BRICS

Basic Research in Computer Science

On the Recursive Enumerability of Fixed-Point Combinators

Mayer Goldberg

BRICS Report Series RS-04-25

ISSN 0909-0878 November 2004

See back inner page for a list of recent BRICS Report Series publications.
Copies may be obtained by contacting:

BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK–8000 Aarhus C
Denmark
Telephone: +45 8942 3360
Telefax: +45 8942 3255
Internet: BRICS@brics.dk

BRICS publications are in general accessible through the World Wide Web and anonymous FTP through these URLs:

`http://www.brics.dk`
`ftp://ftp.brics.dk`
This document in subdirectory `RS/04/25/`

On the Recursive Enumerability of Fixed-Point Combinators

Mayer Goldberg*

November, 2004

Abstract

We show that the set of fixed-point combinators forms a recursively-enumerable subset of a larger set of terms that is (A) not recursively enumerable, and (B) the terms of which are observationally equivalent to fixed-point combinators in any computable context.

1 Introduction

Fixed-point combinators are considered classical material by now in the λ-calculus, combinatory logic, and functional programming. They used to define recursive functions and circular data structures by replacing recursion and circularity with self-application. Many fixed-point combinators are known, including ones by Curry [3, Page 178], Turing [7], Klop [1]. A well-known construction due to Böhm is used to show that there are infinitely-many distinct fixed-point combinators is due to [6].

In this paper we show that the set of fixed-point combinators is recursively enumerable. We found the implications of this result quite surprising: If we think of a fixed-point combinator as a device for defining recursive functions, then the existence of a partial recursive function that identifies fixed-point combinators suggests that we may be able to identify, either statically or dynamically, certain classes of λ-terms as having a potential for non-termination, given that they reduce to terms that contain a fixed-point combinator as a subexpression. The second result presented in this paper sets the theoretical limit for this endevor: It has been known for some time that there exist terms that have the Böhm-tree of a fixed-point combinator, but that in fact fail to satisfy the equation that defines a fixed-point combinator [5]. Such terms can however be used to define recursive functions and circular structures in just the same way as ordinary fixed-point combinators are used. These curious terms are indistinguishable from fixed-point combinators in any applicative context. Put

*Department of Computer Science, Ben Gurion University, P.O. Box 653, Beer Sheva 84105, Israel.

otherwise, such terms are observationally equivalent to fixed-point combinators. We show that the set of fixed-point combinators forms a recursively enumerable subset of a larger set that is not recursively enumerable, and therefore, cannot be enumerated by a recursive function. We refer to terms in this larger set as *non-standard fixed-point combinators*, and terms that fail to satisfy the fixed point equation as *strictly non-standard fixed-point combinators*.

1.1 Notation and Pre-requisites

This work is carried out in the $\lambda\mathbf{K}\beta\eta$-calculus [1]. The $=$ symbol on λ-terms is taken to mean textual identity of language tokens, unless otherwise stated (e.g., taken modulo α-conversion). The one-step $\beta\eta$-reduction is given by $\longrightarrow\!\!\!\!\!\rightarrow$, and the equivalence relation induced by the $\beta\eta$-relation is given by $=_{\beta\eta}$. Equality "by definition" is given by $\equiv$. The set of λ-terms is given by Λ. The set of combinators, given by Λ^0, is the set of terms with no free variables (aka "closed terms"). We use Church numerals [2, 1] to do arithmetic in the λ-calculus. The n-th Church numeral is given by c_n. The successor function on Church numerals is given by **Succ**. The truth values in logic *False*, *True* are modeled in the λ-calculus using the combinators:

$$\begin{aligned} \mathbf{False} &= \lambda xy.y \\ \mathbf{True} &= \lambda xy.x \end{aligned}$$

For any λ-term F, the fixed point of F is a λ-term x such that $Fx =_{\beta\eta} x$. A fixed-point combinator [1, Section 6.1] is a λ-term with the property that when applied to any λ-term x it returns the fixed point of x. More formally, a λ-term Φ is a fixed-point combinator if for any λ-term x, the following holds:

$$\Phi x \quad =_{\beta\eta} \quad x(\Phi x)$$

The set $\mathbb{N}$ is the set of natural numbers not including zero. We use the notation $\{0\} \cup \mathbb{N}$ to refer to the set of non-negative integers.

2 The set of fixed-point combinators is recursively enumerable

We present a construction enumerating the set of fixed-point combinators. A similar construction can be shown for sets of n multiple fixed-point combinators, for any $n > 1$.

2.1 THEOREM: *The set of fixed-point combinators is recursively enumerable.*

Proof: A combinator Φ is a fixed-point combinator if for any term $x \in \Lambda$ we have $\Phi x =_{\beta\eta} x(\Phi x)$. By the ξ-rule [1, Page 23], we abstract over x to get $\lambda x.\Phi x =_{\beta\eta} \lambda x.x(\Phi x)$. By η-reducing the left-hand side we get $\Phi =_{\beta\eta} \lambda x.x(\Phi x)$.

Therefore, by the Church-Rosser theorem, there exist $n, m \in \mathbb{N}, \Psi \in \Lambda$, such that

$$\begin{array}{rl} \Phi & \underbrace{\longrightarrow_{\beta\eta} \cdots \longrightarrow_{\beta\eta}}_{m} \quad \Psi \\ \lambda x.x(\Phi x) & \underbrace{\longrightarrow_{\beta\eta} \cdots \longrightarrow_{\beta\eta}}_{n} \quad \Psi \end{array} \tag{1}$$

This equality can be verified mechanically, terminating if the equality holds, and diverging otherwise. This, together with the fact that the set Λ^0 (of combinators) is recursively enumerable, implies that the set of fixed-point combinators is recursively enumerable. A detailed construction follows.

The set Λ^0 is recursively enumerable, so there exists an effective surjection $\mathfrak{C} : \mathbb{N} \to \Lambda^0$. Let $V_0 = \emptyset$. For any $j > 0$ we define:

$$\begin{array}{rcl} C_j & \equiv & \mathfrak{C}(j) \\ D_j & \equiv & \lambda x.x(C_j x) \end{array}$$

If we can show that $C_j =_{\beta\eta} D_j$, then it follows that C_j is a fixed-point combinator. To do this "in parallel", for $j = 1, 2, \ldots$, we introduce the structure Q_j:

$$Q_j \quad = \quad \langle C_j, \{C_j\}, \{D_j\}, j \rangle$$

We use this structure to maintain all the terms that can be reached from C_j and D_j via any finite number of $\beta\eta$-steps, so as we move from V_j to V_{j+1}, the second and third projections of each of the quadruples "grow" to include all terms that can be reached by an additional $\beta\eta$-step. We now define the set F_j of all quadruples for which we can now show that the first projection is a fixed-point combinator:

$$F_j \quad = \quad \{\langle x_1, x_2, x_3, x_4 \rangle \in V_j : x_2 \cap x_3 \neq \emptyset\}$$

Note that for all quadruples in F_j (and indeed in V_j as well), the relation between the first and fourth projections is given by $x_1 = \mathfrak{C}(x_4)$. Consequently, there is a unique ordering on the quadruples in F_j that is induced by the ascending order of the fourth projection of each quadruple. The first projection of any quadruple in F_j is a fixed-point combinator, and we list these fixed-point combinators in the ordering induced by the ordering on the quadruples. We now remove the structures that correspond to fixed-point combinators:

$$W_j \quad = \quad V_j \setminus F_j$$

And extend the second and third projections respectively, to include all the terms that can be reached after one additional $\beta\eta$-step:

$$\begin{array}{rcl} W'_j & = & \{\langle x_1, x'_2, x'_3, x_4 \rangle : \\ & & \quad \langle x_1, x_2, x_3, x_4 \rangle \in W_j, \\ & & \quad x'_2 = x_2 \cup \{y : x \rightarrow_{\beta\eta} y, \text{ for some } x \in x_2\} \\ & & \quad x'_3 = x_3 \cup \{y : x \rightarrow_{\beta\eta} y, \text{ for some } x \in x_3\}\} \end{array}$$

The set V_{j+1} is defined to contain both the new structure Q_j, as well as the extended quadruples:

$$V_{j+1} \quad = \quad \{Q_j\} \cup W'_j$$

The enumeration of the set of fixed-point combinators is obtained by enumerating the first projections in all quadruples in $\{F_j\}_{j\in\mathbb{N}}$ according to the ordering induced by j, as the primary index, and then by the fourth projections in each F_j, as the secondary index. ■

3 Non-standard fixed-point combinators

All fixed-point combinators have the same infinite extension $\lambda f.f(f(f\cdots))$, which is the same as stating they all have the same Böhm tree [1, Page 217]:

$$\begin{array}{c} \lambda f.f \\ | \\ f \\ | \\ \vdots \end{array}$$

The converse does not hold, i.e., it is not the case that any term with the infinite extension $\lambda f.f(f(f\cdots))$ is a fixed-point combinator. Statman gives an example of such a term in his paper *Some Examples of Non-Existent Combinators* [5, Page 442], where for $M \equiv \lambda x.xx, B \equiv \lambda xyz.x(yz)$, he makes the observation that:

> Note that $BM(B(BM)B)$ has the right Böhm tree to be a fixed point combinator but it is not one.

The term pointed out by Statman is one of many similar terms that fail to satisfy Equation (1) for any finite m, n. Such terms, however, can be used to define recursive functions. This motivates the following definition.

3.1 Definition: *A non-standard fixed-point combinator.* A term Ψ is a non-standard fixed-point combinator if Ψ has the same Böhm-tree as a fixed-point combinator.

Clearly, the set of fixed-point combinators is a proper subset of the set of non-standard fixed-point combinators. We therefore refer to non-standard fixed-point combinator that is not a fixed-point combinator as a *strictly non-standard fixed-point combinator.*

The above definition can be extended to sets of n multiple fixed-point combinators, for $n > 1$.

We can define a non-standard fixed-point combinator Ψ by abstracting f over an expression E_1 that reduces to an application of f over an expression E_2, etc, so that Ψ has the infinite extension $\lambda f.f(f(f\cdots))$. For Ψ to be a strictly

non-standard fixed-point combinator, the set of terms $\{E_n\}_{n\in\mathbb{N}}$ must satisfy $\neg(E_n =_{\beta\eta} E_{n+1})$ for all $n \in \mathbb{N}$.

We define our own construction for a non-standard fixed-point combinator as follows:

$$\begin{array}{rcl} \Psi & \equiv & \lambda f.((\lambda xn.f(xx(\mathbf{Succ}\ n))) \\ & & \quad (\lambda xn.f(xx(\mathbf{Succ}\ n))) \\ & & \quad c_1) \end{array}$$

The corresponding E_n, E_{n+1} are as follows:

$$\begin{array}{rcl} E_n & = & ((\lambda xn.f(xx(\mathbf{Succ}\ n))) \\ & & \quad (\lambda xn.f(xx(\mathbf{Succ}\ n))) \\ & & \quad c_n) \\ E_{n+1} & = & ((\lambda xn.f(xx(\mathbf{Succ}\ n))) \\ & & \quad (\lambda xn.f(xx(\mathbf{Succ}\ n))) \\ & & \quad c_{n+1}) \end{array}$$

and cannot be shown to be equal within a finite number of $\beta\eta$-steps.

The definition of Ψ may appear contrived, since it makes no real use of n, and any good compiler e.g., for Scheme or Common LISP, would optimize it away, reducing Ψ to Curry's well-known fixed-point combinator. It is not difficult, however, to force n to do some useful work, and thus prevent its elimination by an optimizing compiler. For example, we could apply it to f, as in this alternate non-standard fixed-point combinator:

$$\begin{array}{rcl} \Psi' & \equiv & \lambda f.((\lambda xn.nf(xx(\mathbf{Succ}\ n))) \\ & & \quad (\lambda xn.nf(xx(\mathbf{Succ}\ n))) \\ & & \quad c_1) \end{array}$$

3.2 THEOREM: *The set of non-standard fixed-point combinators is not recursively enumerable.*

Proof: Let S be the set of non-standard fixed-point combinators. The structure of our proof is as follows:

- We show that S is not recursive.
- We show that the complement of S, given by $\overline{\mathsf{S}} = \Lambda^0 \setminus \mathsf{S}$, is recursively enumerable.

It follows from the above two items that S is not recursively enumerable, for if both $\mathsf{S}, \overline{\mathsf{S}}$ are recursively enumerable, then both are recursive [4, Page 58]. The details of the proof follow.

If S is recursive, then there exists a combinator E, such that for any combinator M, with encoding $\ulcorner M \urcorner$, we have:

$$(E\ \ulcorner M \urcorner) \quad \longrightarrow\!\!\!\!\!\rightarrow \quad \begin{cases} \mathbf{True} & \text{for } M \in \mathsf{S} \\ \mathbf{False} & \text{otherwise} \end{cases}$$

Let M_0 be any non-standard fixed-point combinator, with encoding $\ulcorner M_0 \urcorner$. Let **Turing** be a combinator that takes an encoding of a Turing machine T, given by $\ulcorner T \urcorner$, and returns the identity combinator $\mathbf{I}$ if T terminates (and diverges if T diverges). Using **Turing**, we define the combinator F as follows:

$$F \equiv \lambda t.(\mathbf{Turing}\ t\ M_0)$$

For any Turing machine T, the application $(E\ \ulcorner (F\ \ulcorner T \urcorner) \urcorner)$ returns **True** if T terminates, and returns **False** if T diverges. This decides the Halting Problem for Turing machines, which is of course, not possible. Hence E does not exist, and S is not recursive.

The set $\overline{\mathsf{S}}$ is the set of terms outside the set of non-standard fixed-point combinators, i.e., the set of terms that do not have the infinite extension $\lambda f.f(f(f\cdots))$. We show that $\overline{\mathsf{S}}$ is recursively enumerableby presenting an effective enumeration of its elements. The proof proceeds much like the proof of Theorem 2.1: Once again, we enumerate the set of combinators, carrying each combinator along in some structure (an ordered triple in this case), and filter out those terms that match our criteria. The details are given below.

We make use of the effective surjection $\mathfrak{C} : \mathbb{N} \to \Lambda^0$ from the proof of Theorem 2.1. Let $V_0 = \emptyset$. For any $j > 0$ we define:

$$\begin{aligned} C_j &\equiv \mathfrak{C}(j) \\ Q_j &\equiv \langle C_j, \{C_j\}, j\rangle \end{aligned}$$

We use this structure to maintain all the terms that can be reached from C_j via any finite number of $\beta\eta$-steps, so as we move from V_j to V_{j+1}, the second projection of each of the triples "grows" to include all the terms that can be reached by an additional $\beta\eta$-step. We now define the set of F_j of all triples for which we can now show that the first projection fails to have the infinite extension $\lambda f.f(f(f\cdots))$. We will make use of the effective predicate $N(x)$ as a "filter":

$$\begin{aligned} F_j &= \{\langle x_1, x_2, x_3\rangle : N(x), \text{ for some } x \in x_2\} \\ N(x) &= \begin{cases} \mathit{True} & \text{for } x \neq \lambda f.f^n((\lambda x.A)B), \text{ modulo} \\ & \alpha\text{-equivalence, and for some} \\ & A, B \in \Lambda, n \in \{0\} \cup \mathbb{N} \\ \mathit{False} & \text{otherwise} \end{cases} \end{aligned}$$

We now remove the structures that correspond to terms that were found not to have the infinite extension $\lambda f.f(f(f\cdots))$:

$$W_j = V_j \setminus F_j$$

We now extend the second projection to include all the terms that can be reached after one additional $\beta\eta$-step:

$$\begin{aligned} W'_j = \{\langle x_1, x'_2, x_3\rangle : \\ x'_2 = x_2 \cup \{y : x \to_{\beta\eta} y, \text{ for some } x \in x_2\}\} \end{aligned}$$

The set V_{j+1} is defined to contain both the new structure Q_j, as well as the extended triples:

$$V_{j+1} \quad = \quad \{Q_j\} \cup W'_j$$

The enumeration of the set $\overline{\mathsf{S}}$ is obtained by enumerating the first projections in all the triples in $\{F_j\}_{j\in\mathbb{N}}$ according to the ordering induced by j, as the primary index, and then by the third projections in each F_j, as the secondary index. ■

Acknowledgments

This work was supported in part by the Danish Research Academy. I am grateful to BRICS for hosting me and for providing a stimulating environment. Thanks are also due to Olivier Danvy, Neil Jones, Menachem Kojman, and Torben Mogensen for their comments and encouragement.

References

[1] Hendrik P. Barendregt. *The Lambda Calculus, Its Syntax and Semantics.* North-Holland, 1984.

[2] Alonzo Church. *The Calculi of Lambda-Conversion.* Princeton University Press, 1941.

[3] Haskell B. Curry, Robert Feys, and William Craig. *Combinatory Logic*, volume I. North-Holland Publishing Company, 1958.

[4] Hartley Jr. Rogers. *Theory of Recursive Functions and Effective Computability.* (publisher The MIT Press), 1987.

[5] Richard Statman. Some Examples of Non-Existent Combinators. *Theoretical Computer Science*, 121:441–448, 1993.

[6] Joseph E. Stoy. *Denotational Semantics: the Scott-Strachey Approach to Programming Language Theory.* The MIT Press Series in Computer Science. MIT Press, 1977.

[7] Alan Turing. The $\mathfrak{p}$-functions in λ-k-conversion. *Journal of Symbolic Logic*, pages 164–164, 1937.

Recent BRICS Report Series Publications

RS-04-25 Mayer Goldberg. *On the Recursive Enumerability of Fixed-Point Combinators*. November 2004. 7 pp.

RS-04-24 Luca Aceto, Willem Jan Fokkink, Anna Ingólfsdóttir, and Sumit Nain. *Bisimilarity is not Finitely Based over BPA with Interrupt.* October 2004. 30 pp.

RS-04-23 Hans Hüttel and Jiří Srba. *Recursion vs. Replication in Simple Cryptographic Protocols*. October 2004.

RS-04-22 Gian Luca Cattani and Glynn Winskel. *Profunctors, Open Maps and Bisimulation.* October 2004. 64 pp. To appear in *Mathematical Structures in Computer Science.*

RS-04-21 Glynn Winskel and Francesco Zappa Nardelli. *New-HOPLA—A Higher-Order Process Language with Name Generation.* October 2004. 38 pp.

RS-04-20 Mads Sig Ager. *From Natural Semantics to Abstract Machines*. October 2004. 21 pp. Presented at the *International Symposium on Logic-based Program Synthesis and Transformation*, LOPSTR 2004, Verona, Italy, August 26–28, 2004.

RS-04-19 Bolette Ammitzbøll Madsen and Peter Rossmanith. *Maximum Exact Satisfiability: NP-completeness Proofs and Exact Algorithms*. October 2004. 20 pp.

RS-04-18 Bolette Ammitzbøll Madsen. *An Algorithm for Exact Satisfiability Analysed with the Number of Clauses as Parameter*. September 2004. 4 pp.

RS-04-17 Mayer Goldberg. *Computing Logarithms Digit-by-Digit*. September 2004. 6 pp.

RS-04-16 Karl Krukow and Andrew Twigg. *Distributed Approximation of Fixed-Points in Trust Structures*. September 2004. 25 pp.

RS-04-15 Jesús Fernando Almansa. *Full Abstraction of the UC Framework in the Probabilistic Polynomial-time Calculus ppc*. August 2004.

RS-04-14 Jesper Makholm Byskov. *Maker-Maker and Maker-Breaker Games are PSPACE-Complete.* August 2004. 5 pp.