

DISCRIMINATING LAMBDA-TERMS USING CLOCKED BÖHM TREES*

JÖRG ENDRULLIS, DIMITRI HENDRIKS, JAN WILLEM KLOP, AND ANDREW POLONSKY

VU University Amsterdam, Dept. of Computer Science, De Boelelaan 1081A, 1081 HV Amsterdam
e-mail address: {j.endrullis,r.d.a.hendriks,j.w.klop,a.polonsky}@vu.nl

ABSTRACT. As observed by Intrigila [16], there are hardly techniques available in the λ -calculus to prove that two λ -terms are not β -convertible. Techniques employing the usual Böhm Trees are inadequate when we deal with terms having the same Böhm Tree (BT). This is the case in particular for fixed point combinators, as they all have the same BT. Another interesting equation, whose consideration was suggested by Scott [24], is $\text{BY} = \text{BYS}$, an equation valid in the classical model $\mathcal{P}\omega$ of λ -calculus, and hence valid with respect to BT-equality $=_{\text{BT}}$, but nevertheless the terms are β -inconvertible.

To prove such β -inconvertibilities, we employ ‘clocked’ BT’s, with annotations that convey information of the tempo in which the data in the BT are produced. Böhm Trees are thus enriched with an intrinsic clock behaviour, leading to a refined discrimination method for λ -terms. The corresponding equality is strictly intermediate between $=_{\beta}$ and $=_{\text{BT}}$, the equality in the model $\mathcal{P}\omega$. An analogous approach pertains to Lévy–Longo and Berarducci Trees.

Our refined Böhm Trees find in particular an application in β -discriminating fixed point combinators (fpc’s). It turns out that Scott’s equation $\text{BY} = \text{BYS}$ is the key to unlocking a plethora of fpc’s, generated by a variety of production schemes of which the simplest was found by Böhm, stating that new fpc’s are obtained by postfixing the term SI , also known as Smullyan’s Owl. We prove that all these newly generated fpc’s are indeed new, by considering their clocked BT’s. Even so, not all pairs of new fpc’s can be discriminated this way. For that purpose we increase the discrimination power by a precision of the clock notion that we call ‘atomic clock’.

1. INTRODUCTION

Böhm Trees constitute a well-known method to discriminate λ -terms M, N : if $\text{BT}(M)$ and $\text{BT}(N)$ are not identical, then M and N are β -inconvertible, $M \neq_{\beta} N$. But how do we prove β -inconvertibility of λ -terms with the same Böhm Tree? This question was raised in Scott [24] for the interesting equation $\text{BY} = \text{BYS}$ between terms that as Scott noted are presumably β -inconvertible, yet BT-equal ($=_{\text{BT}}$). Scott used his Induction Rule to prove

2012 ACM CCS: [Theory of computation]: Models of computation—Computability—Lambda calculus.

Key words and phrases: λ -calculus, Böhm Trees, β -inconvertibility, fixed point combinators.

* This is a modified and extended version of [13] which appeared in the proceedings of LICS 2010. The research has been partially funded by the Netherlands Organisation for Scientific Research (NWO) under grant numbers 612.000.934, 639.021.020, and 612.001.002.

that $BY = BYS$; instead we will employ below the infinitary λ -calculus with the same effect, but with more convenience for calculations as a direct generalization of finitary λ -calculus. Often one can solve such a β -discrimination problem by finding a suitable invariant for all the β -reducts of M, N . Below we will do this by way of preparatory example for the fixed point combinators (fpc's) in the Böhm sequence. But a systematic method for this discrimination problem has been lacking, and such a method is one of the two contributions of this paper.

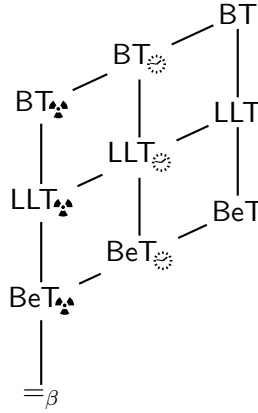


Figure 1: Comparison of (atomic) clock semantics and unlocked semantics. Higher means more identifications.

Actually, the need for such a strategic method was forced upon us, by the other contribution, because Scott's equation $BY = BYS$ turned out to be the key unlocking a plethora of new fpc's. The new generation schemes are of the form: if Y is an fpc, then $YP_1 \dots P_n$ is an fpc, abbreviated as $Y \Rightarrow YP_1 \dots P_n$. So $\square P_1 \dots P_n$ is an 'fpc-generating' vector, and can be considered as a building block to make new fpc's. But are they indeed new? A well-known example of a (singleton)-fpc-generating vector is $\square \delta$, where $\delta = SI$, giving rise when starting from Curry's fpc to the Böhm sequence of fpc's. Here another interesting equation is turning up, namely $Y = Y\delta$, for an arbitrary fpc Y , considered by Statman and Intrigila [16]. In fact, it is implied by Scott's equation, for an arbitrary fpc Y :

$$BY = BYS \implies BYI = BYSI \iff Y = Y\delta$$

The first equation $BY = BYS$ will yield many new fpc's, built in a modular way; the last equation $Y = Y\delta$ addresses the question whether they are indeed new. Finding ad hoc invariant proofs for their novelty is too cumbersome. But fortunately, it turns out that although the new fpc's all have the same BT, namely $\lambda f.f^\omega$, they differ in the way this BT is formed, in the 'tempo of formation', where the ticks of the clock are head reduction steps. We can thus discern a clock-like behaviour of BT's, and we refine BT's to 'clocked' BT's by annotating them with this information. These then enable us to discriminate the terms in question.

This refined discrimination method works best for what we call 'simple' terms (or, for terms that reduce to simple terms). A term is 'simple' if its reduction to the Böhm Tree does not duplicate redexes. The class of simple terms is still fairly extensive; it includes all fpc's that are constructed in the modular way that we present, thereby solving our novelty problem. In fact, we gain some more ground: though our discrimination method works

best for pairs of simple terms, it can also fruitfully be applied to compare a simple term with a non-simple term, and with some more effort, we can even compare and discriminate non-simple terms; see Section 5 for an example.

Even so, many pairs of fpc's cannot yet be discriminated, because they not only have the same BT, they also have the same clocked BT. Therefore, in a final grading up of the precision, we introduce 'atomic clocks', where the actual position of a head reduction step is administrated. All this pertains not only to the BT-semantics, but also to Lévy–Longo Trees (LLT) (or lazy trees), and Berarducci Trees (BeT) (or syntactic trees). Many problems stay open, in particular problems generalizing the equation of Statman and Intrigila, when arbitrary fpc's are considered.

Overview. After defining preliminary notions in Section 2, in Section 3 we are concerned with constructing new fpc's from old, by some generating schemes. In Section 4 we define clocked Böhm Trees. The main results here are Theorems 4.6 and 4.11. The first states that if no reduct of M has a clock that is at least as fast as the clock of N , then M and N are unconvertible, whereas Theorem 4.11 states that if M is a simple term then it suffices that the clock M is not eventually faster than the clock of N . In the paper we are mainly concerned with λ -terms which have simple reducts. An exception is Section 5, where we answer a question of Plotkin (see Related Work below) which involves arbitrary (not necessarily simple) fpc's. Another elaborate example is given in Section 6, where we compute the clocks of three enumerators for Combinatory Logic. In Section 7, we give a refinement of the clock method by not only recording the number of head reduction steps but also their positions. As an application we show that every combination of the fixed point generating vectors $\square(SS)S^{\sim n}$ introduced in Section 3 give rise to *new* fpc's. We briefly mention how the theory can be extended to the other well-known semantics of λ -calculus, namely Lévy–Longo and Berarducci Trees, in Section 8. We conclude in Section 9 with directions for future research.

Related Work. The present paper is an extension and elaboration of [13]. In particular, as an example application of the main theorems (Theorem 4.6), we now answer the following question of Plotkin [22]:

Is there a fixed point combinator Y such that $Y(\lambda z.fzz) =_{\beta} Y(\lambda x.Y(\lambda y.fxy))$?

An idea similar to clocked Böhm Trees is employed in the excellent paper [2], pointed out to us by Tarmo Uustalu at the 2010 LICS conference in Edinburgh. In [2], Aehlig and Joachimski study continuous normalization of the coinductive λ -calculus (see [19, 17]) extended with a void 'wait' constructor \mathcal{R} . This extra constructor is returned whenever the head constructor of a term cannot immediately be read off from the argument. If the head constructor of an application rs is to be found, it depends on a recursive call to investigate whether r is a variable, an abstraction or an application again; in this case an \mathcal{R} is returned. Thus it is guaranteed that the procedure of building a non-wellfounded term over the extended grammar is productive by making the recursion guarded. Every \mathcal{R} is matched either by a β -step necessary to reach the normal form (as represented by the Böhm Tree) or by an application node in the Böhm Tree. When $r = \lambda x.r'$ then \mathcal{R} matches a β -step and so s is used in the substitution $r'[x := s]$. Or we find that r is a variable and so \mathcal{R} matches an application node in the Böhm Tree. So the number of \mathcal{R} 's in the normal form of a term t is precisely related to the number of reduction steps and

the ‘size’ of the resulting Böhm Tree. Summarizing, similar to the clocked Böhm Trees that we introduce in Section 4, in [2] Böhm Trees are enriched with information about how the tree is constructed. However, in [2] the refinement is used for proving their normalization function to be continuous, and not for discriminating λ -terms, the goal we pursue here.

In [15] a heuristic procedure in finitary λ -calculus is given to construct fpc’s in a uniform way.

2. PRELIMINARIES

To make this paper moderately self-contained, and to fix notations, we lay out some ingredients. For λ -calculus we refer to [3] and [7]. For an introduction to Böhm, Berarducci and Lévy–Longo Trees, we refer to [3, 1, 8, 5].

Definition 2.1. Let \mathcal{X} be an countably infinite set of variables. The set $Ter(\lambda)$ of *finite λ -terms* is defined inductively by the following grammar:

$$M ::= x \mid \lambda x.M \mid M \cdot M \quad (x \in \mathcal{X})$$

We use x, y, z, \dots for variables, and M, N, \dots to range over the elements of $Ter(\lambda)$.

Thus λ -terms are variables, abstractions or applications. Usually we suppress the application symbol in a term $M \cdot N$ and just write (MN) . We also adopt the usual conventions for omitting brackets, i.e., we let application associate to the left, so that $N_1 N_2 \dots N_k$ denotes $(\dots (N_1 N_2) \dots N_k)$, and we let abstraction associate to the right: $\lambda x_1 \dots x_n.M$ stands for $(\lambda x_1.(\dots (\lambda x_n.(M))))$.

Definition 2.2. The set $Ter^\infty(\lambda)$ of (finite and) *infinite λ -terms* is defined by interpreting the grammar from Definition 2.1 *coinductively*, that is, $Ter^\infty(\lambda)$ is the largest set X such that every element $M \in X$ is either a variable x , an abstraction $\lambda x.M'$ or an application $M_1 M_2$ with $M', M_1, M_2 \in X$. (See further [23] for a precise treatment of coinductive definition and proof principles.)

Böhm, Lévy–Longo and Berarducci Trees are infinite λ -terms with an additional clause for \perp in the grammar, where \perp stands for the different notions of ‘undefined’ in these semantics, see Definitions 4.1, 8.1 and 8.2.

Definition 2.3. The relation \rightarrow_β on $Ter(\lambda)$ or $Ter^\infty(\lambda)$, called *β -reduction*, is the compatible closure (i.e., closure under term formation) of the β -rule:

$$(\lambda x.M)N \rightarrow M[x := N] \quad (\beta)$$

where $M[x := N]$ denotes the result of substituting N for all free occurrences of x in M . Furthermore, we write \rightarrow_β^n for the n -fold composition of \rightarrow_β , defined by $M \rightarrow_\beta^0 M$, and $M \rightarrow_\beta^{n+1} N$ if $M \rightarrow_\beta P$ and $P \rightarrow_\beta^n N$ for some $P \in Ter(\lambda)$. We use \rightarrow_β to denote the reflexive-transitive closure of \rightarrow_β , $M \twoheadrightarrow_\beta N$ iff $\exists n. M \rightarrow_\beta^n N$; we let $\rightarrow_\beta^{\overline{\overline{\beta}}}$ denote the reflexive closure of \rightarrow_β , $\rightarrow_\beta^{\overline{\overline{\beta}}} = \rightarrow_\beta^0 \cup \rightarrow_\beta^1$. We write $M =_\beta N$ to denote that M is β -convertible with N , i.e., $=_\beta$ is the equivalence closure of \rightarrow_β . For syntactic equality (modulo renaming of bound variables), we use \equiv .

Apart from the ‘many-steps’ relation \rightarrow_β , we introduce ‘multi-steps’ \twoheadrightarrow_β [30, 3], which arise from complete developments.

Definition 2.4. We write \multimap_{β} for *multi-steps*, that is, complete developments of a set of redex occurrences in a term. A *development* of a set of redex occurrences U in a term contracts exclusively descendants (residuals) of U , and it is called *complete* if there are no residuals of U left.

A complete development of U can alternatively be viewed as contracting all redex occurrences in U in an inside-out fashion.

We will often omit the subscript β in \rightarrow_{β} , $\twoheadrightarrow_{\beta}$ and \multimap_{β} .

Definition 2.5. A λ -term M is called a *normal form* if there exists no N with $M \rightarrow N$. We say that a term M *has a normal form* if it reduces to one. For λ -terms M having a normal form we write \underline{M} for the unique normal form N with $M \twoheadrightarrow N$ (uniqueness follows from confluence of the λ -calculus).

Some commonly used combinators are:

$$\mathbf{I} = \lambda x.x \quad \mathbf{K} = \lambda xy.x \quad \mathbf{S} = \lambda xyz.xz(yz) \quad \mathbf{B} = \lambda xyz.x(yz)$$

Definition 2.6. A *position* is a sequence over $\{0, 1, 2\}$. For a finite or infinite λ -term M , the *subterm* $M|_p$ of M at *position* p is defined by:

$$\begin{aligned} M|_{\epsilon} &= M & (MN)|_{1p} &= M|_p \\ (\lambda x.M)|_{0p} &= M|_p & (MN)|_{2p} &= N|_p \end{aligned}$$

$\text{Pos}(M)$ is the set of positions p such that $M|_p$ is defined.

Definition 2.7.

- (1) A term Y is an *fpc* if $Yx =_{\beta} x(Yx)$.
- (2) An fpc Y is *k-reducing* if $Yx \rightarrow^k x(Yx)$.
- (3) An fpc Y is *reducing* if Y is k -reducing for some $k \in \mathbb{N}$.
- (4) A term Z is a *weak fpc (wfpc)* if $Zx =_{\beta} x(Z'x)$ where Z' is again a wfpc.

The definition of weak fpc's in (4) is essentially coinductive [23], that is, implicitly employing a 'largest set' semantics. In long form, the definition means the following: the set of weak fpc's is the largest set $W \subseteq \text{Ter}(\lambda)$ such that for every $Z \in W$ we have $Zx =_{\beta} x(Z'x)$ for some $Z' \in W$.

A wfpc is alternatively defined as a term having the same Böhm Tree as an fpc, namely $\lambda x.x^{\omega} \equiv \lambda x.x(x(x(\dots)))$. Weak fpc's are known in foundational studies of type systems as *looping combinators*; see, e.g., [10] and [14].

Example 2.8. Define by double recursion, Z and Z' such that $Zx = x(Z'x)$ and $Z'x = x(Zx)$. Then Z, Z' are both wfpc's, and $Zx = x(x(Zx))$. So Z delivers its output twice as fast as an ordinary fpc, but the generator flipflops.

As to 'double recursion', [20] collects several proofs of the double fixed point theorem, including some in [3, 28].

Definition 2.9.

- (1) A *head reduction step* \rightarrow_h is a β -reduction step of the form:
 $\lambda x_1 \dots x_n.(\lambda y.M)N_1 \dots N_m \rightarrow \lambda x_1 \dots x_n.(M[y := N])N_1 \dots N_m$ with $n, m \geq 0$.
- (2) Accordingly, a *head normal form (hnf)* is a λ -term of the form:
 $\lambda x_1 \dots \lambda x_n.yN_1 \dots N_m$ with $n, m \geq 0$.

- (3) A *weak head normal form (whnf)* is an hnf or an abstraction, that is, a whnf is a term of the form $xM_1 \dots M_m$ or $\lambda x.M$.
- (4) A term *has a (weak) hnf* if it reduces to one.
- (5) We call a term *root-stable* if it does not reduce to a redex: $(\lambda x.M)N$. A term is called *root-active* if it does not reduce to a root-stable term.

Infinitary λ -calculus $\lambda^\infty\beta$. We will only use the infinitary λ -calculus $\lambda^\infty\beta$ for some simple calculations such as $(\lambda ab.(ab)^\omega)I =_{\lambda^\infty\beta} \lambda b.(Ib)^\omega =_{\lambda^\infty\beta} \lambda b.b^\omega$. Here M^ω denotes the infinite λ -term $M(M(M(\dots)))$ obtained as the solution of $M^\omega = MM^\omega$. For a proper setup of $\lambda^\infty\beta$ we refer to [6, 19, 18, 5]. In a nutshell, $\lambda^\infty\beta$ extends finitary λ -calculus by admitting infinite λ -terms, the set of which is called $Ter^\infty(\lambda)$, and infinite reductions (in [18, 5] possibly transfinitely long, in [6] of length $\leq \omega$). Limits of infinite reduction sequences are obtained by a strengthening of Cauchy-convergence, stipulating that the depth of contracted redexes must tend to infinity. The $\lambda^\infty\beta$ -calculus is not infinitary confluent (CR^∞), but still has unique infinite normal forms (UN^∞). Böhm Trees (BT's) without \perp are infinite normal forms in $\lambda^\infty\beta$. But beware, the reverse does not hold, e.g. $\lambda x.(\lambda x.(\lambda x.\dots))$ is an infinite normal form, but not a BT; it is in fact an LLT (Lévy–Longo Tree, and also a BeT (Berarducci Tree). The notions BT, LLT, BeT are defined e.g. in [5], and in [8]. These notions are also defined in Sections 4 and 8, via their clocked versions.

Definition 2.10. For terms A, B we define $AB^{\sim n}$ and $A^n B$:

$$\begin{aligned} AB^{\sim 0} &= A & A^0 B &= B \\ AB^{\sim n+1} &= ABB^{\sim n} & A^{n+1} B &= A(A^n B) \end{aligned}$$

A context of the form $\square B^{\sim n}$ is called a *vector*. For the vector notation, it is to be understood that term formation gets highest priority, i.e., $ABC^{\sim n} = (AB)C^{\sim n}$.

3. FIXED POINT COMBINATORS

The theory of sage birds (technically called fixed point combinators) is a fascinating and basic part of combinatory logic; we have only scratched the surface.

R. Smullyan [28].

3.1. The Böhm Sequence. There are several ways to make fpc's. For heuristics behind the construction of Curry's fpc $Y_0 = \lambda f.\omega_f\omega_f$, with $\omega_f = \lambda x.f(xx)$, and Turing's fpc $Y_1 = \eta\eta$ with $\eta = \lambda x.f.f(xxf)$, see [3, 20].

It is well-known, as observed by C. Böhm [9, 3], that the class of fpc's coincides exactly with the class of fixed points of the peculiar term $\delta = \lambda ab.b(ab)$, convertible with SI. The notation δ is convenient for calculations and stems from [16]. This term also attracted the attention of R. Smullyan, in his beautiful fable about fpc's figuring as birds in an enchanted forest: "An extremely interesting bird is the owl O defined by the following condition: $Oxy = y(xy)$." [28]. We will return to the Owl in Remark 3.4 below.

Thus the term $Y\delta$ is an fpc whenever Y is. It follows that starting with Y_0 , Curry's fpc, we have an infinite sequence of fpc's $Y_0, Y_0\delta, Y_0\delta\delta, \dots, Y_0\delta^{\sim n}, \dots$; we call this sequence the 'Böhm sequence'. Note that $Y_0\delta =_\beta \eta\eta$, justifying the overloaded notation Y_1 . Now

the question is whether all these ‘derived’ fpc’s are really new, in other words, whether the sequence is free of duplicates. This is *Exercise 6.8.9 in [3].

Note that we could also have started the sequence from another fpc than Curry’s. Now for the sequence starting from an *arbitrary* fpc Y , it is actually an open problem whether that sequence of fpc’s $Y, Y\delta, Y\delta\delta, \dots, Y\delta^{\sim n}, \dots$ is free of repetitions. All we know, applying Intrigila’s theorem, Theorem 3.3 below, is that no two consecutive fpc’s in this sequence are convertible. But let us first consider the Böhm sequence.

Definition 3.1. The *Böhm sequence* is the sequence $(Y_n)_{n \geq 0}$ where Y_n is defined by

$$Y_0 = \lambda f. \omega_f \omega_f \qquad Y_n = \eta \eta \delta^{\sim(n-1)} \quad (n > 0)$$

We show that the Böhm sequence contains no duplicates by determining the set of reducts of every Y_n . For Y_3 ¹ the head reduction is displayed in Figure 2, but this is by no means the whole reduction graph. For future reference we note that the head reduction diagram suggests a ‘clock behaviour’.

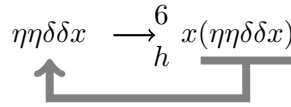


Figure 2: Head reduction of $Y_3 x$.

Theorem 3.2. The Böhm sequence $(Y_n)_{n \geq 0}$ contains no duplicates.

Proof. We define languages $\mathcal{L}_n \subseteq \text{Ter}(\lambda)$ as follows:

$$\begin{aligned} \mathcal{L}_0 &::= \lambda f. f^k(\omega_f \omega_f) & (k \geq 0) \\ \mathcal{L}_1 &::= \eta \eta \mid \lambda f. f^k(\mathcal{L}_1 f) & (k > 0) \\ \mathcal{L}_n &::= \mathcal{L}_{n-1} \delta \mid \lambda b. b^k(\mathcal{L}_n b) \mid \delta \mathcal{L}_n & (n > 1, k > 0) \end{aligned}$$

We show that:

- (1) $Y_n \in \mathcal{L}_n$;
- (2) \mathcal{L}_n is closed under β -reduction; and
- (3) \mathcal{L}_n and \mathcal{L}_m are disjoint, for $n \neq m$.

Then it follows that \mathcal{L}_n contains the set of \rightarrow_β -reducts of Y_n , and using (iii) this implies that $Y_n \not\equiv_\beta Y_m$ for all $n \neq m$.

For (i) note that $Y_0 \in \mathcal{L}_0$, and $Y_n = \eta \eta \delta^{\sim(n-1)}$ which is in \mathcal{L}_n by induction on $n > 0$.

We show (ii): if $M \in \mathcal{L}_n$ and $M \rightarrow N$, then $N \in \mathcal{L}_n$. Using induction, we do not need to consider cases where the rewrite step is inside a variable of the grammar. We write \mathcal{L}_n in terms as shorthand for a term $M \in \mathcal{L}_n$.

(\mathcal{L}_0) We have $\lambda f. f^k(\omega_f \omega_f) \rightarrow \lambda f. f^{k+1}(\omega_f \omega_f) \in \mathcal{L}_0$.

(\mathcal{L}_1) We have $\eta \eta \rightarrow \lambda f. f(\eta \eta f) \in \mathcal{L}_1$,
and $\lambda f. f^k(\lambda f. f^\ell(\mathcal{L}_1 f) f) \rightarrow \lambda f. f^{k+\ell}(\mathcal{L}_1 f) \in \mathcal{L}_1$.

¹Actually Figure 2 displays $Y_3 x$. We will frequently consider Yx instead of Y as then the repetition is immediate, and because we have that if $Mx \not\equiv_\beta M'x$ then also $M \not\equiv_\beta M'$.

(\mathcal{L}_n) Case 1: $\lambda f.f^k(\mathcal{L}_1 f)\delta \rightarrow \delta^k(\mathcal{L}_1 \delta) \in \mathcal{L}_n$ for $n = 2$, and $(\lambda b.b^k(\mathcal{L}_{n-1} b))\delta \rightarrow \delta^k(\mathcal{L}_{n-1} \delta) \in \mathcal{L}_n$ for $n > 2$.

Case 2: $\lambda b.b^k(\lambda c.c^\ell(\mathcal{L}_n c)b) \rightarrow \lambda b.b^{k+\ell}(\mathcal{L}_n b) \in \mathcal{L}_n$.

Case 3: $\delta \mathcal{L}_n \rightarrow \lambda b.b(\mathcal{L}_n b) \in \mathcal{L}_n$.

For $n \neq m$, $n > 1$, (iii) follows by counting the number of passive δ 's, that is, the number of occurrences of the form $P\delta$ for some P . To see that $\mathcal{L}_0 \cap \mathcal{L}_1 = \emptyset$, note that if $M \in \mathcal{L}_1$ is an abstraction, then $M \equiv \lambda f.f^k(Pf)$ containing a subterm Pf which is never the case in \mathcal{L}_0 . \square

A very interesting theorem involving δ was proved by B. Intrigila, affirming a conjecture by R. Statman.

Theorem 3.3 (Intrigila [16]). *There is no ‘double’ fixed point combinator. That is, for no fpc Y we have $Y\delta =_\beta Y$.*

Remark 3.4 (Smullyan’s Owl $\text{SI} =_\beta \delta = \lambda ab.b(ab)$).

We collect some salient facts and questions.

- (1) If Z is a wfpc, both δZ and $Z\delta$ are wfpc’s [28].
- (2) Call an applicative combination of δ ’s a δ -term, that is, a term solely built from δ and application. In spite of δ ’s simplicity, not all δ -terms are strongly normalizing (SN). An example of a δ -term with infinite reduction is $\delta\delta(\delta\delta)$ (Johannes Waldman, Hans Zantema, personal communication, 2007).
- (3) Let M be a non-trivial δ -term, i.e., not a single δ . Then M is SN iff M contains exactly one occurrence of $\delta\delta$. Furthermore, if δ -terms M, M' are SN, then they are convertible if and only if M, M' have the same length [20]. Here the length of a δ -term is the number δ ’s in the term.
- (4) Convertibility is decidable for δ -terms [29].
- (5) We define $\Delta = \delta^\omega$, so $\Delta \equiv \delta\Delta$. Then, the infinite λ -term Δ is an fpc: $\Delta x \equiv \delta\Delta x \rightarrow x(\Delta x)$. The term Δ can be normalized: $\Delta \rightarrow_\omega \lambda f.f^\omega$. There are many more infinitary fpc’s, e.g., for every n , the infinite term $(\text{SS})^\omega \text{S}^{\sim n} \text{I}$ is one, as will be clear from the sequel.
- (6) The term $\delta\delta(\delta\delta)$ has no hnf, and hence its Böhm Tree is trivial, $\text{BT}(\delta\delta(\delta\delta)) \equiv \perp$. However, its Berarducci Tree is not trivial. Zantema remarked that δ -terms, even infinite ones, such as $\Delta\Delta$, are “top-terminating” (Zantema considered the applicative rule for δ only — we expect that his observation remains valid for the $\lambda\beta$ -version).
- (7) Is Intrigila’s theorem also valid for wfpc’s: for no wfpc Z we have $Z\delta =_\beta Z$?

3.2. The Scott Sequence. In [24, p. 360] the equation $\text{BY}_0 = \text{BY}_0\text{S}$ is mentioned as an interesting example of an equation not provable in $\lambda\beta^2$, while easily provable with Scott’s Induction Rule. Scott mentions that he expects that using ‘methods of Böhm’ the non-convertibility in $\lambda\beta$ can be established, but that he did not attempt a proof. On the other hand, with the induction rule the equality is easily established. We will not consider Scott’s Induction Rule, but we will be working in the infinitary lambda calculus, $\lambda^\infty\beta$. It is readily verified that in $\lambda^\infty\beta$ we have:

$$\text{BY}_0 =_{\lambda^\infty\beta} \text{BY}_0\text{S} =_{\lambda^\infty\beta} \lambda ab.(ab)^\omega$$

²This equation is also discussed in [11].

Proposition 3.5. $\text{BY}_0 \neq_{\beta} \text{BY}_0\text{S}$

Proof. Postfixing the combinator I yields BY_0I and BY_0SI . Now $\text{BY}_0\text{I} =_{\beta} \text{Y}_0$ and $\text{BY}_0\text{SI} =_{\beta} \text{Y}_0(\text{SI}) = \text{Y}_1$. Because $\text{Y}_0 \neq_{\beta} \text{Y}_1$ (Theorem 3.2), the result follows. \square

In the same way we can strengthen this non-equation to all fpc's Y , using Theorem 3.3.

Remark 3.6.

- (1) The idea of postfixing an I is suggested by the Böhm Tree $\lambda ab.(ab)^{\omega}$ of BY and BYS . Namely, in $\lambda^{\infty}\beta$ we calculate: $(\lambda ab.(ab)^{\omega})\text{I} = \lambda b.(lb)^{\omega} = \lambda b.b^{\omega}$ which is the Böhm Tree of any fpc.
- (2) Interestingly, Scott's equation $\text{BY} = \text{BYS}$ implies the equation of Statman and Intrigila, $Y = Y\delta$ as one readily verifies, as in the proof of Proposition 3.5.

Actually, the comparison between the terms BY and BYS has more in store for us than just providing an example that the extension from finitary lambda calculus $\lambda\beta$ to infinitary lambda calculus $\lambda^{\infty}\beta$ is not conservative. The BT-equality of BY and BYS suggests looking at the whole sequence $\text{BY}, \text{BYS}, \text{BYSS}, \dots, \text{BYS}^{\sim n}, \dots$. All these terms have the Böhm Tree $\lambda ab.(ab)^{\omega}$, and hence they are not fpc's. But postfixing an I turns them into fpc's.

Definition 3.7. The *Scott sequence* is defined by:

$$\text{BY}_0\text{I}, \text{BY}_0\text{SI}, \text{BY}_0\text{SSI}, \dots, \text{BY}_0\text{S}^{\sim n}\text{I}, \dots$$

We write $\text{U}_n = \text{BY}_0\text{S}^{\sim n}\text{I}$ for the n -th term in this sequence.

The Scott sequence concurs with the Böhm sequence of fpc's only for the first two elements, and then splits off with different fpc's. But there is a second surprise. In showing that U_n is an fpc, we find as a bonus the fpc-generating vector $\square(\text{SS})\text{S}^{\sim n}\text{I}$ (which does preserve the property of fpc's to be reducing).

Theorem 3.8. *Let Y be a k -reducing fpc and $n \geq 0$. Then:*

- (1) $\text{BYS}^{\sim n}\text{I}$ is a (non-reducing) fpc;
- (2) $Y(\text{SS})\text{S}^{\sim n}\text{I}$ is a $(k + 3n + 7)$ -reducing fpc.

The proof of Theorem 3.8 is easy: see the next example.

Example 3.9. Let Y be a k -reducing fpc. Then:

$$\begin{aligned} \text{BYSSSI}x &\rightarrow_h Y(\text{SS})\text{SI}x \rightarrow_h^k \text{SS}(Y(\text{SS}))\text{SI}x \\ &\rightarrow_h^3 \text{SS}(Y(\text{SS})\text{S})\text{I}x \rightarrow_h^3 \text{SI}(Y(\text{SS})\text{SI})x \\ &\rightarrow_h^3 \text{I}x(Y(\text{SS})\text{SI}x) \rightarrow_h^1 x(Y(\text{SS})\text{SI}x) \end{aligned}$$

This shows that $\text{BYS}^{\sim 3}\text{I}$ is a non-reducing fpc, and at the same time that $Y(\text{SS})\text{SI}x$ is reducing.

3.3. Generalized Generation Schemes. The schemes mentioned in Theorem 3.8 for generating new fixed points from old, are by no means the only ones. There are in fact infinitely many of such schemes. They can be obtained analogously to the ones that we extracted above from the equation $\text{BY} = \text{BYS} = \lambda ab.(ab)^{\omega}$, or the equation $Mab = ab(Mab)$. We only treat the case for $n = 3$: consider the equation $Nabc = abc(Nabc)$. Then every solution N is again a 'pre-fpc', namely $N\text{II}$ is an fpc: $N\text{II}x =_{\beta} \text{II}x(N\text{II}x) =_{\beta} x(N\text{II}x)$.

- (1) $Nabc = Y(abc)$, which yields $N = (\lambda yabc.y(abc)))Y = (\lambda yabc.BBByabc)Y$. We obtain $N = BBBY$.
- (2) $N = Y\xi$ with $\xi = \lambda nabc.abc(nabc)$, yielding the fpc-generating vector $\square\xi\mathbb{I}$.
- (3) $Nabc = abc(Nabc) = S(ab)(Nab)c$. So we take $Nab = S(ab)(Nab)$, which yields $Nab = Y(S(ab)) = BBBY(BS)ab$. So $N = BBBY(BS)$, and thus we find the equation $BBBY = BBBY(BS)$, in analogy with the equation $BY = BYS$ above.

Also this equation spawns lots of fpc's as well as fpc-generating vectors. Let's abbreviate BS by A. First one forms the sequence

$$BBBY, BBBYA, BBBYAA, BBBYAAA, \dots$$

These terms all have the BT $\lambda abc.abc(abc)^\omega$. They are not yet fpc's, but only 'pre-fpc's'. But after postfixing this time $\dots\mathbb{I}$ we do again obtain a sequence of fpc's:

$$BBBY\mathbb{I}, BBBY\mathbb{A}\mathbb{I}, BBBY\mathbb{A}\mathbb{A}\mathbb{I}, \dots$$

Again the first two coincide with Y_0, Y_1 , but the series deviates not only from the Böhm sequence but also from the Scott sequence above. As above, the proof that a term in this sequence is indeed an fpc, yields an fpc-generating vector. In this way we find as a new fpc-generating scheme

$$Y \Rightarrow Y(\mathbb{A}\mathbb{A}\mathbb{A})\mathbb{A}^{\sim n}\mathbb{I}$$

We can derive many more of these schemes by proceeding with solving the general equation $Na_1a_2\dots a_n = a_1a_2\dots a_n(Na_1a_2\dots a_n)$, bearing in mind the following proposition.

Proposition 3.10. *If N is a term satisfying $Na_1a_2\dots a_n = a_1a_2\dots a_n(Na_1a_2\dots a_n)$, then $N\mathbb{I}^{\sim(n-1)}$ is an fpc.*

We finally mention an fpc-generating scheme with 'dummy parameters':

$$Y \Rightarrow YQP_1\dots P_n$$

where P_1, \dots, P_n are arbitrary (dummy) terms, and $Q = \lambda yp_1\dots p_nx.x(yp_1\dots p_nx)$.

4. CLOCK BEHAVIOUR OF LAMBDA TERMS

As we have seen, there is vast space of fpc's and there are many ways to derive new fpc's. The question is whether all these fpc's are indeed new. So we have to prove that they are not β -convertible.

For the Böhm sequence we did this by an ad hoc argument based on a syntactic invariant; and this method works fine to establish lots of non-equations between the alleged 'new' fpc's that we constructed above. Still, the question remains whether there are not more 'strategic' ways of proving such inequalities.

In this section we propose a more strategic way to discriminate terms with respect to β -conversion. The idea is to extract from a λ -term more than just its BT, but also how the BT was formed; one could say, in what tempo, or in what rhythm. A BT is formed from static pieces of information, but these are rendered in a clock-wise fashion, where the ticks of the internal clock are head reduction steps.

In the sequel we write $[k]M$ for the term M where the root is *annotated with $k \in \mathbb{N}$* . Here, term formation binds stronger than annotation $[k]$. For example $[k]MN$ stands for the term $[k](MN)$ (that is, annotating the (non-displayed) application symbol in-between M and N , in contrast to $([k]M)N$). Moreover, for an annotated term M we use $\lfloor M \rfloor$ to

denote the term obtained from M by dropping all annotations (including annotations of subterms).

Definition 4.1 (Clocked Böhm Trees). The *clocked Böhm Tree* $\text{BT}_{\circlearrowleft}(M)$ of a λ -term M is an annotated (potentially infinite) term coinductively defined as follows. If M has no hnf, then define $\text{BT}_{\circlearrowleft}(M)$ as \perp . Otherwise, there is a head reduction $M \rightarrow_h^k \lambda x_1 \dots \lambda x_n . y M_1 \dots M_m$ to hnf. Then we define $\text{BT}_{\circlearrowleft}(M)$ as the term $[k] \lambda x_1 \dots \lambda x_n . y \text{BT}_{\circlearrowleft}(M_1) \dots \text{BT}_{\circlearrowleft}(M_m)$.

The notions of subterms and positions (see Definition 2.6) carry over to annotated terms, in particular clocked Böhm Trees, in a straightforward way. The (non-clocked) Böhm Tree of a λ -term M can be obtained by dropping the annotations: $\text{BT}(M) = \lfloor \text{BT}_{\circlearrowleft}(M) \rfloor$.

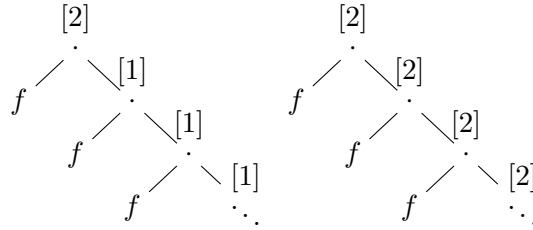


Figure 3: Clocked Böhm Trees of $Y_0 f$ and $Y_1 f$.

Let us consider the fpc's Y_0 of Curry and Y_1 of Turing. We have $Y_0 \equiv \lambda f . \omega_f \omega_f$ where $\omega_f \equiv \lambda x . f(xx)$, and

$$\omega_f \omega_f \rightarrow_h^1 f(\omega_f \omega_f)$$

Therefore we obtain $\text{BT}_{\circlearrowleft}(Y_0 f) = [2] f \text{BT}_{\circlearrowleft}(\omega_f \omega_f)$, and $\text{BT}_{\circlearrowleft}(\omega_f \omega_f) = [1] f \text{BT}_{\circlearrowleft}(\omega_f \omega_f)$.

For $Y_1 \equiv \eta \eta$ where $\eta \equiv \lambda x . \lambda f . f(xxf)$ we get:

$$Y_1 f \equiv \eta \eta f \rightarrow_h^2 f(\eta \eta f)$$

Hence, $\text{BT}_{\circlearrowleft}(Y_1 f) = [2] f \text{BT}_{\circlearrowleft}(Y_1 f)$. Figure 3 displays the clocked Böhm Trees of $Y_0 f$ (left) and $Y_1 f$ (right).

The following definition captures the well-known Böhm equality of λ -terms.

Definition 4.2. λ -terms M and N are *BT-equal*, denoted by $M =_{\text{BT}} N$, if $\text{BT}(M) \equiv \text{BT}(N)$.

If M and N are not BT-equal, then $M \neq_{\beta} N$. Consequently, if for some λ -term F , we have $\text{BT}(MF) \neq \text{BT}(NF)$, then $M \neq_{\beta} N$.

Below, we refine this approach by comparing the clocked Böhm Trees $\text{BT}_{\circlearrowleft}(M)$ and $\text{BT}_{\circlearrowleft}(N)$ instead of the ordinary (non-clocked) Böhm Trees. In general, $\text{BT}_{\circlearrowleft}(M) \neq \text{BT}_{\circlearrowleft}(N)$ does not always imply that $M \neq_{\beta} N$. Nevertheless, for a large class of λ -terms, called ‘simple’ below, this implication will turn out to be true.

In the following definition, we lift relations over natural numbers to relations over clocked Böhm Trees.

Definition 4.3. Let T_1 and T_2 be clocked Böhm Trees, and $R \subseteq \mathbb{N} \times \mathbb{N}$. We define the following notations:

- (1) For $p \in \mathcal{P}os(T_1) \cap \mathcal{P}os(T_2)$ we let $T_1 R_p T_2$ denote that either both $T_1|_p$ and $T_2|_p$ are not annotated, or both are annotated and then $T_1|_p \equiv [k_1]T_1'$ and $T_2|_p \equiv [k_2]T_2'$ with $k_1 R k_2$.
- (2) We write $T_1 R T_2$ if $\lfloor T_1 \rfloor \equiv \lfloor T_2 \rfloor$ and $T_1 R_p T_2$ for every $p \in \mathcal{P}os(T_1)$.
- (3) We write $T_1 R_{\exists} T_2$, and say that R holds eventually, if $\lfloor T_1 \rfloor \equiv \lfloor T_2 \rfloor$ and there exists a depth level $\ell \in \mathbb{N}$ such that $T_1 R_p T_2$ for all positions $p \in \mathcal{P}os(T_1)$ with $|p| \geq \ell$.

Definition 4.4. For λ -terms M and N we say:

- (1) M improves N globally if $\mathbf{BT}_{\odot}(M) \leq \mathbf{BT}_{\odot}(N)$;
- (2) M improves N eventually if $\mathbf{BT}_{\odot}(M) \leq_{\exists} \mathbf{BT}_{\odot}(N)$;
- (3) M matches N eventually if $\mathbf{BT}_{\odot}(M) =_{\exists} \mathbf{BT}_{\odot}(N)$.

The following proposition states that the ordering $>_{\odot}$ on λ -terms defined by $M >_{\odot} N$ if and only if $\mathbf{BT}_{\odot}(M) \geq \mathbf{BT}_{\odot}(N)$ is a ‘semi-model’ of β -reduction [21]. We leave this for future research.

Proposition 4.5. *Clocks are accelerated under reduction, that is, if $M \rightarrow N$, then the reduct N improves M globally ($\mathbf{BT}_{\odot}(M) \geq \mathbf{BT}_{\odot}(N)$). Dually, clocks slow down under expansion (the reverse of reduction).*

Proof. We proceed by an elementary diagram construction. Whenever we have co-initial steps $M \rightarrow_h M_1$ and $M \rightarrow_h M_2$, then by orthogonal projection [30] there exist joining steps $M_1 \rightarrow_h M'$ and $M_2 \rightarrow_h M'$. Note that the head step $M \rightarrow_h M_1$ cannot be duplicated, only erased in case of an overlap. This leads to the elementary diagram displayed in Figure 4.

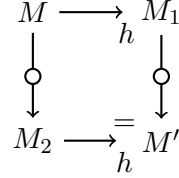


Figure 4: Elementary diagram.

We have $\rightarrow \subseteq \rightarrow^*$. By induction on the length of the rewrite sequence \rightarrow^* it suffices to show that $M \rightarrow N$ implies $\mathbf{BT}_{\odot}(M) \geq \mathbf{BT}_{\odot}(N)$. Let $M \rightarrow N$. If M has no hnf, then the same holds for N , and hence $\mathbf{BT}_{\odot}(M) = \perp = \mathbf{BT}_{\odot}(N)$. Assume that there exists a head rewrite sequence $M \rightarrow_h^k H \equiv \lambda x_1 \dots \lambda x_n. y M_1 \dots M_m$ to hnf. We have $\mathbf{BT}_{\odot}(M) \equiv [k] \lambda x_1 \dots \lambda x_n. y \mathbf{BT}_{\odot}(M_1) \dots \mathbf{BT}_{\odot}(M_m)$.

Using the elementary diagram above (k times), we can project $M \rightarrow N$ over $M \rightarrow_h^k H$, and obtain $H \rightarrow_h^{\ell} H'$, $N \rightarrow_h^{\ell} H' \equiv \lambda x_1 \dots \lambda x_n. y M'_1 \dots M'_m$ with $\ell \leq k$. Then $\mathbf{BT}_{\odot}(N) \equiv [\ell] \lambda x_1 \dots \lambda x_n. y \mathbf{BT}_{\odot}(M'_1) \dots \mathbf{BT}_{\odot}(M'_m)$ and $\ell \leq k$. Since $H \rightarrow_h^{\ell} H'$ and H is in hnf, we get $M_i \rightarrow_h M'_i$ for every $i = 1, \dots, m$. We then can apply the same argument to $M_i \rightarrow_h M'_i$ and by coinduction (or induction on the depth), we obtain $\mathbf{BT}_{\odot}(M) \geq \mathbf{BT}_{\odot}(N)$. \square

While $\mathbf{BT}_{\odot}(M) \not\equiv \mathbf{BT}_{\odot}(N)$ does not imply $M \neq_{\beta} N$, the following theorem allows us to use clocked Böhm Trees for discriminating λ -terms:

Theorem 4.6. *Let M and N be λ -terms. If N cannot be improved globally by any reduct of M , then $M \neq_{\beta} N$.*

Proof. If $M =_{\beta} N$, then $M \rightarrow M' \leftarrow N$ for some M' by confluence. Hence $\mathbf{BT}_{\odot}(M') \leq \mathbf{BT}_{\odot}(N)$ by Proposition 4.5. \square

Note that for distinguishing M and N we can always consider β -equivalent terms $M' =_{\beta} M$ and $N' =_{\beta} N$ instead. For Theorem 4.6 we have to show $\neg(\text{BT}_{\circlearrowright}^{\circ}(M') \leq \text{BT}_{\circlearrowright}^{\circ}(N))$ for all reducts M' of M . This condition is in general difficult to prove. However, the theorem is of use if one of the terms has a manageable set of reducts, and this term happens to have slower clocks. A striking example will be given below in solving a question of Plotkin in Section 5.

For a large class of λ -terms it turns out that clocks are invariant under reduction. We call these terms ‘simple’.

Definition 4.7. A redex $(\lambda x.M)N$ is called:

- (1) *linear* if x has at most one occurrence in M ;
- (2) *call-by-value* if N is a normal form; and
- (3) *simple* if it is linear or call-by-value.

The definition of simple redexes generalizes the well-known notions of call-by-value and linear redexes. Next, we define simple *terms*. Intuitively, we call a term M ‘simple’ if every reduction admitted by M only contracts simple redexes. The following definition further generalizes this intuition by considering only reductions computing the Böhm Tree.

Definition 4.8 (Simple terms). A λ -term M is *simple* if either M has no hnf, or the head reduction to hnf $M \rightarrow_h \lambda x_1 \dots \lambda x_n . y M_1 \dots M_m$ contracts only simple redexes, and M_1, \dots, M_m are simple terms.

Note that this definition is essentially coinductive: the set of simple terms is the largest set X such that if $M \in X$ then either M has no hnf or the reduction to hnf $M \rightarrow_h \lambda x_1 \dots \lambda x_n . y M_1 \dots M_m$ contracts only simple redexes and $M_1, \dots, M_m \in X$ again.

All the fpc’s in this paper are either simple or have simple reducts. The clock of simple λ -terms is invariant under reduction, that is, reduction of a simple term affects only finitely many annotations in the clocked Böhm Tree. For example, by reducing a term we can always make the clock values in a finite prefix equal to 0.

Proposition 4.9. *Let N be a reduct of a simple term M . Then N matches M eventually ($\text{BT}_{\circlearrowright}^{\circ}(M) =_{\exists} \text{BT}_{\circlearrowright}^{\circ}(N)$).*

Proof. The proof is a straightforward extension of the proof of Proposition 4.5 with the observation that for simple terms M , rewriting $M \rightarrow_h^k H \equiv \lambda x_1 \dots \lambda x_n . y M_1 \dots M_m$ to hnf does not duplicate redexes. For simple terms M , the elementary diagrams are of the form displayed in Figure 5. We use $s \xrightarrow{\emptyset} t$ to denote empty steps, that is, $s \equiv t$.

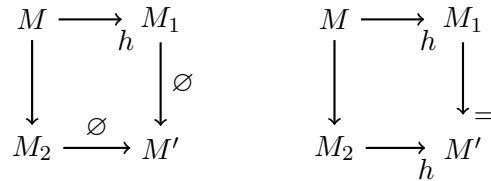


Figure 5: Elementary diagrams for simple M .

We briefly explain the two elementary diagrams. They depict the possible scenarios for joining co-initial steps $M \rightarrow_h M_1$ and $M \rightarrow M_2$, that is, a head reduction step set out against an arbitrary reduction step. If M is a simple term, then either

- (1) both are the same step, and can be joined with empty steps $\xrightarrow{\emptyset}$, or
(2) they can be joined by $M_2 \rightarrow_h M' \leftarrow^= M_1$ since a head step in a simple term with hnf cannot duplicate a redex (but deletion is possible).

We show the following implication by induction on $n \in \mathbb{N}$ (employing the diagrams in Figure 5):

$$M_1 \leftarrow_h M \rightarrow^n N \quad \Longrightarrow \quad M_1 \rightarrow^{\leq n} O \leftarrow_h N \vee M_1 \rightarrow^{< n} N \quad (4.1)$$

For $n = 0$, there is nothing to be shown. Let $n > 0$, and consider $M_1 \leftarrow_h M \rightarrow M_2 \rightarrow^{n-1} N$. Then either (1) $M_1 \equiv M_2$ and consequently $M_1 \rightarrow^{n-1} N$, or (2) $M_1 \rightarrow^= M' \leftarrow_h M_2$ and by induction hypothesis $M' \rightarrow^{\leq n-1} O \leftarrow_h M_2$ or $M' \rightarrow^{< n-1} N$, yielding $M_1 \rightarrow^{\leq n} O \leftarrow_h M_2$ or $M_1 \rightarrow^{< n} N$, respectively.

For λ -terms M and integers $n \in \mathbb{N}$, we define finite approximations $\text{BT}_{\circlearrowleft, n}(M)$ of the clocked Böhm Tree $\text{BT}_{\circlearrowleft}(M)$ of M . We let $\text{BT}_{\circlearrowleft, 0}(M) = M$, and for $n > 0$ define

$$\text{BT}_{\circlearrowleft, n}(M) = [k]\lambda x_1. \dots \lambda x_n. y \text{BT}_{\circlearrowleft, n-1}(M_1) \dots \text{BT}_{\circlearrowleft, n-1}(M_m)$$

if $M \rightarrow_h^k \lambda x_1. \dots \lambda x_n. y M_1 \dots M_m$ to hnf, and $\text{BT}_{\circlearrowleft, n}(M) = \perp$ if M has no hnf. Note that $\text{BT}_{\circlearrowleft}(M) = \lim_{n \rightarrow \infty} \text{BT}_{\circlearrowleft, n}(M)$.

Let M, N be terms, M simple and $M \rightarrow^n N$. Then we claim

$$[\text{BT}_{\circlearrowleft, 1}(M)] \rightarrow^{\leq n'} [\text{BT}_{\circlearrowleft, 1}(N)] \quad (*)$$

where $n' = n$ if $\text{BT}_{\circlearrowleft, 1}(M)$ and $\text{BT}_{\circlearrowleft, 1}(N)$ have the same annotation, and $n' = n-1$ otherwise. If M has no hnf, then N has no hnf, and $\text{BT}_{\circlearrowleft, 1}(M) = \text{BT}_{\circlearrowleft, 1}(N) = \perp$. Thus assume that M admits a head reduction $M \rightarrow_h^k M' \equiv \lambda x_1. \dots \lambda x_n. y M_1 \dots M_m$ to hnf for some $k \in \mathbb{N}$. Then by induction using (4.1) we obtain that either

$$(a) N \rightarrow_h^k N' \leftarrow^{\leq n} M' \quad \text{or} \quad (b) N \rightarrow_h^{< k} N' \leftarrow^{< n} M'$$

for some $N' \equiv \lambda x_1. \dots \lambda x_n. y N_1 \dots N_m$. This proves the claim.

Note that every reduction $\text{BT}_{\circlearrowleft, m}(M) \rightarrow^{\leq n} \text{BT}_{\circlearrowleft, m}(N)$ is contained within the subterms M' of $\text{BT}_{\circlearrowleft, m}(M)$ that are left unreduced by the clause $\text{BT}_{\circlearrowleft, 0}(M') = M'$; everything outside these subterms is in normal form. Using the previous observation and (*), we obtain by induction that for all $m \in \mathbb{N}$: $[\text{BT}_{\circlearrowleft, m}(M)] \rightarrow^{\leq n-d_m} [\text{BT}_{\circlearrowleft, m}(N)]$ where d_m is the number of positions p where the annotation of $\text{BT}_{\circlearrowleft, m}(M)|_p$ differs from the annotation of $\text{BT}_{\circlearrowleft, m}(N)|_p$. As a consequence, the annotations of $\text{BT}_{\circlearrowleft}(M)$ and $\text{BT}_{\circlearrowleft}(N)$ differ at most at n positions. Hence $\text{BT}_{\circlearrowleft}(M) =_{\exists} \text{BT}_{\circlearrowleft}(N)$. \square

Reduction accelerates clocks, and for simple terms the clock is invariant under reduction, see Proposition 4.9. Hence if a term M has a simple reduct N , then N has the fastest clock reachable from M modulo a finite prefix. This justifies the following convention.

Convention 4.10. The (*minimal*) clock of a λ -term M with a simple reduct N is $\text{BT}_{\circlearrowleft}(N)$, the clocked BT of N .

For simple terms we obtain the following theorem:

Theorem 4.11. *Let M and N be λ -terms such that M is simple. If M does not improve eventually on N , then $M \neq_{\beta} N$.*

Proof. Assume $M =_{\beta} N$. Then $M \twoheadrightarrow M' \leftarrow N$ for some M' . We have $\text{BT}_{\circlearrowleft}(M) =_{\exists} \text{BT}_{\circlearrowleft}(M')$ by Proposition 4.9, and $\text{BT}_{\circlearrowleft}(M') \leq \text{BT}_{\circlearrowleft}(N)$ by Proposition 4.5. Hence we obtain $\text{BT}_{\circlearrowleft}(M) \leq_{\exists} \text{BT}_{\circlearrowleft}(N)$. \square

Theorem 4.11 significantly reduces the proof obligation in comparison to Theorem 4.6. We only consider the clocked BT's of M and N , instead of all reducts M' of M .

Note that Theorem 4.11 can also be employed for discriminating non-simple λ -terms if one of the terms has a simple reduct. For the case that both M and N are simple, there is no need to look for reducts since the eventual clocks are invariant under reduction, see Proposition 4.9:

Corollary 4.12. *If simple terms M, N do not match eventually, then they are not β -convertible, that is, $M \neq_\beta N$.*

Proof. Assume $M = N$, then $M \rightarrow O \leftarrow N$ for a common reduct O . Then $\text{BT}_{\text{clock}}(M) =_{\exists} \text{BT}_{\text{clock}}(O) =_{\exists} \text{BT}_{\text{clock}}(N)$ by Proposition 4.9. Hence $\text{BT}_{\text{clock}}(M) =_{\exists} \text{BT}_{\text{clock}}(N)$, that is, M and N match eventually. \square

Remark 4.13. The reason for the qualifier ‘eventually’ in the notions above, in other words, working modulo a finite prefix of the BT, is that by some preliminary reduction we can always make the clock values in any finite prefix equal to 0. So we are interested exclusively in the ‘tail behaviour’, or the behaviour ‘at infinity’, and not in the initial behaviour of the development to the BT.

To give a concrete example: Y_0 and Y_1 , the fpc's of Curry and Turing, can be reduced to reducts M_0, M_1 respectively, that have an initial segment of arbitrary length n of their BT's with clock labels 0 (just reduce first to $\lambda f.f^n(\dots)$). However, the infinite remainders of their BT's, their tails as it were, will reveal the difference in clock values, witnessing the fact that Y_0 eventually improves Y_1 . And this situation is stable under reduction; indeed, for any two reducts M_0, M_1 as above, the first eventually improves the second.

Remark 4.14. Take λ -terms M, N with finite BT's. Then the clock comparison of M and N amounts to the comparison of their non-clocked BT's. In case their BT's are equal and \perp -free, then $M =_\beta N$. If their BT's coincide but are not \perp -free, then we can fine-tune the analysis by comparing their clocked Lévy–Longo or Berarducci Trees (a \perp in a BT may give rise to an infinite Berarducci Tree), see Section 8.

Example 4.15. We compute the clocks of $Y_n x$ with Y_n the n -th term of the Böhm sequence. The clocks of $Y_0 x$ and $Y_1 x$ have been computed before, see Figure 3. We now compute the clock of $Y_n x$ for $n \geq 2$:

$$\begin{aligned} Y_n &\equiv \eta\eta\delta^{\sim n-1}x \\ &\rightarrow_h^2 \delta(\eta\eta\delta)\delta^{\sim n-2}x \\ &\rightarrow_h^{2(n-2)} \delta(\eta\eta\delta^{\sim n-1})x \\ &\rightarrow_h^2 x(\eta\eta\delta^{\sim n-1}x) \end{aligned}$$

Notice that none of these steps duplicate a redex, hence Y_n is a simple term. We find $\text{BT}_{\text{clock}}(Y_n x) = [2n](x\text{BT}_{\text{clock}}(Y_n x))$. Hence, for $n \geq 2$ the clock of Y_n is $2n$. Consequently, all terms in the Böhm sequence have distinct clocks. Thus we have an alternative proof of Theorem 3.2: the Böhm sequence contains no duplicates.

Example 4.16. Let $n \geq 2$. We compute the clocks of the fpc's $U_n = \text{BY}_0 S^{\sim n} \mathbf{I}$ of the Scott sequence. We first reduce $U_n x$ to a simple term:

$$\begin{aligned} U_n x &\rightarrow Y_0(\text{SS})S^{\sim(n-2)} \mathbf{I}x \\ &\rightarrow \omega_{\text{SS}} \omega_{\text{SS}} S^{\sim(n-2)} \mathbf{I}x \\ &\rightarrow \underline{\omega_{\text{SS}}} \underline{\omega_{\text{SS}}} S^{\sim(n-2)} \mathbf{I}x \end{aligned}$$

where $\underline{\omega_{\text{SS}}} \equiv \lambda abc.bc(aabc)$. We abbreviate $\theta = \underline{\omega_{\text{SS}}}$. Then we compute the clocks for $n = 2$, $n = 3$, and $n > 3$:

$$\begin{aligned} \theta\theta \mathbf{I}x &\rightarrow_h^3 \mathbf{I}x(\theta\theta \mathbf{I}x) \rightarrow_h^1 x(\theta\theta \mathbf{I}x) \\ \theta\theta \text{SI}x &\rightarrow_h^3 \text{SI}(\theta\theta \text{SI})x \rightarrow_h^4 x(\theta\theta \text{SI}x) \\ \theta\theta S^{\sim(n-2)} \mathbf{I}x &\rightarrow_h^3 \text{SS}(\theta\theta \text{SS})S^{\sim(n-4)} \mathbf{I}x \\ &\rightarrow_h^{3(n-4)} \text{SS}(\theta\theta \text{SSS}^{\sim(n-4)}) \mathbf{I}x \\ &\rightarrow_h^3 \text{SI}(\theta\theta S^{\sim(n-2)} \mathbf{I})x \\ &\rightarrow_h^4 x(\theta\theta S^{\sim(n-2)} \mathbf{I}x) \end{aligned}$$

respectively. For all three cases, we find:

$$\text{BT}_{\text{c}}^{\text{c}}(\theta\theta S^{\sim(n-2)} \mathbf{I}x) = [3n - 2](x \text{BT}_{\text{c}}^{\text{c}}(\theta\theta S^{\sim(n-2)} \mathbf{I}x)).$$

Using Theorem 4.11 we infer from Example 4.16 and Figure 3 (recall that $U_0 =_\beta Y_0$ and $U_1 =_\beta Y_1$):

Corollary 4.17. *The Scott sequence contains no duplicates.*

5. AN ANSWER TO A QUESTION OF PLOTKIN

Plotkin [22] asked: *Is there a fixed point combinator Y such that*

$$A_Y \equiv Y(\lambda z.fzz) =_\beta Y(\lambda x.Y(\lambda y.fxy)) \equiv B_Y \quad (5.1)$$

or in other notation:

$$\mu z.fzz =_\beta \mu x.\mu y.fxy,$$

with the usual definition $\mu x.M(x) = Y(\lambda x.M(x))$. The terms A_Y and B_Y have the same Böhm Trees, namely the solution of $T = fTT$.

Plotkin's question is pertinent to the question whether absolutely unorderable models of the λ -calculus exist, see Selinger's work [25, 26, 27]. The negative solution of Plotkin's question blocks an appeal to Lemma 3.6 in [27] to show that the generalized Mal'cev equations for all n are inconsistent with the $\lambda\beta$ -calculus. (For $n = 0, 1$ this is established, but the general case is a difficult open problem.)

We show that there is no fpc Y satisfying (5.1). We begin with an example.

Example 5.1. We consider Turing's fpc Y_1 , and compute the clocked BT's of A_{Y_1} and B_{Y_1} . Recall that $Y_1 \equiv \eta\eta$ with $\eta \equiv \lambda x f.f(xxf)$.

$$\begin{aligned} A_{Y_1} &\equiv \eta\eta(\lambda z.fzz) \rightarrow_h^2 (\lambda z.fzz)A_{Y_1} \rightarrow_h^1 fA_{Y_1}A_{Y_1} \\ B_{Y_1} &\equiv \eta\eta(\lambda x.\eta\eta(\lambda y.fxy)) \rightarrow_h^2 (\lambda x.\eta\eta(\lambda y.fxy))B_{Y_1} \\ &\rightarrow_h^1 \eta\eta(\lambda y.fB_{Y_1}y) \rightarrow_h^2 (\lambda y.fB_{Y_1}y)(\eta\eta(\lambda y.fB_{Y_1}y)) \\ &\rightarrow_h^1 fB_{Y_1}(\eta\eta(\lambda y.fB_{Y_1}y)) \end{aligned}$$

Note that for B_{Y_1} developing the left branch takes six steps, whereas the right branch only needs three. We remark that this is not sufficient to conclude that $A_{Y_1} \neq_\beta B_{Y_1}$ since A_Y and B_Y are non-simple even for simple fpc's Y . The clocked BT's for A_{Y_1} and B_{Y_1} are depicted in Figure 6 using hnf-notation, see [3] or [5].

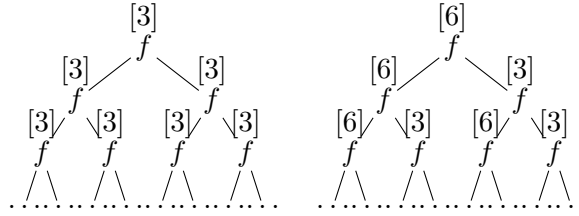


Figure 6: Clocked BT's for A_{Y_1} and B_{Y_1} , in hnf-notation.

To prove $A_Y \neq_\beta B_Y$ for every fpc Y , we construct a term B'_Y convertible with B_Y such that no reduct of A_Y improves globally on B'_Y . Then by Theorem 4.6 we have $A_Y \neq_\beta B'_Y$, and thus $A_Y \neq_\beta B_Y$. We have $B_Y =_\beta B'_Y$ where B'_Y is defined by:

$$B'_Y \equiv Y(\lambda x.fx(fx(Y(\lambda y.fxy))))$$

While the annotations of the clocked Böhm Tree $\text{BT}_{\text{c}}(B'_Y)$ of B'_Y of course depend on Y , we can derive partial knowledge as follows. We have $B'_Y \equiv (Yx)[x:=M]$, where $M \equiv \lambda x.fx(fx(Y(\lambda y.fxy)))$. Then we have for every $Y' =_\beta Yx$:

$$\begin{aligned} Y'[x:=M] &\rightarrow_h^* (xY'')[x:=M] \equiv M(Y''[x:=M]) \\ &\rightarrow_h f(Y''[x:=M]) \underline{(f(Y''[x:=M])(Y(\lambda y.f(Y''[x:=M])y)))} \end{aligned}$$

for some $Y'' =_\beta Yx$. The underlined term in head normal form is obtained without reducing, and hence will be annotated with [0] in the clocked Böhm tree of $Y'[x:=M]$. The same argument applies for $Y''[x:=M]$, and so on. Hence the second argument of every f on the leftmost spine of $\text{BT}_{\text{c}}(Y'[x:=M])$ has annotation [0], as in Figure 7, which depicts the clocked Böhm tree $\text{BT}_{\text{c}}(B'_Y)$ of B'_Y . Therefore we obtain:

Lemma 5.2. *Every annotation at a position of the form $(12)^*2$ in $\text{BT}_{\text{c}}(B'_Y)$ is [0].* \square

Note that in the statement of the above lemma, the positions $(12)^n2$ refer to trees in applicative notation, which is equivalent to 1^n2 in the hnf-notation, as seen in Figure 7.

We continue by proving that no reduct of A_Y improves globally on B'_Y . More precisely, we show that for every reduct A'_Y of A_Y there exists a position p of the form $(12)^*2$ such that the annotation of $\text{BT}_{\text{c}}(A'_Y)$ at p is nonzero.

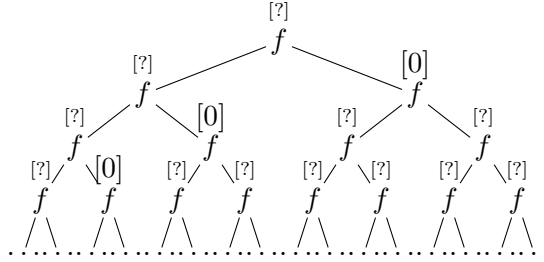


Figure 7: Clocked Böhm Tree for B'_Y in hnf-notation.

Definition 5.3. A reduct M of $A_Y \equiv Y(\lambda z.fzz)$ is called *balanced* if $s \equiv t$ for every subterm $f s t$ in M where f is a descendant of the displayed f (see also Equation (5.1)).

Note that, in case Y is not closed, the term may contain occurrences f . In the definition of ‘balanced’ we are not interested in those occurrences of f , but only the residuals of f displayed in $Y(\lambda z.fzz)$, that is, the f in $\lambda z.fzz$. Let us label this f as f^* , i.e. $Y(\lambda z.f^*zz)$, such that f^* does not occur in Y . Then by ‘residuals of the displayed f ’ we mean the free occurrences of f^* in all reducts of $Y(\lambda z.f^*zz)$.

The following lemma states that every reduct of A_Y can be balanced:

Lemma 5.4. *Let $A_Y \equiv Y(\lambda z.fzz) \rightarrow M$. There exists a balanced N such that $M \rightarrow N$.*

Proof. Let \twoheadrightarrow denote complete developments of the set of all redex occurrences (also known as Gross–Knuth steps). We consider the sequence $A_Y \equiv N_1 \twoheadrightarrow N_2 \twoheadrightarrow \dots$. By cofinality of \twoheadrightarrow [3, 30] there exists $i \in \mathbb{N}$ such that $M \twoheadrightarrow N_i$. We define $N \equiv N_i$.

It remains to be shown that the term N is balanced. This follows from the fact that A_Y is balanced, and that balancedness is preserved under \twoheadrightarrow . The latter can be seen as follows: consider a term $C[f s s]$. Obviously both displayed occurrences of s contain the same redexes, and the same variables bound from above $f s s$ in C . Hence all descendants of both occurrences of s after \twoheadrightarrow will again be identical. \square

Lemma 5.5. *Let $A_Y \twoheadrightarrow A'_Y$. There exists a position p of the form $(12)^*2$ such that $\text{BT}_{\text{c}^*}(A'_Y)$ has a nonzero annotation at position p .*

Proof. Let A''_Y be a reduct of A'_Y such that A''_Y is in hnf. By Lemma 5.4 there exists a balanced reduct A'''_Y of A''_Y . Note that also A'''_Y is in hnf and its head symbol is f .

Let q be the shortest position of the form $(12)^*$ such that $A'''_Y|_{q12}$ is not in hnf. Then $A'''_Y|_q$ is in hnf, and thus $A'''_Y(q11) = f$. Because A'''_Y is balanced we have that $A'''_Y|_{q12} \equiv A'''_Y|_{q2}$, and so $A'''_Y|_{q2}$ is not in hnf, too. Hence the annotation of $\text{BT}_{\text{c}^*}(A'''_Y)$ at the position $p = q2$ is nonzero. The claim follows since $\text{BT}_{\text{c}^*}(A'''_Y) \leq \text{BT}_{\text{c}^*}(A'_Y)$. \square

The combination of Lemmas 5.2 and 5.5 implies that there exists no reduct of A_Y that improves globally on B'_Y . Therefore by Theorem 4.6 we obtain:

Proposition 5.6. *There exists no fpc Y such that $Y(\lambda z.fzz) =_\beta Y(\lambda x.Y(\lambda y.fxy))$.* \square

6. CLOCKED BÖHM TREES AND PERIODIC TERMS

In the last two sections we saw that the concept of clocked Böhm Trees can be successfully used to discriminate between λ -terms with equal Böhm Trees or to analyze the existence of terms satisfying certain equations. In this section we will address the following question: to which classes of λ -terms is the clocks method most readily applicable?

A common feature of terms studied in Sections 4 and 5 is that their Böhm Trees are ‘periodic’: they have inherent cyclical content that allows us to analyze how their clocks are developed towards infinity. For example, Y is an fpc if the term $Y_x \equiv Yx$ satisfies

$$Y_x = x(Y_x) \quad (6.1)$$

That is, the subterm of Y_x occurring at position 2 is equal to the term Y_x itself. If we use Böhm Tree equality instead of β -equality then the equation (6.1) would say that Y is a wfpc, or a looping combinator:

$$\text{BT}(Y_x) = \text{BT}(x(Y_x))$$

The next definition attempts to capture such situations in a general fashion.

In what follows, M_σ denotes the term at position σ in the Böhm Tree of M , with $M_\sigma = \Omega = \mathbf{Y}_0\mathbf{l}$ in the event that $\sigma \notin \text{Pos}(\text{BT}(M))$. By a *Böhm Tree context* $C\Box_1 \cdots \Box_n$ we mean a finite Böhm Tree in which the leaves, in addition to being variables or \perp , are allowed to be either of the holes \Box_i . (Equivalently, it is a $\lambda\perp$ -context which is redex-free.)

Definition 6.1. Let $M \in \text{Ter}(\lambda)$, and $\epsilon \neq \sigma \in \text{Pos}(\text{BT}(M))$.

- (1) M is *periodic at σ* if $M =_\beta M_\sigma$.
- (2) M is *weakly periodic at σ* if $M =_{\text{BT}} M_\sigma$.
- (3) M is *(weakly) locally periodic* if for every infinite path $x = \langle x_0, x_1, \dots \rangle$ through the Böhm Tree of M we have that M is (weakly) periodic at x_n for some n .
- (4) M is *(weakly) globally periodic* if there is a Böhm Tree context $C\Box_1 \cdots \Box_n$ such that $M \equiv C[M_1] \cdots [M_n]$ and $M =_\beta M_i$ ($M =_{\text{BT}} M_i$).

Example 6.2.

- (1) If Y is an fpc, then Yx is periodic at 2.
- (2) If Y is a weak fpc, then Yx is weakly periodic at 2.
- (3) Using the fixed point theorem, let M be such that $M = \lambda z.zMM$. Then M is periodic at 02, 012. Since any infinite path through $\text{BT}(M)$ passes through one of these positions, M is locally periodic. In fact, M is globally periodic with the context $C\Box_1\Box_2 \equiv \lambda z.z\Box_1\Box_2$.

Generally, any term which is defined by an application of the fixed point theorem is periodic by construction. If a looping combinator is used instead of an fpc, then the resulting term will be weakly periodic. This suggests that (w)fpc’s play a central role in the study of general periodic terms.

We prove the following elementary fact:

Proposition 6.3. *Let $M \in \text{Ter}(\lambda)$. Then*

$$M \text{ is (weakly) locally periodic} \iff M \text{ is (weakly) globally periodic}$$

Proof. The statement is proved simultaneously for weak and strong periodicity. For the latter case, the equality should be read as β -convertibility. For the former, it should be read as Böhm Tree equality.

(\Rightarrow) Suppose that $M = C[M_1] \cdots [M_n]$, with $C \square_1 \cdots \square_n$ a Böhm Tree context. Let x be an infinite path through $\text{BT}(M)$. Since $C \square_1 \cdots \square_n$ is finite, x must pass through one of the holes \square_i . Since $M = M_i$ is periodic at this position, x passes through a periodic position. Since x was arbitrary, M is locally periodic.

(\Leftarrow) Suppose that M is locally periodic, and let \mathcal{X} denote the set of infinite paths through the Böhm Tree of M (\mathcal{X} could be called the *Böhm space* of M). Since $\text{BT}(M)$ is a finitely branching tree, the space \mathcal{X} is compact, when given the topology generated by the cylinders

$$X_\sigma = \{\theta \mid \sigma \sqsubseteq \theta\}$$

For $x \in \mathcal{X}$, let σ_x be a periodic position that x passes through (by local periodicity). Then the collection $\Sigma = \{\sigma_x\}_{x \in \mathcal{X}}$ is an open cover of \mathcal{X} . By compactness, there exists a subcollection $\sigma_1, \dots, \sigma_n \in \Sigma$ which covers \mathcal{X} . Let $C \equiv C \square_1 \cdots \square_n$ be obtained from M by unfolding its Böhm Tree up to depth $\max\{|\sigma_1|, \dots, |\sigma_n|\}$ and placing holes \square_i at positions σ_i . We claim that C is the desired Böhm Tree context.

By construction, C is a Böhm Tree context with $M = C[M_{\sigma_1}] \cdots [M_{\sigma_n}]$. We also have that M is periodic at σ_i , hence $M = M_{\sigma_i}$. Furthermore, if $x \in \mathcal{X}$, then x passes through σ_i for some i . Therefore, outside the holes \square_i the Böhm Tree of C is finite.

Indeed, M is globally periodic. □

In light of the proposition above, when M is either locally or globally periodic, we simply say that M is periodic.

The resulting notion of periodicity is somewhat restrictive, because it does not account for terms in which repetition begins later in the Böhm Tree. For example if Y is an fpc, then Yx is periodic, but Y itself is not, because the first node of its Böhm Tree has the abstraction $\lambda f.f$, which is never repeated in the infinite sequence of f 's that follows.

This situation motivates the following definition.

Definition 6.4. Suppose that $\sigma \in \text{Pos}(\text{BT}(M))$ and $\theta \sqsubseteq \sigma$. Let π be such that $\theta\pi = \sigma$.

- (1) M is *periodic at σ with offset θ* if M_θ is periodic at π . In this case, we say that σ is *cyclic* in $\text{BT}(M)$, and call π the *period* and θ the *phase* of the *cycle at σ* .
- (2) M is *weakly periodic at σ with offset θ* if M_θ is weakly periodic at π . We say that σ is *weakly cyclic*, and call π the period and θ the phase of the *weak cycle at σ* .
- (3) M is (*weakly*) *fully periodic* if every infinite path through $\text{BT}(M)$ passes through a (weak) cycle (θ, π) (that is, eventually comes into a periodic subterm of M).

As before, it is easy to see that M is fully periodic if and only if there are finitely many positions $\sigma_1, \dots, \sigma_n \in \text{Pos}(\text{BT}(M))$, $\sigma_i = \theta_i \pi_i$ such that M is periodic at σ_i with offset θ_i and period π_i , and outside these positions the Böhm Tree is finite.

Periodic and fully periodic terms are the perfect targets for exercising the clocks method. We will now illustrate this with an example, where, instead of fpc's, we consider enumerators, also called evaluators, for Combinatory Logic. There are many possibilities for such an enumerator \mathbb{E} , depending on how one does the coding. However, in most schemes \mathbb{E} is a periodic term.

Example 6.5. Let $\ulcorner \cdot \urcorner : CL \rightarrow \text{Ter}(\lambda)$ be defined as follows:

$$\ulcorner \mathbf{K} \urcorner = \lambda z.z\mathbf{K}\mathbf{K}\mathbf{I}$$

$$\ulcorner \mathbf{S} \urcorner = \lambda z.z\mathbf{K}\mathbf{S}\mathbf{I}$$

$$\ulcorner MN \urcorner = \lambda z.z(\mathbf{K}\mathbf{I})\ulcorner M \urcorner \ulcorner N \urcorner$$

Let $\langle M \rangle = \lambda z.zM$, with $z \notin \text{FV}(M)$ and $\omega = \lambda x.xx$. Then some possible evaluators for $\lceil \cdot \rceil$ include:

$$\mathbf{E}_1 \equiv \omega(\lambda w.\langle \lambda abc.ab((wwb)(wvc)) \rangle) = \langle \lambda abc.ab(\mathbf{E}_1 b)(\mathbf{E}_1 c) \rangle \quad (6.2)$$

$$\mathbf{E}_2 \equiv \omega(\lambda w.\langle \lambda abc.ab(\mathbf{S}\langle b \rangle\langle c \rangle)(ww) \rangle) = \langle \lambda abc.ab(\mathbf{S}\langle b \rangle\langle c \rangle \mathbf{E}_2) \rangle \quad (6.3)$$

$$\mathbf{E}_3 \equiv \langle \omega(\lambda wabc.ab(\mathbf{S}bc(ww))) \rangle = \langle \mathbf{E}' \rangle, \quad \text{where } \mathbf{E}' = \lambda abc.ab(\mathbf{S}bc\mathbf{E}') \quad (6.4)$$

It is straightforward to verify that, for $i \in \{1, 2, 3\}$ we have $\mathbf{E}_i \lceil M \rceil = M$ for any $\{\mathbf{K}, \mathbf{S}\}$ -term M . The evaluators look similar, and their Böhm Trees are indeed the same. However, it is not immediate whether the terms are β -convertible or if they are distinct.

Fortunately, since these terms are periodic, their clocked Böhm Trees can be computed easily:

(1) For $N \in \text{Ter}(\lambda)$, let $V_N \equiv \lambda abc.ab((NNb)(NNc))$, and $W = \lambda w.\langle V_w \rangle$. Then we have

$$\begin{aligned} \mathbf{E}_1 &\rightarrow_h WW \rightarrow_h \lambda z.zV_W \\ &\equiv \lambda z.z(\lambda abc.ab((WWb)(WWc))) \\ &\rightarrow_h \lambda z.z(\lambda abc.ab((\lambda z'.z'V_W)b(WWc))) \\ &\rightarrow_h \lambda z.z(\lambda abc.ab(bV_W(WWc))) \\ &\rightarrow \lambda z.z(\lambda abc.ab(bV_W((\lambda z'.z'V_W)c))) \\ &\rightarrow \lambda z.z(\lambda abc.ab(bV_W(cV_W))) \\ &\rightarrow \dots \end{aligned}$$

Hence \mathbf{E}_1 is periodic at positions 02000212 and 02000222 (in hnf-notation 010 and 0110 respectively) with offset 02 and periods 000212 and 000222, which allows us to construct the clocked Böhm Tree of \mathbf{E}_1 .

(2) For $N \in \text{Ter}(\lambda)$, let $V_N \equiv \lambda abc.ab(\mathbf{S}\langle b \rangle\langle c \rangle)(ww)$, $W = \lambda w.\langle V_w \rangle$, $\mathbf{S}'_N = \lambda yz.Nz(yz)$, and $\mathbf{S}''_{N,M} = \lambda z.Nz(Mz)$. Then we have

$$\begin{aligned} \mathbf{E}_2 &\rightarrow_h WW \rightarrow_h \lambda z.zV_W \\ &\equiv \lambda z.z(\lambda abc.ab(\mathbf{S}\langle b \rangle\langle c \rangle)(WW)) \\ &\rightarrow_h \lambda z.z(\lambda abc.ab(\mathbf{S}'_{\langle b \rangle}\langle c \rangle)(WW)) \\ &\rightarrow_h \lambda z.z(\lambda abc.ab(\mathbf{S}''_{\langle b \rangle, \langle c \rangle})(WW)) \\ &\rightarrow_h \lambda z.z(\lambda abc.ab(\langle b \rangle)(WW)(\langle c \rangle)(WW)) \\ &\rightarrow_h \lambda z.z(\lambda abc.ab(WWb(WWc))) \\ &\rightarrow^2 \lambda z.z(\lambda abc.ab((\lambda z'.z'V_W)b((\lambda z'.z'V_W)c))) \\ &\rightarrow^2 \lambda z.z(\lambda abc.ab((bV_W)(cV_W))) \\ &\rightarrow \dots \end{aligned}$$

Again, we are at a point where every position lies on a cycle, giving us a full description of the clocked Böhm Tree of \mathbf{E}_2 .

(3) Let $V_N = \lambda abc.ab(\mathbf{S}bc(NN))$, and $W, S'_N, S''_{N,M}$ as before. Then

$$\begin{aligned}
 E_3 &\equiv \langle \omega W \rangle \rightarrow_h \langle WW \rangle \rightarrow_h \lambda z.zV_W \\
 &\equiv \lambda z.z(\lambda abc.ab(\mathbf{S}bc(WW))) \\
 &\rightarrow_h \lambda z.z(\lambda abc.ab(\mathbf{S}'_b c(WW))) \\
 &\rightarrow_h \lambda z.z(\lambda abc.ab(\mathbf{S}''_{b,c}(WW))) \\
 &\rightarrow_h \lambda z.z(\lambda abc.ab(b(WW)(c(WW))))
 \end{aligned}$$

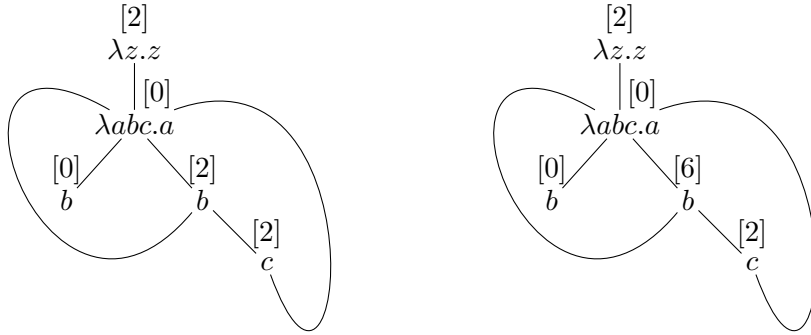


Figure 8: Clocked BT's for E_1 and E_2 , in hnf-notation.

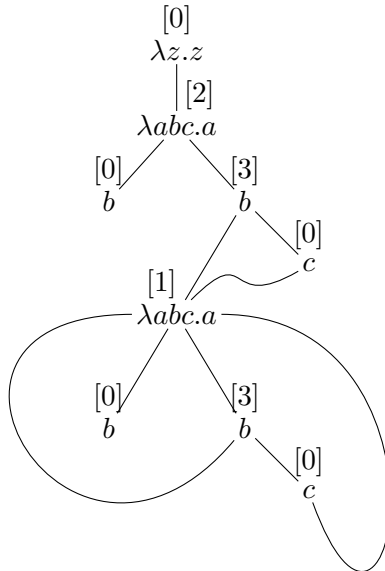


Figure 9: Clocked BT for E_3 , in hnf-notation.

The Böhm Trees of E_i are displayed in Figures 8 and 9. This information, together with Theorem 4.11, allows us to discriminate between the terms. Note that E_1 does not eventually improve on E_3 : the occurrences of the variable c in the clocked Böhm Tree of E_3 are annotated by a 0, while the corresponding subterms of E_1 take two head steps to converge. Furthermore, the head redexes contracted in this reduction WW and $\langle V_W \rangle c$ are both call-by-value. By Theorem 4.11 we conclude that $E_3 \not\equiv_{\beta} E_1$. In contrast, E_2 is improved by E_1 . In fact, E_1 and E_2 can be reduced so that their clocked Böhm Trees match eventually. Upon further inspection, we see that the two terms are indeed convertible.

Remark 6.6. We note that (fully) periodic terms are the λ -calculus analogue of the usual ‘circular’ terms defined by mutually recursive definitions. The `let` binder in Haskell is an example of such a construct — one difference being that, because the semantics of Haskell is graph rewriting, such terms actually denote possibly cyclic graphs rather than their unfoldings as infinite trees. For example, the following is an example of a ‘periodic Haskell term’:

```
streamSys =
  let x = f 0 y
      where f n (b:ys) = n : f (b n) ys
      y = f id z
          where f g (c:cs) = g : f (\n -> g c + n) cs
      z = zz 0
          where zz n = n : zz (n+1)
  in x
```

```
*Main> take 20 streamSys
[0,0,0,1,4,10,20,35,56,84,120,165,220,286,364,455,560,680,816,969]
```

In discussion of such recursive systems, a well-known topic is the distinction between ‘strong equality’ and ‘weak equality’; the former being syntactic convertibility between two such systems, and the latter being bisimulation or ‘behavioral equivalence’. For this reason, the complexity of strong equality is usually Σ_1 , since one only needs to check the existence of a proof object (the conversion sequence) for the equality between two terms, while weak equality is Π_2 , since it asks whether for every number n , there is a conversion that makes the terms coincide up to depth n .

This discussion applies to our notions of fpc’s and wfpc’s, and also periodic and weakly periodic terms. From this point of view, the clocked Böhm Trees offer a refinement of these two types of equality, replacing dichotomy with a spectrum, or hierarchy, of equality, by augmenting the syntactic shape of the terms with the information of how quickly the computation they define is performed. For the case of λ -calculus, this spectrum was displayed in Figure 1.

Concluding this elaboration on cyclic terms, we note that also for the fragment of λ -calculus consisting of μ -terms, with the definition of the corresponding μ -rule $\mu x.A(x) \rightarrow A(\mu x.A(x))$, the notion of clocked Böhm Trees may be interesting, as it yields an equality strictly in between weak and strong equality, as it is called in [12, 4]. There μ -terms are treated extensively, for their application as a representation of recursive types.

7. ATOMIC CLOCKS

We have introduced clocked Böhm Trees for discriminating λ -terms. In this section, we refine the clocks to measure not only the *number* of head steps, but, in addition, the *position* of each of these steps. We call these clocks ‘atomic’. We write $\rightarrow_{h,p}$ for the head reduction step at position p .

Before we give the formal definition (Definition 7.1), we consider a motivating example. We discriminate Y_2 from U_2 . First, we reduce both terms to simple reducts:

$$\begin{aligned} Y_2x &\equiv Y_0\delta\delta x \rightarrow \eta\eta\delta x && \text{where } \eta = \lambda ab.b(abb) \\ U_2x &\equiv Y_0(SS)lx \rightarrow \theta\theta lx && \text{where } \theta = \lambda abc.bc(aabc) \end{aligned}$$

Second, we compute the atomic clocks of these simple reducts, that is, the positions of the head steps, as follows:

$$\begin{aligned} \eta\eta\delta x &\rightarrow_{h,11} (\lambda b.b(\eta\eta b))\delta x \rightarrow_{h,1} \delta(\eta\eta\delta)x \\ &\rightarrow_{h,1} (\lambda b.b(\eta\eta\delta b))x \rightarrow_{h,\epsilon} x(\eta\eta\delta x) \\ \theta\theta lx &\rightarrow_{h,11} (\lambda bc.bc(\theta\theta bc))lx \rightarrow_{h,1} (\lambda c.lc(\theta\theta lc))x \\ &\rightarrow_{h,\epsilon} lx(\theta\theta lx) \rightarrow_{h,1} x(\theta\theta lx) \end{aligned}$$

Thus the atomic clocks of these terms are:

$$\begin{aligned} \mathbf{BT}_{\bullet,\bullet}(\eta\eta\delta x) &= [11, 1, 1, \epsilon](x\mathbf{BT}_{\bullet,\bullet}(\eta\eta\delta x)) \\ \mathbf{BT}_{\bullet,\bullet}(\theta\theta lx) &= [11, 1, \epsilon, 1](x\mathbf{BT}_{\bullet,\bullet}(\theta\theta lx)) \end{aligned}$$

Note that both terms have the (non-atomic) clocked Böhm Tree $T \equiv [4](xT)$. Hence the method from the previous section is not applicable. However, the atomic clocks do allow us to discriminate the terms. Hence $Y_2 \neq_{\beta} U_2$ (by Corollary 4.12 which generalises to the setting of atomic BT’s). Note that the (non-atomic) clocked BT’s can be obtained by taking the length of the lists of positions.

For lists \vec{p}, \vec{q} of positions, we write $\vec{p} \cdot \vec{q}$ for concatenating \vec{p} to \vec{q} . We write $\rightarrow_{h, \langle p_1, \dots, p_n \rangle}$ for the rewrite sequence $\rightarrow_{h,p_1} \cdots \rightarrow_{h,p_n}$ consisting of steps at position p_1, \dots, p_n .

Definition 7.1 (Atomic clock Böhm Trees). Let $M \in \text{Ter}(\lambda)$. The *atomic clock Böhm Tree* $\mathbf{BT}_{\bullet,\bullet}(M)$ of M is an annotated infinite term defined as follows. If M has no hnf, then define $\mathbf{BT}_{\bullet,\bullet}(M)$ as \perp . Otherwise, there is a head reduction

$$M \rightarrow_{h,p_1} \cdots \rightarrow_{h,p_k} \lambda x_1. \dots \lambda x_n. yM_1 \dots M_m$$

of length k to hnf. Then we define $\mathbf{BT}_{\bullet,\bullet}(M)$ as the term:

$$\mathbf{BT}_{\bullet,\bullet}(M) = [\langle p_1, \dots, p_k \rangle] \lambda x_1. \dots \lambda x_n. y\mathbf{BT}_{\bullet,\bullet}(M_1) \dots \mathbf{BT}_{\bullet,\bullet}(M_m)$$

The theory developed for (non-atomic) BT’s in Section 4 generalises to atomic trees, as follows.

Theorem 7.2. For lists of positions \vec{p}, \vec{q} we define $\vec{p} \geq \vec{q}$ whenever \vec{q} is a subsequence of \vec{p} , and $\vec{p} > \vec{q}$ if additionally $\vec{p} \neq \vec{q}$. Here $\langle a_1, \dots, a_n \rangle$ is a subsequence of $\langle b_1, \dots, b_m \rangle$ if there exist indexes $i_1 < i_2 < \dots < i_n$ such that $\langle a_1, \dots, a_n \rangle = \langle b_{i_1}, \dots, b_{i_n} \rangle$.

Using this notation for comparing the atomic annotations (lists of positions), Proposition 4.5, Theorem 4.6, Proposition 4.9, Theorem 4.11, and Corollary 4.12 remain valid.

As an application of using atomic clocks to discriminate λ -terms, we show that every combination of the fpc-generating vectors $\square(\text{SS})S^{\sim n}\mathbf{I}$ from Theorem 3.8 applied to Curry's fpc Y_0 gives rise to inconvertible fpc's. We note that this cannot be proved using non-atomic clocks, as for example we have $\text{BT}_{\text{c}}^{\text{c}}(Y_0(\text{SS})S^{\sim n}\mathbf{I}(\text{SS})S^{\sim m}\mathbf{I}) = \text{BT}_{\text{c}}^{\text{c}}(Y_0(\text{SS})S^{\sim m}\mathbf{I}(\text{SS})S^{\sim n}\mathbf{I})$.

Proposition 7.3. *Let $G_n = \square(\text{SS})S^{\sim n}\mathbf{I}$ the fpc-generating vectors from Theorem 3.8. For $n_1, \dots, n_k \in \mathbb{N}$ we define*

$$Y^{\langle n_1, \dots, n_k \rangle} = G_{n_k}[\dots G_{n_1}[Y_0]\dots]$$

All these fpc's are inconvertible, that is, $\vec{n} \neq \vec{m}$ implies $Y^{\vec{n}} \not\equiv_{\beta} Y^{\vec{m}}$.

Proof. In this proof we abbreviate context filling by simply concatenating the contexts, so that for $Y^{\langle n_1, \dots, n_k \rangle}$ defined above we have $Y^{\langle n_1, \dots, n_k \rangle} = Y_0 G_{n_1} \dots G_{n_k}$.

For $\vec{p} = \langle 1^{m_1}, \dots, 1^{m_k} \rangle$ a list of positions we define $\vec{p} \times n$ as follows:

$$\begin{aligned} \vec{p} \times 0 &= \langle \rangle \\ \vec{p} \times n &= \vec{p} \cdot (\langle 1^{m_1-1}, \dots, 1^{m_k-1} \rangle \times (n-1)) \quad (n > 0) \end{aligned}$$

For example, we have $\langle 1^5 \rangle \times 3 = \langle 1^5, 1^4, 1^3 \rangle$.

We also use the following abbreviations, for $n \in \mathbb{N}$:

$$\begin{aligned} G'_n &= \square S^{\sim n} \mathbf{I} & \theta &= \lambda abc. bc(aabc) \\ \underline{G}_n &= \square \zeta S^{\sim n} \mathbf{I} & \zeta &= \underline{\text{SS}} = \lambda abc. bc(abc) \end{aligned}$$

(Note that $\omega_{\zeta} \rightarrow_{\beta} \theta$.) For $n_1, \dots, n_k \in \mathbb{N}$ we define an fpc Y by

$$Y = \theta \theta G'_{n_1} \underline{G}_{n_2} \dots \underline{G}_{n_k}$$

so that we have the following reduction:

$$Y^{\langle n_1, \dots, n_k \rangle} x \equiv Y_0 G_{n_1} \dots G_{n_k} x \rightarrow Y x \quad (k \geq 1)$$

and we observe that Yx is a simple term (as can be inferred from the reductions below).

Let $V_m = \square N_1 \dots N_m$ be a vector of length m . Then we have, for every $n \geq 0$:

$$\begin{aligned} \theta \theta G'_n V_m &\equiv \theta \theta S^{\sim n} \mathbf{I} V_m \\ &\rightarrow_{h, \langle (1^{n+m+1}) \times 3 \rangle \times n} \text{SI}(\theta \theta S^{\sim n} \mathbf{I}) V_m & (\dagger)_{n,m} \\ &\rightarrow_{h, \langle 1^{m+1} \rangle \times 2} (\lambda c. \text{lc}(\theta \theta S^{\sim n} \mathbf{I} c)) V_m \end{aligned}$$

For M an arbitrary term, we have the following reductions $(*)_{n,m}$ ($n, m \geq 0$):

$$\begin{aligned} (\lambda c. \text{lc}(Mc)) \underline{G}_n V_m &\equiv (\lambda c. \text{lc}(Mc)) \zeta S^{\sim n} \mathbf{I} V_m \\ &\rightarrow_{h, \langle 1^{n+m+1} \rangle} \text{I} \zeta (M \zeta) S^{\sim n} \mathbf{I} V_m \\ &\rightarrow_{h, \langle 1^{n+m+2} \rangle} \zeta (M \zeta) S^{\sim n} \mathbf{I} V_m & (*)_{n,m} \\ &\rightarrow_{h, \langle (1^{n+m+1}) \times 3 \rangle \times n} \text{SI}(M \underline{G}_n) V_m \\ &\rightarrow_{h, \langle 1^{m+1} \rangle \times 2} (\lambda c. \text{lc}(M \underline{G}_n c)) V_m \end{aligned}$$

Moreover, let (\ddagger) denote the following rewrite sequence:

$$(\lambda c. \text{lc}(Yc)) x \rightarrow_{h, \langle 1^0, 1^1 \rangle} x(Yx) \quad (\ddagger)$$

The rewrite sequence $Yx \rightarrow_h x(Yx)$ is composed of k subsequences as follows:

$$(\ddagger)_{n_1, m_1}, (*)_{n_2, m_2}, (*)_{n_3, m_3}, \dots, (*)_{n_k, m_k}, (\ddagger) \quad (*)$$

where m_i is defined by $m_k = 1$ and $m_i = m_{i+1} + n_{i+1} + 2$ ($1 \leq i < k$). For a rewrite sequence σ , let $\langle \sigma \rangle$ denote the sequence of positions of the steps in σ . Note that the atomic clock Böhm Tree of $Y^{\vec{n}}x$ is of the form $\mathbf{BT}_{\blacktriangleleft}(\mathbf{Y}^{\vec{n}}x) = [\langle (*) \rangle] x([\langle (*) \rangle] x(\dots))$.

As the goal is to prove inconvertibility, we may without loss of generality assume that $n_k \neq 0$ since the fact $MN \neq_{\beta} M'N \implies M \neq_{\beta} M'$ allows us to append \mathbf{G}_n with arbitrary $n > 0$. We argue that the starting positions of the subsequences $\langle (*)_{n_i, m_i} \rangle$ in $\langle (*) \rangle$ coincide with the occurrences of subsequences of the form $(\blacklozenge) 1^l, 1^{l+1}, 1^l, 1^{l-1}, 1^{l-2}$ (for some $l \in \mathbb{N}$), that is, an increment followed by four decrements. (In particular, we find exactly the patterns for $l = n_i + m_i + 1$.) This suffices to derive k, n_1, \dots, n_k since then the number of occurrences (\blacklozenge) is $k - 1$, and the n_i 's are a function of the length of the blocks; see also Example 7.4 below. This shows that $\vec{n} \neq \vec{m}$ implies that $\neg(\mathbf{BT}_{\blacktriangleleft}(\mathbf{Y}^{\vec{n}}) =_{\exists} \mathbf{BT}_{\blacktriangleleft}(\mathbf{Y}^{\vec{m}}))$, and hence we conclude $\mathbf{Y}^{\vec{n}} \neq_{\beta} \mathbf{Y}^{\vec{m}}$ by atomic version of Corollary 4.12, see Theorem 7.2.

For the sequences $\langle (*)_{n_i, m_i} \rangle$ where $n_i > 0$, notice that $\langle (*)_{n_i, m_i} \rangle$ starts with the positions $1^{n_i+m_i+1}, 1^{n_i+m_i+2}, 1^{n_i+m_i+1}, 1^{n_i+m_i}, 1^{n_i+m_i-1}$, a subsequence of the form (\blacklozenge) ; this is the only occurrence of four consecutive decrements in $\langle (*)_{n_i, m_i} \rangle$. For sequences $\langle (*)_{n_i, m_i} \rangle$ with $n_i = 0$, we have $i < k$ (since $n_k \neq 0$), and $\langle (*)_{n_i, m_i} \rangle$ is of the form $1^{m_i+1}, 1^{m_i+2}, 1^{m_i+1}, 1^{m_i}$. This combined with the first element $m_i - 1 = m_{i+1} + n_{i+1} + 1$ of $\langle (*)_{n_{i+1}, m_{i+1}} \rangle$ is an occurrence of the form (\blacklozenge) . Finally, we need to check that there are no other occurrences of (\blacklozenge) in $\langle (*) \rangle$. Note that other occurrences of four consecutive decrements can only occur as overlaps between $\langle (*)_{n_i, m_i} \rangle$ and $\langle (*)_{n_{i+1}, m_{i+1}} \rangle$. Each of the sequences $\langle (*)_{n_{i+1}, m_{i+1}} \rangle$ starts with an increment $1^{n_{i+1}+m_{i+1}+1}, 1^{n_{i+1}+m_{i+1}+2}$, thus only the first element can overlap. For $n_i = 0$, we have already analyzed the overlap, and for $n_i > 1$, only the last three elements of $\langle (*)_{n_i, m_i} \rangle$ are not decreasing. This concludes the proof. \square

Example 7.4. Let $\mathbf{Y} = \theta\theta\mathbf{G}'_2\mathbf{G}_0\mathbf{G}_1$ so that $\mathbf{Y}^{\langle 2,0,1 \rangle}x \rightarrow \mathbf{Y}x$ as in the above proof. According to the calculations in the proof, the atomic clock of the simple term $\mathbf{Y}x$ is

$$\langle 1^9, 1^8, 1^7, 1^8, 1^7, 1^6, 1^7, 1^6, \underbrace{1^5, 1^6, 1^5, 1^4}_{(\blacklozenge)}, \overbrace{1^3, 1^4, 1^3, 1^2, 1^1}^{(\blacklozenge)}, 1^2, 1^1, 1^0, 1^1 \rangle$$

where we have indicated the occurrences of (\blacklozenge) , and have alternately used italics to separate the blocks $(\dagger)_{2,6}$, $(*)_{0,4}$, $(*)_{1,1}$, and (\ddagger) .

8. CLOCKED LÉVY–LONGO AND BERARDUCCI TREES

In fact, there are three main semantics for the λ -calculus: \mathbf{BT} , \mathbf{LLT} , and \mathbf{BeT} ; see [1, 6, 8, 18, 5]. In the Böhm Tree semantics, a term is meaningful only if it has a hnf. The Lévy–Longo semantics weakens this condition to whnf's, and thereby allows more terms to be distinguished. The Berarducci Tree semantics is a further weakening where only root-active terms are discarded as meaningless.

The notions from the Sections 4 and 7 generalize directly to \mathbf{LLT} and \mathbf{BeT} semantics. We only treat the non-atomic versions here.

Definition 8.1 (Clocked Lévy–Longo Trees). Let M be a λ -term. The *clocked Lévy–Longo Tree* $\mathbf{LLT}_{\odot}(M)$ of M is an annotated potentially infinite term defined as follows. If M has no whnf, then define $\mathbf{LLT}_{\odot}(M)$ as \perp . Otherwise, there exists a head rewrite

sequence $M \rightarrow_h^k \lambda x.N$ or $M \rightarrow_h^k xM_1 \dots M_m$ to whnf. In this case, we define $\text{LLT}_{\circlearrowleft}(M)$ as $[k]\lambda x.\text{LLT}_{\circlearrowleft}(N)$ or $[k]x\text{LLT}_{\circlearrowleft}(M_1) \dots \text{LLT}_{\circlearrowleft}(M_m)$, respectively.

Definition 8.2 (Clocked Berarducci Trees). Let M be a λ -term. The *clocked Berarducci Tree* $\text{BeT}_{\circlearrowleft}(M)$ of M is an annotated potentially infinite term defined as follows. If M is root-active, let $\text{BeT}_{\circlearrowleft}(M) \equiv \perp$. If $M \rightarrow_h^k N$ rewrites to a root-stable term $N \equiv x$, $N \equiv \lambda x.P$ or $N \equiv PQ$, then define $\text{BeT}_{\circlearrowleft}(M)$ as $[k]x$, $[k]\lambda x.\text{BeT}_{\circlearrowleft}(P)$ or $[k]\text{BeT}_{\circlearrowleft}(P)\text{BeT}_{\circlearrowleft}(Q)$, respectively.

Example 8.3. Consider the terms $M = PP$ with $P = \lambda x.\lambda y.xx$ and $N = QQ$ with $Q = \lambda x.\lambda y.\lambda z.xx$. Then we find

$$\begin{aligned}\text{LLT}_{\circlearrowleft}(M) &\equiv [1]\lambda y.\text{LLT}_{\circlearrowleft}(M) \\ \text{LLT}_{\circlearrowleft}(N) &\equiv [1]\lambda y.[0]\lambda z.\text{LLT}_{\circlearrowleft}(N)\end{aligned}$$

Thus, in $\text{LLT}_{\circlearrowleft}(M)$ every λ requires one head reduction step whereas in $\text{LLT}_{\circlearrowleft}(N)$ every second λ is obtained for ‘free’ (that is, in 0 steps).

We remark that M and N cannot be distinguished in the Böhm Tree semantics since $\text{BT}_{\circlearrowleft}(M) \equiv \text{BT}_{\circlearrowleft}(N) \equiv \perp$.

9. CONCLUDING REMARKS

We conclude with an encompassing conjecture, and some further research questions.

Conjecture. *Building fpc’s with fpc-generating vectors is a free construction, that is, there are no non-trivial identifications.*

A first step is found in Intrigila’s theorem $Y\delta \neq_{\beta} Y$, for any fpc Y . A second step is that the Böhm sequence is duplicate-free. A third step is found in our proof that the Scott sequence is duplicate-free, and Proposition 7.3, which states that there are no identifications when starting the construction with \mathbb{Y}_0 .

Other parts of the conjecture are as follows. Let Y, Y' be fpc’s and $B_1 \dots B_n, C_1 \dots C_k$ be fpc-generating vectors.

- (1) $Y\delta =_{\beta} Y'\delta$ iff $Y =_{\beta} Y'$.
- (2) $YB_1 \dots B_n =_{\beta} Y'B_1 \dots B_n$ iff $Y = Y'$.
- (3) $YB_1 \dots B_n \neq_{\beta} YC_1 \dots C_k$ if $B_1 \dots B_n \not\equiv C_1 \dots C_k$.

For general fpc’s Y, Y' these conjectures may be beyond current techniques, but for the well-known fpc’s of Curry and Turing, and the fpc-generating vectors introduced here, including their versions for $n > 3$, these problems are tractable.

Other directions of research could be

- (1) For atomic clock Böhm Trees (Section 7) we have recorded the positions of head reduction steps building up the head normal form. What about a generalization to other spine reduction strategies [3]? Would this give rise to a stronger discrimination method?
- (2) The notion of simple terms could be refined by focusing on an infinite path in the clocked Böhm Trees. Then duplication of redexes may be allowed along other paths, thereby making the method applicable to a larger class of terms.
- (3) What general condition on a given term’s head reduction is sufficient to ensure that it possesses a ‘minimal clock’?
- (4) Is it possible to characterize fully cyclic terms using a coinductive version of simple type theory?

- (5) Can the notion of a clock itself be given a type-theoretic interpretation?
- (6) Can the clocks method be used to give a quantitative measure for optimization of functional programs?
- (7) Is it possible, using the clocks method, to extend Intrigila’s result [16], and prove that there is no fixed point combinator Y and no $n \in \mathbb{N}$ such that $Y =_{\beta} Y\delta^{\sim n}$? This is an instance of the conjecture above.
- (8) Also interesting is to systematically study all solutions of equations $M\vec{x} = \vec{x}(M\vec{x})$ (we might call them *vector-fpc’s*) where it is understood that $\vec{x} = x_1 \dots x_n$ associates to the right when it occurs at an active position $\vec{x}M = x_1(x_2(\dots(x_nM)\dots))$, and to the left if it occurs in a passive position $M\vec{x} = Mx_1x_2 \dots x_n$. Can these solutions M be discriminated by the methods presented here? Notice that the terms M lead to fpc’s; for example $M\vec{I}^{\sim n-1}$ is an fpc (see Section 3.3), or more generally, $M\vec{N}$ with $\vec{N} = N_1 \dots N_{n-1}$ is an fpc when $\vec{N}x \rightarrow_{\beta} x$.
- (9) Another interesting notion is that of *prime fpc’s*, that is, fpc’s Y not of the form $Y = Y'\vec{P}$ where Y' is an fpc. In λ -calculus no prime fpc’s exist, due to $Y =_{\beta} Y(KY)$, but in the λ l-calculus the notion is non-trivial.

Acknowledgements. We thank Raymond Smullyan for raising our awareness of the mystery and magic surrounding fixed point combinators, Hans Zantema and Johannes Waldmann for communicating some facts about Smullyan’s Owl (SI), Henk Barendregt for intensive discussions which led us to simple terms, Rick Statman for his stimulating interest in this work and suggesting some future elaboration, and Gordon Plotkin for his question and subsequent communication pertinent to Selinger’s work concerning the existence of unorderable models of the λ -calculus.

REFERENCES

- [1] S. Abramsky and C.-H.L. Ong. Full Abstraction in the Lazy Lambda Calculus. *Information and Computation*, 105(2):159–267, 1993.
- [2] K. Aehlig and F. Joachimski. On Continuous Normalization. In *Proc. Workshop on Computer Science Logic (CSL 2002)*, volume 2471 of *LNCS*, pages 59–73. Springer, 2002.
- [3] H.P. Barendregt. *The Lambda Calculus. Its Syntax and Semantics*, volume 103 of *Studies in Logic and The Foundations of Mathematics*. North-Holland, revised edition, 1984.
- [4] H.P. Barendregt, W. Dekkers, and R. Statman. *Lambda Calculus with Types*. Perspectives in Logic. Cambridge University Press, 2011.
- [5] H.P. Barendregt and J.W. Klop. Applications of Infinitary Lambda Calculus. *Information and Computation*, 207(5):559–582, 2009.
- [6] A. Berarducci and B. Intrigila. Church–Rosser λ -theories, Infinite λ -calculus and Consistency Problems. *Logic: From Foundations to Applications*, pages 33–58, 1996.
- [7] I. Bethke. Lambda Calculus. Chapter 10 in [30].
- [8] I. Bethke, J.W. Klop, and R.C. de Vrijer. Descendants and Origins in Term Rewriting. *Information and Computation*, 159(1–2):59–124, 2000.
- [9] C. Böhm. The CUCH as a Formal and Description Language. *Annual Review in Automatic Programming*, 3:179–197, 1963.
- [10] Th. Coquand and H. Herbelin. A-Translation and Looping Combinators in Pure Type Systems. *Journal of Functional Programming*, 4(1):77–88, 1994.
- [11] M. Dezani-Ciancaglini, P. Severi, and F. J. de Vries. Infinitary Lambda Calculus and Discrimination of Berarducci Trees. *Theoretical Computer Science*, 2(298):275–302, 2003.
- [12] J. Endrullis, C. Grabmayer, J.W. Klop, and V. van Oostrom. On Equal μ -Terms. *Theoretical Computer Science*, 412(28):3175–3202, 2011.

- [13] J. Endrullis, D. Hendriks, and J.W. Klop. Modular Construction of Fixed Point Combinators and Clocked Böhm Trees. In *Proc. Symp. on Logic in Computer Science (LICS 2010)*, pages 111–119, 2010.
- [14] H. Geuvers and B. Werner. On the Church–Rosser Property for Expressive Type Systems and its Consequences for their Metatheoretic Study. In *Proc. Symp. on Logic in Computer Science (LICS 1994)*, pages 320–329, 1994.
- [15] M. Goldberg. Constructing Fixed-Point Combinators Using Application Survival. Technical Report BRICS RS-95-35, Dept. of Computer Science, University of Aarhus, 1995.
- [16] B. Intrigila. Non-Existent Statman’s Double Fixed Point Combinator Does Not Exist, Indeed. *Information and Computation*, 137(1):35–40, 1997.
- [17] F. Joachimski. Confluence of the Coinductive λ -Calculus. *Theoretical Computer Science*, 311(1-3):105–119, 2004.
- [18] R. Kennaway and F.-J. de Vries. Infinitary Rewriting. Chapter 12 in [30].
- [19] R. Kennaway, J.W. Klop, M.R. Sleep, and F.-J. de Vries. Infinitary Lambda Calculus. *Theoretic Computer Science*, 175(1):93–125, 1997.
- [20] J.W. Klop. New Fixed Point Combinators from Old. In *Reflections on Type Theory, λ -Calculus, and the Mind. Essays Dedicated to Henk Barendregt on the Occasion of his 60th Birthday*, pages 197–210. 2007. Online version: <http://www.cs.ru.nl/barendregt60>.
- [21] G.D. Plotkin. A Semantics for Type Checking. In *Proc. Conf. on Theoretical Aspects of Computer Software (TACS 1991)*, volume 526 of *LNCS*, pages 1–17. Springer, 1991.
- [22] G.D. Plotkin, 2007. Personal communication at the symposium for H. Barendregt’s 60th birthday.
- [23] D. Sangiorgi and J.J.M.M. Rutten. *Advanced Topics in Bisimulation and Coinduction*, volume 52 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2012.
- [24] D.S. Scott. Some Philosophical Issues Concerning Theories of Combinators. In C. Böhm, editor, *Lambda Calculus and Computer Science Theory*, volume 37 of *LNCS*, pages 346–366, 1975.
- [25] P. Selinger. Order-Incompleteness and Finite Lambda Models. In *Proc. Symp. on Logic in Computer Science (LICS 1996)*, pages 432–439, 1996.
- [26] P. Selinger. *Functionality, Polymorphism, and Concurrency: a Mathematical Investigation of Programming Paradigms*. PhD thesis, University of Pennsylvania, 1997.
- [27] P. Selinger. Order-Incompleteness and Finite Lambda Reduction Models. *Theoretical Computer Science*, 309(1–3):43–63, 2003.
- [28] R. Smullyan. *To Mock a Mockingbird, and Other Logic Puzzles: Including an Amazing Adventure in Combinatory Logic*. Alfred A. Knopf, New York, 1985.
- [29] R. Statman. The Word problem for Smullyan’s Lark Combinator is Decidable. *Journal of Symbolic Computation*, 7:103–112, 1989.
- [30] Terese. *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2003.