Breakable Terms[1]

Tudományos Diákköri Dolgozat

Bárány Vince

témavezetô: Csörnyei Zoltán

ELTE TTK Általános Számítástudományi Tanszék

2002 December 2.

[1]Partially supported by OTKA T037742

Contents

Chapter 1

Introduction

1.1 Introduction

λ-calculus is one of the simplest formalisms imaginable, which is also suitable to represent computation as we know it. *Solvability* of λ-terms is a central notion related to the semantics of such representations, and the model theory of λ-calculus: it is proposed to capture exactly the "meaningful" terms; and it has a very interesting history:

After the foundation of λ-calculus by Alonzo Church his student Kleene showed that of the total numerical functions exactly the recursive functions can be λ-*defined* over the *Church numerals*. Later he also showed that of the partial numerical functions exactly the partial recursive functions can be λ-defined. In his construction, the natural numbers were again represented by the Church numerals, while terms without a *normal form* were taken to represent the undefined. Kleene's construction was implicitly built on the notion of solvability, which was defined only more than three decades later by H. P. Barendregt.

In 1971 Barendregt ([Bar71]) and Wadsworth independently arrived at the same concept using different approaches. Barendregt's semantic notion of solvability was proved to be equivalent to Wadsworth's syntactic definition of having a *head normal form* in λK-calculus, and to having a normal form in λI-calculus, thus closing the circle. But the role of solvability in Kleene's representation [1] has remained uncovered.

So the unsolvables were shown to be adequate representations of the undefined, when λ-defining the partial recursive functions. This already suggest that the unsolvable terms could map to the bottom element of an appropriate semantic domain of λ-calculus. Indeed, the Böhm tree model is one such model as proven by famous results as the *genericity lemma*, the consistent identifiability of all unsolvable terms and the fact that this is a maximal such set. These facts justify Barendregt's proposal to equate meaninglessness of λ-terms with unsolvability.

Motivated by these results joint efforts were made in the 1990's by Ariola, Kennaway, Klop, van Oostrom, Sleep and de Vries to grasp a similar notion of meaninglessness in the more general setting of infinitary term rewriting system. (For reference see [AKKSV94], [KKSV95], [KOV99]). Independently Kuper studied solvability in typed lambda-calculi. (See [Kup94], [Kup95], and [Kup97].)

Taking another path, in my investigations I have been considering possible refinements of the notion of solvability. In this paper we will look at one possible refinement, which I think is particularly interesting because it partitions solvable terms in just two classes. Thus we will define and study a subset of solvable terms, which I have collectively baptized *breakable*. We will show that breakability is an undecidable property of terms, we will investigate several examples making a few observations and giving necessary and sufficient conditions, but the full syntactic characterization of breakability remains an open problem.

[1] For a detailed representation of the (partial) recursive functions using Turing's idea in the λK-calculus and using Kleene's original method in the λI-calculus the reader should consult [Bar84] chapters 8 and 9 respectively, while [Bar92] presents a related and very general result of Statman.

1.2 Preliminaries

We assume the reader is familiar with λ-calculi and basic concepts of the theory of recursive functions. A good introduction to the former is [HS86], while [Bar84] is an indispensable reference on the subject. For the most up to date and thorough presentation of TRSs the reader should turn to [Ter02], which also contains an introduction to λ-calculus. [Rog67] is the usual reference on recursive functions.

In the sequel we give a short overview of the concepts and some "well known" results (without proofs) which we will be using throughout the text.

Let us first formulate some of the basic definitions and fundamental results of λ-calculus.

Basics

Definition 1.2.1. $\lambda K(\lambda I)$ -terms

1. $x \in Var \Longrightarrow x \in \Lambda_K(\Lambda_I)$
2. $M, N \in \Lambda_K(\Lambda_I) \Longrightarrow MN \in \Lambda_K(\Lambda_I)$
3. (a) $x \in Var, M \in \Lambda_K \Longrightarrow \lambda x.M \in \Lambda_K$
   (b) $x \in Var, M \in \Lambda_I, x \in FV(M) \Longrightarrow \lambda x.M \in \Lambda_I$

In the following and throughout this thesis M, N, L, P, Q will denote λK- or λI-terms, depending on the context.

Definition 1.2.2. (Substitution)

1. $x[x := M] \equiv M$
2. $y[x := M] \equiv y \quad (x \not\equiv y)$
3. $NL[x := M] \equiv N[x := M]L[x := M]$
4. $(\lambda x.N)[y := M] \equiv \begin{cases} \lambda x.N & (x \equiv y) \\ \lambda x.N[y := M] & (x \not\equiv y \wedge x \notin FV(M)) \\ \lambda z.N[x := z][y := M] & (x \not\equiv y \wedge x \in FV(M) \wedge z \ fresh) \end{cases}$

Definition 1.2.3. (Conversion, reduction)

1. $\lambda x.M \to_\alpha \lambda y.M[x := y] \quad (y \notin FV(M))$
2. $(\lambda x.M)N \to_\beta M[x := N]$
3. $\lambda x.Mx \to_\eta M$

We consider terms as syntactically equivalent (denoted by $\equiv$) if they are α-convertible, and equal (denoted by $=$) if they are $\alpha\beta$-convertible. This constitutes an intensional calculus, which can be made extensional by adding η-convertibility to equality. This equality will be denoted by $=_\eta$. From now on one-step and many-step β-reduction is simply denoted by $\to$ and $\twoheadrightarrow$ respectively.

Remark 1.2.4. (The variable convention)
We will always assume that all variables bound in a term in question are unique, i.e. different from all other bound as well as free variables of some terms in the same context. (This can always be achieved by α-conversion.)

Definition 1.2.5. Normal forms

1. A term M is in **normal form (nf)** iff it does not contain any $\beta(\eta\delta)$-redexes.

2. A term M is in **head normal form (hnf)** iff it is of the form $M \equiv \lambda\vec{x}.y\vec{N}$, where both $\vec{x}$ and $\vec{N}$ can be empty, and y may or may not be a component of $\vec{x}$.

3. A term M is in **weak head normal form (whnf)** iff it is of the form $M \equiv \lambda x.M'$ or $M \equiv y\vec{M}$.

Remark 1.2.6. In his thesis Kuper observed that a term M is in hnf if it is in whnf and in case it is of the form $\lambda x.M'$, then M' is in hnf as well. He then defined a fourth normal form as follows: a term M is in **fourth normal form (fnf)** iff it is of the form $M \equiv \lambda x.M'$ where M' is arbitrary or $M \equiv y\vec{M}$ where every component of $\vec{M}$ is in fnf as well. (See [Kup94] section 6.)

Let $\mathcal{NF}, \mathcal{HNF}, \mathcal{WHNF}, \mathcal{FNF}$ denote the sets of terms in nf, hnf, whnf, fnf, respectively. It is easy to see that all of these sets are closed under reduction. We say that a term has a certain normal form if it can be reduced to a term in that normal form. It can be easily verified (using **CR**) that this is equivalent to the condition that it can be converted to a term in that normal form. We will denote the set of terms convertible to one of these normal forms by $\mathcal{NF}^=, \mathcal{HNF}^=, \mathcal{WHNF}^=, \mathcal{FNF}^=$, respectively.

Theorem 1.2.7. *(Church-Rosser theorem)*
The $\lambda_{K(I)}$-calculus is confluent, i.e. $\forall M,N \in \Lambda_{K(I)} : M = N \Longrightarrow \exists L \in \Lambda_{K(I)} : M \twoheadrightarrow L \twoheadleftarrow N$.

Remark 1.2.8.
Confluent rewriting systems are also said to have the **Church-Rosser property (CR)**.

Every confluent rewriting system has the unique normal forms property, i.e. no two normal forms are convertible. This implies the following

Theorem 1.2.9. *(Consistency)*
Any confluent rewriting system, with at least two distinct normal forms is consistent. Consequently the λ-calculus is consistent.

Theorem 1.2.10. *(Scott's / Rice's theorem – cf. [Bar84] 6.6)*

1. *$\mathcal{A},\mathcal{B} \subset \Lambda$ non empty, closed under equality $\Longrightarrow$ $\mathcal{A}$ and $\mathcal{B}$ are not recursively separable.*

2. *$\mathcal{A} \subset \Lambda$ non trivial (i.e. $\mathcal{A} \neq \emptyset, \Lambda$), closed under equality $\Longrightarrow$ $\mathcal{A}$ is not recursive.*

Contexts

Contexts are "special terms" with exactly one occurrence of the special symbol [] (the hole), occurring anywhere where a variable is allowed. Formally:

Definition 1.2.11. Contexts are defined recursively as follows

1. $[] \in \mathfrak{C}$

2. $M \in \Lambda, C[] \in \mathfrak{C} \Longrightarrow M(C[]), (C[])M \in \mathfrak{C}$

3. $x \in Var, C[] \in \mathfrak{C} \Longrightarrow \lambda x.(C[]) \in \mathfrak{C}$

The result of substituting a term M into a context $C[]$ is a term obtained by replacing the special symbol [] by M in $C[]$, and is denoted by $C[M]$. Note that free variable occurrences in M may become bound in $C[M]$.

The composition of contexts $C_1[]$ and $C_2[]$ is defined as the context obtained by replacing the hole of $C_1[]$ by the context $C_2[]$, and is denoted by $C_1[C_2[]]$.

Remark 1.2.12. We can extend $\alpha\beta$-conversion to contexts, but care has to be taken to preserve convertibility of terms after substitution, i.e $M = N \wedge C[] = D[] \Longrightarrow C[M] = D[N]$. For this does not hold in general due to the fact that free variables of M and N can be bound by substitution into $C[]$ and $D[]$. For example: $\mathbf{I} = (\lambda x.x[x])\mathbf{I} \neq [x] = x$ so $(\lambda x.x[])\mathbf{I}$ and $[]$ should not be convertible. We can avoid such situations by prohibiting reduction steps which would modify the set of abstractions which contain $[]$ in their scope. For example $(\lambda xy.x)[] \to \lambda y.[]$ is forbidden, but $(\lambda x.x)[] \to []$ is not. It is clear that conversion under these restrictions is an equivalence relation.

Contexts, as defined above, are also referred to as one-hole contexts. We can define multi-hole contexts in a similar way: possibly having more, distinguished (e.g. numbered) holes. Substitution then requires the same number of terms, which replace the holes in some well defined manner (e.g. according to the numbering of the holes, or just "from left to right"), and the resulting term is denoted by $C[\vec{M}]$.

Solvability

Definition 1.2.13. Solvability in λ-calculi
A term M is **solvable** iff. $\exists \vec{x} \in \vec{Var}, \vec{N} \in \vec{\Lambda} : (\lambda \vec{x}.M)\vec{N} = \mathbf{I}$.
We denote the set of solvable and unsolvable terms by $\mathcal{S}$ and $\mathcal{U}$ respectively.

Theorem 1.2.14. *(Solvability and the normal forms)*

1. *(Barendregt) In the λI-calculus: a term is solvable iff it has a nf.*
2. *(Wadsworth) In the λK-calculus: a term is solvable iff it has a hnf.*

Theorem 1.2.15. *(Genericity Lemma) In the λK-calculus*
$\forall M \in \mathcal{U} \forall$ *context* $C[] : C[M] = L \in \mathcal{NF} \Longrightarrow \forall N \in \Lambda : C[N] = L$.

Böhm trees

In [Bar84] ch.10. Böhm trees are defined recursively as follows:

Definition 1.2.16.
The **Böhm tree of a term** M ($BT(M)$) is defined as

$$BT(M) = \begin{cases} \qquad\quad \bot & if\ M\ is\ unsolvable \\ \\ \qquad\quad \lambda\vec{x}.y & if\ M = \lambda\vec{x}.yM_1 \dots M_n \\ \qquad\quad / \dots \backslash & \\ BT(M_1) \quad BT(M_n) & \end{cases}$$

Böhm like trees are trees labeled by $\{\bot\} \cup \{\lambda\vec{x}.y | \vec{x}, y\ variables\}$. $\mathfrak{B}$ denotes the set of Böhm like trees, $\mathfrak{B}_I$ and $\mathfrak{B}_K$ denote the sets of Böhm trees of $\lambda I$and λKterms respectively.

Remark 1.2.17.
Böhm trees can be defined equivalently as normal forms of an extended transfinite calculus obtained by adjoining the symbol $\bot$ (*bottom*) to the alphabet, which can appear everywhere in terms in place of variables, and introducing a new reduction rule, called the $\bot$-rule as

$$\begin{array}{lll} M & \to_\bot \bot & (\forall M \in \mathcal{U}) \\ \lambda x.\bot & \to_\bot \bot & \\ \bot M & \to_\bot \bot & (\forall M \in \Lambda) \end{array}$$

Terms of this extended transfinite system are called Böhm terms, the reduction (by $\alpha\beta(\eta)$ and $\bot$ rules) Böhm reduction, and the normal forms are exactly the Böhm trees as defined above.

Definition 1.2.18. $BT(M) \sqsubseteq BT(N)$ iff $BT(N)$ agrees with $BT(M)$ in every position where $BT(M)$ is defined and is not $\perp$.

Proposition 1.2.19. $(\mathfrak{B}, \sqsubseteq)$ is a coherent algebraic cpo, and compact trees are exactly the finite ones.

Proposition 1.2.20. (Böhm tree model of λ-calculus)

$$M =_\beta N \Longrightarrow BT(M) = BT(N)$$
$$M, N \in \mathcal{NF}, M \not\equiv N \Longrightarrow BT(M) \neq BT(N)$$

Hence $\mathfrak{B}_K$ is a non-trivial model for the λKcalculus.

Finite Böhm like trees are Böhm trees, so every Böhm like tree can be approximated to any finite depth by a Böhm tree. We will use the following notations.

Definition 1.2.21.

- Let $\mathcal{B}$ be a finite Böhm (like) tree. Then $M(\mathcal{B})$ is the term obtained from $\mathcal{B}$ by replacing every $\perp$ by $\blacksquare$ in the tree, and reading it as a term.
- If $\mathcal{B}$ is an infinite Böhm like tree, then let $\mathcal{B}^{(k)}$ be the Böhm tree obtained from $\mathcal{B}$ by relabeling every note at depth k by $\perp$ and let $M^{(k)}(\mathcal{B}) \equiv M(\mathcal{B}^{(k)})$.
- For every $N \in \Lambda$ define $N^{(k)} \equiv M^{(k)}(BT(N))$ and $N^{[k]}$ to be the term obtained from N by performing outermost head reductions on redexes not included in any unsolvable subterms, until no redexes remain at depth less than k, hence

$$N^{(k)} \sqsubseteq N^{[k]} \twoheadleftarrow N$$

1.3 Notations

Sets

$\mathbb{N}$	$\{ 0, 1, 2, \dots \}$
$\mathfrak{C}$	the set of all contexts
Λ, Λ_K	the set of all λK-terms
Λ_I	the set of all λI-terms
Λ^0	the set of closed λ-terms
$\mathcal{NF}$	terms in normal form
$\mathcal{WN}$	weakly normalizing terms
$\mathcal{S}$	solvable (in λ-calculus) or usable (generalized) terms
$\mathcal{U}$	unsolvable (in λ-calculus) or unusable (generalized) terms
$\mathcal{B}$	the set of breakable terms (Definition 2.2.1)
$\overline{\mathcal{A}}$	denotes the complement (depending on context) of the set $\mathcal{A}$

Terms

$M\vec{N}$	is a shorthand notation for $MN_1 \dots N_n$ ($n \geq 1$ is assumed unless explicitely stated otherwise)
$\lambda\vec{x}.M$	is a shorthand notation for $\lambda x_1 \dots x_n.M$ (again $n \geq 1$ is assumed unless stated otherwise)
$F^k M$	$F^0 M \equiv M; F^{k+1} M \equiv F(F^k M)$ (taken from [Bar84] 2.1.9)
$FM^{\sim k}$	$FM^{\sim 0} \equiv F; FM^{\sim k+1} \equiv FM^{\sim k} M$ (taken from [Bar84] 2.1.9)
$FV(M)$	set of free variables in M
$BT(M)$	Böhm tree of M
$M^{(k)}$	the term associated with the k-initial segment of $BT(M)$
$M^{[k]}$	approximation of $BT(M)$ up to depth k by reduction from M (see Definition 1.2.21)

Combinators

I	$\lambda x.x$
K	$\lambda xy.x$
S	$\lambda xyz.xz(yz)$
T	$\lambda xy.x \equiv \mathbf{K}$
F	$\lambda xy.y \equiv \lambda yx.x$
$\mathbf{U^n_k}$	$\lambda x_1 \dots x_n.x_k$
Y	some fixed point combinator
D	$\lambda x.xx$
B	$\lambda xyz.x(yz)$
$\blacksquare$	**DD**

Properties

CR	the Church-Rosser property, i.e. confluence
NF	the normal forms property, i.e. $M = N, N \in \mathcal{NF} \implies M \twoheadrightarrow N$ i.e. $\vec{M} R \vec{N} \implies \forall C[] : C[\vec{M}] R C[\vec{N}]$

Chapter 2

Breakability in λ-calculus

2.1 How solvable is a term?

Recall that a λ-term M is solvable if it has a closure $\lambda x_1 \dots x_k.M$ and there are terms $N_1 \dots N_n$ $(n \geq 0)$ such that

$$(\lambda x_1 \dots x_k.M)\ N_1 \dots N_n = \mathbf{I}$$

or equivalently if M has a head normal form, i.e.

$$M = \lambda u_1 \dots u_l.vP_1 \dots P_t$$

Solvability is a candidate formalization of 'meaningfulness', however this categorization of terms can be further refined and elaborated as many interesting questions concerning the solvability of terms can be asked.

Our current question of interest concerns some sort of quantitative measure of the solvability of terms. We are trying to give an answer in terms of the minimal number of arguments that have to be applied to a given term M (or one of its closures) to obtain $\mathbf{I}$. This approach will turn out to be fruitful, yielding us the definition of breakability.

Let us first give a formal definition of the minimal solvability number of a term.

2.1.1 The minimal solvability number

Definition 2.1.1. let M be a λ-term, then

$$\begin{aligned} S(M) = &\quad \{n \in \mathbb{N} \mid \exists \vec{x}, \vec{N} = N_1 \dots N_n : (\lambda \vec{x}.M)\vec{N} = \mathbf{I}\} \\ m(M) = &\quad min(S(M) \cup \{\infty\}) \quad \textit{is its } \textbf{minimal solvability number} \end{aligned}$$

Remark 2.1.2. It is obvious that $n \in S(M) \Longrightarrow n+1 \in S(M)$ and that $m(M) < \infty \iff M\ solvable.$

Example 2.1.3. The standard combinators

$$\begin{aligned} i) &\quad m(x) = m(\mathbf{I}) = 0 \\ ii) &\quad m(\mathbf{F}) = m(\mathbf{D}) = m(\mathbf{Y}) = 1 \\ iii) &\quad m(\mathbf{K}) = m(\mathbf{S}) = 2 \\ iv) &\quad m(\blacksquare) = \infty \end{aligned}$$

Proof

i), iv) Obvious.

ii) $\mathbf{FD} = \mathbf{DF} = \mathbf{I}$, but $\mathbf{F}, \mathbf{D} \neq \mathbf{I}$; $\mathbf{Y}(\mathbf{KI}) = \mathbf{KI}(\mathbf{Y}(\mathbf{KI})) = \mathbf{I}$, but $\mathbf{Y} \neq \mathbf{I}$

iii) $\mathbf{KI}\blacksquare = \mathbf{I}$ and $\mathbf{SKI} = \mathbf{I}$, but $\mathbf{K}$ can not be solved with just one argument (because it ignores a second), nor can $\mathbf{S}$ be solved with less then two (because it is a triple abstraction). □

An immediate consequence of our definition is the following

Proposition 2.1.4. $M = N \Longrightarrow m(M) = m(N)$

Proof

Since $M = N \Longrightarrow (\lambda \vec{x}.M)\vec{P} = (\lambda \vec{x}.N)\vec{P}$. □

This also means that we can restrict our investigations of the minimal solvability number to terms in head normal form, since unsolvable terms are uninteresting from this point of view, and solvable terms have head normal forms which possess the same minimal solvability number. After making a few observations, we will turn our attention towards the relationship between the minimal solvability number of a term and the structure of its head normal form.

Proposition 2.1.5. $\forall M \in \Lambda^0\ \forall x \in Var : m(\lambda x.M) = m(M)+1$

Proof
Since M is closed, $(\lambda\vec{y}x.M)\vec{Y}X\vec{N} = M\vec{N} = (\lambda\vec{y}.M)\vec{Y}\vec{N}$, which implies that $S(\lambda x.M) = S(M)+1 = \{n+1 | n \in S(M)\}$, i.e. $m(\lambda x.M) = m(M)+1$. □

Corollary 2.1.6. $\forall M \in \Lambda^0 : m(\mathbf{K}M) = m(M)+1$

Proof
Because $\mathbf{K}M \to \lambda x.M$. □

It is easy to show (see e.g. [Bar84]), that if $M[\vec{x} := \vec{P}]$ is solvable, then so is M. Using our notation we can be a little more specific.

Proposition 2.1.7. $m(M) \leq m(M[x := P])+1$

Proof
$(\lambda\vec{y}.M[x := P])\vec{Q} = \mathbf{I} \Longrightarrow (\lambda x\vec{y}.M)P\vec{Q} = \mathbf{I}$
and we can conclude the result by definition. □

Remark 2.1.8.
Applying the result multiple times we get $m(M) \leq m(M[\vec{x} := \vec{P}]) + |\vec{x}|$.

Definition 2.1.9. A head normal form $\lambda x_1 \ldots x_n.y\vec{N}$ is **head closed** iff $y = x_i$ for some $1 \leq i \leq n$, otherwise it is **head free** ($n = 0$ is allowed, in which case the head normal form is automatically head free).

Remark 2.1.10. A closed head normal form is automatically head closed.

Lemma 2.1.11. $xN_1 \ldots N_n \neq y \iff n > 0$ or $x \neq y$

Proof
Since reductions can only take place within individual N_i-s. □

Regarding the relationship between the structural complexity of a term and its minimal solvability number, we can make the following simple observation.

Proposition 2.1.12. If a term M has a
i) head closed head normal form $\lambda x_1 \ldots x_n.x_i\vec{N}$ then $n-1 \leq m(M) \leq n$
ii) head free head normal form $\lambda x_1 \ldots x_n.y\vec{N}$ then $n \leq m(M) \leq n+1$
iii) in the special cases of $M \equiv x$ and $M \equiv x\vec{N}$ we have that $m(x) = 0$ and $m(x\vec{N}) = 1$ respectively.

Proof
i) $(\lambda x_1 \ldots x_n.x_i N_1 \ldots N_k)X_1 \ldots X_{i-1}\mathbf{U}^{\mathbf{k+1}}_{\mathbf{k+1}}X_{i+1} \ldots X_n = \mathbf{U}^{\mathbf{k+1}}_{\mathbf{k+1}}N'_1 \ldots N'_k = \mathbf{I}$
$\Longrightarrow m(M) \leq n$
$(\lambda x_1 \ldots x_n.N)X_1 \ldots X_{n-2} = \lambda x_{n-1}x_n.N' \neq \mathbf{I}$
$\Longrightarrow m(M) \geq n-1$
ii) Immediate from i) by Lemma 2.1.11 and considering the closure $\lambda y.M$ of M which is a head closed head normal form.
iii) The first case is obvious, the second follows with the help of Lemma 2.1.11. □

Remark 2.1.13. Every solvable term has a huge set of trivial solutions as suggested by the first part of i) in the proof, namely
$(\lambda x_1 \ldots x_n.x_i N_1 \ldots N_k)X_1 \ldots X_{i-1}\mathbf{U}^{\mathbf{k+1}}_{\mathbf{k+1}}X_{i+1} \ldots X_n$
where X_i are arbitrary terms; and a "canonical" trivial solution :

$(\lambda x_1 \ldots x_n.x_i N_1 \ldots N_k)\blacksquare_{(1)} \ldots \blacksquare_{(i-1)} \mathbf{U^{k+1}_{k+1}} \blacksquare_{(i+1)} \ldots \blacksquare_{(n)}$
Of course there might be other solutions of n arguments in individual cases, but in general not (as in $\lambda\vec{x}.x_i(\blacksquare\vec{x})$ for example).

As the Examples in 2.1.3 show, this proposition cannot be strengthened but it gives rise to a refined categorization of solvable terms.

2.2 Breakable terms

Definition 2.2.1. (Breakable terms)
A closed term M having head normal form $\lambda x_1 \ldots x_n.x_i\vec{N}$ is **breakable** iff $m(M) = n-1$ and **unbreakable** otherwise. A general term is breakable iff it has a closure which is breakable, otherwise it is unbreakable. The set of breakable terms is denoted by $\mathcal{B}$.

Remark 2.2.2. on the definition

i) In the light of Remark 2.1.13, another way to put this is as follows: a term is breakable if and only if it has a "less-than-trivial" solution (one consisting of less arguments than there are top-most abstractions in the term).

ii) Note that all breakable terms are solvable.

iii) Note that due to Proposition 2.1.4 $m(M)$ does not depend on the particular head normal form. From Proposition 2.1.4 it also follows that if $M = N$ then M is breakable if and only if N is breakable.

iv) From iii) and the fact that breakability is non-trivial, it follows by Rice's theorem (Theorem 1.2.10) that breakability is undecidable.

Example 2.2.3. Breakability of the standard combinators
i) $\mathbf{I}, \mathbf{S}, \mathbf{F}, \mathbf{B} \in \mathcal{B}$
ii) $\mathbf{K}, \mathbf{D} \in \mathcal{S} \setminus \mathcal{B}$
iii) $\mathbf{U^n_k} \in \mathcal{B} \iff n = k$
iv) $\blacksquare \notin \mathcal{B}$

Let us look at two other simple, but warning examples.

Example 2.2.4.
i) $\lambda xz.x(xz) \in \mathcal{B}$
ii) $\lambda xz.x(zx) \notin \mathcal{B}$

Proof
i) $(\lambda xz.x(xz))(\mathbf{I}) = \lambda z.\mathbf{I}(\mathbf{I}z) = \lambda z.z = \mathbf{I}$
ii)

$$\begin{aligned}
&\lambda xz.x(zx) \in \mathcal{B} \iff \\
&\exists P : \lambda z.P(zP) = \mathbf{I} \Longrightarrow \\
&\exists P \forall Q : P(QP) = Q \Longrightarrow \\
&(P(\mathbf{I}P) = \mathbf{I} \textit{ and } Px = P(\mathbf{K}xP) = \mathbf{K}x) \Longrightarrow \\
&\mathbf{I} = P(\mathbf{I}P) = \mathbf{K}(\mathbf{I}P) = \mathbf{K}P
\end{aligned}$$

which is a contradiction, because $m(\mathbf{K}) = 2$ (see Example 2.1.3). □

Proposition 2.2.5. In all of the following cases we allow $\vec{x}$, but not $\vec{N}$ to be empty:
i) $\lambda\vec{x}z.z$ is breakable
ii) $\lambda\vec{x}z.z\vec{N}$ is unbreakable
iii) $\lambda\vec{x}z.y\vec{N}$ is unbreakable if $y \neq z \notin FV(\vec{N})$

Proof
Cases i) and iii) are trivial, ii) follows from Lemma 2.1.11. □

Corollary 2.2.6. *Every fixed-point combinator* $\mathbf{Y}$ *is unbreakable.*

Proof
In the next Lemma we will show that every fixed point combinator reduces to a form $\lambda f.fY^*$ which proves our claim using Proposition 2.2.5 ii).
□

Lemma 2.2.7. $\forall F \in \Lambda : \mathbf{Y}F = F(\mathbf{Y}F) \Longrightarrow \mathbf{Y} \twoheadrightarrow \lambda f.fY^*$

Proof
$\mathbf{Y}$ is solvable, in fact $m(\mathbf{Y}) = 1$ as seen in Example 2.1.3, so it is of the form $\mathbf{Y} = \lambda x_1 \ldots x_n.yY_1 \ldots Y_k$.
We see, that in the above equation even $\twoheadrightarrow$ holds. Moreover $1 \leq n \leq 2$ by Proposition 2.1.12.
First we show, that $n = 1$ and $y \equiv x_1$.
We know by **CR** that $\mathbf{Y}F$ and $F(\mathbf{Y}F)$ have a common reduct X. Take F of order zero ($\blacksquare$ for example), that is $F \not\twoheadrightarrow \lambda x.F'$. Then $F(\mathbf{Y}F) \twoheadrightarrow X \Longrightarrow X \equiv FX'$, and so $\mathbf{Y}F \twoheadrightarrow FX'$, which is – being F of order zero – hereditarily an application term.
Now suppose $n = 2$. Then since $\mathbf{Y}F \twoheadrightarrow \lambda x_2 \ldots$, which is hereditarily an abstraction term, we would have by **CR** that it has a reduct which is an application term, which is a contradiction.
So $n = 1$ and $\mathbf{Y} \twoheadrightarrow \lambda x_1.yY_1 \ldots Y_k$. Suppose now, that $y \not\equiv x_1$. Then $\mathbf{Y}F = yY_1' \ldots Y_k' = F(\mathbf{Y}F)$ and again by **CR** we have a contradiction.
This proves, that $\mathbf{Y} \twoheadrightarrow \lambda f.fY_1 \ldots Y_k$. By a similar argument we will now show that $k = 1$.
We have that $\mathbf{Y}F = FY_1' \ldots Y_k' = F(\mathbf{Y}F)$ for all F. Now assuming that F is of order zero, so it cannot "eat" any of its arguments, we get that in every reduct of the second term F has k arguments while in every reduct of the third term it has one. Then by **CR** we get that $k = 1$.
□

We can further generalize this result as follows.

Theorem 2.2.8. $\forall \mathbf{Y}\, fixed\ point\ combinator : \mathbf{Y} \twoheadrightarrow \lambda f.f^k(Y^{(k)})$ *where* k *is an arbitrary natural number.*

Proof
We have seen by **CR** and the solvability of $\mathbf{Y}$ that $\mathbf{Y} \twoheadrightarrow \lambda f.fY^*$. Now denote Y^* by $Y^{(1)}$. Observing that $Y^{(k)}[f := F] = \mathbf{Y}F$ implies that $Y^{(k)}$ is solvable (take $F \equiv \mathbf{KI}$) and repeating the above argument using **CR** we can prove $Y^{(k)} \twoheadrightarrow fY^{(k+1)}$ and $Y^{(k+1)}[f := F] = \mathbf{Y}F$ again, hence we can prove the claim by induction for every $k \in \mathbb{N}$. □

Corollary 2.2.9.
All fixed point combinators have the same infinite Böhm tree [1] *determined by the following recursive formula:*

$$BT(\mathbf{Y}) = \lambda f.f(BT(\mathbf{Y}))$$

This means that all fixed point combinators can be consistently identified (see also [Bar84] theorem 19.3.4), because the Böhm tree model ([Bar84] section 18.3) is one such model in which they are represented by the same object.

[1] Note that $\mathbf{BD}(\mathbf{B}(\mathbf{BD})\mathbf{B})$ has the same Böhm tree but is not a fixed point combinator (see [Sta93])

In the remark following Definition 2.2.1, we noted that equal (i.e. β-convertible) terms are either both breakable or both unbreakable. We have just proven that all fixed point combinators are unbreakable, while our efforts have lead to proving that they all share the same Böhm tree (Corollary 2.2.6 and 2.2.9). A simple consequence of Proposition 18.3.4 of [Bar84] is that in general the following connection holds:

Proposition 2.2.10. Let M, N be arbitrary terms. Then
$BT(M) = BT(N) \Longrightarrow (M \in \mathcal{B} \iff N \in \mathcal{B})$.

Proof
$BT(M) = BT(N)$ implies by Proposition 18.3.4 of [Bar84] that

$$BT((\lambda\vec{x}.M)P_1 \dots P_k) = BT(M) \cdot BT(P_1) \cdot \dots \cdot BT(P_k) = BT((\lambda\vec{x}.N)P_1 \dots P_k)$$

for every $\vec{x}$ and $\vec{P}$, hence

$$\begin{aligned}
&(\lambda\vec{x}.M)\vec{P} = \lambda z.z \iff \\
&BT((\lambda\vec{x}.M)\vec{P}) = BT(\lambda z.z) = \lambda z.z \iff \\
&BT((\lambda\vec{x}.N)\vec{P}) = BT(\lambda z.z) = \lambda z.z \iff \\
&(\lambda\vec{x}.N)\vec{P} = \lambda z.z
\end{aligned}$$

and from $BT(M) = BT(N)$ we also know that any head normal forms of M and N have the same leading abstractions, and the result follows by definition. □

2.2.1 Breakability and the term formation rules

With respect to substitution, application and abstraction, the following closure properties are known to hold for solvable and unsolvable terms:

$$\begin{array}{lcl}
M[x := P] \textit{ is solvable} & \Longrightarrow & M \textit{ is solvable} \\
M \textit{ is unsolvable} & \Longrightarrow & \forall N \in \Lambda : MN \textit{ is unsolvable} \\
M \textit{ is solvable} & \iff & \forall x : \lambda x.M \textit{ is solvable}
\end{array}$$

Below we will investigate the relationship between breakability and the term formation rules obtaining similar results.

Proposition 2.2.11. $M[x := P]$ is breakable $\Longrightarrow M$ is breakable

Proof
The case $x \notin FV(M)$ is void, otherwise use Proposition 2.1.7. □

Examining the behavior of breakability in connection with application, we find the following discouraging examples.

Example 2.2.12. (Breakability and application)

I breakable,	**I** breakable	**II** $=$ **I** breakable
$\lambda xz.x\mathbf{K}z$ breakable,	**I** breakable	$(\lambda xz.x\mathbf{K}z)\mathbf{I} = \lambda z.\mathbf{K}z = \mathbf{K}$ unbreakable
F breakable,	**K** unbreakable	**FK** $=$ **I** breakable
I breakable,	**K** unbreakable	**IK** $=$ **K** unbreakable
K unbreakable,	**I** breakable	**KI** $=$ **F** breakable
KK unbreakable,	**I** breakable	**KKI** $=$ **K** unbreakable
D unbreakable,	**K** unbreakable	**DK** $=$ **KK** unbreakable

All terms in the above examples are closed. For the last case we only have an example with an open term:

$(\lambda x.y)$ unbreakable,	■ unbreakable	$(\lambda x.y)$■ $= y$ breakable

Conjectrure 2.2.13. *For all M, N closed unbreakable terms MN is unbreakable as well.*

Remark 2.2.14. If it were the case, as I conjecture, that the set $\mathcal{U}^0$ of closed unbreakable terms is closed under application, then $\mathcal{U}^0$ would be a set of terms closed under term formation rules (see Proposition 2.2.17). One could then ask to find a sufficiently simple or better yet a minimal generator set for $\mathcal{U}^0$ under the term formation rules. Of course $\mathcal{U}^0$ and hence all of its generator sets are not even recursively enumerable.

As hinted by the definition breakability is in some sense all about "keeping the inner-most abstraction" when solving a term. This is expressed by the next three propositions.

Proposition 2.2.15. Let $M \equiv yM_1 \ldots M_k$.
Then for any permutation $(i_1, \ldots, i_n)$ of $\{1, \ldots, n\}$:
$\lambda x_1 \ldots x_n z.M$ is breakable $\iff$ $\lambda x_{i_1} \ldots x_{i_n} z.M$ is breakable.

Proof
Since permutations can be inverted, it is sufficient to prove only one direction of implication, e.g. "$\Longrightarrow$".
Let us first note that $\lambda x_1 \ldots x_n z.M$ is closed $\iff$ $\lambda x_{i_1} \ldots x_{i_n} z.M$ is closed. According to this we consider two cases:
Let us assume first that $\lambda x_1 \ldots x_n z.M$ is closed. Then by Definition 2.2.1 it is breakable iff there are terms $N_1, \ldots, N_n$ such that $(\lambda x_1 \ldots x_n z.M)N_1 \ldots N_n = \mathbf{I}$. But now (assuming by the variable convention, that $x_1, \ldots, x_n \notin FV(N_1 \ldots N_n)$) $(\lambda x_{i_1} \ldots x_{i_n} z.M)N_{i_1} \ldots N_{i_n} = (\lambda x_1 \ldots x_n z.M)N_1 \ldots N_n = \mathbf{I}$, i.e. $(\lambda x_{i_1} \ldots x_{i_n} z.M)$ is breakable as well.
If the two terms are not closed, then by definition they are breakable iff they have a breakable closure. So let $\lambda \vec{y} x_1 \ldots x_n z.M$ be a breakable closure of $\lambda x_1 \ldots x_n z.M$. Then using the first case we conclude that $\lambda \vec{y} x_{i_1} \ldots x_{i_n} z.M$ and thus $\lambda x_{i_1} \ldots x_{i_n} z.M$ is breakable as well.
□

Remark 2.2.16. The inner-most variable z must not move, otherwise the proposition would not hold. For example: $K \equiv \lambda xy.x$ is unbreakable while $F \equiv \lambda yx.x$ is breakable.

Proposition 2.2.17. For any term $M \in \Lambda^0$:
M is breakable $\iff$ $\lambda x.M$ is breakable

Proof
By Definition 2.2.1 and Proposition 2.1.5. □

In case of abstraction terms the condition that M be closed can be dropped.

Proposition 2.2.18. For any term $M \in \Lambda$:
$\lambda y.M$ is breakable $\iff$ $\lambda xy.M$ is breakable

Proof
$\lambda y.M \in \mathcal{B} \iff$ (by definition)
$\exists \vec{u} : \lambda \vec{u} y.M \in \Lambda^0 \cap \mathcal{B} \iff$ (by Proposition 2.2.17)
$\exists \vec{u} : \lambda x \vec{u} y.M \in \Lambda^0 \cap \mathcal{B} \iff$ (by Proposition 2.2.15)
$\exists \vec{u} : \lambda \vec{u} xy.M \in \Lambda^0 \cap \mathcal{B} \iff$ (by definition)
$\lambda xy.M \in \mathcal{B}$.
□

2.2.2 Breaking out the inner-most variable

Solvable terms in general have a head normal form $\lambda \vec{x}.y\vec{N}$ where both $\vec{x}$ and $\vec{N}$ can be empty. We have seen that in case $\vec{x}$ is not empty, the breakability of the term depends only on the inner-most abstraction $\lambda x_n.y\vec{N}$. This observation leads to the following definition.

Definition 2.2.19.
We say that a term in hnf $\lambda\vec{x}.y\vec{N}$ is **in application head normal form (ahnf)** iff $\vec{x}$ is empty, i.e. there are no initial abstractions, i.e. it is of the form $y\vec{N}$.
An application term in ahnf $y\vec{N}$ is **breakable for (the variable)** z iff $\lambda z.y\vec{N}$ is breakable.
We denote the set of ahnf terms breakable for z by $\mathcal{B}_z$.

Example 2.2.20.
xy is breakable for y but not for x nor any other variable
$x(yx)$ is not breakable for any variable (Example 2.2.4 proves this for y)
xyz is breakable for y and z only

Using this new notation, we can summarize our results in the following theorem.

Theorem 2.2.21.

1. *An abstraction term M (in head normal form)*
   $M \equiv \lambda\vec{x}z.y\vec{N} \in \mathcal{B} \iff$
   $\lambda z.y\vec{N} \in \mathcal{B} \iff$
   $y\vec{N} \in \mathcal{B}_z \iff$
   $\exists\vec{P} : (y\vec{N})[\vec{y} := \vec{P}] \twoheadrightarrow z$ (*where* $\{\vec{y}\} = FV(y\vec{N}) - \{z\}$)

2. *An application term M (in head normal form)*
   $M \equiv y\vec{N} \in \mathcal{B} \iff$
   $\exists z \in Var : y\vec{N} \in \mathcal{B}_z \iff$
   $\exists z \in Var : \lambda z.y\vec{N} \in \mathcal{B}$

The examples in 2.2.4 illustrate that breakability is not a trivial notion at all. For example to "break" the term $\lambda xz.x(xxz)x$, we have to find an appropriate term X, such that $(\lambda xz.x(xxz)x)X = \lambda z.X(XXz)X = \lambda z.z$, i.e. $X(XXz)X = z$. This is very similar to the so called Böhm-out technique used by Böhm to prove the separability of normal forms. The Böhm-out technique is described in detail in [Bar84] section 10.3, here we will mention only one result (Proposition 10.3.7 in [Bar84]) informally: an instance of any subtree of the Böhm tree of a term can be obtained by an appropriate solving transformation (i.e. by a sequence of substitutions and applications of variables). When "breaking" a term we have to do something similar, for example to break the term $\lambda xz.x(xxz)x$ we have to sort of "Böhm out" the one occurrence of z appearing in the body of the term, but using more restricted transformations.

In the sequel we are going to look at some more examples, but first we make a few simple observations. The following definition is taken from [Bar84] 10.3.5.

Definition 2.2.22. A hnf $M \equiv \lambda\vec{x}.y\vec{N}$ is called **head original** if $y \notin FV(\vec{N})$.

The advantage of a hnf being head original is that we can freely substitute any term in its head variable, that is without consequences on any of its other subterms. This is expressed by the following proposition.

Proposition 2.2.23. If an ahnf $M \equiv yN_1 \ldots N_n$ is head original and $\exists i : N_i \in \mathcal{B}_z$, then $M \in \mathcal{B}_z$.

This proposition follows from the next stronger statement.

Proposition 2.2.24. Let $M \equiv yN_1 \ldots N_n$. If $\exists i : (N_i \in \mathcal{B}_z \wedge y \notin FV(N_i))$, then $M \in \mathcal{B}_z$.

Note that for $M \equiv yN_1 \ldots N_n$ to be breakable for z it is not necessary nor sufficient that $\exists i : N_i \in \mathcal{B}_z$. In fact, the above proposition can be strengthen as follows:

Proposition 2.2.25. Let $M \equiv yN_1 \ldots N_n$. If $\exists i : (N_i = \lambda\vec{x}.N_i' \wedge N_i' \in \mathcal{B}_z \wedge y \notin FV(N_i))$, then $M \in \mathcal{B}_z$.

Proof
Let $FV(N'_i) \setminus \{z, \vec{x}\} = \{u_1, \ldots, u_k\}$ and $FV(N_1 \ldots N_n) \setminus FV(N'_i) = \{v_1, \ldots, v_l\}$. Then by assumption, there are terms $\vec{X}, \vec{U}$ (we can also assume that they are closed) such that $N'_i[\vec{u} := \vec{U}][\vec{x} := \vec{X}] = (N_i[\vec{u} := \vec{U}])\vec{X} = z$. Let $V_1, \ldots, V_l$ be arbitrary terms and $Y \equiv \lambda w_1 \ldots w_n.w_i X_1 \ldots X_m$. Then $M[\vec{u} := \vec{U}][\vec{v} := \vec{V}][y := Y] = YN'_1 \ldots N'_{i-1}(N_i[\vec{u} := \vec{U}])N'_{i+1} \ldots N'_n = (N_i[\vec{u} := \vec{U}])\vec{X} = z$, and so $M \in \mathcal{B}_z$. □

Head originality is of course a very strong condition. To break terms which are not head original, we need more sophisticated techniques. Let us see some more intricate examples.

Example 2.2.26.
i) $\forall \Delta \in \Lambda^0 : \lambda xz.x\Delta(xz), \lambda xz.x(xz)(x\Delta), \lambda xz.x(xxz)\Delta \in \mathcal{B}$
ii) $\lambda xz.x(xx)(zx), \lambda xz.x(zz), \lambda xz.x(xx)(zz), \lambda xz.x(xz\Delta) \notin \mathcal{B}$

Proof
i) In the following $[,]$ denotes pairing, i.e. $[U,V] = \lambda f.fUV$
Let $X = \lambda v.[U,v]$, and $U = \mathbf{U}^3_2$. Then
$(\lambda xz.x\Delta(xz))X = \lambda z.X\Delta'(Xz) = \lambda z.[U,\Delta'][U,z] =$
$\lambda z.[U,z]U\Delta' = \lambda z.UUz\Delta' = \lambda z.z$
Let $X = \lambda v.[U,v]$, and $U = \lambda pqr.rF$. Then
$(\lambda xz.x(xz)(x\Delta))X = \lambda z.X(Xz)(X\Delta') = \lambda z.X[U,z][U,\Delta'] =$
$\lambda z.[U,[U,z]][U\Delta'] = \lambda z.[U,\Delta']U[U,z] = \lambda z.UU\Delta'[U,z] = \lambda z.z$
Let $X = \lambda uv.u[U,v]$, and $U = \mathbf{U}^3_3$. Then
$(\lambda xz.x(xxz)\Delta)X = \lambda z.X(XXz)\Delta' = \lambda z.X(X[U,z])\Delta' = \lambda z.X[U,z][U,\Delta'] =$
$\lambda z.[U,z][U,[U,\Delta']] = \lambda z.[U,[U,\Delta']]Uz = \lambda z.UU[U,\Delta']z = \lambda z.z$
ii)

$$\begin{aligned}
&\lambda xz.x(xx)(zx) \in \mathcal{B} \iff \\
&\exists P \forall Q : P(PP)(QP) = Q \Longrightarrow \\
&\mathbf{I} = P(PP)(\mathbf{I}P) = P(PP)(\mathbf{K}PP) = \mathbf{K}P, \quad \textit{contradiction}
\end{aligned}$$

$$\begin{aligned}
&\lambda xz.x(zz) \in \mathcal{B} \iff \\
&\exists P \forall Q : P(QQ) = Q \Longrightarrow \\
&\mathbf{I} = P(\mathbf{II}) = P(\mathbf{KI}(\mathbf{KI})) = \mathbf{KI}, \quad \textit{contradiction}
\end{aligned}$$

$$\begin{aligned}
&\lambda xz.x(xx)(zz) \in \mathcal{B} \iff \\
&\exists P \forall Q : P(PP)(QQ) = Q \Longrightarrow \\
&\mathbf{I} = P(PP)(\mathbf{II}) = P(PP)(\mathbf{KI}(\mathbf{KI})) = \mathbf{KI}, \quad \textit{contradiction}
\end{aligned}$$

□

Example 2.2.27. $x(xz\Delta) \notin \mathcal{B}_z$

Proof
Suppose that $M \equiv x(xz\Delta)$ is breakable for z by $[x := X]$. Then X is solvable, i.e. X has a hnf $\lambda u_1 \ldots u_n.v\vec{N}$, which has to be head closed, meaning $n > 0$. It is easy to see, that if $n > 1$ then $M[x := X] \equiv \lambda u_2 \ldots u_n.M' \neq z$. We have proved that $X = \lambda u.u\vec{N}$. Now substituting it into M we get $M[x := X] \equiv (\lambda u.u\vec{N})((\lambda u.u\vec{N})z\Delta') = ((\lambda u.u\vec{N})z\Delta')\vec{N} = z\vec{N}\Delta'\vec{N} \neq z$ by Lemma 2.1.11, contradiction. □

2.2.3 Towards generalizing breakability

So far we have been examining breakability through syntactical properties of terms. But breakability is no less a semantical notion as solvability. As suggested by the previous examples, there is a correspondence between the breakability of a given ahnf for a given variable and a particular instance of a class of problems. Another way to look at breakability is the following:

$$
\begin{array}{l}
M(\equiv y\vec{N}) \in \mathcal{B}_z \iff \\
\exists \vec{x},\vec{P}\ (|\vec{x}| = |\vec{P}|) : (\lambda\vec{x}z.M)\vec{P} = \mathbf{I} \iff \\
\exists \vec{x},\vec{P} : (\lambda z.(M[\vec{x} := \vec{P}])) = \mathbf{I} \iff \\
\exists M^*\ instance\ of\ M : (\lambda z.M^*)z = z \iff \\
\exists \lambda z.M^*\ instance\ of\ \lambda z.M : (\lambda z.M^*)z = z \iff \\
\exists \lambda z.M^*\ instance\ of\ \lambda z.M : z\ is\ a\ fixed\ point\ of\ \lambda z.M^*
\end{array}
$$

There are different ways of generalizing this, giving the following classes of problems (below M^* always denotes an instance of M, i.e. $M^* = M[x := N]$ for some x and N):

- given Q find an M s.t. $MQ = Q$
  (trivial: take $M = \mathbf{I}$; also note that if M is closed, $MQ = Q$ then $\forall Q^* : MQ^* = Q^*$)
- given Q find an M s.t. $MP = P \iff \exists Q^* = P$
- given Q and M find an instance M^* s.t. $M^*Q = Q$
- given Q and M find M^* s.t. $M^*P = P \iff \exists Q^* = P$

Note that as viewed above, breakability is a special case of the last problem class, with $Q \equiv z$.

We are not going to address these problems in this paper, just mentioned them as interesting questions for the curious mind.

Chapter 3

Conclusion

3.1 Summary

In this paper we have looked at one possible refinement of solvability in λ-calculus arriving at the notion of breakability.

- We have proven basic results in connection with the term formation rules similar to those known for solvability.
- We have presented several examples to illustrate the difficulties of "breaking" a term.
- We showed that no universal method exists to break every breakable term, as breakability is undecidable.
- The major open question remaining: is there a syntactic equivalent of breakability similar to Wadsworth theorem equating solvable terms and terms having a head normal form? In other words is there recursive enumeration of breakable terms similar to head reduction? This question is still to be answered.

Bibliography

[AKKSV94] Zena Ariola, Richard Kennaway, Jan Willem Klop, Ronan Sleep and Fer-Jan de Vries: *Syntactic definitions of undefined: On defining the undefined*, Lecture Notes in Computer Science, volume 789, pages 543-554, Springer-Verlag, Berlin 1994.

[Bar71] H.P. Barendregt: *Some Extensional Term Models for Combinatory Logics and λ-calculi*, PhD thesis, Rijksuniversiteit te Utrecht, 1971.

[Bar84] H.P. Barendregt: *The Lambda Calculus — Its Syntax and Semantics — Revised Edition*, Studies in Logic and the Foundations of Mathematics, Volume 103, North-Holland, 1984.

[Bar92] Henk Barendregt: *Representing 'undefined' in lambda calculus*, J. Functional Programming **2** (3): 367-374, July 1992.

[HS86] Hindley, J.R. and Seldin, J.P.: *Introduction to Combinators and the Lambda Calculus*, Cambridge University Press, Cambridge, 1986.

[KKSV95] Richard Kennaway, Jan Willem Klop, Ronan Sleep and Fer-Jan de Vries: *Transfinite Reductions in Orthogonal Term Rewriting Systems*, Information and Computation, volume 119, pages 18-38, Academic Press, 1995.

[Ket02] Jeroen Ketema: *Indiscernability versus Transitivity*, unpublished note, Vrije Universiteit Amsterdam, 2002.

[Kup94] Jan Kuper: *Partiality in Logic and Computation — Aspects of Undefinedness*, PhD thesis, Universiteit Twente, 1994.

[Kup95] Jan Kuper: *Usability: formalising (un)definedness in typed lambda calculus*, in: Leszek Pacholski, Jerzy Tiuryn (editors) – Computer Science Logic, LNCS 933, Springer-Verlag, 1995.

[Kup97] Jan Kuper: *On the Jacopini Technique*, Information and Computation, vol. 138, pp. 101-123, 1997.

[KOV99] Richard Kennaway, Vincent van Oostrom, Fer-Jan de Vries: *Meaningless Terms in Rewriting*, The Journal of Functional and Logic Programming, February 1999.

[Rog67] Rogers, H.: *Theory of Recursive Functions and Effective Operations*, McGraw Hill, New York, 1967.

[Sta93] Rick Statman: *Some examples of non-existent combinators*, Theoretical Computer Science 121 (1993) 441-448.

[Ter02] Terese: *Term Rewriting Systems*, Cambridge University Press, 2002.