument is the only means, as already recalled, by which the argument may literally take the lead and affect the term behaviour.

On the other hand, a term reducible to a hnf, even if it does not have a nf, may take an argument and put it in head position, wherefrom it can modify the term itself, e.g., eliminate the divergent subterms.

Now suppose M and N are two terms separated by the head equivalence, i.e., there exists a context $C[\ ]$ such that, say, $C[M]$ reduces to a hnf while $C[N]$ does not; then it is always possible, owing to the two above recalled properties, to build a context $C'[\ ]$ such that $C'[M]$ reduces to a nf while $C'[N]$ does not, thus discriminating w.r.t. the normal equivalence.

In fact, let $C[M]$ be reducible to a hnf $\lambda x_1 \ldots x_n.x_k M_1 \ldots M_m$, where the head variable x_k is one of the $x_1 \ldots x_n$ (if it is not, one can always add to the context an outermost abstraction to bind it), and where some or all of the M_i may be diverging. Then it is sufficient to append to the context $C[\ ]$ a suitable eraser that, when bound to the head variable, cancels all the M_i (if not only the offending ones), or – which is the same thing – a normalizing term (say, a variable) preceded by a suitable selector to pick it up: $C'[\ ] \equiv C[\ ]L_1 \dots L_{k-1}(U_1^{m+1}z)$, where the L_i are arbitrary terms needed to saturate the abstractions, and $U_1^{m+1}z$ is the selector with its argument (at their place the eraser $\lambda x_1 \dots x_m.z$ could have directly been written, saving a β-reduction step).

Thus $C'[M] \to_\beta^* \lambda x_1 \dots x_n.z$, while $C'[N]$ diverges – for the reasons mentioned – since $C[N]$ has no hnf.

The next question is whether the head observational equivalence is *strictly* less discriminating than the normal equivalence and, as the answer is positive, whether the equivalence relation it therefore induces on Böhm η-trees can be independently characterized in a natural way, for example by the property of having the same normal form, for some new kind of tree normal form.

Two terms M and N may be observationally distinct w.r.t. to the normal equivalence ($M \not\simeq_n N$) but undistinguishable w.r.t. to the head equivalence ($M \simeq_h N$) only if there is context $C[\ ]$ such that, say, $C[M]$ has no normal form while $C[N]$ has, and at the same time both $C[M]$ and $C[N]$ have head normal forms (for if they were both without hnf, of course they would both also be non-normalizing).

We may for simplicity and – as we saw – without loss of generality restrict ourselves to closed terms, and therefore to applicative contexts. Then M must be a term such that, for any sequence of arguments $L_1 \dots L_k$, either both expressions $ML_1 \dots L_k$ and $NL_1 \dots L_k$ have hnf or both haven't, but with at least one such sequence having no nf on M, while normalizing on N. This kind of common behaviour of M and N on any arguments requires a sort of η-convertibility between them; but for M to be nonterminating, the conversion must consist of an infinite number of steps, namely an infinite η-expansion of N. An infinite expansion of the form $\lambda \dots y_h y_{h+1} y_{h+2} \dots .N \dots y_{h+2} y_{h+1} y_h \cdots$, either obtained from the interior or from the exterior, will not do, since in either case it could only be produced by a recursion in head position, and so by a term without hnf. The expansion can therefore only be performed, at each step, on the newly introduced variable:

$$N \ {}_\eta\!\leftarrow \lambda y_0.Ny_0 \ {}_\eta\!\leftarrow \lambda y_0.N(\lambda y_1.y_0y_1) \ {}_\eta\!\leftarrow \lambda y_0.N(\lambda y_1.y_0(\lambda y_2 y_1.y_2)) \ {}_\eta\!\leftarrow \dots$$

The expanding transformation Q is thus defined by the recursive relation $Qz \to \lambda y.z(Qy)$, or $Q \to \lambda zy.z(Qy)$; its translation by means of the fixed point combinator gives $\mathbf{Y}(\lambda qzy.z(qy))$ which with the usual steps of β-reduction finally yields $Q \equiv RR$, where $R \equiv \lambda xzy.z(xxy)$.

Using the notation $T \geq_\eta x$ to indicate that the tree T is a finite or infinite (in the above sense) η-expansion of the variable x, and following [3,2] (where the relation $\geq_\eta$ is also formally defined), one may introduce the notion of η_∞-normal form of a Böhm tree, or Böhm η_∞-tree, and state for the head observational equivalence a characterization theorem analogous to 2.6.

Definition 2.8 η_∞**-normal form of a Böhm tree**.
The η_∞-normal form $\eta_\infty(T)$ of a Böhm tree T is defined by cases as follows:

(i) $\eta_\infty(\perp) = \perp$

(ii) $\eta_\infty(\lambda x_1 \ldots x_n.x) = \lambda x_1 \ldots x_n.x$

(iii)
$$\eta_\infty\left(\begin{array}{c}\lambda x_1\ldots x_n.x\\ \diagup \quad \diagdown\\ T_1 \quad \ldots \quad T_m\end{array}\right) = \begin{cases} \eta_\infty\left(\begin{array}{c}\lambda x_1\ldots x_{n-1}.x\\ \diagup\quad\diagdown\\ T_1 \quad\ldots\quad T_{m-1}\end{array}\right) & \begin{array}{l}\text{if } T_m \geq_\eta x_n,\\ x_n \neq x \text{ and}\\ x_n \notin FV(T_i) \text{ for}\\ 1\leq i\leq m{-}1.\end{array}\\ \begin{array}{c}\lambda x_1\ldots x_n.x\\ \diagup\quad\diagdown\\ \eta_\infty(T_1)\quad\ldots\quad\eta_\infty(T_m)\end{array} & \text{otherwise}\end{cases}$$

Curien [8] shows that the η_∞-normal form of a Böhm tree can be obtained by means of a finite number of finite eta-reductions and of a *finite* number of infinite eta-reductions of variables.

Definition 2.9 Böhm η_∞**-trees**.
The Böhm η_∞-tree $\mathfrak{BT}_{\eta_\infty}(M)$ of a term M is defined as:

$$\mathfrak{BT}_{\eta_\infty}(M) = \eta_\infty(\mathfrak{BT}(M)).$$

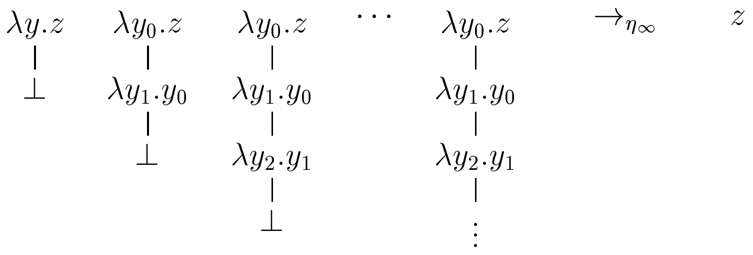


Fig. 5. The term Qz: approximants, Böhm tree, Böhm η_∞-tree.

In Fig. 5 are shown the approximants and the resulting infinite Böhm tree of the term Qz, whose η_∞-normal form is the variable z. The respective closures $\lambda z.Qz$ and $\lambda z.z$, are obviously non-equivalent w.r.t. the normal observational equivalence, for they are trivially discriminated by the empty context (one of them diverges and the other converges). On the other hand, no context can tell them apart if values are head normal forms, as stated in the general form by Wadsworth's result.

Theorem 2.10 *(Wadsworth [32]).*
For any two terms M and N:

$$M \simeq_h N \quad \Longleftrightarrow \quad \mathfrak{BT}_{\eta_\infty}(M) = \mathfrak{BT}_{\eta_\infty}(N)$$

i.e., expressed with the contrapositive:

$$\mathfrak{BT}_{\eta\infty}(M) \neq \mathfrak{BT}_{\eta\infty}(N) \iff$$
$$\exists C[\,]\,.\,C[M] \text{ has hnf } \wedge\ C[N] \text{ has no hnf, } \textit{or the converse.}$$

A short and self-contained proof of the above theorem can be found in [8].

2.3 Weak head normal forms as values

In order to model the implementation of the actual programming languages, where program execution does not include program transformation, in particular to model the implementation of functional programming languages, where the evaluation does not evaluate function bodies, a third kind of normal form was introduced in pure λ-calculus too [26]: the *weak head normal form*, henceforth often abbreviated as *whnf*, which is either a variable applied to (possibly zero!) arguments $xM_1 \ldots M_m$, or an abstraction $\lambda x.M$. The evaluation to weak head normal form of a closed term stops as soon as it reaches an abstraction, that is, a functional value.

Correspondingly, a third kind of observational equivalence may be considered, where the discriminating context behaviour is whether it reaches a weak head normal form or not.

Definition 2.11 Weak head observational equivalence.
We say that two terms M and N are *observationally equivalent w.r.t. to weak head convergence*, i.e., when the set of values is the set of weak head normal forms, and we write $M \simeq_w N$, iff

$$\forall C[\,].C[M] \text{ has whnf } \iff C[N] \text{ has whnf.}$$

Interestingly, such equivalence turns out to be strictly finer than the normal equivalence, and thus the finest of the three so far examined. To begin with, one can show that if two terms are distinct w.r.t. to the normal equivalence, so they are w.r.t. the weak head equivalence.

Let M, N be two such terms, and $C[\,]$ a discriminating context for them; let therefore $M_0 \equiv C[M]$ and $N_0 \equiv C[N]$ be two closed terms such that, say, M_0 has a nf, which of course also is a whnf, while N_0 has no nf. If N_0 has no whnf either, the two terms are distinct w.r.t. to the weak head equivalence too, and no further argument[5] is required.

The only interesting case is when both M_0 and N_0 have whnfs respectively $\lambda x.M'_0$ and $\lambda x.N'_0$, with M'_0 in nf, and N'_0 diverging. The term N'_0 may be an application not reducing to a whnf, or it may in turn reduce to an abstraction whose body is diverging; depending on whether this recursive structure is finite or infinite, there are two possibilities. Either a term not having a whnf may be extracted by application to a sufficiently long sequence of free variables $z_1 \ldots z_n$; these, on the other hand, can be fed to M_0 without altering its normalizing behaviour: thus $M_0 z_1 \ldots z_n$ has whnf and $N_0 z_1 \ldots z_n$ has not. Or $N_0 z_1 \ldots z_n$ always reduces to an abstraction for any finite sequence $z_1 \ldots z_n$,

[5] if the pun is allowed …

and in such case it does so even if some or all of the z_i are replaced by diverging terms L_i; on the other hand M_0, being in (closed) nf, may be transformed, by application to suitable arguments, into any desired term, in particular into one without whnf; thus a context $(\lambda x_1 \ldots x_k.[\ ])L_1 \ldots L_n$ may be built such that $(\lambda x_1 \ldots x_k.M_0)L_1 \ldots L_n$ has no whnf and $(\lambda x_1 \ldots x_k.N_0)L_1 \ldots L_n$ has. In either case there is thus a context separating M_0 from N_0 w.r.t. the weak head equivalence.

Terms always reducing to an abstraction whatever the number of arguments passed to them, like the one hypothesized above, indeed exist. The simplest of them is a term Ξ such that $\Xi \rightarrow_\beta^* \lambda z.\Xi$, known as the *ogre*, since it eats every successive argument fed to it; its explicit expression is obtained, as usual, by applying the $\mathbf{Y}$ combinator to its recursive definition: $\Xi \equiv \mathbf{Y}(\lambda yz.y) \rightarrow_\beta^* (\lambda xz.xx)(\lambda xz.xx)$. We may informally write the equality $\Xi = \lambda z_1.\lambda z_2.\lambda z_3 \ldots$, whose intuitive meaning will be given a formal description in Subsect. 3.2.

The weak head equivalence, which – as we just argued – does not discriminate less than normal equivalence, can immediately be seen to actually discriminate more. It distinguishes – trivially by the empty context – the term Ω from $\lambda x.\Omega$, which on the contrary are equated by the two previous equivalences because they are both unsolvable (and thus represented by the same Böhm tree $\perp$). In a sense, it makes the difference between a piece of running code that loops forever without doing anything, and a defined procedure that does the same but it's not invoked by any other part of the program.

Moreover, it considers as different any two abstractions with a different number of abstracted variables (i.e., any two functions with a different number of parameters) even if they are η-convertible (it is sufficient to take a context consisting of a number of arguments equal to the lesser of the two arities, so as to saturate one of two abstractions, but not the other).

The example of $\lambda x.\Omega$ and Ω shows that there are pairs of terms with identical Böhm trees that are separated by the weak head equivalence; however, there also exist equivalent terms having distinct Böhm trees, like $M \equiv \lambda x.xx$ and $N \equiv \lambda x.x(\lambda y.xy)$. They are equivalent since no applicative context $[\ ]L_1 \ldots L_n$ reduces to whnf on one of them and does not on the other. It is sufficient to consider the form of L_1: if it reduces to an abstraction, then one immediately sees that ML_1 and NL_1 become β-convertible, so they behave the same for any further common sequence of arguments; if on the contrary L_1 never reduces to an abstraction, then the reducts of ML_1 and NL_1, which are respectively L_1L_1 and $L_1(\lambda y.L_1y)$, either both have whnfs (if L_1 has whnf) or they can never become redexes, and thus neither of them can reduce to a whnf.

Hence the weak head observational equivalence, though finer than the normal equivalence, is orthogonal w.r.t. Böhm tree equality:

$$\begin{array}{rcccc} M \simeq_w N & \Rightarrow & M \simeq_n N & \Rightarrow & M \simeq_h N \\ \mathfrak{BT}(M) = \mathfrak{BT}(N) & \Rightarrow & M \simeq_n N & \Rightarrow & M \simeq_h N \end{array}$$

but $\mathfrak{BT}(M) = \mathfrak{BT}(N) \not\Rightarrow M \simeq_w N$ and $M \simeq_w N \not\Rightarrow \mathfrak{BT}(M) = \mathfrak{BT}(N)$.

The issue to be explored, at this point, was whether and how it is possible to find some kind of observational equivalence that exactly corresponds to Böhm tree equality.

3 Observing pure λ-terms in extended λ-calculi

In this section we review the various notions of trees which were successively introduced, beside Böhm trees, to represent the evaluation of terms. A unified view is obtained by considering the different kinds of trees as corresponding to different possible formalizations of the intuitive notion of *stable relevant minimal information* coming out of a computation (dually, they also naturally induce different notions of *meaningless term* [19]).

When in a reduction sequence a term reduces to one of the following forms, the underlined parts will remain stable during the rest (if any) of the computation: $\underline{x}M_1 \ldots M_m$, $\underline{\lambda x.}M$, $P \,\underline{@}\, Q$ (where @ is the explicit representation, normally omitted, of the operation of application, and P is a term which will never reduce to an abstraction). Having a *stable* part in a computation, however, does not necessarily mean that we consider it *relevant.* For instance, we may decide that an abstraction $\underline{\lambda x.}M$ is only relevant when M is of the form $\lambda y_1 \ldots y_n.zN_1 \ldots N_m$ $(n, m \geq 0)$: this leads us to the notion of hnf; but other choices of what is to be taken as relevant are possible.

As we will see, depending on whether we assume as stable relevant minimal information the *head normal form*, the *weak head normal form*, or the *top normal form*, we respectively obtain *Böhm trees*, *Lévy-Longo trees*, or *Berarducci trees.* For each of these we will examine λ-calculus extensions that allow the corresponding version of Böhm's theorem to be established, i.e., the definitions of observational equivalences that discriminate terms exactly as trees do.

3.1 Böhm's theorem for Böhm trees

As we saw in the previous sections, it is impossible within the pure λ-calculus to build a context that discriminates two η-convertible Böhm trees in every case. The reason basically is that, as is well known, in the pure calculus every term can be considered a function and thus applied to an argument; the ability to discriminate between x and its η-expansion $\lambda y.xy$ may then be obtained by enriching the context calculus with primitive elements that are not functions, such as a numeric constant $\mathsf{0}$ (obviously not considered as an abbreviation of corresponding Church numeral [3]), and consequently a notion of an error situation which results from the application of a non-function to an argument. Termination with error has therefore to be considered different from normal termination and assimilated to divergence, so that $\lambda x.x$, where x may be anything, is discriminated from $\lambda xy.xy$, where x must be a function, by a simple context $[\ ]\mathsf{0}$ that passes as argument the non-functional object. These two terms, however, being abstractions of different arities, are already separated by the weak head equivalence.

A variable may have multiple occurrences in a term, some of them in functional positions, and some not. It is therefore essential, in this approach, to allow different occurrences of the same variable to become ultimately bound to different terms; to this end, a natural solution is the introduction of an operator of nondeterministic choice, which of course immediately makes the calculus to become non-confluent, where in general only some of the many possible reduction paths will lead to a correct termination. As a consequence, the observed convergence has to be the *may-convergence*, which means that a term is convergent if it is the starting point of at least one converging reduction sequence, i.e., if at least one of the computations it may generate is terminating with a value.

Actually, the nondeterministic choice operator gives too much freedom for replacing variables. We need to control that every time one occurrence of a variable is replaced by a combinator different from the expected one, the whole term cannot converge. This control can be realized by adding a standard numerical system [3], i.e., besides the constant $\mathsf{0}$, three numeric unary functional constants: a constructor s, a destructor p, and a test $\mathsf{zero?}$.

The above sketched approach is the one adopted by [10], where the nondeterministic operator is denoted by the symbol $+$ (not to be confused with the arithmetic operator!). The rules added to the pure calculus (δ-rules, following the established terminology) are:

$$\begin{array}{lll} \mathsf{p}\,\mathsf{0} \to \mathsf{0} & \mathsf{zero?}\,\mathsf{0} \to \mathbf{T} & M + N \to M \\ \mathsf{p}(\mathsf{s}\mathsf{n}) \to \mathsf{n} & \mathsf{zero?}(\mathsf{s}\mathsf{n}) \to \mathbf{F} & M + N \to N \end{array}$$

where n stands for $\underbrace{\mathsf{s}(\mathsf{s}\ldots(\mathsf{s}\,\mathsf{0})\ldots)}_{n \text{ times}}$, and $\mathbf{T} \equiv \lambda xy.x$, $\mathbf{F} \equiv \lambda xy.y$.

The set of values is the set of numerals, the other terms being a sort of *Not a Number* expressions, and convergence is correspondingly intended as convergence to a numeral; the resulting context equivalence exactly coincides with Böhm tree equality.

We denote the extended calculus by $\Lambda_{\mathbb{N}+}$, and the usual many-step reduction relation by $\to^*_{\beta\mathbb{N}}$; to simplify the statement of the main property, we introduce a couple of straightforward definitions.

Definition 3.1 May-convergence to a numeral, or $\Lambda_{\mathbb{N}+}$-convergence.

We say that a term M in the extended calculus *may converge to a numeral*, or $\Lambda_{\mathbb{N}+}$*-converges*, and we write $M\downarrow_{\mathbb{N}+}$, if there exists a numeral n such that $M \rightarrow^*_{\beta\mathbb{N}} \mathsf{n}$; we say that M $\Lambda_{\mathbb{N}+}$-diverges, and we write $M\uparrow_{\mathbb{N}+}$, if it does not $\Lambda_{\mathbb{N}+}$-converge.

Definition 3.2 $\Lambda_{\mathbb{N}+}$**-observational equivalence.** We say that two terms M and N are $\Lambda_{\mathbb{N}+}$*-observationally equivalent*, and we write $M \simeq_{\mathbb{N}} N$, iff

$$\forall C[\,] \in \Lambda_{\mathbb{N}+}. C[M]\downarrow_{\mathbb{N}+} \Longleftrightarrow C[N]\downarrow_{\mathbb{N}+}.$$

Theorem 3.3 *(Dezani et al. [10]).*
For any two pure terms M and N:

$$M \simeq_{\mathbb{N}} N \quad \Longleftrightarrow \quad \mathfrak{BT}(M) = \mathfrak{BT}(N)$$

i.e., expressed with the contrapositive:

$$\mathfrak{BT}(M) \neq \mathfrak{BT}(N) \Longleftrightarrow \exists C[\,] \in \Lambda_{\mathbb{N}+}. C[M]\downarrow_{\mathbb{N}+} \wedge C[N]\uparrow_{\mathbb{N}+} \quad \textit{or the converse.}$$

The above equivalence may therefore be considered a kind of “external” operational semantics that is finally sound and complete w.r.t. the semantics consisting of Böhm trees; or, put in the opposite way, an operational semantics for which the Böhm tree semantics is fully abstract.

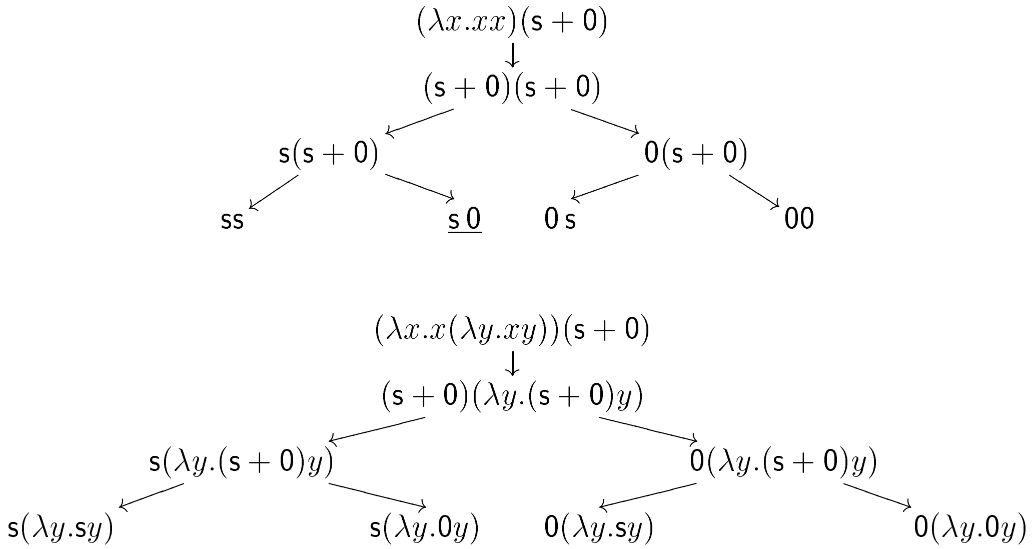


Fig. 6. Reduction trees of $\Delta \equiv \lambda x.xx$ and $\Delta_\eta \equiv \lambda x.x(\lambda y.xy)$ in the context $[\,](\mathsf{s}+\mathsf{0})$

As an example, the Fig. 6 reports the reduction trees of the two terms obtained by filling the context $[\,](\mathsf{s}+\mathsf{0})$ respectively with the term $\Delta \equiv \lambda x.xx$ and with its η-expansion $\Delta_\eta \equiv \lambda x.x(\lambda y.xy)$. In the first case there is a path leading to the numeral $\mathsf{s\,0}$, while in the second case every computation ends in Not a Number; so the context is able to discriminate, as remarked above, between x, which may be anything, and $\lambda y.xy$ which, being a function, cannot be a numeral.

The two terms have different Böhm trees, but being η-convertible they cannot be distinguished by any observational equivalence based on pure λ-calculus contexts and respecting Böhm tree equality; actually, they are equated by all three equivalences based on pure λ-calculus contexts: $\Delta \simeq_\diamond \Delta_\eta (\diamond = n, h, w)$.

An analogous result of an observational equivalence exactly matching the Böhm tree semantics is obtained in [28,29] through an encoding of the λ-calculus into the π-calculus, followed by the use of an appropriate (e.g., taking divergence into account) bisimulation between processes.

3.2 Lévy-Longo trees vs. Böhm trees

As recalled respectively in Subsect. 2.1 and in Subsect. 2.3, besides the ordinary β-reduction two other kinds of reduction – and correspondingly of normal form – have been considered in the pure λ-calculus: the head reduction, with the associate notion of head normal form, and the weak head reduction, or lazy reduction, with the associate notion of weak head normal form.

Referring to a usual formal presentation of λ-calculus, with contextual rules written explicitly:

$$\begin{array}{llll}
(\beta) & & & (\lambda x.M)N \to M[N/x] \\
(\eta) & & & \lambda x.Mx \to M \quad \text{if } x \notin FV(M) \\
(\mu) & M \to N & \Rightarrow & LM \to LN \\
(\nu) & M \to N & \Rightarrow & ML \to NL \\
(\xi) & M \to N & \Rightarrow & \lambda x.M \to \lambda x.N
\end{array}$$

the ordinary reduction, also generically called β-reduction, is the one induced by the rules β, μ, ν, ξ; by excluding the rule μ, one obtains the head reduction, induced by the rules β, ν, ξ; finally, the weak head reduction is obtained by further excluding the rule ξ, i.e., it is the one induced by β, ν. For each reduction relation there is the corresponding notion of normal form, which is a term where none of the rules of the respective set applies. By adding the η-rule to the ordinary and head reduction one respectively obtains the $\beta\eta$-reduction and η-head reduction.

It is well known that in the case of ordinary and head reduction more than one rule may in general apply to a given term, and therefore different reduction strategies can be defined, not all of them guaranteed to lead to the respective normal forms whenever they exist (actually, head reduction is usually intended to be associated with the normalizing outermost strategy); the weak head reduction, on the other hand, is completely deterministic.

The normalizing leftmost outermost strategy for ordinary reduction, simply called normal reduction in the following, can be recursively defined, as reminded in Subsect. 2.1, by means of the head reduction. It also admits a recursive definition using the weak head reduction: to normally reduce a term, reduce it by weak head reduction; if this terminates, then either the resulting whnf is a variable, in which case the normal reduction also terminates, or it has the form $\lambda x.M$ or $xM_1 \ldots M_m$, in which cases recursively weak head reduce the subterms, respectively M or $M_1, \ldots, M_m$.

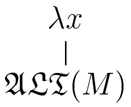
This corresponds, of course, to another possible way of inductively defining the notion of β-normal form:

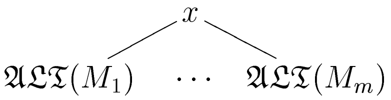
- a variable x is a (weak head normal form that also is) a normal form;
- a weak head normal form $\lambda x.M$ is a normal form if M is a normal form;
- a weak head normal form $xM_1 \ldots M_m$ is a normal form if $M_1, \ldots, M_m$ are normal forms.

The Lévy-Longo tree [22,21,20] of a β-normal form is the tree representation of this inductive structure of nested weak head normal forms, like the Böhm tree was for nested head normal forms. Here too the above definition, if read coinductively, defines the notion of a generalized, possibly infinite, β-normal form, existing for every term. Approximate Lévy-Longo trees are the analogous of the approximate Böhm trees, and so are the notions – w.r.t. their Böhm-tree homologous – of partial order and limit.

Definition 3.4 Approximate Lévy-Longo trees.

(i) if $M = x$, then $\mathfrak{ALT}(M) = x$

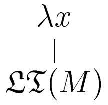
(ii) if $M = \lambda x.M$, then $\mathfrak{ALT}(M) =$ 

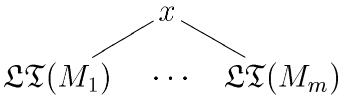
(iii) if $M = xM_1 \ldots M_m$ $(m > 0)$, $\mathfrak{ALT}(M) =$ 

(iv) otherwise (i.e., if M is not in whnf) $\mathfrak{ALT}(M) = \bot$.

Definition 3.5 Lévy-Longo trees.

(i) if $M \to_\beta^* x$, then $\mathfrak{LT}(M) = x$

(ii) if $M \to_\beta^* \lambda x.M$, then $\mathfrak{LT}(M) =$ 

(iii) if $M \to_\beta^* xM_1 \ldots M_m$ $(m > 0)$, $\mathfrak{LT}(M) =$ 

(iv) otherwise (i.e., if M does not have a whnf) $\mathfrak{LT}(M) = \bot$.

Observe that, owing to the recursive definition of the leftmost outermost strategy through head or weak head reduction, when the computation reaches a hnf or respectively a whnf, the successive reduction, only acting on subterms, does not change the term's top-level form; or, in the language of trees, it does not change the root node.

A term reduction is therefore, as already pointed out in Subsect. 2.1, the step-by-step construction of a possibly infinite structure, through the accumulation of stable atoms – i.e., not subject to future change – of relevant information. Depending on which syntactical structure is assumed as the atomic (or minimal) relevant information, descriptions of different granularities are obtained for the computed result, as anticipated at the beginning of the section.

In the representation based on head normal forms, the atom of relevant information is the whole underlined part in $\underline{\lambda x_1 \dots x_n.x}M_1 \dots M_m$ (with $m, n \geq 0$). In the representation based on weak head normal forms, this atom is further split into smaller separate components $\underline{\lambda x_1}, \dots, \underline{\lambda x_n}, \underline{x}$, since the minimal relevant information is the underlined part in $\underline{\lambda x}.M$ or $\underline{x}M_1 \dots M_m$.

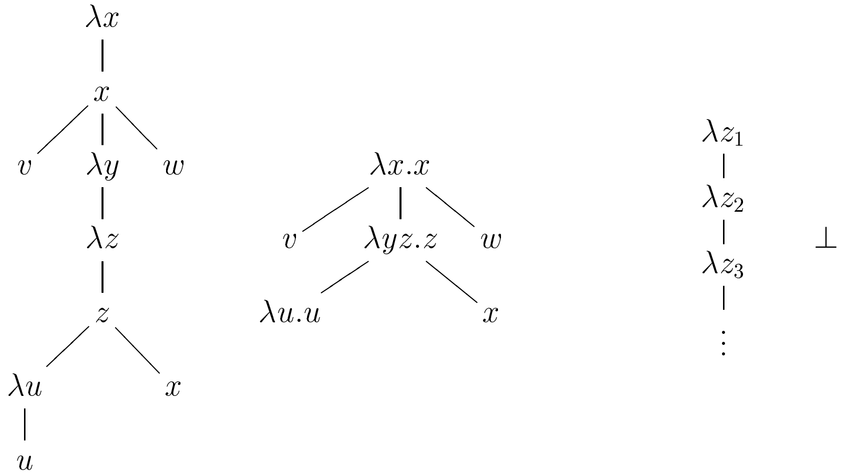


Fig. 7. Lévy-Longo and Böhm trees of the term $\lambda x.xv(\lambda yz.z(\lambda u.u)x)w$ and of the term $\Xi \equiv (\lambda xz.xx)(\lambda xz.xx)$

Lévy-Longo trees are thus finer than Böhm trees, in the sense that there is a homomorphic node mapping from the Lévy-Longo tree to the Böhm tree of the same term. For example, in the left part of Fig. 7 are shown the Lévy-Longo tree and the Böhm tree of the normal form $\lambda x.xv(\lambda yz.z(\lambda u.u)x)w$ (instance of one considered in Sect. 1). However, the fact that the relevant information labelling one Böhm tree node generally happens to be distributed over several nodes in the corresponding Lévy-Longo tree is a mere superficial syntactic appearance; the actual difference between the two structures lies in the unsolvable terms, i.e., in that a $\perp$-labelled node in a Böhm tree may correspond to a non-$\perp$-subtree in its Lévy-Longo correspondent. For example, in Fig. 7 is also shown an infinite Lévy-Longo tree corresponding to a finite Böhm tree: the term Ξ introduced in Subsect. 2.3, which has no hnf, whose Böhm tree is therefore simply $\perp$.

As a consequence, two different Lévy-Longo trees corresponding to identical Böhm trees may only differ in an unsolvable node, represented by a Böhm tree $\perp$-node.

3.3 Böhm's theorem for Lévy-Longo trees

As we saw in the preceding subsection, Lévy-Longo tree equality strictly implies Böhm tree equality, and therefore it strictly implies head and normal observational equivalences. It is also strictly finer than weak head observational equivalence: in the first place, it cannot be less discriminating, roughly since if $M \not\simeq_w N$, that is, say, $C[M]$ has whnf while $C[N]$ has not, then M and N must differ in some homologous subterms, which cannot be both without whnf; therefore $\mathfrak{LT}(M) \neq \mathfrak{LT}(N)$ (for a rigorous proof see [1]). Secondly, $\Delta \equiv \lambda x.xx$ and $\Delta_\eta \equiv \lambda x.x(\lambda y.xy)$ are an example of two terms having different Lévy-Longo trees which are equated by the weak head equivalence. Summarizing, we have:

$$\mathfrak{LT}(M) = \mathfrak{LT}(N) \;\Rightarrow\; \mathfrak{BT}(M) = \mathfrak{BT}(N)$$
$$\mathfrak{LT}(M) = \mathfrak{LT}(N) \;\Rightarrow\; M \simeq_w N \;\Rightarrow\; M \simeq_n N \;\Rightarrow\; M \simeq_h N.$$

Remark however that the two terms Ω and Ξ, which have different Lévy-Longo trees but identical Böhm trees, are already discriminated by the weak head equivalence, while Δ and Δ_η, which have different Lévy-Longo trees but are observationally equivalent, have different Böhm trees.

As a matter of fact, if two terms have different Lévy-Longo trees but identical Böhm trees, they must differ – as remarked above – in an unsolvable node (like in the trees of Ω and $\lambda x.\Omega$ or Ξ, where the difference is directly in the root), and are therefore separated by the weak head equivalence; hence, one has in general:

$$\mathfrak{LT}(M) = \mathfrak{LT}(N) \quad \Longleftrightarrow \quad \mathfrak{BT}(M) = \mathfrak{BT}(N) \text{ and } M \simeq_w N.$$

The exact discrimination of Lévy-Longo trees cannot thus be achieved by a purely observational equivalence in the pure λ-calculus; put in other words, an operational semantics for pure λ-calculus that is sound and complete w.r.t. the Lévy-Longo tree semantics cannot be defined observationally within the calculus itself.

Such an equivalence is obtained by Sangiorgi in [27,29] by first adopting the Milner encoding of the lazy λ-calculus into the π-calculus, let it be denoted by $\pi(\;)$, and then considering within the π-calculus a standard observational equivalence between processes, as the weak bisimilarity, or a barbed congruence, let it be generically denoted by the symbol $\simeq_\pi$: the result is proved that $\pi(M) \simeq_\pi \pi(N)$ iff $\mathfrak{LT}(M) = \mathfrak{LT}(N)$.

Observational equivalences sound and complete w.r.t. the Lévy-Longo tree equality have been defined outside the π-calculus by resorting to various standard (or less standard) extensions of λ-calculus. In [27] the operator $+$ of nondeterministic choice, already presented in Subsect. 3.1, is added to the calculus, obtaining the set of terms Λ_+ and the reduction relation $\rightarrow_{\beta+}$: the usual applicative bisimilarity (between closed terms) is then considered.

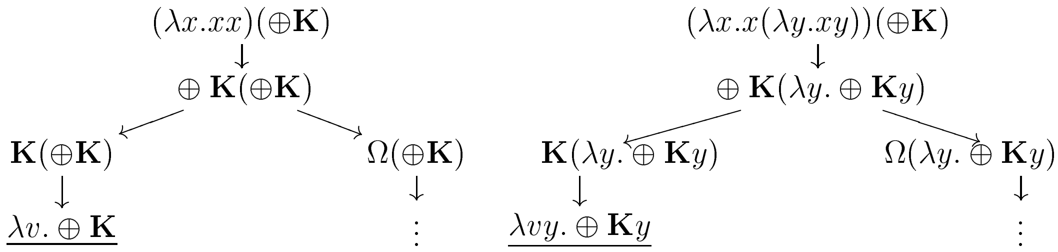

$(\lambda x.xx)(\oplus\mathbf{K})$
$\oplus\,\mathbf{K}(\oplus\mathbf{K})$
$\mathbf{K}(\oplus\mathbf{K})$
$\Omega(\oplus\mathbf{K})$
$\lambda v.\oplus\mathbf{K}$
$(\lambda x.x(\lambda y.xy))(\oplus\mathbf{K})$
$\oplus\,\mathbf{K}(\lambda y.\oplus\mathbf{K}y)$
$\mathbf{K}(\lambda y.\oplus\mathbf{K}y)$
$\Omega(\lambda y.\oplus\mathbf{K}y)$
$\lambda vy.\oplus\mathbf{K}y$

Fig. 8. Reduction trees of Δ and Δ_η applied to the argument $\oplus\mathbf{K}$

Definition 3.6 Λ_+**-observational equivalence**
We say that two closed term M and N are Λ_+*-observationally equivalent, or applicatively bisimilar* in the Λ_+-calculus, and we write $M \simeq_+ N$, iff:

(i) M has whnf $\iff$ N has whnf ;

(ii) for all closed $L \in \Lambda_+$ one has $ML \simeq_+ NL$;

(iii) if $M \to^*_{\beta+} M'$, there exists an N' such that $N \to^*_{\beta+} N'$ and $M' \simeq_+ N'$,
if $N \to^*_{\beta+} N'$, there exists an M' such that $M \to^*_{\beta+} M'$ and $M' \simeq_+ N'$.

Observe that the third clause, not present in the observational equivalence definitions we previously introduced, is necessary because of the non-confluence of the extended calculus; it is a typical feature of bisimilarity proper, as defined in concurrent (and thus non-confluent) calculi, in contrast with the ordinary contextual equivalences in λ-calculi.

The operator $+$ is only needed in expressions of the form $M+\Omega$, to produce a nondeterministic branching between a possibly converging term and the diverging Ω; hence a restricted unary form suffices, denoted $\oplus$, with the rules:

$$\oplus M \to M \qquad \oplus M \to \Omega.$$

More formally, if we denote by $\simeq_\oplus$ the applicative bisimilarity in $\Lambda_\oplus$, defined analogously to Def. 3.6, we have

$$M \simeq_+ N \iff M \simeq_\oplus N.$$

The main result of [27] is then:

Theorem 3.7 *(Sangiorgi [27]).*
For any two pure terms M and N:

$$M \simeq_\oplus N \quad\iff\quad \mathfrak{LT}(M) = \mathfrak{LT}(N).$$

For example, we can discriminate Δ and Δ_η by applying them to $\oplus\mathbf{K}$, with $\mathbf{K} \equiv \lambda uv.u$. Fig. 8 shows the reduction trees: now, if obeying to the third clause of Def. 3.6 we apply the two corresponding (underlined) terms to an arbitrary argument L we get:

$$\begin{tikzcd}
 & (\lambda v. \oplus \mathbf{K})L \arrow[d] & & (\lambda vy. \oplus \mathbf{K}y)L \arrow[d] \\
 & \oplus \mathbf{K} \arrow[dl] \arrow[dr] & & \lambda y. \oplus \mathbf{K}y \\
\mathbf{K} & & \Omega \arrow[d] & \\
 & & \vdots &
\end{tikzcd}$$

where obviously Ω and the value $\lambda y. \oplus \mathbf{K}y$ are not bisimilar, hence neither are Δ and Δ_η in the first place.

Another way of obtaining a Böhm's theorem for Lévy-Longo trees is the one of Boudol and Laneve [6], who introduce a "resource-conscious" refinement of λ-calculus in which every argument comes with a *multiplicity*. More precisely, the argument of a β-redex may happen to be available only a finite number of times, in contrast with its always infinite availability in the ordinary λ-calculus. In such *calculus of multiplicities* [6] the β-rule is consequently modified as follows:

$$(\beta\mu) \qquad (\lambda x.M)N^m \to M < N^m/x >$$

where $< N^m/x >$ is the *explicit substitution* that can replace at most m occurrences of x in M by N. The ordinary β-rule is recovered by putting $m = \infty$.

A term of the form $x < N^0/x >$ is a *deadlock*, since we are required to replace an occurrence of x with the term N that is not available. For example, the reduction paths of Δ and Δ_η when applied to $\mathbf{I}^1$ are:

$$\begin{aligned}
&(\lambda x.xx)\mathbf{I}^1 \to_{\beta\mu} (xx) < \mathbf{I}^1/x > \to_{\beta\mu} (\mathbf{I}x) < \mathbf{I}^0/x > \to_{\beta\mu} z < x^\infty/z >< \mathbf{I}^0/x > \\
&\to_{\beta\mu} x < x^\infty/z >< \mathbf{I}^0/x > \\
&(\lambda x.x(\lambda y.xy))\mathbf{I}^1 \to_{\beta\mu} (x(\lambda y.xy)) < \mathbf{I}^1/x > \to_{\beta\mu} (\mathbf{I}(\lambda y.xy)) < \mathbf{I}^0/x > \\
&\to_{\beta\mu} z < (\lambda y.xy)^\infty/z >< \mathbf{I}^0/x > \to_{\beta\mu} (\lambda y.xy) < (\lambda y.xy)^\infty/z >< \mathbf{I}^0/x >,
\end{aligned}$$

where $\to_{\beta\mu}$ is the reduction relation induced by the rule $(\beta\mu)$.

Let Λ_μ denote the set of terms with multiplicities: a contextual equivalence is then obtained by choosing as set of values the set of abstractions (abs).

Definition 3.8 **Λ_μ-observational equivalence.** We say that two terms M and N are *Λ_μ-observationally equivalent*, and we write $M \simeq_\mu N$, iff

$$\forall C[\ ] \in \Lambda_\mu.\ C[M] \to^*_{\beta\mu} \text{ an abs } \iff C[N] \to^*_{\beta\mu} \text{ an abs.}$$

Theorem 3.9 *(Boudol and Laneve [6]).*
For any two pure terms M and N:

$$M \simeq_\mu N \quad \Longleftrightarrow \quad \mathfrak{LT}(M) = \mathfrak{LT}(N)$$

i.e., expressed with the contrapositive:

$$\mathfrak{LT}(M) \neq \mathfrak{LT}(N) \iff$$
$$\exists C[\,] \in \Lambda_\mu \,.\, C[M] \to^*_{\beta\mu} \text{ an abs } \wedge\ C[N] \not\to^*_{\beta\mu} \text{ an abs } \textit{or the converse.}$$

For example $\Delta \not\simeq_\mu \Delta_\eta$, since $\Delta_\eta \mathbf{I}^1$ reduces to an abstraction, while this is false for $\Delta\mathbf{I}^1$.

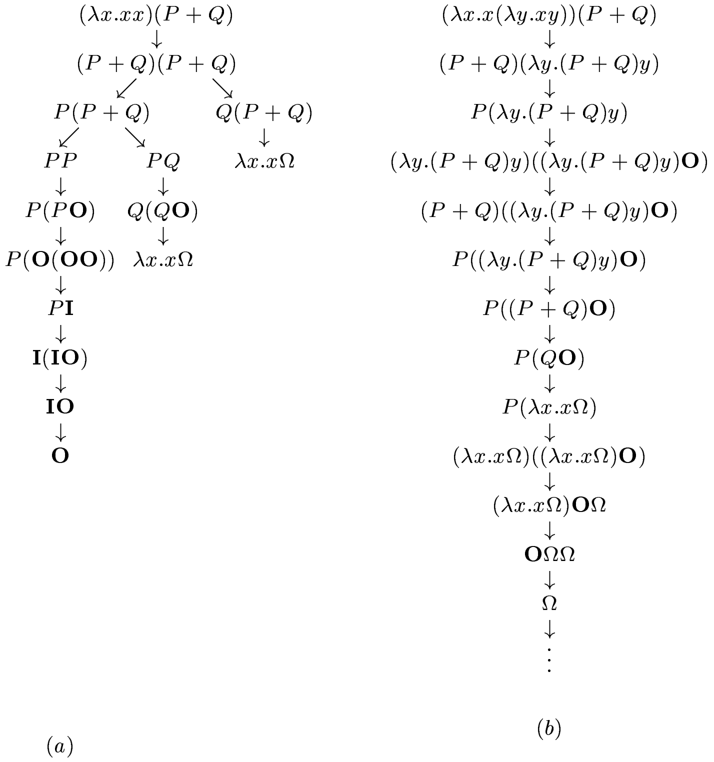


Fig. 9. (a) The reduction tree of $\Delta(P + Q)$. (b) An infinite reduction path out of $\Delta_\eta(P + Q)$.

Finally, Dezani et al. [12] consider the behaviour of pure terms within contexts of the concurrent λ-calculus defined in [9]. This calculus is obtained from the pure λ-calculus (with call-by-value and call-by-name variables) by adding the nondeterministic choice operator $+$ and a parallel operator $\|$, whose main reduction rule is

$$\frac{M \to M' \quad N \to N'}{M \| N \to M' \| N'}\ (\|)$$

where → stands for one-step reduction.

Let $\to_{\beta+\|}$ be the so obtained reduction relation. We compare terms by taking as values call-by-value variables, abstractions, and parallel compositions of an arbitrary term with a value, the last being quite natural in view of the above rule for $\|$. If we denote call-by-value variables by Greek letters we obtain the following grammar for the set V of values:

$$V ::= \xi \mid \lambda x.M \mid \lambda\xi.M \mid V\|M \mid M\|V$$

where M is any term.

We consider *must-convergence*, which means that a term is convergent if *all* reduction sequences starting from it reach a value.

Definition 3.10 $\Lambda_{+\|}$-convergence.
We say that a term M in the extended calculus $\Lambda_{+\|}$-*converges*, and we write $M{\downarrow_{+\|}}$, if there is no infinite reduction path out of M and moreover whenever $M \to^*_{\beta+\|} N$ there exists a value V such that $N \to^*_{\beta+\|} V$; we say that M $\Lambda_{+\|}$-diverges, and we write $M{\uparrow_{+\|}}$, if it does not $\Lambda_{+\|}$-converge.

Definition 3.11 $\Lambda_{+\|}$-observational equivalence. We say that two terms M and N are $\Lambda_{+\|}$-*observationally equivalent*, and we write $M \simeq_{+\|} N$, iff

$$\forall C[\,] \in \Lambda_{+\|}.C[M]{\downarrow_{+\|}} \Longleftrightarrow C[N]{\downarrow_{+\|}}.$$

Theorem 3.12 *(Dezani et al. [12]).*
For any two pure terms M and N:

$$M \simeq_{+\|} N \quad \Longleftrightarrow \quad \mathfrak{LT}(M) = \mathfrak{LT}(N)$$

i.e., expressed with the contrapositive:

$$\mathfrak{LT}(M) \neq \mathfrak{LT}(N) \Longleftrightarrow \exists C[\,] \in \Lambda_{+\|}.C[M]{\downarrow_{+\|}} \wedge C[N]{\uparrow_{+\|}} \quad \textit{or the converse.}$$

Fig. 9 shows that $\Delta \not\simeq_{+\|} \Delta_\eta$ by applying them to the term $P + Q$, where $P \equiv \lambda\xi.\xi(\xi\mathbf{O})$, $\mathbf{O} \equiv \lambda zt.t$, and $Q \equiv \lambda yx.x\Omega$.

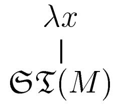
3.4 Böhm's theorem for Berarducci Trees

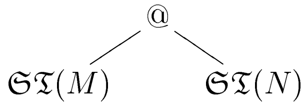
Böhm trees and Lévy-Longo trees may be viewed as two particular kinds of syntax trees of possibly infinite normal forms, where some parts of the syntactic structure are hidden as non-relevant: in particular, following the concrete syntax of λ-calculus, the binary application operator is left implicit.

The third kind of trees representing possibly infinite normal forms is obtained by directly starting from abstract syntax trees of terms, where we explicitly represent application with the symbol @.

Definition 3.13 Syntax trees.

(i) $\mathfrak{ST}(x) = x$;

(ii) $\mathfrak{ST}(\lambda x.M) =$ 

(iii) $\mathfrak{ST}(MN) =$ 

For example the syntax trees of the terms $\lambda x.f(xx)$, $\lambda x.xx$ and $\lambda x.xxx$ are shown in Fig. 10, and Fig. 11 gives the syntax trees of the self-applications of these terms.

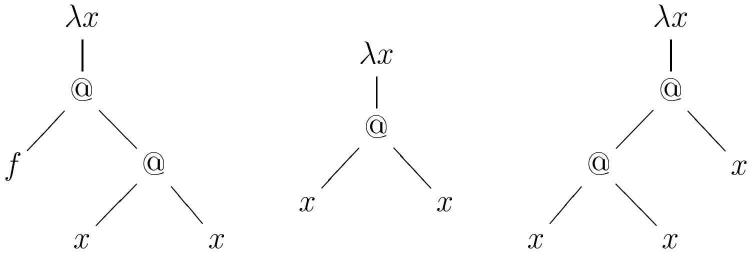


Fig. 10. The syntax trees of $\lambda x.f(xx)$, $\lambda x.xx$ and $\lambda x.xxx$

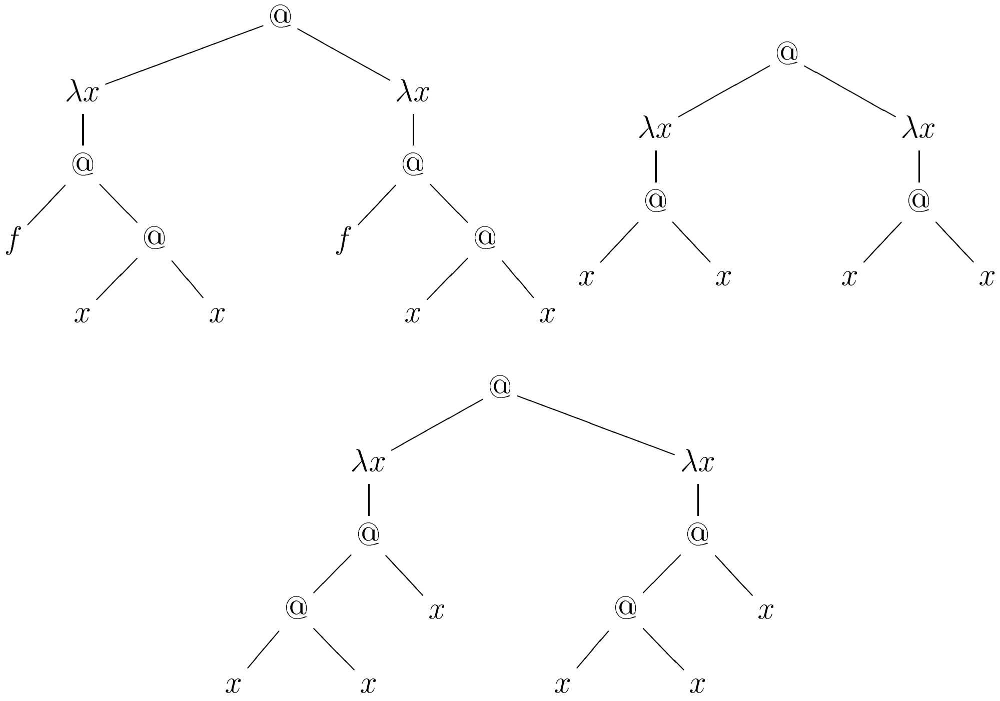


Fig. 11. The syntax trees of $(\lambda x.f(xx))(\lambda x.f(xx))$, $(\lambda x.xx)(\lambda x.xx)$ and $(\lambda x.xxx)(\lambda x.xxx)$

Now if we reduce these terms we get:

$$(\lambda x.f(xx))(\lambda x.f(xx)) \to_\beta f((\lambda x.f(xx))(\lambda x.f(xx))) \to_\beta f(f(\ldots)) \to_\beta \ldots$$

$$(\lambda x.xx)(\lambda x.xx) \to_\beta (\lambda x.xx)(\lambda x.xx) \to_\beta (\lambda x.xx)(\lambda x.xx) \to_\beta \ldots$$

$$(\lambda x.xxx)(\lambda x.xxx) \to_\beta (\lambda x.xxx)(\lambda x.xxx)(\lambda x.xxx) \to_\beta \ldots$$

so the trees of Fig. 11 are not informative of their behaviours. As a matter of fact $(\lambda x.f(xx))(\lambda x.f(xx))$ generates an infinite number of applications in which the terms in function position are always f, $(\lambda x.xxx)(\lambda x.xxx)$ generates an infinite number of applications in which the terms in argument position are always $\lambda x.xxx$, while $(\lambda x.xx)(\lambda x.xx)$ always reduces to itself.

In analogy with the previously presented kinds of trees, consider for every term the possibly infinite sequences of syntax trees associated to reduction sequences starting from that term; for a notion of limit to be defined, the application operator must also be taken into account – beside abstraction and variables – as an atom of relevant information, whose occurrences become part of the limit tree when they are stable.

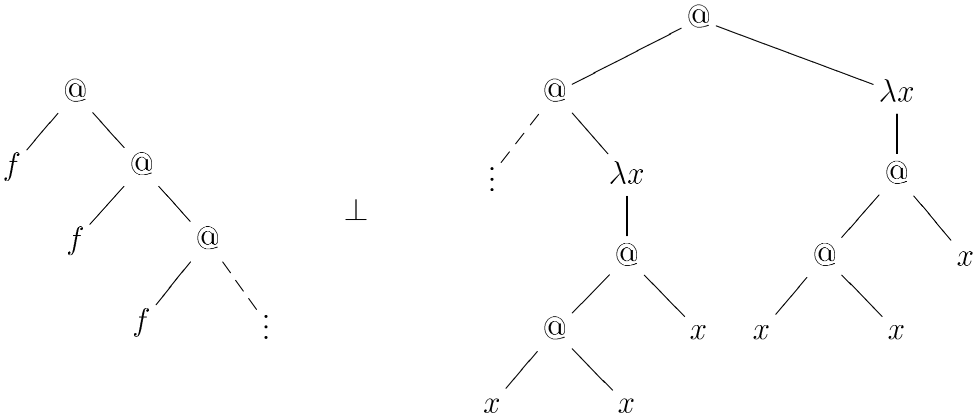
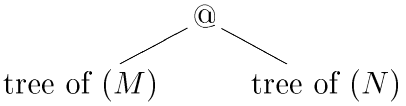


Fig. 12. The Berarducci trees of $(\lambda x.f(xx))(\lambda x.f(xx))$, $(\lambda x.xx)(\lambda x.xx)$ and $(\lambda x.xxx)(\lambda x.xxx)$

The question is then: when is an application occurrence stable? Given a term MN ($\equiv M@N$), we have two cases: if M reduces to a λ-abstraction, i.e., $M \to_\beta^* \lambda x.M_0$, then a step of β-reduction of course destroys the occurrence of the application operator, and the tree of MN must be the tree of $M_0[N/x]$; on the contrary, if M does not reduce to an abstraction, the application is *stable* and the tree of MN is . This leads to the key notions of *zero term* and of *top normal form*, and through them to the definition of *Berarducci trees*, introduced in [4] and independently in [17] at the same time.

Definition 3.14 Zero Terms and Top Normal Forms.

(i) A term is called a *zero term* iff it cannot β-reduce to an abstraction.

(ii) A term is called a *top normal form (tnf)* if it is a variable, or an abstraction, or an application of the form MN, where M is a zero term.

It is easy to verify that zero terms are either unsolvable terms of order zero [1], like Ω and $(\lambda x.xxx)(\lambda x.xxx)$, or they are reducible to terms of the form $xM_1 \ldots M_n$ where $n \geq 0$, i.e., to applications of a free variable to any number (also zero!) of arguments.
Examples of top normal forms are $(\lambda x.xxx)(\lambda x.xxx)(\lambda x.xxx)$ and $\lambda x.\Omega$.

Alternatively one can define the top normal forms as the normal forms w.r.t. the *top reduction*, i.e., the reduction induced by the rules (β) and (ν_t), where:

$$(\nu_t) \quad M \to N \Rightarrow ML \to NL \ \text{(provided } M \text{ is not a zero term).}$$

If we observe terms by putting them within pure λ-calculus contexts and taking as values tnfs, we obtain a corresponding contextual equivalence.

Definition 3.15 Top observational equivalence.
We say that two terms M and N are *observationally equivalent w.r.t. to top convergence*, i.e., when the set of values is the set of top normal forms, and we write $M \simeq_t N$, iff

$$\forall C[\,].C[M] \text{ has tnf } \iff C[N] \text{ has tnf.}$$

This last equivalence is not comparable with the weak head equivalence $\simeq_w$: on the one hand it distinguishes Ω from $(\lambda x.xxx)(\lambda x.xxx)$ which are equated by $\simeq_w$; on the other hand it equates $\lambda x.\Omega\mathbf{I}$ and $\Omega\mathbf{I}$ which are separated by $\simeq_w$ (the first of the two terms has whnf, the second has not; both are tnfs, and moreover, by definition, they still have tnfs to whatever sequence of arguments they might be applied).

From the above it is natural to define the Berarducci trees of terms as the trees we can draw as soon as we reach a top normal form.

Definition 3.16 Berarducci trees.

(i) if $M \to_\beta^* x$, then $\mathfrak{BeT}(M) = x$

(ii) if $M \to_\beta^* \lambda x.N$, then $\mathfrak{BeT}(M) = \begin{array}{c}\lambda x\\ | \\ \mathfrak{BeT}(N)\end{array}$

(iii) if $M \to_\beta^* M_1M_2$, where M_1 is a zero term, then

$$\mathfrak{BeT}(M) = \begin{array}{ccc} & @ & \\ \diagup & & \diagdown \\ \mathfrak{BeT}(M_1) & & \mathfrak{BeT}(M_2)\end{array}$$

(iv) otherwise (i.e. if M does not have a tnf) $\mathfrak{BeT}(M) = \bot$.

Berarducci trees are more discriminating than Lévy-Longo trees, and hence than Böhm trees, since for example $(\lambda x.xx)(\lambda x.xx)$ and $(\lambda x.xxx)(\lambda x.xxx)$ have different Berarducci trees (shown in Fig. 12), while they have identical Lévy-Longo trees and Böhm trees, namely $\bot$.

Berarducci tree equality is also at least as discriminating as the top observational equivalence $\simeq_t$, as follows from an argument analogous to the one used for Lévy-Longo trees and weak head equivalence in Subsect. 3.3 (for a proof see [11]).

Observing terms within pure λ-calculus will therefore equate some terms having different Berarducci trees, whatever kind of normal forms is chosen as the set of values. To separate terms having different Berarducci trees we need to böhm out also arguments of unsolvable terms, as for example in case we want to find a context which discriminates $\Omega\mathbf{II}$ and $\Omega\Omega\mathbf{I}$, whose Berarducci trees are represented in Fig. 13. To this purpose the paper [11] considers the set of terms Λ_{OA} obtained from the pure λ-calculus by adding the two constants O and A. The constants O and A select the operator and the argument of a closed, stable application. These constants have the following reduction rules:

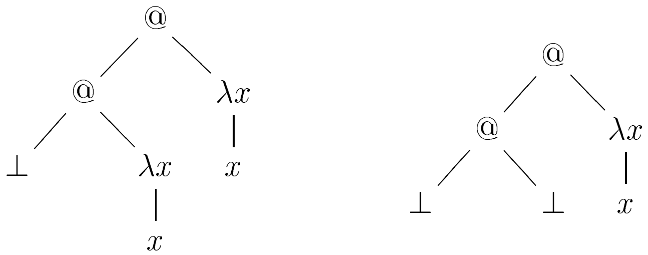


Fig. 13. Berarducci trees of $\Omega\mathbf{II}$ and $\Omega\Omega\mathbf{I}$

$$\mathsf{O}(MN) \to M \text{ if } MN \text{ is a closed term and } M \text{ is a zero term}$$
$$\mathsf{A}(MN) \to N \text{ if } MN \text{ is a closed term and } M \text{ is a zero term.}$$

For instance, $\Omega\mathbf{II}$ and $\Omega\Omega\mathbf{I}$ are discriminated by the context $\mathsf{A}(\mathsf{O}[\,])$. In fact, if we denote by $\to_{\beta\mathsf{OA}}$ the induced reduction relation, we have:

$$\mathsf{A}(\mathsf{O}(\Omega\mathbf{II})) \to_{\beta\mathsf{OA}} \mathsf{A}(\Omega\mathbf{I}) \to_{\beta\mathsf{OA}} \mathbf{I}$$
$$\mathsf{A}(\mathsf{O}(\Omega\Omega\mathbf{I})) \to_{\beta\mathsf{OA}} \mathsf{A}(\Omega\Omega) \to_{\beta\mathsf{OA}} \Omega.$$

With the Böhm-out technique employed in the calculi presented so far, one had to solve (either by using suitable combinators, or by means of the nondeterministic choice) the problem of replacing different occurrences of the same variable by different selectors; such problem disappears in this last λ-calculus extension, because the selection is performed by the two constants O and A.

Definition 3.17 **Λ_{OA}-observational equivalence.** We say that two terms M and N are Λ_{OA}-*observationally equivalent*, and we write $M \simeq_{\mathsf{OA}} N$, iff

$$\forall C[\,] \in \Lambda_{\mathsf{OA}} \,.\, C[M] \text{ has a tnf } \Longleftrightarrow C[N] \text{ has a tnf.}$$

Theorem 3.18 *(Dezani et al. [11]).*
For any two pure terms M and N:

$$M \simeq_{\mathsf{OA}} N \quad \Longleftrightarrow \quad \mathfrak{BeT}(M) = \mathfrak{BeT}(N)$$

i.e., expressed with the contrapositive:

$$\mathfrak{BeT}(M) \neq \mathfrak{BeT}(N) \Longleftrightarrow$$
$$\exists C[\,] \in \Lambda_{\mathsf{OA}} \,.\, C[M] \text{ has tnf } \wedge\ C[N] \text{ has no tnf } \textit{or the converse.}$$

4 Conclusion

In Fig. 14 are summarized the observational equivalences and tree representations discussed in the present paper. An arrow between two points means that the starting point induces a λ-theory finer than the arriving point: for example the arrow between $\simeq_w$ and $\simeq_n$ says that $M \simeq_w N$ implies $M \simeq_n N$. Similarly for double arrows. We want to remark that all the proofs of the horizontal arrows use (some variant of) the Böhm-out technique.

$$\begin{tikzcd}
 & & & \mathfrak{BT}_{\eta\infty} \arrow[r, Leftrightarrow] & \simeq_h \\
 & & & \mathfrak{BT}_{\eta} \arrow[r, Leftrightarrow] \arrow[u, Rightarrow] & \simeq_n \arrow[u, Rightarrow] \\
 & & \simeq_{\mathbb{N}} \arrow[r, Leftrightarrow] & \mathfrak{BT} \arrow[u, Rightarrow] \arrow[ur, Rightarrow] & \simeq_w \arrow[u, Rightarrow] \\
\simeq_{+||} \arrow[r, Leftrightarrow] & \simeq_{\mu} \arrow[r, Leftrightarrow] & \simeq_{\oplus} \arrow[r, Leftrightarrow] \arrow[u, Rightarrow] & \mathfrak{LT} \arrow[u, Rightarrow] \arrow[ur, Rightarrow] & \simeq_t \\
 & & \simeq_{\mathsf{OA}} \arrow[r, Leftrightarrow] \arrow[u, Rightarrow] & \mathfrak{BeT} \arrow[u, Rightarrow] \arrow[ur, Rightarrow] &
\end{tikzcd}$$

Fig. 14. Relations between observational equivalences and equalities of trees

It should be clear that the present overview leaves many questions unanswered or in wait for more satisfactory answers. In particular, all the various λ-calculus extensions need to be better justified, in the sense that it must be determined how they actually depend on the structure of the kinds of trees respectively addressed.

To conclude, it is worthwhile to mention that there exist precise correspondences between the tree representations of terms and the local structures (or equivalently the λ-theories) of certain λ-models ([3], Chapter 19). In particular, such correspondences amount to the fact that two terms have the same tree representation if and only if they are equal in the λ-model. For example,

- the Böhm η_∞-trees represent the local structure of Scott's D_∞ model as defined in [30] (this result was proved in [32]);
- the Böhm η-trees represent the local structure of the inverse limit model defined in [7];
- the Böhm trees represent the local structure of Scott's P_ω model as defined in [31] (a discussion on this topic can be found in [3], Chapter 19);
- the Lévy-Longo trees were introduced by Longo in [22] (following [21]), to prove that they represent the local structure of Engeler's models as defined in [13].

Acknowledgement

We are very grateful to Pierre-Louis Curien, Roger Hindley, Fer-Jan de Vries and the anonymous referees for their useful remarks and suggestions.

References

[1] S. Abramsky and C.-H. L. Ong. Full abstraction in the lazy lambda calculus. *Inform. and Comput.*, 105(2):159–267, 1993.

[2] F. Barbanera, M. Dezani-Ciancaglini, and F.-J. de Vries. Types for trees. In *Proceedings of PROCOMET'98*, pages 11–29. Chapman & Hall, London, 1998.

[3] H. P. Barendregt. *The lambda calculus. Its syntax and semantics.* North-Holland Publishing Co., Amsterdam, revised edition, 1984.

[4] A. Berarducci. Infinite λ-calculus and non-sensible models. In *Logic and algebra (Pontignano, 1994)*, pages 339–377. Dekker, New York, 1996.

[5] C. Böhm. Alcune proprietà delle forme β-η-normali nel λ-K-calcolo. *Pubblicazioni dell'IAC*, 696:1–19, 1968.

[6] G. Boudol and C. Laneve. The discriminating power of multiplicities in the λ-calculus. *Inform. and Comput.*, 126(1):83–102, 1996.

[7] M. Coppo, M. Dezani-Ciancaglini, and M. Zacchi. Type theories, normal forms, and D_∞-lambda-models. *Inform. and Comput.*, 72(2):85–116, 1987.

[8] P.L. Curien. Sur l'eta-expansion infinie. *Comptes Rendus de l'Académie des Sciences*, to appear, 2001.

[9] M. Dezani-Ciancaglini, U. de'Liguoro, and A. Piperno. A filter model for concurrent λ-calculus. *SIAM J. Comput.*, 27(5):1376–1419 (electronic), 1998.

[10] M. Dezani-Ciancaglini, B. Intrigila, and M. Venturini-Zilli. Böhm's theorem for Böhm trees. In *ICTCS'98 (Prato, 1998)*, pages 1–23. World Scientific, Oxford, 1998.

[11] M. Dezani-Ciancaglini, P. Severi, and F.-J. de Vries. Böhm's theorem for Berarducci trees. In *CATS'00 (Canberra, 2000)*, volume 31(1) of *Electronic Notes in Theoretical Computer Science*, pages 143–166. Elsevier, 2000.

[12] M. Dezani-Ciancaglini, J. Tiuryn, and P. Urzyczyn. Discrimination by parallel observers: the algorithm. *Inform. and Comput.*, 150(2):153–186, 1999.

[13] E. Engeler. Algebras and combinators. *Algebra Universalis*, 13(3):389–392, 1981.

[14] G. Huet. An analysis of Böhm's theorem. *Theoret. Comput. Sci.*, 121:145–167, 1993.

[15] M. Hyland. A syntactic characterization of the equality in some models for the lambda calculus. *J. London Math. Soc. (2)*, 12(3):361–370, 1975/76.

[16] B. Jacobs and J. Rutten. A tutorial on (co)algebras and (co)induction. *Bulletin of EATCS*, 62:222–259, 1997.

[17] R. Kennaway, J. W. Klop, R. Sleep, and F.-J. de Vries. Infinite lambda calculus and Böhm models. In *Rewriting Techniques and Applications*, pages 257–270. Lecture Notes in Comput. Sci. 914. Springer-Verlag, 1995.

[18] R. Kennaway, J. W. Klop, R. Sleep, and F.-J. de Vries. Infinitary lambda calculus. *Theoretical Computer Science*, 175(1):93–125, 1997.

[19] R. Kennaway, V. van Oostrom, and F.-J. de Vries. Meaningless terms in rewriting. *J. Funct. Logic Programming*, Article 1:35 pp, 1999. (electronic) http://www.cs.tu-berlin.de/journal/jflp/articles/1999/A99-01/A99-01.html.

[20] J.-J. Lévy. An algebraic interpretation of the $\lambda\beta$K-calculus and a labellel λ-calculus. In *λ-Calculus and Computer Science Theory*, pages 147–165. Lecture Notes in Comput. Sci. 37. Springer-Verlag, 1975.

[21] J.-J. Lévy. An algebraic interpretation of the $\lambda\beta K$-calculus, and an application of a labelled λ-calculus. *Theoret. Comput. Sci.*, 2(1):97–114, 1976.

[22] G. Longo. Set-theoretical models of λ-calculus: theories, expansions, isomorphisms. *Ann. Pure Appl. Logic*, 24(2):153–188, 1983.

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

[24] J.-H. Morris. *Lambda calculus models of programming languages.* PhD thesis, M.I.T., 1968.

[25] D. Park. Concurrency and automata on infinite sequences. In *5th GI Conference on Theoretical Computer Science*, pages 167–183. Lecture Notes in Comput. Sci. 184. Springer Verlag, 1981.

[26] S. L. Peyton Jones. *The Implementation of Functional Programming Languages.* Prentice Hall, 1987.

[27] D. Sangiorgi. The lazy lambda calculus in a concurrency scenario. *Inform. and Comput.*, 111(1):120–153, 1994.

[28] D. Sangiorgi. Lévy-Longo Trees and Böhm Trees from encodings of λ-calculus into π-calculus. Notes, 1995.

[29] D. Sangiorgi. Interpreting functions as pi-calculus processes: a tutorial. Revised version of TR RR-3470, INRIA Sophia Antipolis. Available as ftp://ftp-sop/meije/theorie-par/davides/functionPItutorial.ps.gz, 1999.

[30] D. Scott. Continuous lattices. In *Toposes, algebraic geometry and logic (Conf., Dalhousie Univ., Halifax, N. S., 1971)*, pages 97–136. Lecture Notes in Math., Vol. 274. Springer, Berlin, 1972.

[31] D. Scott. Data types as lattices. *SIAM J. Comput.*, 5(3):522–587, 1976. Semantics and correctness of programs.

[32] C. P. Wadsworth. The relation between computational and denotational properties for Scott's D_∞-models of the lambda-calculus. *SIAM J. Comput.*, 5(3):488–521, 1976.