A Story of Meaningless Terms

MSc. Thesis

Vince Bárány

Supervisors:
prof. Jan Willem Klop,
dr. Roel de Vrijer

Division of Mathematics and Computer Science
Vrije Universiteit, Amsterdam

2002

Preface

This thesis represents the final work for the M.Sc. degree in Computer Science at the Vrije Universiteit, Amsterdam in the course of the international master's program of the Division of Mathematics and Computer Science.

Acknowledgments

I would like to thank the people who helped me in preparing this thesis.

First of all, I would like to thank my supervisors Jan Willem Klop and Roel de Vrijer for introducing me not only to the theory of rewriting, but to the extraordinary study of meaninglessness, and for their guidance and support along the way.

I especially want to thank Jan Kuper for reading a large part of my draft, for his conceptual questions, and for inviting me for a personal discussion, providing me with ideas and motivation for an important section of the final thesis.

I would also like to thank Jeroen Ketema for reading the section on λ-calculus with the perfect scrutiny, not overlooking any peculiar details. I especially thank my friends Mihály Petreczky and Clemens Grabmayer for always being available for discussions, and never giving up on me. I thank my sister Bori for correcting grammatical errors in the text. My thanks go to Femke van Raamsdonk and everyone at the Theoretical Computer Science group of VU for welcoming me at their weekly seminar, which has been a great source of inspiration to me.

I would like to express my love for the Vreeke family and my neighbors at "unit 96" for being my family in the past ten month in Amsterdam. Without their day-by-day support this work could not have been done.

Contents

Chapter 1

Introduction

1.1 Introduction

The study of meaningless terms dates back to the early years of Church's foundation of λ-calculus. First 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 over the Church numerals.

Right here we have to stop and point out that representing a partial function in λ-calculus is not as trivial as representing total functions, for the question immediately arises: which terms should represent "the" *undefined*? An obvious requirement is that these terms representing the undefined should not in any way interfere with the terms representing "real" values, that is the numerals during computations. In other words they should not have any *meaning* with respect to the numerals. The safest choice would then be to have "the" *meaningless* terms represent the undefined. So which are the meaningless terms?

Church and Kleene considered those terms without a normal form to be meaningless and this is the notion their definition of λ-definability of a partial recursive function incorporated. However, we have to note that they were using Church's original λI-calculus and that Kleene's clever construction for representing primitive recursion and minimalization did not incorporate fixed point combinators. Thus, in this scenario, their selection of terms without a normal form to represent the undefined was adequate. Fixed point combinators on the other hand are good examples of terms without a normal form which are extremely useful tools in recursive constructions, and the advantages of the λK-calculus (in this thesis usually just referred to as λ-calculus) are also too tempting to be ignored.

Unsolvable terms and terms without *head normal forms* were independently proposed to be considered meaningless and to represent the undefined in the λK-calculus by Barendregt ([Bar71]) and Wadsworth, respectively. Wadsworth later showed that the two notions are equivalent in the λK-calculus, and Barendregt proved that in the λI-calculus unsolvability is equivalent to having no normal form.

Many different numeral systems have since been shown to be adequate (see [Bar84] section 6.4 and exercises 6.8.19-6.8.22), together with alternative representations of function composition, primitive recursion and minimalization (such as the idea of using a fixed point combinator due to Turing). Also many other notions of meaninglessness have been proposed to represent the undefined, such as *easy terms*, *terms of order zero*, terms with no *weak head normal form*, etc. A very general result of Statman proves that any of these classes of terms can be used to represent the undefined in an appropriate representation of partial recursive functions (see [Bar92]).

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. In section 2.1 of this thesis we will investigate the role of solvability in the latter.

Aside of their adequacy for representing the undefined, the unsolvable terms have some other valuable properties. Famous results include the *genericity lemma*, the consistent identifiability of all unsolvable terms, the fact that this is a maximal such set, and the separability of distinct normal forms proved by Böhm. Solvability is a semantic notion but as we mentioned above, it has a syntactical equivalent: the property of having a head normal form.

All these notions can be generalized to typed λ-calculi, to $\lambda\delta$-calculi, and also to term rewriting systems (TRS), both in the finitary and infinitary setting.

Joint efforts have been made by Ariola, Kennaway, Klop, van Oostrom, Sleep and de Vries to characterize meaninglessness and to establish axioms for a set of terms to possess some of the above mentioned desired properties of the unsolvables. Most of their axioms are truly elementary or intuitive, rather easily verifiable as they illustrate on examples such as the set of easy terms or zero terms. They generalize the notion of genericity, and *Böhm tree*, and prove that this genericity, as well as the existence of a Böhm tree model, and the consistent identifiability of a set of terms follow from an appropriate subset of their axioms.

In his thesis [Kup94] Kuper takes a different approach. In the first part of his thesis he devises a first order partial logic theory equipped with Hilbert's ε-operator, Kleene-equality and test for existence. In the framework of this logic he then defines an applicative theory of partial functions with λ-abstraction in which α-conversion, β-reduction and ξ-rule of λ-calculus can be formalized and proven as theorems, while the η-rule of extensionality can be added as an axiom. An equivalent formalization of the λ-calculus is then obtained by a restriction to the set of so called "pure" terms. In the second part of his thesis, Kuper examines various normal forms, *strictness* (an important and useful concept which naturally appears in partial logic), solvability, and easiness in the framework of a simply typed $\lambda\delta$-calculus armed with an explicit recursion abstraction, named $\lambda\mu E$-calculus. His investigations lead to the definition of a *fourth normal form* and *usability* as a generalization of solvability, and to proofs of the genericity and the consistent identifiability of the *unusable* terms. In [KOV99] the authors derive these results after showing that the set of unusable terms satisfies their axioms.

In this thesis we are going to investigate all these concepts, with a special emphasis on solvability, in accordance with its importance, looking at all the above mentioned approaches to generalizations and how they are related. These investigations lead us to the definition of *breakable* terms (an interesting subclass of solvable terms) and some results about strictness in various frameworks, a generalized theory of usability and a connection between certain axioms of [KOV99] in relation with usability.

1.2 Preliminaries

We assume the reader is familiar with λ-calculi and term rewriting systems (TRS) 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.

1.2.1 λ-calculi

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 $M = N \wedge C[] = D[] \implies C[M] = D[N]$. 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 (**GL**))*
In the λK-calculus: $\forall M \in \mathcal{U} \forall$ context $C[] : C[M] = L \in \mathcal{NF} \implies \forall N \in \Lambda : C[N] = L$.

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

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

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 $\perp$-rule as

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

Terms of this extended transfinite system are called Böhm terms, the reduction (by $\alpha\beta(\eta)$ and $\perp$ 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 $\mathbf{\Omega}$ 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

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

1.2.2 Term Rewriting Systems

Term rewriting systems, TRSs, are a special kind of abstract reduction systems. An abstract reduction system, ARS, is a pair $(A, \to)$ where A is a nonempty set and $\to$ is just a binary relation over A. For a TRS this pair is defined as follows:

Definition 1.2.22. (Term Rewriting Systems)
A **TRS** is a pair (Σ, R), where Σ is a signature, and R is a binary relation over $\mathfrak{T} = \mathfrak{T}(\Sigma)$, the set of all terms over Σ, such that for every **reduction rule** $(l, r) \in R$:

- l is not a variable, and
- $FV(r) \subseteq FV(l)$

The ARS associated with a TRS is

$$(\,A, \to\,) = (\,\mathfrak{T}(\Sigma), \{(C[\sigma(l)], C[\sigma(r)]) \mid (l, r) \in R, \sigma : \textit{ substitution}\}\,)$$

that is $\to$ is the closure of R under substitutions and contexts, where a substitution is a map $\mathfrak{T}(\Sigma) \longrightarrow \mathfrak{T}(\Sigma)$ compatible with the term formation rules and as such is defined by its restriction to the set of variables.

Definition 1.2.23. (Left-linear TRSs)
A TRS is **left-linear** if no variable occurs twice inside the left hand side of any of its reduction rules.

Left-linearity is the minimal requirement of a TRS when talking about meaninglessness. In [KOV99] the authors argue that in a non left-linear TRS intuitively every term should be considered meaningful, as it can contribute to a computation in which a non left-linear, i.e. matching rule is applied. Matching rules require syntactic matching of subterms, and with such a powerful "hardware" every term is transparent to the system. For this reason we will always implicitly assume that our TRSs are left-linear.

Probably the most important property of abstract reduction systems is confluence, or Church-Rosser property stating that any two diverging reduction sequences starting from the same object can be rejoined. Formally:

Definition 1.2.24. (Confluence and Church-Rosser property)

Confluence : $\forall s, t, t' \in \mathfrak{T} : t' \twoheadleftarrow s \twoheadrightarrow t \Longrightarrow \exists r \in \mathfrak{T} : t' \twoheadrightarrow r \twoheadleftarrow t.$

CR : $\forall t, s \in \mathfrak{T} : s = t \Longrightarrow \exists r \in \mathfrak{T} : s \twoheadrightarrow r \twoheadleftarrow t.$

It is easy to see that confluence and **CR** are equivalent. We have already stated the theorem of Church and Rosser proving confluence of λ-calculus. Various different proofs of their theorem have since been given, all building on a much more general property of λ-calculus as a reduction system, namely orthogonality of its reduction rules.

Definition 1.2.25. (Overlapping redexes)
Let (Σ, R) be a TRS, and $(l, r), (l', r') \in R$ two (not necessarily different) reduction rules. If $\Delta \equiv \sigma(l)$ is a redex by the first rule and $\Delta' \equiv \sigma'(l')$ is a redex by the second rule such that Δ' is a proper subterm of Δ at position π, then the two redexes **overlap** if π is also a position of l and $l|\pi$ is not a variable.

Definition 1.2.26. (Orthogonal TRSs)
A TRS is **orthogonal** if it is left-linear and non-ambiguous, i.e. has no overlapping redexes.

Intuitively in an orthogonal TRS, reduction steps performed on a term do not interfere with one another, the reduction of one redex does not destroy other redexes (it may erase them of course). This motivates the following well known result.

Theorem 1.2.27. *Orthogonal TRSs are confluent*

Contexts are defined in the same way as in λ-calculus, and there is no difference in notation either. There is one important difference though that we have to point out which is that since variable binding is not supported in TRSs, we do not have the problems as we did in λ-calculus that arise from the fact that the [] symbol could get into the scope of abstractions.

Böhm trees can be defined analogously as well under certain assumptions. This will be studied in section 3.2.

1.3 Notations

Sets

$\mathbb{N}$	$\{\ 0,\ 1,\ 2, \ldots\}$
$\mathfrak{T}$	the set of all terms of a TRS
$\mathfrak{C}$	the set of all contexts
$\mathcal{C}^{\mathcal{A}}_{\mathcal{B}}$	$\{C[] \in \mathfrak{C} \mid \forall t \in \mathcal{A} : C[t] \in \mathcal{B}\}$
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 2.2.18)
$\overline{\mathcal{A}}$	denotes the complement (depending on context) of the set $\mathcal{A}$
$\mathcal{A}^{\rightarrow}$	closure of $\mathcal{A}$ under rewriting
$\mathcal{A}^{\leftarrow}$	closure of $\mathcal{A}$ under $\leftarrow$
$\mathcal{A}^{=}$	closure of $\mathcal{A}$ under conversion
$\mathcal{TM}(\mathcal{U})$	totally meaningful terms w.r.t $\mathcal{U}$ (Definition 3.1.13)
$\mathcal{M}^{\leftarrow}(\mathcal{U})$	$\mathcal{TM}(\mathcal{U})^{\leftarrow}$

Terms

$M\vec{N}$	is a shorthand notation for $MN_1 \ldots N_n$ ($n \geq 1$ is assumed unless explicitely stated otherwise)
$\lambda\vec{x}.M$	is a shorthand notation for $\lambda x_1 \ldots x_n.M$ (again $n \geq 1$ is assumed unless stated otherwise)
$F^k M$	$F^0M \equiv M; F^{k+1}M \equiv F(F^kM)$ (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	$\lambda xy.y \equiv \lambda yx.x$
$\mathbf{U^n_k}$	$\lambda x_1 \ldots x_n.x_k$
Y	some fixed point combinator
D	$\lambda x.xx$
B	$\lambda xyz.x(yz)$
$\mathbf{\Omega}$	**DD**

Properties

CR	the Church-Rosser property, i.e. confluence
NF	the normal forms property, i.e. $M = N, N \in \mathcal{NF} \Longrightarrow M \twoheadrightarrow N$
$\mathcal{A} \vDash R$	$\mathcal{A}$ is closed under R (e.g. $\mathcal{WN} \vDash \leftarrow$, $\mathcal{TM}(\mathcal{U}) \vDash \rightarrow$)
$R \vDash \pi$	the relation R satisfies the property π (e.g. $\rightarrow_\beta \vDash$ **CR**)
$R \vDash []$	R is 'closed under contexts': $M\,R\,N \Longrightarrow \forall C[] : C[M]\,R\,C[N]$
$R \vDash [\ldots]$	R is 'closed under multi-hole contexts', i.e. $\vec{M}\,R\,\vec{N} \Longrightarrow \forall C[] : C[\vec{M}]\,R\,C[\vec{N}]$

Chapter 2

λ-calculi

2.1 The λI-calculus

2.1.1 An analysis of Kleene's representation

Introduction

As mentioned in the introduction, Kleene's original representation of the partial recursive functions was formulated in the λI-calculus without the use of a fixed point combinator and representing the undefined via terms without a normal form. Below we will analyze this construction focusing on the representation of minimalization and conclude that it essentially uses the (uniform) solvability of the representing terms.

Representation in the λI-calculus

Below we follow the presentation found in [Bar84] section 9.2 with minor adjustments for our ease of use. We begin by defining the numerals.

Definition 2.1.1.
$\ulcorner 0 \urcorner \equiv \lambda xy.x\mathbf{II}y$
$\ulcorner n+1 \urcorner \equiv \lambda xy.x^{n+1}y$

Note that we are using the Church numerals with the exception of $\ulcorner 0 \urcorner$ which had to be adapted to the restriction of λI-calculus. (Remember that in the λI-calculus abstraction from a term M by a variable x is only allowed if x is free in M. So for example $\lambda xy.y$ (the original Church numeral for zero) is not a valid term in λI-calculus.)

Right here in the definition of $\ulcorner 0 \urcorner$ we can observe how solvability plays a silent role in the construction. We know that all the Church numerals are in normal form, are closed, and have only two bound variables, consequently they can be uniformly solved by two arguments of $\mathbf{I}$. The definition of $\ulcorner 0 \urcorner$ exploits this property to create a term that ignores its first argument if it can be solved by two arguments of $\mathbf{I}$.

This observation gives the basic idea and this "trick" is the heart of the whole construction. Conditionals can only be represented by terms which ignore some of their arguments. In the λK-calculus this can be simply expressed with the terms $\mathbf{T} \equiv \lambda xy.x$ and $\mathbf{F} \equiv \lambda xy.y$. In the λI-calculus however, this is only locally possible, that is with respect to a set of uniformly solvable terms.

Definition 2.1.2. $\mathcal{S}_k = \{M \mid M\mathbf{I}^{\sim k} = \mathbf{I}\}$

The boolean values, for example, have the following representations for all $k \in \mathbb{N}$:

Definition 2.1.3. The boolean values local to $\mathcal{S}_k$ are
$\mathbf{T}_k \equiv \lambda xy.y\mathbf{I}^{\sim k}x$
$\mathbf{F}_k \equiv \lambda xy.x\mathbf{I}^{\sim k}y$
$\mathbf{T}_I \equiv \mathbf{T}_2$
$\mathbf{F}_I \equiv \mathbf{F}_2$

Again, all of the above terms are in $\mathcal{S}_2$, so we can represent conditionals (locally to the same sets) as follows:

Definition 2.1.4. Representing conditionals on terms of $\mathcal{S}_k$
$\mathbf{R}_k \equiv \lambda xyz.x\mathbf{T}_k\mathbf{F}_k yz$

One can easily check that if $M, N \in \mathcal{S}_k$ then $\mathbf{R}_k\mathbf{T}_I MN = M$ and $\mathbf{R}_k\mathbf{F}_I MN = N$.

We are now ready to define the representation of minimalization. We need a term μ_I such that $\mu_I P = \ulcorner n \urcorner$ if n is the smallest natural number such that $P\ulcorner n \urcorner = \mathbf{T}_I$ for every P which only takes the values $\mathbf{T}_I$ and $\mathbf{F}_I$ on the set of numerals.

The key to defining μ_I is the construction of a recursor term H such that $HP\ulcorner n \urcorner = \ulcorner n \urcorner$ if $P\ulcorner n \urcorner = \mathbf{T}_I$ and $HP\ulcorner n \urcorner = HP\ulcorner n+1 \urcorner$ if $P\ulcorner n \urcorner = \mathbf{F}_I$. μ_I is then defined as follows:

$$\begin{array}{rl} \mu_I & \equiv \lambda p.Hp\ulcorner 0 \urcorner \\ H & \equiv \lambda pn.W(pn)nWp \\ W & \equiv \lambda t.\mathbf{R}_k tAB \\ A & \equiv \lambda nwp.w\mathbf{T}_I\mathbf{I}^{\sim k}(pn)\mathbf{I}^{\sim 2}n \\ B & \equiv \lambda nwp.w(p(S_I^+ n))(S_I^+ n)wp \end{array}$$

Where S_I^+ denotes a representation of the successor function (we have omitted its definition here) such that $S_I^+\mathbf{I} \in \mathcal{S}_2$ and consequently $S_I^+ \in \mathcal{S}_3$. Thus $B\ulcorner n \urcorner WP = W(P\ulcorner n+1 \urcorner)\ulcorner n+1 \urcorner WP = HP\ulcorner n+1 \urcorner$ and $B \in \mathcal{S}_3$. In order to achieve $A\ulcorner n \urcorner WP = \ulcorner n \urcorner$ we have to neutralize the arguments W and P. Note that to neutralize (i.e. solve) P we need to assume something about P. Usually this assumption would be solvability by certain fixed arguments, but then our construction would not be universal. Fortunately we do have a weaker, but sufficient assumption, namely that $P\ulcorner n \urcorner$ is one of $\mathbf{T}_I$ or $\mathbf{F}_I$ and as such falls inside $\mathcal{S}_2$, meaning $P\ulcorner n \urcorner\mathbf{II} = \mathbf{I}$ for all $n \in \mathbb{N}$. It is interesting how the solvability of A and W are related. W is the branching point of the construction, and as we have noted above it can be expressed using $\mathbf{R}_k$ if the two branches, that is A and B are members of $\mathcal{S}_k$. So if A is in $\mathcal{S}_k \quad (k \geq 3)$ then we can define W as above and conclude that $W\mathbf{T}_I \in \mathcal{S}_k$. Consequently we can also define A as above and conclude that it is in fact a member of $\mathcal{S}_3$. So we can actually let $k = 3$ in the above definition.

Conclusion. The role of solvability

As we can see Kleene's construction is actually built on (uniform) solvability of certain terms used in the construction. It is correct because those terms representing the undefined, i.e. terms without a normal form are not solvable at all. In fact, as proven by Barendregt (see e.g. [Bar84] 9.4.21) the solvable terms are exactly the ones having a normal form (in the λI-calculus), so the hereditary property of unsolvable terms means that terms without a normal form (i.e. terms representing the undefined) will not produce a normal form (i.e. something meaningful in this representation) even when applied to some arguments.

The conclusion is that (uniform) solvability is a crucial property in the λI-calculus, for *operators* represented by terms can not ignore any of their arguments, the best they can do is to *neutralize* them. This of course can only be done if some information about the arguments is known and can be built into the term representation. This also means that this "trick" can only work on a set of terms for which the same information is appropriate.

This information in our case is the solvability by a certain sequence of arguments, and the set to which it applies is the set of terms uniformly solvable by these arguments. Formally, let

$$\mathcal{S}^{\vec{M}} = \{X \mid X\vec{M} = \mathbf{I}\}$$

for every sequence $\vec{M}$ of terms. Then if $O[x_1, \ldots, x_n]$ is an "erasing" operator, i.e. there is an operator R such that

$$O[X_1, \ldots, X_n] = R[X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n]$$

for every $X_j \in \Lambda (i \neq j)$ and $X_i \in \mathcal{S}^{\vec{M}}$ then O can be λ-*defined locally to* $\Lambda^{i-1} \times \mathcal{S}^{\vec{M}} \times \Lambda^{n-i}$ (provided that R is λ-defined) by neutralizing the ith argument as follows:

$$O \equiv \lambda x_1 \ldots x_n.(x_i \vec{M}) R[x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n]$$

On the universality of solvability

As we have pointed out, solvability was exploited in a clever trick in Kleene's representation to λ-define conditionals locally to a set of terms uniformly solvable by some arguments. Below we will argue that this is not just an ad hoc technique, in fact solvability is seems to be the only tool that we have to λ-define erasing operators, and even then only locally to a set of terms satisfying a solvability condition.

First we have to note that solvability by certain arguments is not the only possible assumption one could build on when constructing a term representation of an erasing operator. In general, assuming that the argument to be ignored will yield a fixed term when applied to a fixed sequence of parameters is just as useful. Formally this would mean information about membership in

$$\mathcal{S}^{\vec{M};N} = \{X \mid X\vec{M} = N\}$$

It is obvious however, that $X\vec{M} = \mathbf{I} \Longrightarrow X\vec{M}N = N$, hence $\mathcal{S}^{\vec{M}} \subseteq \mathcal{S}^{\vec{M}N;N}$, and conversely if $N\vec{P} = \mathbf{I}$ (i.e. $N \in \mathcal{S}^{\vec{P}}$) then $\mathcal{S}^{\vec{M};N} \subseteq \mathcal{S}^{\vec{M}\vec{P}}$. This can be informally stated as the assumption of solvability by given arguments is in this sense universal among these generalized assumptions of solvability for a given term by given arguments.

But as we have seen in the formulation of A above, the assumption of much more intricate solvability conditions (such as information about the solvability of a certain combination of the arguments) can also be sufficient in some cases. For instance, by assuming that two arguments neutralize one another we could define the two projections as follows: $\lambda xy.yxx$ and $\lambda xy.xyy$, where arguments X, Y are expected to satisfy $XY = YX = \mathbf{I}$.

Definition 2.1.5. (Local representation)
Let $\mathcal{A} \subseteq \Lambda_I^k$ be a set of k-tuples of λI-terms.
We say that a λK-term $M \in \Lambda_K$ **can be λI-defined locally to $\mathcal{A}$** if

$$\exists M_{\mathcal{A}} \in \Lambda_I : \forall \vec{X} \in \mathcal{A} : M\vec{X} = M_{\mathcal{A}}\vec{X}$$

In this case we say that $M_{\mathcal{A}}$ **locally λI-defines** M **over** $\mathcal{A}$, ($M_{\mathcal{A}} \propto_{\mathcal{A}} M$ for short) and we let $\Lambda_{\mathcal{A}}$ denote the set of such M which can be locally defined over $\mathcal{A}$.

Example 2.1.6.

$\lambda xy.y\mathbf{II}x$	locally defines	$\mathbf{K} \equiv \lambda xy.x$	over $\{(X,Y) \mid Y\mathbf{II} = \mathbf{I}\} = \Lambda \times \mathcal{S}_2$
$\lambda xy.yxx$	locally defines	$\mathbf{K}$	over $\{(X,Y) \mid YX = \mathbf{I}\}$
$\lambda xy.xxy$	locally defines	$\mathbf{F} \equiv \lambda x.\mathbf{I}$	over $\{(X,Y) \mid XX = \mathbf{I}\}$ and
		$\lambda xy.\mathbf{I}$	over $\{(X,Y) \mid XXY = \mathbf{I}\}$

In a most general form sufficient assumptions about the (mutual) neutralization of n arguments can be expressed by a term A with $FV(A) = x_1, \ldots, x_n$ and the equation $A[\vec{x} := \vec{X}] = \mathbf{I}$ for all expected vectors $\vec{X}$ of arguments. Let $A' \equiv \lambda\vec{x}.A$. Then if N is a λI-term, then any λK-term of the form $M_K \equiv \lambda\vec{x}.N$ can be translated to a λI-term $M_I \equiv \lambda\vec{x}.(A'\vec{x})N$ behaving the same way as M_K on the set

$$\mathcal{S}_{A'} = \{\vec{X} \mid A'\vec{X} = \mathbf{I}\}$$

Of course if N is not a λI-term , then it has to be locally defined as well over some appropriate set. Here we will not go into the investigation of the applicability of this method.

To show that uniform solvability can also be expressed in this latter form, just note that $X\vec{M} = \mathbf{I} \iff (\lambda x.x\vec{M})X = \mathbf{I}$, hence $\mathcal{S}^{\vec{M}} = \mathcal{S}_{(\lambda x.x\vec{M})}$.

It seems that this is the most general formalization of neutralizability conditions and that this is as close as we can get to translating λK-terms into λI-terms. Notice that this latter technique differs from that exploiting uniform solvability only in that it asserts an equation of the form $A'\vec{X} = \mathbf{I}$ (Xs appearing as arguments) instead of the system $X_i\vec{M} = \mathbf{I}$ ($i = 1, \ldots, n$) of equations (Xs in function position), but they still express solvability of some terms by others. For an elaboration of this idea, see Definition 2.2.46.

2.1.2 Strictness w.r.t. the unsolvables

In this section we will show that all λI-terms represent strict operations with respect to the set of unsolvable λI-terms. This also justifies the representation of the undefined by the unsolvables and the identification: meaningless $\iff$ unsolvable in the λI-calculus.

Lemma 2.1.7. (Application is strict in its left argument.)
$\forall Q \in \mathcal{U}_I : \forall M \in \Lambda_I : QM \in \mathcal{U}_I$

Proof
If QM was solvable, let's say $(\lambda\vec{x}.QM)\vec{P}\vec{R} = (Q[\vec{x} := \vec{P}])(M[\vec{x} := \vec{P}])\vec{R} = \mathbf{I}$ then Q would be solvable as well, because $(\lambda\vec{x}.Q)\vec{P}(M[\vec{x} := \vec{P}])\vec{R} = \mathbf{I}$. □

Lemma 2.1.8. (Abstraction is strict.)
$\forall Q \in \mathcal{U}_I : \forall x \in FV(Q) : \lambda x.Q \in \mathcal{U}_I$

Proof
If $\lambda x.Q$ is solvable, then by definition (1.2.13) $\exists\vec{y}, \vec{N}$ such that $(\lambda\vec{y}x.Q)\vec{N} = \mathbf{I}$, but then by definition again, Q is solvable as well, contradiction. □

Lemma 2.1.9. (Substitution is strict in its right argument.)
M is in nf, $x \in FV(M)$, Q unsolvable $\Longrightarrow M[x := Q]$ has no nf.

Proof
The proof is by induction on the complexity of M.
Base case: $M \equiv x \Longrightarrow M[x := Q] \equiv Q$, which is unsolvable, i.e. has no nf.
Induction: Since M is in nf it can be written as $\lambda\vec{y}.zM_1 \ldots M_n$, where $\forall i = 1, \ldots, n$: M_i is in nf. If $x = z$ then by Lemma 2.1.7 we get, that $M[x := Q] \equiv QM_1[x := Q] \ldots M_n[x := Q]$ is unsolvable, i.e. has no nf. On the other hand $x \neq z$ implies $\exists i : x \in FV(M_i)$. Since M_i is also in nf, we know by induction that $M_i[x := Q]$ has no nf, which means that $M[x := Q] \equiv \lambda\vec{y}.zM_1[x := Q] \ldots M_n[x := Q]$ cannot have a nf either. □

Theorem 2.1.10. *(Every λI-term is strict with respect to the unsolvables.)*
$\forall M \in \Lambda_I : \forall Q \in \mathcal{U}_I : MQ \in \mathcal{U}_I$

Proof
If M is unsolvable, then we can use Lemma 2.1.7, so we can assume that M is solvable, i.e. it has a nf. Moreover we can assume that M is in nf, since $M = M' \Longrightarrow MQ \in \mathcal{U}_I \iff M'Q \in \mathcal{U}_I$
M can be in one of two forms:

1. $M \equiv xM_1 \dots M_n \Longrightarrow MQ \equiv xM_1 \dots M_n Q$, and so every reduct of MQ includes a reduct of Q as a subterm, consequently MQ cannot have a nf.

2. $M \equiv \lambda x.M'$, where M' is in nf, $x \in FV(M')$ and $MQ = M'[x := Q]$. In this case MQ has no nf by Lemma 2.1.9.

□

Corollary 2.1.11. *(Every λI context is strict with respect to the unsolvables.)*
$\forall C[] \in \mathfrak{C}_I : \forall Q \in \mathcal{U}_I : C[Q] \in \mathcal{U}_I$

Proof
[] is obviously strict.
$C[]$ strict, $M \in \Lambda_I \Longrightarrow C[]M$ is strict by Lemma 2.1.7.
$C[]$ strict, $x \in FV(C[]) \Longrightarrow \lambda x.C[]$ is strict by Lemma 2.1.8.
$C[]$ strict, $M \in \Lambda_I \Longrightarrow MC[]$ is strict by Theorem 2.1.10.
Consequently all contexts are strict. □

This also explains why Kleene did not use a fixed point combinator. Since fixed point combinators have no nf, they cannot be used to formulate or compute terms which have a nf, e.g. Church numerals.

2.2 The λK-calculus

2.2.1 λ-definability, Statman's result

In the previous section we have looked at an example how to represent the partial recursive functions in the λI-calculus with the Church numerals representing natural numbers and terms without a normal form representing the undefined. We also showed how this representation used the fact that terms without a nf are exactly the unsolvables.

In the λK-calculus the two notions do not coincide anymore, but it still holds that all terms having a normal form are solvable. This ensures that Kleene's construction gives a valid representation of the partial recursive functions in the λK-calculus (just λ-calculus from now on) in which undefined is represented by terms without a normal form.

In [Bar71] section 1.3 (see also [Bar84] section 8.4) we find a different construction, using the unsolvable terms as representing the undefined.

As we have remarked in the introduction, given a representation of the natural numbers, the initial functions, composition, recursion and minimalization, the undefined has to be represented by terms which are invariably meaningless with respect to this representation, meaning that these constructions cannot yield (a representation of) a meaningful value (e.g. a natural number) from (a representation of) the undefined. This condition is referred to as *strictness*.

We can obtain a necessary condition of meaninglessness by turning this argument around and saying: members of a set $\mathcal{A}$ of terms can be considered meaningless if there is a representation of the partial recursive functions, such that terms belonging to $\mathcal{A}$ represent the undefined. The following "master" result of Statman (described in [Bar92]) proves that this is possible under a few, not too strong restrictions on $\mathcal{A}$.

Theorem 2.2.1. *(Statman)*
Let $\mathcal{A} \subseteq \Lambda^0$ be non empty, closed under β-conversion and such that $\Lambda^0 \setminus \mathcal{A}$ is recursively enumerable (r.e.). Then all partial recursive functions can be λ-defined w.r.t $\mathcal{A}$ as a set of undefined elements.

Of course we shall also require that $\Lambda^0 \setminus \mathcal{A}$ contains at least one adequate numeral system (one that allows a successor function and a test for zero, consequently every recursive function to be λ-defined).

2.2.2 Strictness

In Theorem 2.1.10 and Corollary 2.1.11 we proved that all λI-terms and λI-contexts are strict with respect to the unsolvables. This is not true in the λK-calculus, in fact we have the following negative result:

Theorem 2.2.2. *(Non-strict λK-terms.)*
Let $\mathcal{A} \subset \Lambda_K$ be non-trivial (i.e. $\mathcal{A} \neq \emptyset, \Lambda_K$) and closed under β-reduction. Then $\exists M \in \Lambda_K : \forall N \in \mathcal{A} : MN \notin \mathcal{A}$ (i.e. M is not (a bit) strict w.r.t $\mathcal{A}$).

Proof
Let $L \in \Lambda_K \setminus \mathcal{A}$ and $M \equiv \mathbf{K}L$ and $N \in \mathcal{A}$. Then $MN \equiv \mathbf{K}LN \to L \notin \mathcal{A}$, which implies that $MN \notin \mathcal{A}$ either, because $\mathcal{A}$ is closed under reduction. $\square$

Remark 2.2.3. Note that Lemma 2.1.7 and 2.1.8 still hold in the λK-calculus, i.e. abstraction is strict with respect to the unsolvables, and application is also strict in its left argument, while it is obviously not strict in its right argument, as shown by the example in the proof of the theorem.

Corollary 2.2.4. *(Non-strict λK-contexts.)*
Let $\mathcal{A} \subset \Lambda_K$ be non-trivial (i.e. $\mathcal{A} \neq \emptyset, \Lambda_K$) and closed under β-reduction. Then $\exists C[] \in \Lambda_K[] : \forall N \in \mathcal{A} : C[N] \notin \mathcal{A}$.

Proof
Just take $C[] \equiv M[]$, where M is as in Theorem 2.2.2. □

Since non-triviality and closure under reduction (see Axiom U1. in 3.1.9) are the simplest and most natural requirements of a set of terms to be considered as the set of (in some sense) meaningless terms, the previous theorem ensures that for any formalization of meaninglessness, strictness does not hold for every λK-term. However, for every notion of meaninglessness, we can speak of the set of terms or contexts which are strict with respect to the set of such meaningless terms.

Partial recursive functions are strict with respect to undefinedness, thus asking *'which terms (or contexts) are strict with respect to a given (candidate) set of (meaningless) terms'*, (e.g. the set of unsolvables) is in some sense the converse of the approach phrased by *'which terms are meaningless with respect to a given set of (meaningful) terms'*, e.g. the set of terms representing partial recursive functions.

In the previous section we followed the latter idea, in contrast strict contexts play an important role in [Kup94] in a generalization of solvability. We will investigate this technique later in section 2.3 and an adaptation to TRSs in section 3.3.

2.2.3 Solvability

Remember that a λ-term M is solvable if it has a closure $\lambda x_1 \ldots x_k.M$ and there are terms $N_1 \ldots N_n$ $(n \geq 0)$ such that $(\lambda x_1 \ldots x_k.M)\ N_1 \ldots N_n = \mathbf{I}$, or equivalently if M has a head normal form (i.e. $M = \lambda x_1 \ldots x_k.yN_1 \ldots N_n$). 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.

In this section we are going to focus on solvability in the λK-calculus, with a motivation to refine this notion by partitioning solvable terms into various subclasses.

How solvable is a term?

Our first question 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.

Definition 2.2.5. let M be a λ-term, then

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

Remark 2.2.6. It is obvious that $n \in S(M) \Longrightarrow n+1 \in S(M)$ and that $m(M) < \infty \iff M$ *solvable*.

Example 2.2.7. The standard combinators

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

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) $\mathbf{KI\Omega} = \mathbf{I}$ 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 2.2.8. $M = N \Longrightarrow m(M) = m(N)$

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

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

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

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 2.2.12.
Applying the result multiple times we get $m(M) \leq m(M[\vec{x} := \vec{P}]) + |\vec{x}|$.

Definition 2.2.13. 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 2.2.14. A closed head normal form is automatically head closed.

Lemma 2.2.15. $xN_1 \ldots N_n \neq y \iff n > 0$ 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 2.2.16. 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) $(\lambda x_1 \ldots x_n.x_iN_1 \ldots N_k)X_1 \ldots X_{i-1}\mathbf{U^{k+1}_{k+1}}X_{i+1} \ldots X_n = \mathbf{U^{k+1}_{k+1}}N'_1 \ldots N'_k = \mathbf{I}$
$\Longrightarrow m(M) \leq n$
$(\lambda x_1 \ldots x_n.N)X_1 \ldots X_{n-2} = \lambda x_{n-1}x_n.N' \neq \mathbf{I}$
$\Longrightarrow m(M) \geq n-1$
ii) Immediate from i) by Lemma 2.2.15 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 2.2.15. □

Remark 2.2.17. Every solvable term has a huge set of trivial solutions as suggested by the first part of i) in the proof, namely
$(\lambda x_1 \ldots x_n.x_iN_1 \ldots N_k)X_1 \ldots X_{i-1}\mathbf{U^{k+1}_{k+1}}X_{i+1} \ldots X_n$
where X_i are arbitrary terms; and a "canonical" trivial solution :
$(\lambda x_1 \ldots x_n.x_iN_1 \ldots N_k)\mathbf{\Omega}_{(1)} \ldots \mathbf{\Omega}_{(i-1)}\mathbf{U^{k+1}_{k+1}}\mathbf{\Omega}_{(i+1)} \ldots \mathbf{\Omega}_{(n)}$
Of course there might be other solutions of n arguments in individual cases, but in general not (as in $\lambda\vec{x}.x_i(\mathbf{\Omega}\vec{x})$ for example).

As the Examples in 2.2.7 show, this proposition cannot be strengthened but it gives rise to a refined categorization of solvable terms.

Breakable terms

Definition 2.2.18. (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 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 2.2.19. on the definition

i) In the light of Remark 2.2.17, 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 2.2.8 $m(M)$ does not depend on the particular head normal form. From Proposition 2.2.8 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 2.2.20. 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) $\mathbf{\Omega} \notin \mathcal{B}$

Example 2.2.21. A more complicated example:
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)

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

which is a contradiction, because $m(\mathbf{K}) = 2$ (see Example 2.2.7). □

Proposition 2.2.22. In all of the following cases we allow $\vec{x}$ 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 2.2.15. □

Corollary 2.2.23. *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 2.2.22 ii).
□

Lemma 2.2.24. $\forall F \in \Lambda : \mathbf{Y}F = F(\mathbf{Y}F) \Longrightarrow \mathbf{Y} \twoheadrightarrow \lambda f.fY^*$

Proof
$\mathbf{Y}$ is solvable, in fact $m(\mathbf{Y}) = 1$ as seen in Example 2.2.7, 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 2.2.16.
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 ($\mathbf{\Omega}$ 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 terms it has one. Then by **CR** we get that $k = 1$. □

We can further generalize this result as follows.

Theorem 2.2.25. *$\forall \mathbf{Y}$ 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 **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 same argument by **CR** we get that $Y^{(k)} \twoheadrightarrow fY^{(k+1)}$ and $Y^{(k+1)}[f := F] = \mathbf{Y}F$ again, we can prove the claim by induction for every $k \in \mathbb{N}$. □

Corollary 2.2.26.
All fixed point combinators have the same (infinite) Böhm tree [1] . 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.

Returning to breakability, next we show that analogous to the fact (used in the proof above) that solvability of an instance of a term implies the solvability of the term, breakability behaves the same way.

Proposition 2.2.27. $M[x := P]$ is breakable $\Longrightarrow$ M is breakable

Proof
The case $x \notin FV(M)$ is obvious, otherwise use Proposition 2.2.11. □

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 2.2.28. Let $M \equiv yM_1 \dots M_k$.
Then for any permutation $(i_1, \dots, i_n)$ of $\{1, \dots, n\}$:
$\lambda x_1 \dots x_n z.M$ is breakable $\Longleftrightarrow$ $\lambda x_{i_1} \dots x_{i_n} z.M$ 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 $\lambda x_1 \dots x_n z.M$ is closed $\Longleftrightarrow$ $\lambda x_{i_1} \dots x_{i_n} z.M$ is closed. According to this we consider two cases:
Let us assume first that $\lambda x_1 \dots x_n z.M$ is closed. Then by Definition 2.2.18 it is breakable iff there are terms $N_1, \dots, N_n$ such that $(\lambda x_1 \dots x_n z.M)N_1 \dots N_n = \mathbf{I}$. But now (assuming by the variable convention, that $x_1, \dots, x_n \notin FV(N_1 \dots N_n)$) $(\lambda x_{i_1} \dots x_{i_n} z.M)N_{i_1} \dots N_{i_n} = (\lambda x_1 \dots x_n z.M)N_1 \dots N_n = \mathbf{I}$, i.e. $(\lambda x_{i_1} \dots x_{i_n} z.M)$ 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 $\lambda \vec{y} x_1 \dots x_n z.M$ be a breakable closure of $\lambda x_1 \dots x_n z.M$. Then using the first case we conclude that $\lambda \vec{y} x_{i_1} \dots x_{i_n} z.M$ and thus $\lambda x_{i_1} \dots x_{i_n} z.M$ is breakable as well.
□

Remark 2.2.29. 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 2.2.30. For any term $M \in \Lambda^0$:
M is breakable $\Longleftrightarrow$ $\lambda x.M$ is breakable

[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])

Proof
By Definition 2.2.18 and Proposition 2.2.9. $\square$

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

Proposition 2.2.31. 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 2.2.30)
$\exists \vec{u} : \lambda x\vec{u}y.M \in \Lambda^0 \cap \mathcal{B} \iff$ (by Proposition 2.2.28)
$\exists \vec{u} : \lambda \vec{u}xy.M \in \Lambda^0 \cap \mathcal{B} \iff$ (by definition)
$\lambda xy.M \in \mathcal{B}$.
$\square$

It is easy to see (see Remark 2.2.3) that application is strict in its left argument with respect to the set of unsolvable terms, i.e.

$$Q \textit{ unsolvable}, M \textit{ arbitrary} \quad \Longrightarrow \quad QM \textit{ unsolvable}$$

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

Example 2.2.32. (Breakability and application)

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

Remark 2.2.33. If it were the case, as I conjecture, that the set $\mathcal{B}^0$ of closed breakable terms is closed under application, then $\mathcal{B}^0$ would be a set of terms closed under term formation rules (see Proposition 2.2.30). One could then ask to find a sufficiently simple or better yet a minimal generator set for $\mathcal{B}^0$ under the term formation rules. Of course $\mathcal{B}^0$ and hence all of its generator sets are not recursive, thus the most one can hope for is to find an algorithm, which recursively enumerates one of them.

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 2.2.34.
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 2.2.35.
xy is breakable for y but not for x nor any other variable
$x(yx)$ is not breakable for any variable (Example 2.2.21 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 2.2.36.

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 2.2.21 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 remark following Definition 2.2.18, we noted that equal (i.e. β-convertible) terms are either both breakable or both unbreakable. Later we proved that all fixed point combinators are unbreakable, while our efforts have lead to proving that they all share the same Böhm tree (See Corollary 2.2.23 and 2.2.26). A simple consequence of Proposition 18.3.4 of [Bar84] is that in general, the following connection holds:

Proposition 2.2.37. Let M, N 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. □

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 2.2.38. 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 2.2.39. If an ahnf $M \equiv yN_1 \dots N_n$ 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 2.2.40. Let $M \equiv yN_1 \dots N_n$. If $\exists i : (N_i \in \mathcal{B}_z \wedge y \notin FV(N_i)$, then $M \in \mathcal{B}_z$.

Note that for $M \equiv yN_1 \dots N_n$ 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 2.2.41. Let $M \equiv yN_1 \dots N_n$. 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 $FV(N'_i) \setminus \{z, \vec{x}\} = \{u_1, \dots, u_k\}$ and $FV(N_1 \dots N_n) \setminus FV(N'_i) = \{v_1, \dots, v_l\}$. 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 $V_1, \dots, V_l$ be arbitrary terms and $Y \equiv \lambda w_1 \dots w_n.w_i X_1 \dots X_m$. Then $M[\vec{u} := \vec{U}][\vec{v} := \vec{V}][y := Y] = YN'_1 \dots N'_{i-1}(N_i[\vec{u} := \vec{U}])N'_{i+1} \dots N'_n = (N_i[\vec{u} := \vec{U}])\vec{X} = z$, and so $M \in \mathcal{B}_z$. $\square$

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 2.2.42.
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)

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

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

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

□

Example 2.2.43. $x(xz\Delta) \notin \mathcal{B}_z$

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 2.2.15, contradiction. □

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:

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

There are different ways of generalizing this, giving the following classes of problems (below M^* always denotes an instance of M):

- 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 thesis, just mentioned them as interesting questions for the curious mind.

Solvability of application terms

What can we conclude of the solvability of MN from the solvability of M and N ? This is another question with no satisfying answer, but from which we can derive new properties of terms. There is, in fact, one trivial case:

Proposition 2.2.44. M is unsolvable $\implies$ MN is unsolvable.

However, we can easily find examples for every other case:

Example 2.2.45.

$\mathbf{I}$ solvable,	$\mathbf{\Omega}$ unsolvable	$\mathbf{I\Omega} = \mathbf{\Omega}$ unsolvable
$\mathbf{F}$ solvable,	$\mathbf{\Omega}$ unsolvable	$\mathbf{F\Omega} = \mathbf{I}$ solvable
$\mathbf{I}$ solvable,	$\mathbf{I}$ solvable	$\mathbf{II} = \mathbf{I}$ solvable
$\mathbf{D}$ solvable,	$\mathbf{D}$ solvable	$\mathbf{DD} = \mathbf{\Omega}$ unsolvable

Where there is no straight answer, there is the possibility of a new definition.

Definition 2.2.46.
i) M is **solvable with** N iff MN is solvable
ii) $W(M) = \{N|MN\ \mathit{is\ solvable}\}$
iii) Two terms M and N are **equi-solvable** iff $W(M) = W(N)$
iv) and **co-solvable** iff $N \in W(M) and M \in W(N)$
v) and **counter-solvable** iff $N \in W(M) \iff M \in W(N)$
etc.

Remark 2.2.47. A term M is unsolvable $\iff W(M) = \emptyset$

Definition 2.2.48. A term M is

- **easily solvable** iff $W(M) = \Lambda$
- **self solvable** iff $M \in W(M)$

Example 2.2.49. Let's see the standard combinators again:
i) $\mathbf{I}$, $\mathbf{K}$, $\mathbf{S}$ are self solvable, $\mathbf{D}$ is not, but it is solvable with exactly the self-solvable terms.
ii) $\mathbf{F}$ is easily solvable $\mathbf{I}$, $\mathbf{K}$, $\mathbf{S}$ are not.
iii) In fact $W(\mathbf{I}) = W(\mathbf{K}) = W(\mathbf{S}) = \{solvables\}$, that is $\mathbf{I}$, $\mathbf{K}$ and $\mathbf{S}$ are equi-solvable, they are solvable exactly by the solvable terms, whereas $\mathbf{D}$ is not solvable by itself although it is solvable.
iv) $\mathbf{I}$, $\mathbf{K}$ and $\mathbf{S}$ are all counter-solvable with $\mathbf{\Omega}$.
v) $\mathbf{F}$ and $\mathbf{D}$ are co-solvable even in the strongest sense: $\mathbf{FD} = \mathbf{DF} = \mathbf{I}$.

Proposition 2.2.50. For every term M
M solvable with $\mathbf{\Omega} \iff M$ easily solvable $\Longrightarrow M$ self solvable $\Longrightarrow M$ solvable

Proof
The first $\Longrightarrow$ is by the Genericity Lemma, the others are trivial. □

Corollary 2.2.51. $\mathcal{S} \subsetneq W(M) \Longrightarrow W(M) = \Lambda$

Proposition 2.2.52.
i) Every term of the form $\lambda x\vec{y}.xx\vec{N}$ is not self solvable ($\vec{y}, \vec{N}$ can be empty)
ii) For every M: M solvable $\iff \mathbf{K}M$ solvable $\iff \mathbf{K}M$ easily solvable
iii) For every term M: M solvable $\iff \mathbf{S}M$ solvable

Proof
i) Such a term applied to itself will allow an infinite head-reduction.
ii) $\mathbf{K}M$ is solvable iff M is solvable, in which case it is obviously easily solvable too.
iii) If $MN_1 \ldots N_n = I$ (we can assume $n >= 2$) then $\mathbf{S}M(\mathbf{K}N_2)N_1N_3 \ldots N_n = MN_1 \ldots N_n = I$. The other direction is even more obvious. □

Using the sets $W(M)$ associated to every term M we can define different pre(orders) on λ-terms. One straightforward way to do that is as follows.

Definition 2.2.53. Let $P \leq_W Q$ iff $W(P) \subseteq W(Q)$

This is obviously a preorder on terms with exactly the unsolvable terms being minimal and the easily solvable terms being maximal with respect to it. We have seen that $W(I) = W(\lambda x.x) = \{\text{solvables}\}$, $W(D) = W(\lambda x.xx) = \{\text{self solvables}\}$, and $W(\lambda x.x\mathbf{\Omega}) = \{\text{easily solvables}\}$ can be checked similarly. By generalizing the latter two examples, we can define decreasing sequences of subsets of solvable terms as follows.

Definition 2.2.54.
$\mathcal{S}^n\mathcal{S} = W(\lambda x.xx^n)$
$\mathcal{E}^n\mathcal{S} = W(\lambda x.x\mathbf{\Omega}^n)$

Proposition 2.2.55.
i) $\forall n \in \mathbb{N} :\ \mathcal{S}^{n+1}\mathcal{S} \subsetneq \mathcal{S}^n\mathcal{S}$, $\mathcal{E}^{n+1}\mathcal{S} \subsetneq \mathcal{E}^n\mathcal{S}$, $\mathcal{E}^n\mathcal{S} \subsetneq \mathcal{S}^n\mathcal{S}$
ii) $\lambda\vec{x}.x_i\vec{N} \in \mathcal{E}^{i-1}\mathcal{S} \setminus \mathcal{E}^i\mathcal{S}$
iii) $\lambda x_1 \ldots x_n.x_n x_n x_1 \ldots x_{n-1} \in \mathcal{S}^{n-1}\mathcal{S} \setminus \mathcal{S}^n\mathcal{S}$
iv) $\mathbf{I} \in \mathcal{S}^n\mathcal{S} \setminus \mathcal{E}^n\mathcal{S}$

Proof
The inclusions in i) are obvious from the definition, and ii) - iv) prove that they are proper. ii) - iv) can be simply checked to hold. □

Hence the sets $\mathcal{S}^n\mathcal{S}$ and $\mathcal{E}^n\mathcal{S}$ of the above definition give another answer to the question of this section "How solvable is a term?".

2.2.4 Other Notions of Meaninglessness

In the previous sections we got acquainted with solvable and unsolvable terms. In this section we will give a brief overview of some of the other notions of meaninglessness which appeared in the literature.

Easy Terms

One could consider the theory $\mathcal{L}$ over λ-terms consisting of the sets of equations derivable in the λ-calculus. The importance of the Churh-Rosser theorem (Theorem 1.2.7) is exactly that it proves the consistency of this theory. [2] A standard technique in logic is to look at extensions of a theory and their consistency. Extensions are obtained by adding new equations to the original set of defining equations. Applying this to $\mathcal{L}$, we can define the following relation on terms.

Definition 2.2.56. (Consistent equalities)
Let M,N be λ-terms. Then $Con(M = N)$ denotes the proposition that the theory $\mathcal{L} + M = N$, obtained by adding the equality $M = N$ to $\mathcal{L}$, is consistent.

Definition 2.2.57. (Easy terms)
A term $M \in \Lambda$ is **easy** if $\forall N \in \Lambda : Con(M = N)$.
Let $\mathcal{E}$ denote the set of easy terms.

A famous result of Böhm states that distinct normal forms are separable, (cf. [Bar84] 10.4.2) and consequently cannot be consistently identified. Barendregt proved that the set of unsolvable terms can be consistently identified, hence any two of them can. In our notation:

$$M, N \in \mathcal{NF}\ ,\ M \neq N \Longrightarrow \neg Con(M = N)$$

[2] Church-Rosser types of theorems, i.e. confluence theorems are typically used to prove consistency of a theory by orienting its defining equations to obtain a rewriting system, and (after some possible adjustments, e.g. Knuth-Bendix completion) proving its confluence, thus the consistency of the theory, provided that there are at least two distinct normal forms with respect to this rewriting system.

$$M, N \in \mathcal{U} \Longrightarrow Con(M = N)$$

Hence normal forms are not easy, but in fact this is true for all solvable terms.

Proposition 2.2.58. Solvable terms are not easy, i.e. $\mathcal{E} \subseteq \mathcal{S}$.

Proof
Let $MP_1 \dots P_n = \mathbf{I}$, and let $U = \lambda x_1 \dots x_n.\mathbf{K}$. Then $UP_1 \dots P_n = \mathbf{K}$, hence $\mathcal{L} + M = U \vdash \mathbf{I} = \mathbf{K}$ so by Böhm 's separability theorem $\neg Con(M = U)$. Thus M is not easy. Since this applies to all solvable M, $\mathcal{E} \subseteq \mathcal{S}$ hold. ☐

Remark 2.2.59.
Actually the inclusion is proper, since for example $\mathbf{\Omega}_3 \equiv (\lambda x.xxx)(\lambda x.xxx)$ and $\mathbf{K}^\infty \equiv \mathbf{YK}$ are unsolvable, but not easy ($\neg Con(\mathbf{\Omega}_3 = \mathbf{I})$, and $\neg Con(\mathbf{K}^\infty = \mathbf{I})$).

The first result on easy terms is due to Jacopini, who showed that $\mathbf{\Omega}$ is easy. [BB79] also contains a model-theoretical proof of this fact. Other results on easy terms presented in [Int91] and [BI93] used Church-Rosser type of arguments. [Kup97] is an explanatory survey and generalization of Jacopini's technique.

Terms of Order Zero

Since we have already used terms of zero order in the proof of Lemma 2.2.24, we will mention them here as well.

Definition 2.2.60. (Terms of order zero) A term M is **of order zero** if it can not be reduced to an abstraction term, i.e. $\forall x, N : M \not\twoheadrightarrow \lambda x.N$.
Let $\mathcal{Z}$ denote the set of zero order terms.

Immediate from definition that $\mathcal{Z}$ is closed under reduction but it is easy to show by **CR** that if a term reduces to a term of order zero, then it can not reduce to an abstraction term, consequently $\mathcal{Z}$ is closed under conversion. It is also closed under application of arbitrary arguments, i.e. application is strict in its left argument with respect to terms of order zero:

Proposition 2.2.61. $M \in \mathcal{Z}, N \in \Lambda \Longrightarrow MN \in \mathcal{Z}$

Proof
Since M can not reduce to an abstraction term, MN can not reduce to a redex, so all of its reducts are application terms, which is another way of saying that MN is of order zero. ☐

Corollary 2.2.62. *Terms of order zero are unsolvable, i.e.* $\mathcal{Z} \subseteq \mathcal{S}$.

Proof
If M is of order zero, then so is $M\vec{P}$ for every sequence $\vec{P}$ of terms, therefore $M\vec{P} \neq \mathbf{I}$, since $\mathbf{I}$ is of course not zero order. ☐

Remark 2.2.63.
Again, the inclusion is proper, since for example $\mathbf{K}^\infty$ is unsolvable, but not zero order (in fact it is more of infinite order, so to speak, since its Böhm tree consists of an infinite chain of abstractions).

Let $\mathcal{Z}^0$ denote the set of closed terms of order zero. Then it is also closed under conversion, it is non-trivial, and the complement $\Lambda^0 \setminus \mathcal{Z}^0$ is obviously recursively enumerable, hence Statman's theorem (Theorem 2.2.1) applies, so it is possible to λ-define the partial recursive functions in such a way, that the undefined is represented by terms of $\mathcal{Z}^0$. The same holds for the set $\mathcal{E}^0$ of closed easy terms as well, as seen in [Bar92].

2.3 Other λ-calculi

In this section we give a brief overview of some of the concepts and results of [Kup94] and consider various generalizations. (Note that [Kup95] also covers everything we need in this section and is probably easier to get hold of.)

Kuper identified three ways of extracting information out of a term:

- by (partially) evaluating it, i.e. bringing it to (some) normal form
- by evaluating it inside a context which 'actively uses' it
- by equating it to some other term and trying to derive a contradiction

and followed three approaches to characterizing meaningfulness and meaninglessness accordingly. The intuition behind the second technique is that there are terms which cannot be reduced to a certain kind of normal form (e.g. a representation of a recursive function or an infinite list of numerals which have no normal form) but can act on some appropriate arguments within a context to yield a term with a normal form, that is a term meaningful in the first sense. It is not clear, however, how to define which contexts should be considered to actively use their argument. Kuper gave an inductive syntactical definition, but with the same intuition in mind we will give an alternative definition.

Kuper demonstrated these concepts in $\lambda\mu\mathbf{E}$ -calculus: a simply typed calculus with numeral and arithmetical as well boolean, test, and conditional constants and a special recursion abstractor μ. The most important feature of this system is the special constant $\mathtt{u}\!:\!\mathbf{Nat}$ denoting recognizable undefinedness and $\mathtt{E}_? : \mathbf{Nat} \to \mathbf{Bool}$ as a test of (recognizable) existence (i.e. $\mathtt{E}_?\underline{\mathtt{n}} \to \mathbf{true}$, $\mathtt{E}_?\mathtt{u} \to \mathbf{false}$). In other respects his choice of the system was motivated by rather practical reasons, and most of the demonstrated ideas can easily be adopted to other $\lambda\delta$-calculi. With respect to the result, of course, one cannot generally talk about all $\lambda\delta$-calculi, but here we are primarily interested in the concepts and notions used. Some results do hold, however regardless of types and constants, some of which will be mentioned.

Let us finally remark that the more general approach of [KOV99] also applies to (typed) $\lambda(\delta)$-calculi, but will be discussed in a later chapter. We cannot say that the latter work, axiomatizing some of the most important properties of unsolvable terms in a very general framework, obsoletes Kuper's ideas. The two approaches rather complement one another, as we will see in sections 3.1.2 and 3.3.

2.3.1 Strictness

We have previously remarked that the partial recursive functions are strict with respect to undefinedness, meaning that if at least one of the arguments of a partial recursive function is undefined, then so is the result of applying the function to the arguments. In partial logic all the function symbols are in the same way expected and interpreted to represent strict operations with respect to undefinedness. This is expressed by one of the axiom schemes of the deduction calculi for partial logic (cf. [Kup94] page 12).

This is of course a very natural expectation, as the intended meaning of the statement *'term M is undefined'* is that absolutely no information is available regarding the value or meaning of the term M, consequently one cannot obtain any information of any term of which M is a subterm. This view seems, at first, to fail as soon as constant functions are present in a theory. However, if we think of the knowledge of having no information regarding a term, i.e. the knowledge of the undefinedness of a term to be, in itself, some information about that term, then having absolutely no information regarding a term can mean no information even about its possible undefinedness. In the introduction to his thesis Kuper speaks of two kinds of undefinedness:

- recognizable, and
- unrecognizable

the former being the case when the undefinedness of a term can be learned by a user of the system or even the system itself, while in the latter case not even this knowledge can be obtained. A simple example of a recognizably undefined term is 1/0, while unrecognizable undefinedness can usually be thought of as non-termination of a computation. Constant functions can in some 'lazy' formalisms be represented as to ignore the appropriate arguments, while in others as to evaluate them. Thus the behavior of constant functions applied to undefined arguments is an implementational question rather than logical. This justifies, in my view, the notion of interpretation of formulae of partial logic with respect to strictness, and the axiom scheme concerning strictness of the function symbols.

We have already seen that λK-calculus does not behave in this expected way: one cannot represent the undefined with any set of terms even under minimal restrictions to ensure strictness of all terms and contexts (see Theorem 2.2.2). This is due to the liberal abstraction forming rule of the λK-calculus, which allows such 'void' abstractions as can, for instance, be seen in $\mathbf{F} \equiv \lambda xy.y \equiv \lambda x.\mathbf{I}$. This puts λK-calculus equipped with a lazy reduction strategy among the lazy formalisms mentioned above.

We begin by introducing the first syntactical notion of strict contexts (to which we will refer to as "K-strict contexts" to avoid confusion with a later definition):

Definition 2.3.1. **K-strict contexts** are defined inductively as follows:

1. $[]$ is K-strict
2. if $C[]$ K-strict then $f(C[])$ is K-strict for all constants f
3. if $C[]$ K-strict then $(C[])M$ is K-strict for all terms M
4. if $C[]$ K-strict then $\lambda x.(C[])$ is K-strict for all variables x.

We will denote the set of K-strict contexts by $\mathcal{K}$.

Remark 2.3.2. The original definition included another case for abstraction by the μ recursor which we did not adopt here, since it is specific to $\lambda\mu\mathbf{E}$ -calculus.

Observe that the above definition of K-strict contexts differs from that of the contexts (Definition 1.2.11) only in that application from the left is restricted to constants. This is in accordance with our remarks after Theorem 2.2.2, that with respect to the unsolvables abstraction is strict and application is strict in its left argument, but not in its right. But also note that the validity of this formation rule with regard to strictness depends entirely on the behavior of the particular constant, and as such should, in general, be only used with restrictions on the constants.

The following property of K-strict contexts is straightforward.

Proposition 2.3.3. (K-strict contexts are closed under composition.)
$C[C'[]]$ is K-strict $\iff$ $C[]$ and $C'[]$ are both K-strict.

Recall that solvability of a term M was defined (in 1.2.13) as $\exists\vec{x}, \vec{N} : (\lambda\vec{x}.M)\vec{N} = \mathbf{I}$. Observe that this can (in pure λ-calculus) be rephrased in terms of K-strict contexts as follows: $\exists C[] \in \mathcal{K} : C[M] = \mathbf{I}$. The above definition of K-strict contexts was motivated by the same intuition as that underlying solvability, which explains this strong resemblance, but more importantly allows a generalization of solvability to $\lambda\mu\mathbf{E}$ -calculus, discussed in the next subsection.

We have to note however, that this definition of K-strict contexts is purely syntactical, and intimately related to $\lambda\mu\mathbf{E}$ -calculus. Therefore we will first consider two alternative formalizations: one based on the idea of strictness as used in partial logic and another based on the same intuition underlying usability. But taking a turn later (in section 3.3) we will look at a way of defining K-strict contexts for TRSs in general based on redexes.

2.3.2 Usability: generalizing solvability

We already saw how solvability, being a semantical notion, lost some of its syntactical properties while moving from $\lambda I$to λK-calculus (e.g. strictness of all terms with respect to the unsolvables, or the equivalence of having a normal form to the equivalence of having a head normal form). This trend inevitably continues when extending the syntax even further, e.g. by adding constants and corresponding δ-rules. The same holds for typed λ-calculus.

One straightforward way to define solvability in typed λ-calculi is to say

$$\exists \tau \in Typ, \vec{x}, \vec{N} : ((\lambda \vec{x}.M)\vec{N}) = \mathbf{I}_\tau$$

Solvability automatically generalizes to $\lambda\delta$-calculi, but as pointed out in Example 7.2.5 of [Kup94] not all normal forms are necessarily solvable anymore: take any constant of ground type for example.

Kuper considers the following generalizations of solvability:

Definition 2.3.4. (taken from [Kup94] 7.2.4)

1. M is **strongly solvable** if there is a type σ such that

$$\exists \vec{x}, \vec{N} : (\lambda \vec{x}.M)\vec{N} = \mathbf{I}_\sigma$$

2. M is **solvable** if there is a type σ such that for all L of type σ

$$\exists \vec{x}, \vec{N} : (\lambda \vec{x}.M)\vec{N} = L$$

3. M is **weakly solvable** if there is (a type σ and) a term L in normal form (of type σ), such that

$$\exists \vec{x}, \vec{N} : (\lambda \vec{x}.M)\vec{N} = L$$

It is easy to see that in the λK-calculus all three notions are equivalent, but as our example showed, this equivalence in general does not hold in typed $\lambda\delta$-calculi. The following implications however, are straightforward:

Lemma 2.3.5. (cf. [Kup94] 7.2.6) For all M
M strongly solvable $\Longrightarrow$ M solvable $\Longrightarrow$ M weakly solvable.

In [Kup94] section 7.2 the notion of relative usability is defined in terms of K-strict contexts. It was introduced by Barendregt, and in [Kup94] it is used to generalize the notion of solvability to $\lambda\mu\mathbf{E}$ -calculus.

Definition 2.3.6. (Relative usability, usable terms)

1. M is **usable to compute** N ($M \gg N$) if there is a K-strict context $C[]$ such that $C[M] \twoheadrightarrow N$.

2. M is **usable** if it is usable to compute some term in normal form.

Remark 2.3.7. We will refer to the relation $\gg$ as 'relative usability' or 'usability relation', and denote the set of unusable terms with $\mathcal{U}$. (The same notation used for the set of unsolvables in pure λ-calculus.)

Indeed, usability coincides with solvability in pure λ-calculus, and Kuper showed that in the $\lambda\mu\mathbf{E}$ -calculus it is equivalent to weak solvability (cf. [Kup94] 7.2.23) and further that the genericity lemma holds for the set of unusable terms (cf. [Kup94] 7.3.17), consequently they can be consistently identified (cf. [Kup94] 7.4.1).

Let us now list some of the properties of usability and relative usability proved in [Kup94] Lemma 7.2.14 - 7.2.18. Note that these properties follow from definition, independent of $\lambda\mu\mathbf{E}$ -calculus.

Proposition 2.3.8.

1. $C[]\ K - strict \Longrightarrow M \gg C[M]$ in particular $M \gg fM, MN, \lambda x.M$
2. $M \twoheadrightarrow N \Longrightarrow M \gg N$
3. $M \gg M$
4. $M \gg N, N \gg L \Longrightarrow M \gg L$
5. $M \gg M[x := N]$
6. $\lambda x.M \gg M$
7. $M \gg N, N\ usable \Longrightarrow M\ usable$, i.e. the unusables are closed under $\gg$.
8. $C[]\ K - strict, C[M]\ usable \Longrightarrow M\ usable$, i.e. the unusables are closed under transformation by K-strict contexts, i.e. K-strict contexts are indeed strict with respect to the unusables.
9. If the λ-theory $\mathcal{L}$ is **CR** , then $\mathcal{L} \vdash M = N \Longrightarrow (M\ usable \iff N\ usable)$.

If two contexts are $\beta\delta$-convertible, then so are their instances (see Remark 1.2.12). Consequently if a context is convertible to a K-strict context, then it is also strict with respect to the unsolvables. The converse of this implication does not hold as can be seen by item ii) of the next example.

Example 2.3.9.

i) $\mathbf{I}[]$ is not K-strict, but $\mathbf{I}[] = []$ is

ii) $\mathbf{D}[]$ is not convertible to a K-strict context, but is strict with respect to the unusables

It seems unlikely to be able to give a perfect syntactic characterization of all contexts which are strict with respect to the unusables, but we can prove the following simple proposition, stating that variables cannot appear at the root of strict contexts. Combining that with the definition of K-strict contexts, which allows constants, λ-abstraction and [] to appear at the root of a strict context, we see that only in the case of having a complex term at the root of a context do we have uncertainty about its strictness, just as in the examples above.

Proposition 2.3.10.
Let $C[] \equiv xM_1 \ldots M_{i-1}(C'[])M_{i+1} \ldots M_n$. Then $\forall M \in \Lambda_{\mu\mathbf{E}} : C[M]\ is\ usable$, i.e. $C[]$ is not strict with respect to the unusables.

Proof
Let $M \in \Lambda_{\mu\mathbf{E}}$ be arbitrary. Since $D[] \equiv (\lambda x.[]\mathbf{I})\mathbf{U}^{\mathbf{n+1}}_{\mathbf{n+1}}$ is strict and $D[C[M]] = \mathbf{I}$, $C[M]$ is usable. □

So far we have seen how to use K-strict contexts to define usability, a generalization of solvability to $\lambda\mu\mathbf{E}$ -calculus. We have also noted, without any effort to prove, that this technique can also be used in various other λ-calculi, but we have also pointed out the difficulties of such generalizations, mainly due to the syntactical definition of K-strict contexts.

2.3.3 Alternative definitions of usability

Attempting to break the syntactical barrier, we will now give two alternative formalizations, as promised.

Definition 2.3.11. (Contexts as term transformations.)

1. let $\mathcal{A}$ be a set of terms and $C[]$ a context, then $C[\mathcal{A}] = \{C[M] \mid M \in \mathcal{A}\}$
2. let $\mathcal{A}$ and $\mathcal{B}$ be sets of terms, then $\mathcal{C}^{\mathcal{A}}_{\mathcal{B}} = \{C[] \mid C[\mathcal{A}] \subseteq \mathcal{B}\}$

Definition 2.3.12. (Strict contexts.)
Let $\mathcal{A}$ be a set of terms. Then a context $C[]$ is **$\mathcal{A}$-strict** if $C[] \in \mathcal{C}^{\mathcal{A}}_{\mathcal{A}}$

The following properties of $\mathcal{A}$-strict contexts are trivial.

Proposition 2.3.13. For every set $\mathcal{A}$ of terms
$[] \in \mathcal{C}^{\mathcal{A}}_{\mathcal{A}}$ and
$C[], D[] \in \mathcal{C}^{\mathcal{A}}_{\mathcal{A}} \Longrightarrow C[D[]] \in \mathcal{C}^{\mathcal{A}}_{\mathcal{A}}$

Our next definition is motivated by the same intuition underlying usability. Namely, a term has some meaning if we can extract some information out of it by a non-trivial mechanism definable in the original system, i.e. by a separating context. Intuitively a separating context represents an automaton, which (partially) recursively separates a set of terms from the rest, and a term is separable if it can be separated from at least one other term by such a method. Note that these definitions apply to any reduction system.

Definition 2.3.14. (Separating contexts.)
A context $C[]$ is a **separating context** if there are terms P, Q and a normal form N such that $C[P] \twoheadrightarrow N \not\twoheadleftarrow C[Q]$.
We will denote the set of separating contexts by $\mathcal{C}^p$.

Definition 2.3.15. (Separable terms.)
A term M is **separable** if there is a separating context $C[]$ such that $C[M]$ has a normal form, otherwise it is **inseparable**.
We will denote the set of separable terms by $\mathcal{P}$.

Let us remark that the set of K-strict contexts and separating contexts cannot be compared by $\subseteq$. For instance $\mathbf{I}[]$ is separating but not K-strict, while $\underline{\texttt{if}}[]\mathbf{\Omega\Omega}$ is K-strict, but not separating. On the other hand, the set of usable terms coincides with the set of separable terms. In order to prove this we employ another definition and result of [Kup94].

Definition 2.3.16. (Generic terms, Definition 7.3.1 in [Kup94])
A term M is **generic** if $\forall C[] : (C[M] \twoheadrightarrow N \in \mathcal{NF} \Longrightarrow \forall L : C[L] \twoheadrightarrow N)$

Theorem 2.3.17. *(Theorems 7.2.23 and 7.3.2 in [Kup94])*

i) In λK-calculus:
M is not generic $\iff$ M is usable $\iff$ M is solvable.

ii) In $\lambda\mu\mathbf{E}$ -calculus:
M is not generic $\iff$ M is usable $\iff$ M is weakly solvable.

The following theorem connects separable terms with generic terms, hence with usable terms as well.

Theorem 2.3.18. *In any combinatory reduction system:*
M is generic $\iff$ M is inseparable.

Proof
$\Longrightarrow$:
Let M be generic, and $C[]$ be a separating context. Suppose that $C[M] \twoheadrightarrow N$, where N is in normal form. Then by definition of separating contexts there exists a term Q for which $C[Q] \not\twoheadrightarrow N$, contradicting genericity of M. Hence M has no normal form in a separating context, i.e. it is inseparable.
$\Longleftarrow$:
Let M be not generic. Then there exists a context $C[]$ and a term Q such that $C[M]$ has a normal form, but $C[Q]$ does not have the same normal form. Then by definition $C[]$ is a separating context, and M is separable. $\square$

Corollary 2.3.19. *In $\lambda\mu\mathbf{E}$ -calculus:*
M is separable $\iff$ M is usable

Proof
Using Theorem 2.3.17.ii). $\square$

So in λ and in $\lambda\mu\mathbf{E}$ the three notions: unusable, generic, and inseparable are equivalent, and the last two coincide in every combinatory reduction system, which gives a characterization of (non) generic terms via the class of separating contexts, just like (un)usable terms were defined using strict contexts. This inspires the following general notion:

Definition 2.3.20. Let $\mathcal{C}$ be a set of contexts. Then a term M is $\mathcal{C}$ **-usable** if $\exists C[] \in \mathcal{C}$ s.t. $C[M]$ has a normal form, otherwise it is $\mathcal{C}$ **-unusable**.
Let $S(\mathcal{C})$ and $U(\mathcal{C})$ denote the set of $\mathcal{C}$ -usable and $\mathcal{C}$ -unusable terms respectively.

Our previous results imply that in $\lambda\mu\mathbf{E}$ K-strict contexts and separating contexts give rise to the same partitioning of terms. K-strict contexts are important because they are syntactically defined, while separating contexts formalize our intuition of a context actively using its argument in a more abstract form. It is interesting that the two classes of contexts are incomparable.

Definition 2.3.21. Let $\mathcal{C}$ and $\mathcal{C}'$ be sets of contexts. Then
$\mathcal{C} \sim \mathcal{C}' \iff S(\mathcal{C}) = S(\mathcal{C}')$
$\mathcal{C} \preceq \mathcal{C}' \iff S(\mathcal{C}) \subseteq S(\mathcal{C}')$
$\mathcal{C} \prec \mathcal{C}' \iff S(\mathcal{C}) \subset S(\mathcal{C}')$

Remark 2.3.22. Clearly $\prec$ is a strict partial order (it is irreflexive and transitive), and $\preceq$ is a preorder (reflexive and transitive), and $\sim\, = \,\preceq \cap \succeq$ is an equivalence relation.

Lemma 2.3.23.

i) $\mathcal{C} \subseteq \mathcal{C}' \Longrightarrow \mathcal{C} \preceq \mathcal{C}'$.

ii) If $[] \in \mathcal{C}$ then every weakly normalizing term is $\mathcal{C}$ -usable, i.e. $\mathcal{NF}^{=} \subseteq S(\mathcal{C})$.

iii) If $\mathcal{C}$ is closed under composition, then every $C[] \in \mathcal{C}$ is $U(\mathcal{C})$-strict, i.e. $\mathcal{C} \subseteq \mathcal{C}_{U(\mathcal{C})}^{U(\mathcal{C})}$.

Proof

i) From $\mathcal{C} \subseteq \mathcal{C}'$ it follows that $\mathcal{C}$-usable terms are $\mathcal{C}'$-usable as well.

ii) by definition.

iii) Let $M \in U(\mathcal{C})$ and $C[] \in \mathcal{C}$ and suppose that $C[M] \in U(\mathcal{C})$. Then by definition $\exists D[] \in \mathcal{C}$ such that $D[C[M]]$ has a normal form. But by assumption $D[C[]] \in \mathcal{C}$, hence M is $\mathcal{C}$-usable, contradiction.

□

Definition 2.3.24. We will say, that a set $\mathcal{C}$ of contexts is a **(context) monoid** if it is indeed a monoid with context composition and the empty context, i.e. if

1. $[] \in \mathcal{C}$ and
2. $\mathcal{C}$ is closed under composition of contexts

Lemma 2.3.25. Let $\mathcal{C}$ be a context monoid. Then $\mathcal{C} \sim \mathcal{C}_{U(\mathcal{C})}^{U(\mathcal{C})}$

Proof
"$\preceq$": By Lemma 2.3.23 iii) and i).
"$\succeq$": We have to show that $S(\mathcal{C}) \supseteq S(\mathcal{C}_{U(\mathcal{C})}^{U(\mathcal{C})})$.
Suppose indirectly that $M \in S(\mathcal{C}_{U(\mathcal{C})}^{U(\mathcal{C})}) \cap U(\mathcal{C})$. Then there exists a context $C[] \in \mathcal{C}_{U(\mathcal{C})}^{U(\mathcal{C})}$ such that $C[M]$ has a normal form. But then again, by $U(\mathcal{C})$-strictness of $C[]$, $C[M] \in U(\mathcal{C})$, which contradicts Lemma 2.3.23 ii). □

Our previous results have shown that the set $\mathcal{K}$ of K-strict contexts, the set $\mathcal{C}^p$ of separating contexts and the set $\mathcal{C}_{\mathcal{A}}^{\mathcal{A}}$ of $\mathcal{A}$-strict contexts for every set $\mathcal{A}$ of terms all form monoids, and in the $\lambda\mu\mathbf{E}$ -calculus (by Corollary 2.3.19 and Lemma 2.3.25) we have:

$$\mathcal{C}^p \sim \mathcal{K} \sim \mathcal{C}_{\mathcal{U}}^{\mathcal{U}}$$

Thus we have proven that all of our approaches of this section give rise to the same definition of *meaningful* and *meaningless* terms, and so in this respect are equivalent. But of course they are very different in their definitions. To understand these differences, let us investigate the structure of the set of context monoids.

Definition 2.3.26. Let $\mathbb{M}$ denote the set of context monoids.

Proposition 2.3.27. $(\mathbb{M}, \subseteq)$ is a complete algebraic lattice, with $\perp = \{[]\}$, $\top = \mathfrak{C}$ and $< \bigcup_{i \in I} \mathcal{C}_i >$ as supremum ($< \mathcal{G} >$ denotes the monoid generated by $\mathcal{G}$). A monoid $\mathcal{C} \in \mathbb{M}$ is compact iff it is finitely generated.

Proof
$(\mathbb{M}, \subseteq)$ is of course a lattice, with $\wedge = \cap$ and $\mathcal{C}_1 \vee \mathcal{C}_2 = < \mathcal{C}_1 \cup \mathcal{C}_2 >$. $\{[]\}$ and $\mathfrak{C}$ are the trivial monoids, and they are the bottom and top element of $\mathbb{M}$ respectively.

$\mathbb{M}$ is also complete, as the $\vee$ operation can be extended to arbitrary subsets of $\mathbb{M}$ as $\bigvee_{i\in I}\mathcal{C}_i =< \bigcup_{i\in I}\mathcal{C}_i >$ It is obvious that a monoid $\mathcal{C}$ is compact if it is finitely generated, since in any covering $\bigvee_i \mathcal{C}_i$ of $\mathcal{C}$ there are monoids $\mathcal{C}_{i_1},\ldots\mathcal{C}_{i_n}$ covering the generating elements of $\mathcal{C}$. To prove the converse, just take a monoid generated by the independent elements $C_1[], C_2[],\ldots$ and cover it with $\{\bigvee_{i=1}^n < C_i[] > : n \in \mathbb{N}\}$. It is then clear that $\mathcal{C}$ can not be covered with only a finite number these monoids. This construction also gives a covering by compact sub-monoids, proving that $\mathbb{M}$ is indeed algebraic. □

Remark 2.3.28. By the same argument, the same holds for the substructures of any algebraic structure, provided that the intersection of arbitrary substructures is again a substructure. This is usually the case, as closure properties are inherited through intersection. This might explain the name "algebraic lattice".

Note that $(P(\Lambda),\subseteq)$ is also a complete algebraic lattice (in which exactly the finite subsets of terms are compact). Moreover it is a Boole algebra, hence by duality so is $(P(\Lambda),\supseteq)$ So S and U are maps between complete algebraic lattices.

Proposition 2.3.29. ($\subseteq$-continuity)
$S : (\mathbb{M},\subseteq) \to (P(\Lambda),\subseteq)$, and $U : (\mathbb{M},\subseteq) \to (P(\Lambda),\supseteq)$ are continuous maps of complete lattices.

Proof
Observe that by definition $S(\mathcal{C}) = \bigcup_{C[]\in\mathcal{C}} S(< C[] >)$, for every monoid $\mathcal{C}$. Then S is continuous, since $\mathbb{M}$ is algebraic.
By duality U is continuous as well when considered as a map into the dual lattice. □

Definition 2.3.30. Let

- $T : P(\Lambda) \to \mathbb{M}$ be defined as $T(\mathcal{A}) = \mathcal{C}_{\Lambda\setminus\mathcal{A}}^{\Lambda\setminus\mathcal{A}}$
- $\mathbb{S} = \{S(\mathcal{C}) : \mathcal{C} \in \mathbb{M}\}$
- $\mathbb{C} = \{TS(\mathcal{C}) = \mathcal{C}_{U(\mathcal{C})}^{U(\mathcal{C})} : \mathcal{C} \in \mathbb{M}\}$

Lemma 2.3.31. (Strict $\prec$-monotonicity)
$S : (\mathbb{M},\prec) \to (\mathbb{S},\subset)$, and $T : (\mathbb{S},\subset) \to (\mathbb{M},\prec)$ are strictly monotonic maps of strict partial orders.

Proof
S is continuous and strictly monotonic by definition of $\prec$, and T is strictly monotonic by Lemma 2.3.25. □

Lemma 2.3.32.

i) $\forall\mathcal{C} \in \mathbb{M} : TSTS(\mathcal{C}) = TS(\mathcal{C})$

ii) $TS : \mathbb{M} \to \mathbb{C}$ is a surjective projection.

iii) $\forall\mathcal{C} \in \mathbb{M} : \mathcal{C} \in \mathbb{C} \iff TS(\mathcal{C}) = \mathcal{C}$

Proof

i) Immediate from Lemma 2.3.25.

ii) Surjective by definition of $\mathbb{C}$, projection by item i).

iii) If $\mathcal{C} \in \mathbb{C}$, then $\mathcal{C} = TS(\mathcal{C}')$ for some monoid $\mathcal{C}'$, and by item i) $\mathcal{C} = TS(\mathcal{C}') = TSTS(\mathcal{C}')$, which is just $TS(\mathcal{C})$. The other direction is trivial.

$\square$

Proposition 2.3.33. The following partial orders are isomorphic.

$$(\mathbb{C}, \preceq) \cong (\mathbb{M}/\sim, \preceq) \cong (\mathbb{S}, \subseteq)$$

Proof
$(\mathbb{S}, \subseteq)$ is of course a partial order, and $(\mathbb{M}/\sim, \preceq)$ is by definition isomorphic to it via S.
Lemma 2.3.25 says that all $\sim$ equivalence classes of $\mathbb{M}$ are represented in $\mathbb{C}$, and by definition this representation is unique. Thus $(\mathbb{C}, \preceq)$ is indeed a partial order isomorphic to $(\mathbb{M}/\sim, \preceq)$ via TS.
$\square$

Remark 2.3.34. The composition of these isomorphisms gives STS as another isomorphism between $(\mathbb{M}/\sim, \preceq)$ and $(\mathbb{S}, \subseteq)$. From Lemma 2.3.25 we know that $STS = S$. Note that by Lemma 2.3.23 iii) $\mathbb{C}$ contains the $\subseteq$-maximal element of every $\sim$ equivalence class.

Corollary 2.3.35. $(\mathbb{C}, \prec)$ *is a retract of* $(\mathbb{M}, \prec)$ *by the retraction map* TS*, and is isomorphic to* $(\mathbb{S}, \subset)$ *in the category of strict partial orders.*

So $S|_{\mathbb{C}}$ and $T|_{\mathbb{S}}$ are bijections and inverses of each other; $T|_{\mathbb{S}}$ selects the $\subseteq$-maximal monoids of every $\sim$ equivalence class of $\mathbb{M}$; and $\mathbb{C}$ consists of exactly these representatives. By Lemma 2.3.23 i) $(\mathbb{C}, \preceq)$ is a refinement of $(\mathbb{C}, \subseteq)$, which is also a partially ordered set; and $S|_{\mathbb{C}}$ as a map from $(\mathbb{C}, \subseteq)$ to $(\mathbb{S}, \subseteq)$ is strictly monotonic.

Remark 2.3.36. Note that if $\mathcal{C} \in \mathbb{M}$ is such that $S(\mathcal{C})$ contains one generic term, then necessarily $S(\mathcal{C}) = \Lambda$. By Theorem 2.3.18 this means that $(\mathbb{S}, \subseteq)$ contains a unique second maximal element: the set of non-generic or separable terms obtained as $S(\mathcal{C}^p)$. (Its maximal element is of course $\Lambda = S(\mathfrak{C})$, and the minimal is $\mathcal{NF}^= = S(\{[]\})$. Consequently $\mathfrak{C}$ is the maximal, $TS(\mathcal{C}^p)$ is the second maximal, and $TS(\{[]\})$ is the minimal element of $(\mathbb{C}, \preceq)$ and by refinement of $(\mathbb{C}, \subseteq)$ as well. Note that some rewrite systems may have no generic terms, in which case the maximal and "second" maximal elements of $\mathbb{S}$ (and $\mathbb{C}$) coincide.

Taking a look at our three monoids again in $\lambda\mu\mathbf{E}$, we see, that $\mathcal{C}_{\mathcal{U}}^{\mathcal{U}} = TS(\mathcal{K})$ is in the retract $\mathbb{C}$; $\mathcal{K}$ is infinitely generated by the contexts: $f([]), ([])M, \lambda x.([]), \mu x.([])$ for every constant f, term M, and variable x; and since they are both equivalent to $\mathcal{C}^p$, the representative $TS(\mathcal{K})$ of their equivalence class is the maximal non-trivial context monoid in $\mathbb{C}$ and $\mathcal{S} = S(\mathcal{K})$ is the maximal non-trivial set of terms in $\mathbb{S}$.

Chapter 3

Term Rewriting Systems

The first attempt to generalize the notion of solvability to (infinitary) term rewriting systems (TRSs) was made in [AKKSV94]. The leading idea of that paper was the construction of a Böhm$_{\mathcal{U}}$ tree model by identifying a set $\mathcal{U}$ of terms satisfying certain basic properties. Unfortunately their axioms were not sufficient (even for orthogonal TRSs), as pointed out in [KOV99], to ensure the uniqueness of Böhm$_{\mathcal{U}}$ trees (i.e. Böhm$_{\mathcal{U}}$ normal forms) of Böhm terms. The problem was that aside of their axioms, the finite **CR** property of the Böhm$_{\mathcal{U}}$ -reduction is required for uniqueness of Böhm$_{\mathcal{U}}$ trees, and orthogonality, hence finite **CR** property of only the original reduction is in itself not enough.

Proceedings of this work was presented in [KOV99]. This paper is concerned with left linear TRSs and λ-calculus both in the finite and infinitary setting, for authors argue that it does not make much sense to talk about meaningless terms in other than left linear systems, because every non-left linear rule can "effectively use" (e.g. in the sense of 2.3.20) any term if it occurs twice appropriately in a redex of that rule.

The latter work also corrected the above mentioned error of the former by the introduction of two new axioms. Interestingly one of the new axioms turned out to be strong enough to prove, together with even weaker versions of the two axioms of [AKKSV94], genericity and consistent identifiability of a set $\mathcal{U}$ of terms satisfying them, without the use of Böhm trees.

Hence, giving less attention to Böhm$_{\mathcal{U}}$ trees, the authors of [KOV99] take a slightly different approach, analyzing reductions, tracking meaningful and meaningless subterms. Their axioms are given to make this tracking easier, but also have a good intuitive meaning. With the intention to identify meaningless terms, and the intuition in mind that meaninglessness is invariant under reductions, the basic idea is to classify reduction steps into two classes: those that occur inside a meaningless subterm, and those which do not.

Remark 3.0.1. In the sequel by reduction we will always refer to strongly converging reduction, but if we want to make it explicit, then we will use the notation **s.c.r.**.

3.1 Axioms of Meaninglessness

Remember that in λK-calculus a term is solvable iff it has a head normal form, i.e. it can be reduced (via head reductions: reducing head redexes) to a term which contains no head redex. A head redex is a redex containing the head symbol of the term, which is the root symbol of the body of the term, i.e. the remainder of the term after removing the leading abstractions. By capturing this idea we can generalize (weak) head normal forms as follows:

Definition 3.1.1. (Root stable terms; cf. [AKKSV94] Definition 3.)
A term is **root stable** if it can not be reduced to a redex. A term **has a root stable form** if it can be reduced to a root stable term, otherwise it is **root active**. $\mathcal{R}$ denotes the set of root active terms.

Remark 3.1.2. From the definition it follows that the root symbol of a root stable term cannot change with further reductions. Note that in this respect root stable terms are rather a generalizations of the weak head normal forms.

Definition 3.1.3. The axioms of [AKKSV94] are

Axiom A1. $\mathcal{U}$ and its complement are closed under s.c.r. ($\mathcal{U}$,$\overline{\mathcal{U}} \vDash s.c.r.$)

Axiom A2. $\mathcal{R} \subseteq \mathcal{U}$.

Before posting the axioms of [KOV99], let us give some preliminary definitions (also taken from [KOV99]). Our first term relation is intuitively in connection with prefix ordering, while the second one is related to identification of a set of terms.

Definition 3.1.4. Let $\mathcal{U}$ be a set of terms, and define

1. $s \overset{\mathcal{U}}{\to} t$ iff t can be obtained from s by replacing some (possibly zero) disjoint subterms belonging to $\mathcal{U}$ with arbitrary terms.
2. $s \overset{\mathcal{U}}{\leftrightarrow} t$ iff t can be obtained from s by replacing some (possibly zero) disjoint subterms belonging to $\mathcal{U}$ with arbitrary terms of $\mathcal{U}$.
3. $\overset{\mathcal{U}}{=}$ is the transitive closure of $\overset{\mathcal{U}}{\leftrightarrow}$.
4. $s \overset{in\mathcal{U}}{\longrightarrow} t$ iff $s \to t$ and the reduced redex is contained in a subterm s' of s with $s' \in \mathcal{U}$.
5. $s \overset{out\mathcal{U}}{\longrightarrow} t$ iff $s \to t$ and not $s \overset{in\mathcal{U}}{\longrightarrow} t$.

Remark 3.1.5. Sometimes will will use a subscript $t \overset{\mathcal{U}}{\to}_A s$ or $t \overset{\mathcal{U}}{\leftrightarrow}_A s$ to denote that replacements are made at the positions in the set A.

Definition 3.1.6. (Hypercollapsing terms; taken from [KOV99] Definition 2.)
A rewrite rule is collapsing if for every reduction by the rule, the reduct is a descendant of a subterm of the redex. A collapsing redex is a redex of a collapsing rule. A term is **hypercollapsing** if each of its reducts reduces to a collapsing redex.
$\mathcal{H}$ denotes the set of hypercollapsing terms.

Remark 3.1.7. Note that in a TRS a rule is collapsing iff its right hand side is a variable. Hypercollapsing terms are also considered in [AKKSV94], where it is shown that $\mathcal{H}$ satisfies axiom A1., but not A2. in general, as hypercollapsing terms are special cases of root active terms, i.e. $\mathcal{H} \subseteq \mathcal{R}$. Hypercollapsing terms are important, because they are responsible for the failure of transfinite **CR** in orthogonal transfinite rewriting systems. In [KKSV95] it is proven that every orthogonal TRS is transfinitely confluent up to $\mathcal{H}$.

Definition 3.1.8. (Overlapping subterms; taken from [KOV99] Definition 1.)
A redex $t \equiv \sigma(l)$ overlaps its subterm at position π, if π is a non-empty position of l and $l|\pi$ is not a variable.

Definition 3.1.9. The axioms of [KOV99] are

Axiom U1. (Closure)
$\mathcal{U}$ is closed under reduction.

Axiom U2. (Overlap — Orthogonality)
If t is a redex and t overlaps $s \in \mathcal{U}$, then $t \in \mathcal{U}$.

Axiom U3. (Substitution — Instantiation)
$\mathcal{U}$ is closed under substitution (i.e. instantiation).

Axiom U4. (Minimality)

(1) $\mathcal{H} \subseteq \mathcal{U}$

(2) $\mathcal{R} \subseteq \mathcal{U}$

Axiom U5. (Indiscernability — Maximality)
$\forall t, s : t \overset{\mathcal{U}}{\leftrightarrow} s \Longrightarrow (t \in \mathcal{U} \iff s \in \mathcal{U})$

Clearly, axiom U1. is the most basic, and cannot be avoided. It states that all descendants of a meaningless subterm are themselves meaningless (for λ-calculus or CRSs axiom U3. is required as well for this to hold), therefore it allows us to ignore $\xrightarrow{in\mathcal{U}}$ steps, when considering terms or reductions modulo $\mathcal{U}$.

Axiom U2. is a very powerful condition. Intuitively it ensures that $\xrightarrow{out\mathcal{U}}$ steps can not delete meaningless subterms other than by erasing rules. It also yields, that $\xrightarrow{in\mathcal{U}}$ commutes with $\xrightarrow{out\mathcal{U}}$ and that the $\to_{\mathcal{U}_\perp}$ reduction (see Definition 3.2.1) commutes with $\to_{\mathcal{U}}$ (provided $\mathcal{U}$ satisfies axiom U1. as well, and not forgetting left-linearity).

Axiom U3. is used only in connection with the λ-calculus. It plays a role whenever a rewriting system allows variable binding, and reductions involve substitution, such as combinatory reduction systems. In any CRS axioms U1. and U3. together ensure that descendants of a term in $\mathcal{U}$ are themselves in $\mathcal{U}$.

We have two different minimality conditions in axiom U4. As remarked in 3.1.7 $\mathcal{H} \subseteq \mathcal{R}$, hence U4.(2) implies U4.(1). The stronger versions, (2) is to ensure, as we will later see, the existence of Böhm$_{\mathcal{U}}$ trees, while (1) is, as mentioned above, required for transfinite confluence up to $\mathcal{U}$ of orthogonal TRSs.

Observe that if $t \overset{\mathcal{U}}{\leftrightarrow} s$ then after identifying all terms in $\mathcal{U}$, t and s will be identified as well. In fact, t and s will be identified if and only if $t \overset{\mathcal{U}}{=} s$. In the light of this remark axiom U5., indiscernability, expresses that identifying terms of $\mathcal{U}$ does not necessitate identifying them with more terms outside of $\mathcal{U}$, i.e. that $\mathcal{U}$ is closed or maximal in this respect. This intuitively explains the following equivalence:

Proposition 3.1.10. (see [KOV99] Lemma 1 and 15, [Ket02], [Ter02] chapter 12.9)
In both finitary and infinitary frameworks:
$\mathcal{U}$ satisfies indiscernability if and only if $\overset{\mathcal{U}}{\leftrightarrow}$ is transitive.

Indiscernability has many interesting characterizations, which we shall later investigate in connection with genericity in section 3.1.2.

3.1.1 Genericity and Relative Consistency

Let us now list some of the basic results. Proofs can be found in [KOV99] (Lemma 2.3 is erroneous) and [Ter02] chapter 12.9 (corrects the former). The following result from Lemma 3.1.11 to Theorem 3.1.20 apply to both finitary and infinitary left-linear TRSs.

Lemma 3.1.11. Let $\mathcal{U} \vDash U2.$. Then

1. If $s \overset{\mathcal{U}}{\leftarrow} s' \overset{out\mathcal{U}}{\twoheadrightarrow} t$ then $\exists t' : s \twoheadrightarrow t' \overset{\mathcal{U}}{\leftarrow} t$.

2. If $s \overset{\mathcal{U}}{\leftrightarrow} s' \overset{out\mathcal{U}}{\twoheadrightarrow} t$ then $\exists t' : s \twoheadrightarrow t' \overset{\mathcal{U}}{\leftrightarrow} t$.

Lemma 3.1.12. Let $\mathcal{U} \vDash U1., U2.$. Then
if $s \overset{\mathcal{U}}{=} s' \overset{out\mathcal{U}}{\twoheadrightarrow} t$ then $\exists t' : s \twoheadrightarrow t' \overset{\mathcal{U}}{=} t$.

Definition 3.1.13. (Totally meaningful terms.)
A term is **totally meaningful (with respect to $\mathcal{U}$)** if none of the subterms of none of its reducts is in $\mathcal{U}$. The set of terms totally meaningful with respect to $\mathcal{U}$ is denoted by $\mathcal{TM}(\mathcal{U})$.

Lemma 3.1.14. Let $\mathcal{U} \vDash U1., U2.$. Then
if $s \overset{in\mathcal{U}}{\twoheadrightarrow} s' \overset{out\mathcal{U}}{\twoheadrightarrow} t$ then $\exists t' : s \overset{out\mathcal{U}}{\twoheadrightarrow} t' \overset{in\mathcal{U}}{\twoheadrightarrow} t$.

Corollary 3.1.15. *Let* $\mathcal{U} \vDash U1., U2..$ *Then*
if $s \twoheadrightarrow t$ *then* $\exists t' : s \overset{out\mathcal{U}}{\twoheadrightarrow} t' \overset{in\mathcal{U}}{\twoheadrightarrow} t$.

Corollary 3.1.16. *Let* $\mathcal{U} \vDash U1., U2..$ *Then*
if $s \twoheadrightarrow t$ *and* t *totally meaningful, then* $s \overset{out\mathcal{U}}{\twoheadrightarrow} t$.

Lemma 3.1.17. Let $\mathcal{U} \vDash U1., U2..$ Then
if $s \overset{\mathcal{U}}{=} s' \overset{out\mathcal{U}}{\twoheadrightarrow} t$ and t is totally meaningful, then $s \twoheadrightarrow t$.

Our results thus far are already sufficient to prove the following generalization of the genericity lemma of unsolvable terms of λ-calculus.

Definition 3.1.18. (Generic set.)
A set $\mathcal{U}$ is **generic**[1] if for every term $u \in \mathcal{U}$ and every context $C[]$
$C[u] \twoheadrightarrow t \in \mathcal{TM}(\mathcal{U}) \Longrightarrow \forall s : C[s] \twoheadrightarrow t$

Remark 3.1.19. In the definition $\twoheadrightarrow$ refers to a finite or strongly converging transfinite reduction sequence, depending on the framework.

Theorem 3.1.20. *(Genericity Lemma)*
In a left-linear (finite or transfinite) term rewriting system every set $\mathcal{U}$ *satisfying axioms U1. and U2. is generic.*

Proof
Let $u \in \mathcal{U}$, $t \in \mathcal{TM}(\mathcal{U})$, and $C[u] \twoheadrightarrow t$. Then by Lemma 3.1.11 $C[u] \overset{out\mathcal{U}}{\twoheadrightarrow} t$. For arbitrary term s: $C[u] \overset{\mathcal{U}}{\rightarrow} C[s]$, hence by Corollary 3.1.16 $C[s] \overset{out\mathcal{U}}{\twoheadrightarrow} t' \overset{\mathcal{U}}{\leftarrow} t$. But since t is totally meaningful with respect to $\mathcal{U}$, we have $t' \equiv t$. □

Another important question is the one regarding the consistency of the contracted system obtained by identifying a set $\mathcal{U}$ of terms. Intuitively, if a set $\mathcal{U}$ of terms have the same *meaning* then they are "the same", i.e. should be consistently identifiable. Formally this means that there is a model of the theory associated with the rewriting system in which the terms belonging to $\mathcal{U}$ represent the same object.

Definition 3.1.21. (Consistency and relative consistency with respect to $\mathcal{U}$)
A TRS is **consistent** if the associated theory is consistent, i.e. if there exists two inconvertible normal forms, i.e. not related by $(\twoheadrightarrow \cup \twoheadleftarrow)^*$.
A TRS is **consistent with respect to** $\mathcal{U}$ if any two totally meaningful terms related by $(\twoheadrightarrow \cup \overset{\mathcal{U}}{=} \cup \twoheadleftarrow)^*$ are also related by $(\twoheadrightarrow \cup \twoheadleftarrow)^*$ in the original system.

Regarding the consistency of identifying elements of $\mathcal{U}$ and confluence modulo $\mathcal{U}$ in orthogonal TRSs, [KOV99] proves the following results.

Definition 3.1.22. (Confluence up to and modulo $\mathcal{U}$)
A TRS is **confluent up to** $\mathcal{U}$ if for every pair $s_1 \twoheadleftarrow s \twoheadrightarrow s_2$ of diverging reductions $\exists t_1, t_2 : s_1 \twoheadrightarrow t_1 \overset{\mathcal{U}}{=} t_2 \twoheadleftarrow s_2$, i.e. $\twoheadleftarrow \cdot \twoheadrightarrow \subseteq \twoheadrightarrow \cdot \overset{\mathcal{U}}{=} \cdot \twoheadleftarrow$.
A TRS is **confluent modulo** $\mathcal{U}$ if moreover $\twoheadleftarrow \cdot \overset{\mathcal{U}}{=} \cdot \twoheadrightarrow \subseteq \twoheadrightarrow \cdot \overset{\mathcal{U}}{=} \cdot \twoheadleftarrow$.

Lemma 3.1.23. In any left-linear TRS, if $\mathcal{U}$ satisfies axioms U1. and U2. and the system is confluent up to $\mathcal{U}$, then it is also relatively consistent with respect to $\mathcal{U}$
.

[1] Compare this definition with Definition 2.3.16

Theorem 3.1.24. *(Confluence modulo $\mathcal{U}$, and relative consistency)*
An orthogonal term rewriting system is confluent modulo $\mathcal{U}$, hence relatively consistent with respect to $\mathcal{U}$, if $\mathcal{U}$ satisfies axioms U1. and U2. (and U4.(1) and U5. in the transfinite case).

In the transfinite case however, [KKSV95] proves a stronger theorem.

Theorem 3.1.25. *(Consistency up to $\mathcal{H}$)*
Transfinite orthogonal TRSs are confluent up to $\mathcal{H}$.

Corollary 3.1.26.
Transfinite orthogonal TRSs are confluent up to any set $\mathcal{U}$ satisfying axiom U4.(1), hence relatively consistent with respect to $\mathcal{U}$, if in addition $\mathcal{U}$ satisfies axioms U1. and U2. as well.

As we have previously remarked, the consistency of identifying a set $\mathcal{U}$ of terms is essentially a model theoretic question. Böhm trees, under sufficient conditions on $\mathcal{U}$, provide one way to construct a model in which elements of $\mathcal{U}$ are identified. In section 3.2 we will revisit the problem of genericity and consistent identifiability of a set of terms using Böhm terms, but first let us present some alternative characterizations of our axioms.

3.1.2 Genericity and Indiscernability

In this section we will look at various levels of genericity and investigate the connection between them and indiscernability.

Remember that in 3.1.4 the relations $\stackrel{\mathcal{U}}{\to}$ and $\stackrel{\mathcal{U}}{\leftrightarrow}$ were defined to allow any number of replacements of disjoint subterms belonging to $\mathcal{U}$ by other terms in $\mathcal{U}$ and by arbitrary terms, respectively. The following definition restricts the number of replacements to one.

Definition 3.1.27.

i) $t \stackrel{\mathcal{U}}{\to}_1 s$ iff s can be obtained from t by replacing at most one subterm belonging to $\mathcal{U}$ with an arbitrary term.

ii) $t \stackrel{\mathcal{U}}{\leftrightarrow}_1 s$ iff s can be obtained from t by replacing at most one subterm belonging to $\mathcal{U}$ with an arbitrary term of $\mathcal{U}$.

The following closure properties are trivial. (See 1.3 for the notations.)

Proposition 3.1.28.

$$\begin{array}{l}
\stackrel{\mathcal{U}}{\to}_1,\ \stackrel{\mathcal{U}}{\leftrightarrow}_1\ \models\ []\\
\stackrel{\mathcal{U}}{\to},\ \stackrel{\mathcal{U}}{\leftrightarrow}\ \models\ [\ldots]\\
\stackrel{\mathcal{U}}{\to}_1 \subseteq \stackrel{\mathcal{U}}{\to} \subseteq (\stackrel{\mathcal{U}}{\to}_1)^*\\
\stackrel{\mathcal{U}}{\leftrightarrow}_1 \subseteq \stackrel{\mathcal{U}}{\leftrightarrow} \subseteq (\stackrel{\mathcal{U}}{\leftrightarrow}_1)^*
\end{array}$$

Corollary 3.1.29.

$$\begin{array}{lcl}
\mathcal{U} \models \stackrel{\mathcal{U}}{\to} & \iff & \mathcal{U} \models \stackrel{\mathcal{U}}{\to}_1\\
\mathcal{U} \models \stackrel{\mathcal{U}}{\leftrightarrow} & \iff & \mathcal{U} \models \stackrel{\mathcal{U}}{\leftrightarrow}_1
\end{array}$$

Remember that $\mathcal{TM}(\mathcal{U})$ denotes the set of totally meaningful terms with respect to $\mathcal{U}$ (Definition 3.1.13). In addition we introduce some notations, and give their basic properties below.

Definition 3.1.30.

$$\begin{array}{rcl} \mathcal{M}^{\leftarrow}(\mathcal{U}) & = & \{t \mid t \twoheadrightarrow s \in \mathcal{TM}(\mathcal{U})\} \\ \mathcal{M}(\mathcal{U}) & = & \{t \mid t = s \in \mathcal{TM}(\mathcal{U})\} \\ \mathcal{L}^{\leftarrow}(\mathcal{U}) & = & \{t \mid t \twoheadrightarrow s \in \mathcal{U}\} \\ \mathcal{L}(\mathcal{U}) & = & \{t \mid t = s \in \mathcal{U}\} \end{array}$$

Proposition 3.1.31.

i) $\mathcal{TM}(\mathcal{U}) \cap \mathcal{U} = \emptyset$

ii) $\mathcal{TM}(\mathcal{U}) \subseteq \mathcal{M}^{\leftarrow}(\mathcal{U}) \subseteq \mathcal{M}(\mathcal{U})$

iii) $\mathcal{M}^{\leftarrow}(\mathcal{U}) \vDash \leftarrow$ and if $\mathcal{U} \vDash \rightarrow$ then $\mathcal{M}^{\leftarrow}(\mathcal{U}) \subseteq \overline{\mathcal{U}}$

iv) $\mathcal{M}(\mathcal{U}) \vDash =$ and if $\rightarrow \vDash$ **CR** then $\mathcal{M}^{\leftarrow}(\mathcal{U}) = \mathcal{M}(\mathcal{U})$

v) $\mathcal{U} \subseteq \mathcal{L}^{\leftarrow}(\mathcal{U}) \subseteq \mathcal{L}(\mathcal{U})$

vi) $\mathcal{L}^{\leftarrow}(\mathcal{U}) \vDash \leftarrow$ and if $\mathcal{U} \vDash \leftarrow$ then $\mathcal{L}^{\leftarrow}(\mathcal{U}) = \mathcal{U}$

vii) $\mathcal{L}(\mathcal{U}) \vDash =$ and if $\mathcal{U} \vDash =$ then $\mathcal{L}(\mathcal{U}) = \mathcal{U}$

viii) if $\rightarrow \vDash$ **CR** and $\mathcal{U} \vDash \rightarrow$ then $\mathcal{L}^{\leftarrow}(\mathcal{U}) = \mathcal{L}(\mathcal{U})$

Proof
Easy. □

Definition 3.1.32. A set $\mathcal{U}$ of terms is

1. (Repeating Definition 3.1.18)
   generic ($\mathcal{U} \vDash$ *gen.*) iff whenever $u \in \mathcal{U}$ and $C[u] \twoheadrightarrow t$ where t is totally meaningful with respect to $\mathcal{U}$, then $C[s] \twoheadrightarrow t$ for every term s.
2. **weakly generic** ($\mathcal{U} \vDash$ *weakgen.*) iff whenever $t \in \mathcal{U}$ and $C[t]$ reduces to a totally meaningful term with respect to $\mathcal{U}$, then so does $C[s]$ for every term s.
3. **widely generic** ($\mathcal{U} \vDash$ *widegen.*) iff whenever $t \in \mathcal{U}$ and $C[t] \notin \mathcal{U}$ then $C[s] \notin \mathcal{U}$ for every term s.
4. **indiscernible** ($\mathcal{U} \vDash$ *ind.*) iff whenever $t \overset{\mathcal{U}}{\leftrightarrow} s$ then $t \in \mathcal{U} \iff s \in \mathcal{U}$.

Remark 3.1.33. Weak genericity trivially follows from genericity. However, this is not the case with wide genericity, as can be seen intuitively because genericity does not say anything about those terms which do not reduce to a totally meaningful term. Of course if we assume that $\overline{\mathcal{U}} = \mathcal{M}^{\leftarrow}(\mathcal{U})$ (see next proposition) and $\mathcal{U} \vDash \rightarrow$ then genericity implies wide genericity, but this assumption is too strong as it does not apply to solvability (e.g. $x\mathbf{\Omega} \notin \mathcal{U}$ but it does not reduce to a totally meaningful term with respect to $\mathcal{U}$). Below we will give a weaker side-condition using strict contexts and usability as established in section 2.3.3.

Proposition 3.1.34.

1. $\mathcal{U} \vDash$ *weakgen.* $\iff \mathcal{M}^{\leftarrow}(\mathcal{U}) \vDash \overset{\mathcal{U}}{\rightarrow}$
2. $\mathcal{U} \vDash$ *widegen.* $\iff \overline{\mathcal{U}} \vDash \overset{\mathcal{U}}{\rightarrow}$
3. $\mathcal{U} \vDash$ *ind.* $\iff \mathcal{U} \vDash \overset{\mathcal{U}}{\leftrightarrow}$

Proof

1. $\mathcal{U} \vDash \mathit{weakgen.} \iff$ (per.def.)
   $[\forall t \in \mathcal{U}, s, C[] : C[t] \in \mathcal{M}^{\leftarrow}(\mathcal{U}) \Longrightarrow C[s] \in \mathcal{M}^{\leftarrow}(\mathcal{U})] \iff$ (per.def.)
   $\mathcal{M}^{\leftarrow}(\mathcal{U}) \vDash \stackrel{\mathcal{U}}{\to}_1 \iff$ (by Corollary 3.1.29)
   $\mathcal{M}^{\leftarrow}(\mathcal{U}) \vDash \stackrel{\mathcal{U}}{\to}$

2. $\mathcal{U} \vDash \mathit{widegen.} \iff$ (per.def.)
   $[\forall t \in \mathcal{U}, s, C[] : C[t] \in \overline{\mathcal{U}} \Longrightarrow C[s] \in \overline{\mathcal{U}}] \iff$ (per.def.)
   $\overline{\mathcal{U}} \vDash \stackrel{\mathcal{U}}{\to}_1 \iff$ (by Corollary 3.1.29)
   $\overline{\mathcal{U}} \vDash \stackrel{\mathcal{U}}{\to}$

3. $\mathcal{U} \vDash \mathit{ind.} \iff$ (per.def.)
   $[t \stackrel{\mathcal{U}}{\leftrightarrow} s \Longrightarrow (t \in \mathcal{U} \iff s \in \mathcal{U})] \iff$ ($\stackrel{\mathcal{U}}{\leftrightarrow} \vDash \mathit{symm.}$)
   $[t \stackrel{\mathcal{U}}{\leftrightarrow} s \Longrightarrow (t \in \mathcal{U} \Longrightarrow s \in \mathcal{U})] \iff$ (logically)
   $[t \in \mathcal{U} \wedge t \stackrel{\mathcal{U}}{\leftrightarrow} s \Longrightarrow s \in \mathcal{U}] \iff$ (per.def.)
   $\mathcal{U} \vDash \stackrel{\mathcal{U}}{\leftrightarrow}$

□

Corollary 3.1.35. *(Indiscernability follows from wide genericity.)*
$\mathcal{U} \vDash \mathit{widegen.} \Longrightarrow \mathcal{U} \vDash \mathit{ind.}$

Proof
$\mathcal{U} \vDash \mathit{widegen.} \iff \overline{\mathcal{U}} \vDash \stackrel{\mathcal{U}}{\to} \Longrightarrow \overline{\mathcal{U}} \vDash \stackrel{\mathcal{U}}{\leftrightarrow} \iff \mathcal{U} \vDash \stackrel{\mathcal{U}}{\leftrightarrow} \iff \mathcal{U} \vDash \mathit{ind.}$ □

Remark 3.1.36. In the above proof we used the simple fact that $\stackrel{\mathcal{U}}{\leftrightarrow}$ is a sub-relation of $\stackrel{\mathcal{U}}{\to}$ and that it is symmetric. Later we will show that if $\stackrel{\mathcal{U}}{\leftrightarrow}$ is the maximal such sub-relation, then the implication can be reversed, that is indiscernability implies wide genericity.

Another characterization of indiscernability and wide genericity is given by the following lemma.

Lemma 3.1.37.

i) $\stackrel{\mathcal{U}}{\leftrightarrow}$ is transitive $\iff$ $\mathcal{U}$ is indiscernible

ii) $\stackrel{\mathcal{U}}{\to}$ is transitive $\iff$ $\mathcal{U}$ is widely generic

Proof

i) In [KOV99] it is shown that in the finitary case the transitivity of $\stackrel{\mathcal{U}}{\leftrightarrow}$ is equivalent to the indiscernability of $\mathcal{U}$. That the equivalence also holds in the infinitary case is proven by [Ket02] ([KOV99] only proved one direction). (The proof of ii) below follows the same line of reasoning.)

ii) $\Longleftarrow$: let $t \stackrel{\mathcal{U}}{\to}_A s \stackrel{\mathcal{U}}{\to}_B r$ (i.e. the replacements are made in the positions in A and B). We claim that $t \stackrel{\mathcal{U}}{\to}_C r$ where C consists of the minimal elements of $A \cup B$. Let $\alpha \in A$ and $\beta \in B$ be two positions of replacement. If $\alpha \leq \beta$ or the two positions are disjoint, then we can do the replacements in one step in their minimal positions. If $\alpha \geq \beta$ then by wide genericity we can still do the

two replacements in one step in position β.
$\Longrightarrow$: let $t, C[s] \in \mathcal{U}$ then $C[t] \overset{\mathcal{U}}{\to} C[s] \overset{\mathcal{U}}{\to} a$ where a is an arbitrary atom (e.g. a variable) and by transitivity $C[t] \overset{\mathcal{U}}{\to} a$ which implies that $C[t] \in \mathcal{U}$ since a is an arbitrary atom.

$\square$

Corollary 3.1.38.
$\overset{\mathcal{U}}{\to}$ *trans.* $\iff$ $\mathcal{U}$ *widely generic* $\Longrightarrow$ $\mathcal{U}$ *indiscernible* $\iff$ $\overset{\mathcal{U}}{\leftrightarrow}$ *trans.*

Definition 3.1.39. Let R be any binary relation, then

i) its **symmetric closure**, denoted $SC(R)$, is the minimal symmetric relation which includes R, or equivalently $(a,b) \in SC(R) \iff (a,b) \in R \vee (b,a) \in R$

ii) its **symmetric interior**, denoted $SI(R)$, is the maximal symmetric relation included in R, or equivalently $(a,b) \in SI(R) \iff (a,b) \in R \wedge (b,a) \in R$

Lemma 3.1.40.

i) R reflexive $\iff$ $SI(R)$ reflexive

ii) R transitive $\Longrightarrow$ $SI(R)$ transitive

iii) R reflexive and transitive $\Longrightarrow$ $SI(R)$ is an equivalence relation

Proof

i) Trivial.

ii) Suppose $a \, SI(R) \, b \, SI(R) \, c$. Then by definition $a \, R \, b \, R \, c$ and $c \, R \, b \, R \, a$ both hold, so $a \, R \, c$ and $c \, R \, a$ by transitivity of R, which means by definition again that $a \, SI(R) \, c$.

iii) By i) and ii) since $SI(R)$ is by definition symmetric.

$\square$

Corollary 3.1.41. *Every reflexive transitive relation (preorder) is a partial order (because it is by definition antisymmetric) with respect to its symmetric interior as equality. Conversely the symmetric interior of every partial order coincides with equality.*

Lemma 3.1.42. $\mathcal{U} \models$ *widegen.* $\Longrightarrow SI(\overset{\mathcal{U}}{\to}) = \overset{\mathcal{U}}{\leftrightarrow}$.

Proof
$\subseteq$: Suppose $t \overset{\mathcal{U}}{\to}_A s$ and $s \overset{\mathcal{U}}{\to}_B t$ (i.e. the replacements are made in the positions in A and B respectively, where A and B are sets of disjoint positions of t and s) and let C be the set of minimal positions of $A \cup B$. Now consider $\alpha \in C \cap A$ and $\alpha \leq \beta_1, \ldots, \beta_k \in B$. We know that $s|\beta_1, \ldots, s|\beta_k \in \mathcal{U}$, $s|\alpha \equiv C[s|\beta_1, \ldots, s|\beta_k]$ and $t|\alpha \equiv C[t|\beta_1, \ldots, t|\beta_k] \in \mathcal{U}$. From wide genericity using Proposition 3.1.34 (b) we have that $s|\alpha \in \mathcal{U}$ as well, so $t|\alpha \overset{\mathcal{U}}{\leftrightarrow} s|\alpha$. The case $\alpha \in C \cap B$ is analogous and this proves $t \overset{\mathcal{U}}{\leftrightarrow}_C s$ since the minimal positions are disjoint.
$\supseteq$: $t \overset{\mathcal{U}}{\leftrightarrow} s$ trivially implies both $t \overset{\mathcal{U}}{\to} s$ and $s \overset{\mathcal{U}}{\to} t$.
$\square$

Lemma 3.1.43. $\mathcal{U} \models$ *widegen.* $\iff \mathcal{U} \models$ *ind.* $\wedge\, SI(\overset{\mathcal{U}}{\to}) = \overset{\mathcal{U}}{\leftrightarrow}$

Proof
$\Longrightarrow$: Corollary 3.1.35 and Lemma 3.1.42 prove this.
$\Longleftarrow$: Suppose $t, C[s] \in \mathcal{U}$. Then $C[t] \overset{\mathcal{U}}{\to} C[s] \wedge C[s] \overset{\mathcal{U}}{\to} C[t]$, which means (since $SI(\overset{\mathcal{U}}{\to}) = \overset{\mathcal{U}}{\leftrightarrow}$) that $C[t] \overset{\mathcal{U}}{\leftrightarrow} C[s]$ and by indiscernability of $\mathcal{U}$ we get that $C[t] \in \mathcal{U}$ as well. ☐

Lemma 3.1.44.
If $\mathcal{U} \models \to,$ *weakgen.* and $\forall t \notin \mathcal{U} : \exists C[] \in \mathcal{C}^{\mathcal{U}}_{\mathcal{U}} : C[t] \in \mathcal{M}^{\leftarrow}(\mathcal{U})$, then $\mathcal{U} \models$ *widegen.*

Proof
Let $t \in \mathcal{U}$ and $C[t] \notin \mathcal{U}$ and s arbitrary term. Then there is a context $D \in \mathcal{C}^{\mathcal{U}}_{\mathcal{U}}$ such that $D[C[t]] \in \mathcal{M}^{\leftarrow}(\mathcal{U})$. Now weak genericity implies that $D[C[s]] \in \mathcal{M}^{\leftarrow}(\mathcal{U})$ as well. Closure of $\mathcal{U}$ under rewriting ensures that $\mathcal{M}^{\leftarrow}(\mathcal{U}) \cap \mathcal{U} = \emptyset$, proving $D[C[s]] \notin \mathcal{U}$ and $C[s] \notin \mathcal{U}$ since $D[]$ is strict with respect to $\mathcal{U}$. ☐

Example 3.1.45. The set $\mathcal{U}$ of unsolvable terms in λ-calculus is widely generic.

Proof
We know that $\mathcal{U}$ is generic and closed under reduction. Let $M \notin \mathcal{U}$ that is M is solvable. Then it has a solution $\lambda\vec{x}.M\vec{N} \twoheadrightarrow \mathbf{I}$ which means that $C[M] \in \mathcal{M}^{\leftarrow}(\mathcal{U})$ where $C[] \equiv \lambda\vec{x}.[]\vec{N}$, which is a strict context. Now the result follows from Lemma 3.1.44. ☐

Remark 3.1.46. In section 3.3.3 we introduce the notion of *total* sets (Definition 3.3.14) with the help of usability, and in Proposition 3.3.15 we show that a set $\mathcal{U}$ is total if and only if it satisfies the side-conditions of Lemma 3.1.44, namely

$$\mathcal{U} \models \to \quad \textit{and} \quad \forall t \notin \mathcal{U} : \exists C[] \in \mathcal{C}^{\mathcal{U}}_{\mathcal{U}} : C[t] \in \mathcal{M}^{\leftarrow}(\mathcal{U})$$

Hence Lemma 3.1.44 can be stated as weak genericity implies wide genericity for every total set.

Theorem 3.1.47.
$\mathcal{U} \models$ *gen.* $\Longrightarrow \mathcal{U} \models$ *weakgen.*
$\mathcal{U} \models$ *weakgen. and* $\mathcal{U}$ *is total* $\Longrightarrow \mathcal{U} \models$ *widegen.*
$\mathcal{U} \models$ *widegen.* $\Longrightarrow \mathcal{U} \models$ *ind.*

Corollary 3.1.48. *If* $\mathcal{U}$ *is* $\mathcal{M}^{\leftarrow}(\mathcal{U})$*-monoidic, then if it satisfies axioms U1.,U2. (and U3. for* λ*-calculus) then it also satisfies axiom U5.*

Since genericity of $\mathcal{U}$ follows from axioms U1. and U2. (and U3. for λ-calculus – see Theorem 3.1.20), we conclude, that Axioms U1., U2. (and U3. for λ-calculus) together with $\mathcal{U}$ is $\mathcal{M}^{\leftarrow}(\mathcal{U})$-monoidic imply axiom U5. Using the following generalization of usability to arbitrary rewriting systems, there is yet another way of saying this.

Definition 3.1.49. A term s is **usable with respect to** $\mathcal{U}$ [2] if
$\exists t \in \mathcal{TM}(\mathcal{U}), C[] \in \mathcal{C}^{\mathcal{U}}_{\mathcal{U}} : C[s] \twoheadrightarrow t$, or equivalently, if
$\exists C[] \in \mathcal{C}^{\mathcal{U}}_{\mathcal{U}} : C[s] \in \mathcal{M}^{\leftarrow}(\mathcal{U})$.

[2] Same as in Definition 3.3.1 with the exception of the notion of strict context.

Theorem 3.1.50. *If every term outside of $\mathcal{U}$ is usable with respect to $\mathcal{U}$, then if it satisfies axioms U1.,U2. (and U3. for λ-calculus) then it satisfies axiom U5 as well.*

Proof
Just apply Lemma 3.1.44. □

3.2 Böhm Trees

The construction of *Böhm$_{\mathcal{U}}$ trees* associated with a set $\mathcal{U}$ is the direct analogue of that introduced by Barendregt with the help of unsolvability in λ-calculus (cf. [Bar84] ch.10.). First the signature of the TRS is extended with a constant (nullary function symbol) $\perp$ intuitively denoting the undefined. The terms of this extended system (*Böhm terms* in [KOV99]) are partially ordered by the relation $\sqsubseteq$ (*prefix*) defined by $\forall t : \perp \sqsubseteq t$ and stipulating that every function symbol is monotonic. Böhm$_{\mathcal{U}}$ trees are defined as the normal forms with respect to the $\mathcal{U}$-reduction (Böhm reduction in [KOV99]) on Böhm terms, defined as follows.

Definition 3.2.1.

- $\forall t \in \mathcal{U} : \perp \sqsubseteq_{\mathcal{U}} t$ and $\forall s, t \in \mathfrak{T}, C[] \in \mathfrak{C} : \vec{s} \sqsubseteq_{\mathcal{U}} \vec{t} \Longrightarrow C[\vec{s}] \sqsubseteq_{\mathcal{U}} C[\vec{t}]$
- $\mathcal{U}_{\perp} = \{t \mid \exists s \in \mathcal{U} : t \sqsubseteq_{\mathcal{U}} s\}$[3]
- $\to_{\mathcal{U}_{\perp}}$ **reduction** is defined by the **$\perp$-rule**: $t \to_{\mathcal{U}_{\perp}} \perp$ for all $\perp \not\equiv t \in \mathcal{U}_{\perp}$.
- **$\mathcal{U}$-reduction** (or **Böhm reduction**) is the union: $\to_{\mathcal{U}} = \to \cup \to_{\mathcal{U}_{\perp}}$.
- Normal forms with respect to the $\mathcal{U}$-reduction are called **Böhm$_{\mathcal{U}}$ trees**.

First we will give a few lemmas, identifying basic properties of Böhm reduction, prefix ordering and sets satisfying axiom A1. or U1.

Lemma 3.2.2. (First Lifting Lemma)
Let $s_0 \to^{\infty} s_{\alpha}$ be a s.c.r., and $s_0 \sqsubseteq t_0$.
Then there exists a s.c.r. $t_0 \to^{\infty} t_{\alpha}$, such that $\forall \beta \leq \alpha : s_{\beta} \sqsubseteq t_{\beta}$.

Proof
We will proceed by induction on α.
Case $\alpha = 0$ is void.
Successor case: Let $s_{\alpha} \to s_{\alpha+1}$, $s_{\alpha} \sqsubseteq t_{\alpha}$, and $s_{\alpha} \equiv D[l^{\sigma}]$, where l^{σ} is the redex contracted at step α. Then since l can not overlap any occurrences of $\perp$ (simply because $\perp$ is a new symbol disjoint from the alphabet of the TRS), there exists a context $D'[]$ and a substitution ρ such that $D[] \sqsubseteq D'[]$ and $t_{\alpha} \equiv D'[l^{\rho}]$. Clearly $l^{\sigma} \sqsubseteq l^{\rho}$, hence $r^{\sigma} \sqsubseteq r^{\rho}$ and $s_{\alpha+1} \equiv D[r^{\sigma}] \sqsubseteq D'[r^{\rho}] \equiv t_{\alpha+1}$ also hold.
Limit case: Since in the above construction reductions in s_{β} and t_{β} take place at the same positions, for every $\beta < \alpha$, and $s_0 \to^{\infty} s_{\alpha}$ is s.c.r., so is $t_0 \to^{\infty} t_{\alpha}$, and by continuity $s_{\alpha} \sqsubseteq t_{\alpha}$. □

Lemma 3.2.3. (Second Lifting Lemma)
Let $s_0 \to_{\mathcal{U}}^{\alpha} s_{\alpha}$ be a s.c.r., and $s_0 \sqsubseteq t_0$.
Then there exists a s.c.r. $t_0 \to^{\beta} t_{\beta}$, such that $\beta \leq \alpha$ and $s_{\alpha} \sqsubseteq t_{\beta}$.

[3] Note the difference between $\sqsubseteq$ and $\sqsubseteq_{\mathcal{U}}$, that in the former $\perp$ is less then or equal to every term, while in the latter, this only holds for terms in $\mathcal{U}$; but both are closed under multi-hole contexts. Consequently $\sqsubseteq_{\mathcal{U}} \subseteq \sqsubseteq$. In [AKKSV94] $\mathcal{U}_{\perp}$ is defined using $\sqsubseteq$ instead of $\sqsubseteq_{\mathcal{U}}$. Our definition is equivalent to the one given in [KOV99]. Note that with this notation $\sqsubseteq_{\mathcal{U}_{\perp}}$ is just the inverse of parallel $\to_{\mathcal{U}_{\perp}}$.

Proof
First we will construct a "semi-reduction" sequence $t'_0, \ldots, t'_\alpha$ of length α, consisting of $\to$ and $\equiv$ steps. We proceed by induction.
Base case : Let $t'_0 \equiv t_0$.
Successor case: If $s_\alpha \to s_{\alpha+1}$, then let $t'_{\alpha+1}$ be obtained as in the First Lifting Lemma, by contraction of a redex in t'_α at the same position as in s_α. If $s_\alpha \to_{\mathcal{U}_\perp} s_{\alpha+1}$, then let $t'_{\alpha+1} \equiv t'_\alpha$. It is clear, that in both cases $s_{\alpha+1} \sqsubseteq t'_{\alpha+1}$
Limit case: Take t'_α to be the limit of $(t'_\beta,\ \beta < \alpha)$. It is easy to see, that this limit exists (and then it is of course unique), because if for some $\gamma < \alpha$ there are no $t'_\beta \to t'_{\beta+1}$ steps, where $\gamma < \beta < \alpha$ then $(t'_\beta,\ \beta < \alpha)$ is quasi-constant, thus converging; otherwise the depths of the contracted redexes tend to infinity, just as in $(s_\beta \to_{\mathcal{U}} s_\beta + 1, \beta < \alpha)$.
Now $t_0, \ldots, t_\beta$ is obtained by removing multiple consecutive occurrences of terms from $t'_0, \ldots, t'_\alpha$. We have to show, that this is a reduction sequence and that it is strongly convergent.
For every $\delta \leq \beta$ let δ' be the least ordinal for which $t_\delta \equiv t'_{\delta'}$, and let $\delta^* = \bigcup_{\gamma<\delta} \gamma' \leq \delta'$. The following are easy to check:

i) If δ is a limit ordinal, then so is δ^*.

ii) If δ' is a limit ordinal, then so is δ, and $\delta' = \delta^*$

By observation ii), we know that $(\delta+1)'$ is a successor ordinal. Let δ^+ denote its predecessor. Then $t_\delta \equiv t'_{\delta'} \equiv \ldots \equiv t'_{\delta+} \to t'_{(\delta+1)'} \equiv t_{\delta+1}$. This proves that $t_0, \ldots, t_\beta$ is indeed a reduction sequence.
By observation i), we know that if $\delta \leq \beta$ is a limit ordinal, then $(\gamma', \gamma < \delta)$ converges to δ^*, which is also a limit ordinal, hence by strong convergence of $s_0 \to^\alpha_{\mathcal{U}} s_\alpha$, the depths of the contracted redexes in $t_\gamma \to t_{\gamma+1}$, being at the same position as in $s_{\gamma^+} \to s_{(\gamma+1)'}$, tend to infinity as γ approaches δ from below. This proves that $t_0 \to^\beta t_\beta$ is indeed a strongly convergent reduction sequence.
$s_\alpha \sqsubseteq t'_\alpha \equiv t_\beta$ holds by construction. □

In [KOV99] Lemma 27. the authors prove the following version of the Second Lifting Lemma, with the same line of reasoning. Note that in the case of λ-calculus, $\mathcal{U}$ is required to satisfy axiom U3.

Lemma 3.2.4. If $s \twoheadrightarrow_{\mathcal{U}} t$ then $\exists t' : s \twoheadrightarrow s' \twoheadrightarrow_{\mathcal{U}_\perp} t$.

In [KOV99] it is also proven that $\mathcal{U}_\perp$ satisfies any one of axioms U1.-U5. if and only if $\mathcal{U}$ does, respectively. This is also true for axiom A1.

Lemma 3.2.5. (Lemma 5 in [AKKSV94])
$\mathcal{U}$ is closed under finite or strongly converging transfinite reduction if and only if $\mathcal{U}_\perp$ is.

Proof
"if": trivial, since $\mathcal{U} \subseteq \mathcal{U}_\perp$, and a term in $\mathcal{U}$ can not reduce to a term in $\mathcal{U}_\perp \setminus \mathcal{U}$.
"only if": by the First Lifting Lemma. □

Corollary 3.2.6.

i) $\mathcal{U} \models U1. \iff \mathcal{U}_\perp \models U1.$

ii) $\mathcal{U} \models A1. \iff \mathcal{U}_\perp \models A1.$

Proof
Because $(\overline{\mathcal{U}_\perp})_\perp = \overline{\mathcal{U}}_\perp$ (the first complement is taken in the set of Böhm terms, the second in the original set of term), we can apply the lemma to $\overline{\mathcal{U}}$ and then to $\overline{\mathcal{U}_\perp}$. □

Corollary 3.2.7. *$\mathcal{U}$ is closed under finite or strongly converging transfinite reduction if and only if $\mathcal{U}_\perp$ is closed under finite or strongly converging transfinite Böhm reduction, respectively.*

Proof
Because $\mathcal{U}_\perp$ is by definition closed under strongly converging reductions by the $\perp$-rule. □

In the following lemma we give an equivalent characterization of the "second half" of Axiom A1.: $\overline{\mathcal{U}} \models s.c.r.$[4]

Lemma 3.2.8.

i) $\overline{\mathcal{U}}$ is closed under s.c.r. $\iff$ $(t \to^{\infty}_{\mathcal{U}} \perp \iff t \in \mathcal{U}_\perp)$.

ii) $\mathcal{U}$ is closed under s.c.r. $\implies \forall t \in \mathcal{U}_\perp : t$ has exactly one $\to_{\mathcal{U}}$ normal form, and that is $\perp$.

Proof

i) $\implies$: ([AKKSV94] Theorem 13.)
If $t \in \mathcal{U}_\perp$, then clearly $t \to_{\mathcal{U}} \perp$. If $t \to^{\infty}_{\mathcal{U}} \perp$, then the reduction has to end in a step $s \to_{\mathcal{U}} \perp$, where $t \to^{\infty}_{\mathcal{U}} s$, $s \in \mathcal{U}_\perp$. Then by assumption and Lemma 3.2.5 $t \in \mathcal{U}_\perp$ as well.
$\impliedby$:
Suppose indirectly that $\overline{\mathcal{U}}$ is not closed under s.c.r. Then there is a s.c.r. $\overline{\mathcal{U}} \ni t \to^{\infty} s \in \mathcal{U}$. But then $\overline{\mathcal{U}} \ni t \to^{\infty} s \to_{\mathcal{U}_\perp} \perp$, contradicting the assumption.

ii) Every $\to_{\mathcal{U}}$ normal form, i.e. every Böhm$_{\mathcal{U}}$ tree is either identical to $\perp$, or is not in $\mathcal{U}_\perp$. $\perp$ is a Böhm$_{\mathcal{U}}$ tree of every term in $\mathcal{U}_\perp$. Now suppose that $s \in \mathcal{U}_\perp$ has a Böhm$_{\mathcal{U}}$ tree $t \not\equiv \perp$. Then by the previous observation $t \notin \mathcal{U}_\perp$, contradicting our assumption and Corollary 3.2.7.

□

As proved in [KOV99] (page 22.), the existence of a Böhm$_{\mathcal{U}}$ tree of every Böhm term follows from just Axiom A2 (U4.(2)). Indeed, Axiom A2. implies that every Böhm term $\to_{\mathcal{U}}$-reduces (in a strongly converging, possibly infinite reduction sequence) to a Böhm term root stable with respect to $\to_{\mathcal{U}}$. Continuing this process indefinitely for subterms, subterms of subterms, etc. we get a reduction sequence strongly converging to a Böhm$_{\mathcal{U}}$ tree. A consequence of Lemma 3.2.8 is the uniqueness of Böhm trees of terms in $\mathcal{U}_\perp$. But uniqueness of Böhm trees in general do not follow from A1. and A2., not even in orthogonal TRSs. The counterexample given in [KOV99] is the following one rule TRS: $F(A) \to B$, with $\mathcal{U} = \{A\}$. It is orthogonal, $\mathcal{U}$ satisfies axioms A1. and A2, but $B \leftarrow F(A) \to_{\mathcal{U}_\perp} F(\perp)$ are two different Böhm trees of $F(A)$. The error can be fixed if we assume axiom U2. which ensures that $\to_{\mathcal{U}}$ and the $\perp$-rule commute, and axiom U5. which implies that the $\perp$-rule is transfinitely confluent. Orthogonality, or confluence of $\to$ is not required, since by axiom U4.(1) $\to$ is confluent up to $\mathcal{U}$.

Lemma 3.2.9. ([KOV99] Lemma 26.)
If $\mathcal{U}$ satisfies axiom U5., then the $\perp$-rule is transfinitely Church-Rosser, and if $s \overset{\mathcal{U}}{=} t$ then s and t have a common $\to_\perp$ reduct.

[4] $\overline{\mathcal{U}}$ denotes complement in the original set of terms in this lemma.

Theorem 3.2.10. *([KOV99] Theorem 2.)*
If $\mathcal{U}$ satisfies axiom U1., U2., U4.(2), U5. (and U3. in case of λ-calculus) then $\to_{\mathcal{U}}$ is transfinitely Church-Rosser, hence every term t has a unique Böhm$_{\mathcal{U}}$ tree, denoted by $\mathcal{B}_{\mathcal{U}}(t)$.

Once we have a (non-trivial) Böhm$_{\mathcal{U}}$ tree model, we have proved that a set of terms having the same Böhm$_{\mathcal{U}}$ tree can be consistently identified, e.g. the terms of $\mathcal{U}$ in particular. The Second Lifting Lemma then translates to a generalized genericity lemma. Note that if $\mathcal{U}$ satisfies Axiom A1. then non-triviality of the Böhm$_{\mathcal{U}}$ tree model is equivalent to the non-triviality of $\mathcal{U}$.

Definition 3.2.11. (Totally defined Böhm terms.)
A Böhm term is **totally defined (with respect to $\mathcal{U}$)** if none of its subterms is in $\mathcal{U}_\perp$.

Totally defined terms were defined in [AKKSV94] Definition 17., totally meaningful terms (see Definition 3.1.13) were introduced in [KOV99] Definition 3. Clearly, every totally meaningful term is a totally defined Böhm term, and the set of totally meaningful terms is closed under finite reduction, while totally defined terms are not. Take the reduction rule $A \to B(C)$, and $\mathcal{U} = \{C\}$ satisfying Axioms A1. and A2. for instance: A is totally defined, but $B(C)$ is not. On Böhm$_{\mathcal{U}}$ trees however, the two notions are equivalent. The following lemmas investigate these two notions in connection with strongly converging ($\to_{\mathcal{U}}$) reductions.

Lemma 3.2.12. $\overline{\mathcal{U}} \models$ *s.c.r.* $\Longrightarrow \mathcal{TM}(\mathcal{U}) \models$ *s.c.r.*

Proof
Let $t \in \mathcal{TM}(\mathcal{U})$.
We will prove by induction on the ordinal α, that $t \equiv t_0 \to^\alpha t_\alpha \Longrightarrow t_\alpha \in \mathcal{TM}(\mathcal{U})$.
Base Case: $\alpha = 0$ is true.
Successor Case: Let $t \equiv t_0 \to^\alpha t_\alpha \to t_{\alpha+1}$. Then t_α is totally meaningful by the induction hypothesis, hence by definition so are all of its finite reducts, such as $t_{\alpha+1}$.
Limit Case: Let $t \equiv t_0 \to^\alpha t_\alpha$, where α is a limit ordinal. By the induction hypothesis s_β is totally meaningful, for every $\beta < \alpha$. Suppose that $s_\alpha|\pi \in \mathcal{U}_\perp$ for some position π. By strong convergence $\exists\beta < \alpha$ such that in $s_\beta \to^\infty s_\alpha$ no reductions occur at depth less that the length of π. Hence $\mathcal{U}_\perp \not\ni s_\beta|\pi \to^\infty s_\alpha|\pi \in \mathcal{U}_\perp$, contradicting Lemma 3.2.5 and our assumption that $\overline{\mathcal{U}}$ is closed under s.c.r. □

Corollary 3.2.13.
If $\overline{\mathcal{U}}$ is closed under s.c.r. then $\mathcal{TM}(\mathcal{U})$ is closed under strongly converging Böhm reduction. In fact, for arbitrary Böhm terms t, s:
s totally meaningful, $s \to_{\mathcal{U}}^\infty t \Longrightarrow s \to^\infty t$, t totally meaningful

Proof
Let $s \equiv s_0 \to_{\mathcal{U}}^\alpha s_\alpha \equiv t$. Suppose that this reduction contains a $\to_{\mathcal{U}_\perp}$-reduction step. Let β be the least ordinal, s.t. $s_\beta \to_{\mathcal{U}_\perp} s_{\beta+1}$. Then by Lemma 3.2.12 s_β is totally meaningful, hence can not contain any $\to_{\mathcal{U}_\perp}$ redexes, contradiction. □

Totally defined terms are maximal with respect to the prefix ordering. We have the following simple consequence of the Second Lifting Lemma combined with the $\sqsubseteq$-maximality of totally defined terms:

Lemma 3.2.14. Let t, s, and r be arbitrary Böhm terms. Then
$s \to_{\mathcal{U}}^\infty t$, $s \sqsubseteq r$, t totally defined $\Longrightarrow r \to^\infty t$

Proof
Let $s \to_{\mathcal{U}}^{\infty} t$, $s \sqsubseteq r$, and t totally defined. Then by the Second Lifting Lemma $s \to^{\infty} t'$, $t \sqsubseteq t'$, which means that $t' \equiv t$, since t is maximal with respect to the prefix ordering. $\square$

The genericity of $\perp$ in the sense, that if $C[\perp] \to^{\infty} t$ and t is totally defined, then $C[s] \to^{\infty} t$ for every term s, is immediate. For other terms in $\mathcal{U}_{\perp}$, we can prove the following

Theorem 3.2.15. *(Transfinite Genericity Lemma)*
Let $\mathcal{U}$ be a set of terms such that every term has a unique Böhm$_{\mathcal{U}}$ tree.
Then if $s \in \mathcal{U}$ and $C[s] = t$ and t is totally meaningful, then $\forall r : C[r] = t$[5]

Proof
Since t is totally meaningful, it has a totally defined Böhm$_{\mathcal{U}}$ tree $\mathcal{B}_{\mathcal{U}}(t)$. $C[s] = t$ implies that $\mathcal{B}_{\mathcal{U}}(C[s]) \equiv \mathcal{B}_{\mathcal{U}}(t)$, and $\mathcal{B}_{\mathcal{U}}(C[\perp]) \equiv \mathcal{B}_{\mathcal{U}}(t)$ also, because $C[s] \to_{\mathcal{U}_{\perp}} C[\perp]$ and Böhm$_{\mathcal{U}}$ trees are unique by assumption. By Lemma 3.2.14 $C[s], C[r] \to^{\infty} \mathcal{B}_{\mathcal{U}}(t)$, proving that $C[s]$ and $C[r]$ are transfinitely convertible. $\square$

The following proposition relates the two notions of meaningfulness in TRSs.

Proposition 3.2.16. If every term t has a unique Böhm$_{\mathcal{U}}$ tree $\mathcal{B}_{\mathcal{U}}(t)$, then for every Böhm term t:
$\mathcal{B}_{\mathcal{U}}(t)$ is totally defined $\iff$ t can be reduced to a totally meaningful term.

Proof
If $\mathcal{B}_{\mathcal{U}}(t)$ is totally defined, then it is totally meaningful as well, and by Lemma 3.2.14 $t \to^{\infty} \mathcal{B}_{\mathcal{U}}(t)$.
If $t \to^{\infty} t'$ and t' is totally meaningful then by Corollary 3.2.13 $\mathcal{B}_{\mathcal{U}}(t')$ is totally meaningful, hence totally defined. By uniqueness of Böhm$_{\mathcal{U}}$ trees we know that $\mathcal{B}_{\mathcal{U}}(t) \equiv \mathcal{B}_{\mathcal{U}}(t')$. $\square$

Remember that $\stackrel{\mathcal{U}}{\to}$ was defined as $s \stackrel{\mathcal{U}}{\to} t$ iff t can be obtained from s by replacing some subterms in $\mathcal{U}$ by arbitrary terms (Definition 3.1.4). The similarity between the definition of $\stackrel{\mathcal{U}}{\to}$ and the prefix ordering of Böhm$_{\mathcal{U}}$ terms suggest a strong connection. Prefix ordering can be translated to (ordinary) terms in the presence of unique Böhm$_{\mathcal{U}}$ trees.

Definition 3.2.17. (Prefix order on terms)
Assume that every term t has a unique Böhm$_{\mathcal{U}}$ term $\mathcal{B}_{\mathcal{U}}(t)$. Then define

$$\forall s, t \in \mathfrak{T} : \quad s \sqsubseteq_{\mathcal{U}} t \iff \mathcal{B}_{\mathcal{U}}(s) \sqsubseteq \mathcal{B}_{\mathcal{U}}(t)$$

Proposition 3.2.18. Let $\mathcal{U} \vDash U1, U2, [U3,]U4(2), U5$. Then

$$s\ (\stackrel{\mathcal{U}}{\to})^{*}\ t \Longrightarrow s\ \sqsubseteq_{\mathcal{U}}\ t$$

Proof
Note that since $\sqsubseteq_{\mathcal{U}}$ is transitive, we only have to prove $s \stackrel{\mathcal{U}}{\to} t \Longrightarrow s \sqsubseteq_{\mathcal{U}} t$.
First we prove this for s having a finite Böhm tree by induction on the depth of the Böhm tree of s.
Base case ($d(\mathcal{B}_{\mathcal{U}}(s)) = 0$ i.e. $\mathcal{B}_{\mathcal{U}}(s) = \perp$ i.e. $s \in \mathcal{U}$) :
trivial, since $s \sqsubseteq_{\mathcal{U}} t$ for every term t.
Induction case ($d(\mathcal{B}_{\mathcal{U}}(s)) = n+1$) : $\mathcal{B}_{\mathcal{U}}(s) = F(B_1, \ldots, B_m)$. Then by Lemma 3.2.4 and Corollary 3.1.15 we know that $s \stackrel{out\mathcal{U}}{\twoheadrightarrow} s' \equiv F(s_1, \ldots, s_m)$, where $\mathcal{B}_{\mathcal{U}}(s_i) = B_i$

[5] Equality in this lemma denotes transfinite convertibility.

and $d(\mathcal{B}_{\mathcal{U}}(s_i)) \leq n$ $(i = 1, \ldots, m)$. Now by Lemma 3.1.11 $\exists t' : t \twoheadrightarrow t' \wedge s' \overset{\mathcal{U}}{\to} t'$. Since replacements in s' can only occur in the s_i subterms we get that $t' \equiv F(t_1, \ldots, t_m)$ and $s_i \overset{\mathcal{U}}{\to} t_i$ $(i = 1, \ldots, m)$. By the induction hypothesis $s_i \sqsubseteq_{\mathcal{U}} t_i$ and this trivially implies $s \sqsubseteq_{\mathcal{U}} t$.
Now suppose that $\mathcal{B}_{\mathcal{U}}(s)$ is infinite. We know that $\mathcal{B}_{\mathcal{U}}(s) = \cup_k \mathcal{B}_{\mathcal{U}}(s^{(k)})$ where $s^{(k)}$ is the term obtained from $\mathcal{B}_{\mathcal{U}}(s)$ by replacing every subterm at depth k and every occurrence of $\perp$ by some term $u \in \mathcal{U}$. So it is sufficient to prove $s^{(k)} \sqsubseteq_{\mathcal{U}} t$ for every $k \in \mathbb{N}$ [6].
First we connect s to $s^{(k)}$. Again by Lemma 3.2.4 and Corollary 3.1.15 $\exists s' : s \overset{out\mathcal{U}}{\twoheadrightarrow} s' \overset{in\mathcal{U}}{\twoheadrightarrow} \twoheadrightarrow_{\mathcal{U}_\perp} \mathcal{B}_{\mathcal{U}}(s)$, consequently for every position π of $\mathcal{B}_{\mathcal{U}}(s)$: $\mathcal{B}_{\mathcal{U}}(s)|\pi \equiv \perp$ iff $s'|\pi \in \mathcal{U}$, i.e. s' has the "same structure" as $\mathcal{B}_{\mathcal{U}}(s)$, so for every $k \in \mathbb{N}$ there is a $s^{[k]}$ such that $s \overset{out\mathcal{U}}{\twoheadrightarrow} s^{[k]} \overset{out\mathcal{U}}{\twoheadrightarrow} s'$ and $s^{[k]}$ is the k-prefix of s' and so it follows, that $s^{(k)} \overset{\mathcal{U}}{\to} s^{[k]}$. From $s \overset{out\mathcal{U}}{\twoheadrightarrow} s^{[k]}$ and $s^{(k)} \overset{\mathcal{U}}{\to} s^{[k]}$ we get by Lemma 3.1.11 that $\exists t^{[k]} : t \twoheadrightarrow t^{[k]}$, $s^{[k]} \overset{\mathcal{U}}{\to} t^{[k]}$, hence $s^{(k)}(\overset{\mathcal{U}}{\to})^* t^{[k]}$ and since $\mathcal{B}_{\mathcal{U}}(s^{(k)})$ is finite we can use this proposition in the finite case giving us $s^{(k)} \sqsubseteq_{\mathcal{U}} t^{[k]}$ and thus $s^{(k)} \sqsubseteq_{\mathcal{U}} t$ as needed, since $\mathcal{B}_{\mathcal{U}}(t^{[k]}) = \mathcal{B}_{\mathcal{U}}(t)$. □

Remark 3.2.19. The inverse implication does not hold as shown by the following example, again from λ-calculus. Let $s \equiv \lambda y.(\lambda x.y(xx))\mathbf{D}$ and $t \equiv \lambda y.y\mathbf{I}$. Then $BT(s) = \lambda y.y\perp \subseteq \lambda y.y\mathbf{I} = BT(t)$ so $s \sqsubseteq_{\mathcal{U}} t$ but $s(\overset{\mathcal{U}}{\not\to})^* t$ since s does not contain any unsolvable subterms, only after a reduction.

The reason behind the fact that the implication can not be reversed is that Lemma 3.1.11 does not hold with $\overset{out\mathcal{U}}{\twoheadrightarrow}$ (and $\twoheadrightarrow$) in the inverse direction and possibly there are terms which do not contain any subterms in $\mathcal{U}$, but some of their reducts do. The difference lies in that $\sqsubseteq_{\mathcal{U}}$ is compatible with the equivalence relation generated by $\mathcal{B}_{\mathcal{U}}()^{-1}$ and thus with convertibility while $\overset{\mathcal{U}}{\to}$ is in general not.

3.3 Strict Contexts and Usability

In this section we will generalize the approach presented in section 2.3 to TRSs along the tracks laid down in subsection 2.3.3. We have already seen a benefit of this approach to the axiomatic theory in Theorem 3.1.47.

The following definition is a combination of ideas from [Kup94], [KOV99] and subsection 2.3.3. It defines usability as in Definition 2.3.6 and genericity as in Definition 2.3.16 using strict contexts taken from Definition 2.3.12 instead of K-strict contexts of Definitions 2.3.1.

Definition 3.3.1. Let $\mathcal{V}$ be a set of terms. Then

- a context $C[]$ is a **strict context (with respect to $\mathcal{V}$)** iff $C[] \in \mathcal{C}^{\mathcal{V}}_{\mathcal{V}}$.
- a term t is **usable (with respect to $\mathcal{V}$) to compute** a term s $(t \gg_{\mathcal{V}} s)$ iff there is a strict context $C[]$ s.t. $C[t] \twoheadrightarrow s$.
- a term is **usable (with respect to $\mathcal{V}$)** iff it is usable to compute a totally meaningful term with respect to $\mathcal{V}$. That is there is a strict context $C[]$ s.t. $C[t] \twoheadrightarrow s \in \mathcal{TM}(\mathcal{V})$, otherwise it is **unusable with respect to $\mathcal{V}$** .

[6]Note that we can not prove $s \overset{\mathcal{U}}{\to} t \Longrightarrow \forall k \in \mathbb{N} : s^{(k)} \overset{\mathcal{U}}{\to} t$ as in λ-calculus for instance: $s \equiv x(\mathbf{SKK})(\mathbf{YK})\mathbf{\Omega}$, $t \equiv x(\mathbf{SKK})(\mathbf{YK})\mathbf{I}$, and $s^{(k+1)} \equiv x\mathbf{I}(\mathbf{YK})^{(k)}\mathbf{\Omega}$

- a term t is **generic (with respect to $\mathcal{V}$)** if for every context $C[]$: $C[t] \twoheadrightarrow t' \in \mathcal{TM}(\mathcal{V}) \Longrightarrow \forall s : C[s] \twoheadrightarrow t'$
- $S(\mathcal{V})$, $U(\mathcal{V})$, and $G(\mathcal{V})$ denote the set of terms usable, unusable, and generic with respect to $\mathcal{V}$ respectively.

These generalized notions possess most of the basic properties of the strict contexts and usability of $\lambda\delta$-calculi discussed earlier in Section 2.3. Remember that $\mathcal{V}$ -strict contexts form a context monoid.

Proposition 3.3.2. For any set $\mathcal{V}$ of terms:

i) $C[] \in \mathcal{C}_{\mathcal{V}}^{\mathcal{V}} \Longrightarrow t \gg_{\mathcal{V}} C[t]$

ii) $t \twoheadrightarrow s \Longrightarrow t \gg_{\mathcal{V}} s$

iii) $t \gg_{\mathcal{V}} t$

iv) $t \gg_{\mathcal{V}} s, s \gg_{\mathcal{V}} L \Longrightarrow t \gg_{\mathcal{V}} L$

v) $t \gg_{\mathcal{V}} s, s \in S(\mathcal{V}) \Longrightarrow t \in S(\mathcal{V})$

vi) $C[] \in \mathcal{C}_{\mathcal{V}}^{\mathcal{V}}, C[t] \in S(\mathcal{V}) \Longrightarrow t \in S(\mathcal{V})$,

vii) If $\rightarrow$ is **CR** , then $t = s \Longrightarrow (t \in S(\mathcal{V}) \iff s \in S(\mathcal{V}))$.

Proof

i) per def. since $C[t] \twoheadrightarrow C[t]$

ii) take $C[] \equiv [] \in \mathcal{C}_{\mathcal{V}}^{\mathcal{V}}$

iii) immediate from ii)

iv) since strict contexts are closed under composition

v) by transitivity (iv)

vi) immediate from i) and v)

vii) follows from ii) and the fact that $\mathcal{TM}(\mathcal{V}) \vDash \rightarrow$

$\square$

Corollary 3.3.3.

- *$\gg_{\mathcal{V}}$ is a reflexive, transitive relation i.e. a preorder; $\twoheadrightarrow \subseteq \gg_{\mathcal{V}}$. (Note that $\twoheadrightarrow$ is closed under contexts, but $\gg_{\mathcal{V}}$ is not.)*
- *$U(\mathcal{V}) \vDash \gg_{\mathcal{V}}$ (consequently $U(\mathcal{V}) \vDash \rightarrow$).*
- *$\rightarrow \vDash$ **CR** $\Longrightarrow U(\mathcal{V}) \vDash =$*
- *$\mathcal{C}_{\mathcal{V}}^{\mathcal{V}} \subseteq \mathcal{C}_{U(\mathcal{V})}^{U(\mathcal{V})}$ i.e. all contexts strict with respect to $\mathcal{V}$ are also strict with respect to the set of $\mathcal{V}$ -unusable terms.*

3.3.1 Θ-usability

Observe that nowhere in the above reasoning and in section 2.3.3 did we use the fact that $\mathcal{C}$-usability was defined in terms of normal forms. Thus we can generalize our theory by parameterizing it with a set Θ of terms (replacing $\mathcal{NF}^{=}$ and "has a normal form" by "$\in \Theta$" in every definition, lemma or theorem) and our results will carry over.

Definition 3.3.4. (Θ-usability)
Let $S, U : P(\mathfrak{T}) \times \mathbb{M} \longrightarrow P(\mathfrak{T})$ be defined as

$$\begin{array}{rcl} S_\Theta(\mathcal{C}) = S(\Theta, \mathcal{C}) & = & \{t : \exists C[] \in \mathcal{C} : C[t] \in \Theta\} \\ U_\Theta(\mathcal{C}) = U(\Theta, \mathcal{C}) & = & \{t : \forall C[] \in \mathcal{C} : C[t] \notin \Theta\} \\ T_\Theta(\mathcal{U}) = T(\mathcal{U}) & = & \mathcal{C}_{\mathfrak{T}\setminus\mathcal{U}}^{\mathfrak{T}\setminus\mathcal{U}} \\ \mathbb{M}_\Theta & = & \mathbb{M} \\ \mathbb{S}_\Theta & = & \{S_\Theta(\mathcal{C}) : \mathcal{C} \in \mathbb{M}_{(\Theta)}\} \\ \mathbb{C}_\Theta & = & \{T_{(\Theta)} S_\Theta(\mathcal{C}) : \mathcal{C} \in \mathbb{M}_{(\Theta)}\} \\ \mathcal{C}^p{}_\Theta & = & \{C[] : \exists t, s : C[t] \in \Theta, C[t] \neq C[s]\} \\ \mathcal{P}_\Theta & = & S_\Theta(\mathcal{C}^p{}_\Theta) \\ \mathcal{G}_\Theta & = & \{t : \forall C[] : C[t] \in \Theta \Longrightarrow \forall s : C[s] = C[t]\} \end{array}$$

$$\begin{array}{rcl} (\Theta, \mathcal{C}) \sim (\Theta', \mathcal{C}') & \Longleftrightarrow & S(\Theta, \mathcal{C}) = S(\Theta', \mathcal{C}') \\ (\Theta, \mathcal{C}) \preceq (\Theta', \mathcal{C}') & \Longleftrightarrow & S(\Theta, \mathcal{C}) \subseteq S(\Theta', \mathcal{C}') \\ (\Theta, \mathcal{C}) \prec (\Theta', \mathcal{C}') & \Longleftrightarrow & S(\Theta, \mathcal{C}) \subset S(\Theta', \mathcal{C}') \\ \Theta \sim_{\mathcal{C}} \Theta' & \Longleftrightarrow & (\Theta, \mathcal{C}) \sim (\Theta', \mathcal{C}) \\ \Theta \preceq_{\mathcal{C}} \Theta' & \Longleftrightarrow & (\Theta, \mathcal{C}) \preceq (\Theta', \mathcal{C}) \\ \Theta \prec_{\mathcal{C}} \Theta' & \Longleftrightarrow & (\Theta, \mathcal{C}) \prec (\Theta', \mathcal{C}) \\ \mathcal{C} \sim_\Theta \mathcal{C}' & \Longleftrightarrow & (\Theta, \mathcal{C}) \sim (\Theta, \mathcal{C}') \\ \mathcal{C} \preceq_\Theta \mathcal{C}' & \Longleftrightarrow & (\Theta, \mathcal{C}) \preceq (\Theta, \mathcal{C}') \\ \mathcal{C} \prec_\Theta \mathcal{C}' & \Longleftrightarrow & (\Theta, \mathcal{C}) \prec (\Theta, \mathcal{C}') \end{array}$$

Remark 3.3.5. We introduced two notations in the definition above: $S_\Theta(\mathcal{C})$ and $S(\Theta, \mathcal{C})$ (and $U_\Theta(\mathcal{C})$ and $U(\Theta, \mathcal{C})$) denote the same set, and we will use both notations without preference.

Remark 3.3.6. In Definition 3.3.1 we defined a term s usable with respect to a set $\mathcal{V}$ as

$$\begin{array}{l} \exists t \in \mathcal{TM}(\mathcal{V}) : s \gg_{\mathcal{V}} t \quad i.e. \\ \exists t \in \mathcal{TM}(\mathcal{V})\ \exists C[] \in \mathcal{C}_{\mathcal{V}}^{\mathcal{V}} :\ C[s] \twoheadrightarrow t \end{array}$$

so we see, that $S(\mathcal{V})$ and $U(\mathcal{V})$ are just shorthand for the following

$$S(\mathcal{V}) = S(\mathcal{M}^{\leftarrow}(\mathcal{V}), \mathcal{C}_{\mathcal{V}}^{\mathcal{V}}), \quad U(\mathcal{V}) = U(\mathcal{M}^{\leftarrow}(\mathcal{V}), \mathcal{C}_{\mathcal{V}}^{\mathcal{V}})$$

In other words, terms (un)usable with respect to $\mathcal{V}$ are exactly the $(\mathcal{M}^{\leftarrow}(\mathcal{V}), \mathcal{C}_{\mathcal{V}}^{\mathcal{V}})$-(un)usable terms.

Remark 3.3.7. In Definition 3.3.1 and in Definition 3.3.4 above we have defined two kinds of genericity:

$$\begin{array}{rcl} G(\mathcal{V}) & = & \{t | \forall C[] : (C[t] \to t' \in \mathcal{TM}(\mathcal{V}) \Longrightarrow \forall s : C[s] \twoheadrightarrow t'\} \\ \mathcal{G}_\Theta & = & \{t | \forall C[] : (C[t] \in \Theta \Longrightarrow \forall s : C[s] = C[t]\} \end{array}$$

Both of them are generalizations of the concept of genericity in the λ-calculus, the first one is important, because it is how genericity was used in [KOV99] and the second one is of interest to us, because of the special role it plays in our world of context monoids. Unfortunately, we do not have as strong a correspondence between the two notions as we did with the two kinds of usability in Remark 3.3.6 above. One relation is obvious though, and will be important:

$$G(\mathcal{V}) \subseteq \mathcal{G}_{\mathcal{M}^{\leftarrow}(\mathcal{V})}$$

Inclusion in the other direction does not hold in general, not even under **CR** , only if we assume that totally meaningful terms are in normal form, and that the system has the **NF** property (i.e. that $\forall t, s : t = s, s \in \mathcal{NF} \Longrightarrow t \twoheadrightarrow s$).

$$\mathbf{NF}, \mathcal{TM}(\mathcal{V}) \subseteq \mathcal{NF} \quad \Longrightarrow \quad \mathcal{G}_{\mathcal{M}^{\leftarrow}(\mathcal{V})} \subseteq G(\mathcal{V})$$

Theorem 3.3.8. *(Results)*

$$\begin{array}{rrcll}
i) & \mathcal{C} \subseteq \mathcal{C}' & \Longrightarrow & \mathcal{C} \preceq_\Theta \mathcal{C}' & (\mathit{Lemma}\ 2.3.23\ i)) \\
ii) & [] \in \mathcal{C} & \Longrightarrow & \Theta \subseteq S_\Theta(\mathcal{C}) & (\mathit{Lemma}\ 2.3.23\ ii)) \\
iii) & \mathcal{C}\mathcal{C} \subseteq \mathcal{C} & \Longrightarrow & \mathcal{C} \subseteq TS_\Theta(\mathcal{C}) & (\mathit{Lemma}\ 2.3.23\ iii)) \\
iv) & S_\Theta T S_\Theta & = & S_\Theta & (\mathit{Lemma}\ 2.3.25)
\end{array}$$

$$\begin{array}{rll}
v) & \forall \mathcal{C} \in \mathbb{M} : (\mathcal{C} \in \mathbb{C}_\Theta \iff TS_\Theta(\mathcal{C}) = \mathcal{C}) & (\mathit{Lemma}\ 2.3.32\ iii)) \\
vi) & (\mathbb{C}_\Theta, \preceq_\Theta) \cong (\mathbb{M}/\sim_\Theta, \preceq_\Theta) \cong (\mathbb{S}_\Theta, \subseteq) & (\mathit{Proposition}\ 2.3.33)
\end{array}$$

$$\begin{array}{rrcll}
vii) & \mathcal{G}_\Theta & = & \overline{\mathcal{P}_\Theta} = U_\Theta(\mathcal{C}^p{}_\Theta) & (\mathit{Theorem}\ 2.3.18) \\
viii) & S_\Theta(\mathcal{C}) \subsetneq \Lambda & \Longrightarrow & S_\Theta(\mathcal{C}) \subseteq \mathcal{P}_\Theta) & (\mathit{Remark}\ 2.3.36) \\
ix) & \emptyset \subsetneq U_\Theta(\mathcal{C}) & \Longrightarrow & \mathcal{G}_\Theta \subseteq U_\Theta(\mathcal{C}) & (\mathit{Remark}\ 2.3.36)
\end{array}$$

Proof
i),ii), and iii) are simple consequences of the definitions, proofs are identical to those in Lemma 2.3.23.
iv) is also straightforward using i),ii), and iii), just as in Lemma 2.3.25.
vii) holds by definition.
All the rest follows from these items by identical reasoning as in their original proof.
□

So far we have only studied the relationship of context monoids with respect to the Θ-usable terms they determine for some fixed Θ, but it is just as reasonable (and will be important later) to study how S and U behave with different Θ when a monoid $\mathcal{C}$ is fixed.

Proposition 3.3.9.
$S : (P(\mathfrak{T}), \subseteq) \times (\mathbb{M}, \subseteq) \longrightarrow (P(\mathfrak{T}), \subseteq)$ and
$U : (P(\mathfrak{T}), \subseteq) \times (\mathbb{M}, \subseteq) \longrightarrow (P(\mathfrak{T}), \supseteq)$ are continuous in both arguments.

Proof
Proposition 2.3.29 proves this for the second argument, and continuity in the first argument, i.e.

$$S(\bigcup_i \Theta_i, \mathcal{C}) = \bigcup_i S(\Theta_i, \mathcal{C})$$

is straightforward from the definition. □

Corollary 3.3.10. *Monotonicity also holds in both arguments.*

3.3.2 Monoid problem

A general problem is to find, given a set of terms $\mathcal{V}$ a set of contexts $\mathcal{C}$ — preferably a context monoid — such that $\mathcal{V} = U(\Theta, \mathcal{C})$. Of course such a monoid, or even set of contexts does not always exist, but when one does, then we say that $\mathcal{V}$ is Θ-*syntactic* or Θ-*monoidic* respectively.

Definition 3.3.11. (Syntactic and Monoidic sets of terms.)
A set $\mathcal{V}$ of terms is **Θ-syntactic** if there is a set $\mathcal{C}$ of contexts, such that $\mathcal{V} = U(\Theta, \mathcal{C})$.
A set $\mathcal{V}$ of terms is **Θ-monoidic** if there is a $\mathcal{C} \in \mathbb{M}$, such that $\mathcal{V} = U(\Theta, \mathcal{C})$.

Proposition 3.3.12. For every set $\mathcal{V}$ of terms:

$$\begin{array}{ll} i) & [\exists \mathcal{C} \in \mathbb{M} : \mathcal{V} = U(\Theta, \mathcal{C})] \iff \mathcal{V} = U(\Theta, \mathcal{C}^{\mathcal{V}}_{\mathcal{V}}) \\ ii) & \mathcal{V} = U(\overline{\mathcal{V}}, \mathcal{C}^{\mathcal{V}}_{\mathcal{V}}) \end{array}$$

Proof
i)
$\Longleftarrow$: By Proposition 2.3.13 $\mathcal{C}^{\mathcal{V}}_{\mathcal{V}} \in \mathbb{M}$
$\Longrightarrow$: If $\mathcal{C}$ is a context monoid, then by Lemma 2.3.25 $U(\Theta, \mathcal{C}) = U(\Theta, \mathcal{C}^{U(\Theta,\mathcal{C})}_{U(\Theta,\mathcal{C})})$, hence $\mathcal{V} = U(\Theta, \mathcal{C}) \Longrightarrow \mathcal{V} = U(\Theta, \mathcal{C}^{\mathcal{V}}_{\mathcal{V}})$.
ii)
For every term $t \in \mathcal{V}$ and context $C[] \in \mathcal{C}^{\mathcal{V}}_{\mathcal{V}}$: $C[t] \in \mathcal{V}$ holds by definition, hence $\mathcal{V} \subseteq U(\overline{\mathcal{V}}, \mathcal{C}^{\mathcal{V}}_{\mathcal{V}})$; and of course $\overline{\mathcal{V}} \subseteq S(\overline{\mathcal{V}}, \mathcal{C}^{\mathcal{V}}_{\mathcal{V}})$ via the empty context. $\square$

Remark 3.3.13. As remarked in 2.3.36, if $\mathcal{C}$ is a context monoid, then either $S_\Theta(\mathcal{C}) \subseteq \mathcal{P}_\Theta$ or $S_\Theta(\mathcal{C}) = \Lambda$. Hence if $\mathcal{V}$ is Θ-monoidic and non-trivial, then $\mathcal{V}$ contains all the Θ-generic terms, i.e. if $\mathcal{V}$ does not contain all Θ-generic terms, and is not empty, then it is not Θ-monoidic.

3.3.3 Total sets

In Proposition 3.1.31 we noticed, that if $\mathcal{V} \vDash \rightarrow$, then $\mathcal{V} \cap \mathcal{M}^{\leftarrow}(\mathcal{V}) = \emptyset$, hence $\mathcal{V} \cap S(\mathcal{M}^{\leftarrow}(\mathcal{V}), \mathcal{C}^{\mathcal{V}}_{\mathcal{V}}) = \emptyset$, i.e.

$$\mathcal{V} \vDash \rightarrow \quad \Longrightarrow \quad \mathcal{V} \subseteq U(\mathcal{M}^{\leftarrow}(\mathcal{V}), \mathcal{C}^{\mathcal{V}}_{\mathcal{V}}) = U(\mathcal{V})$$

We get an important special case of the monoid problem by requiring equality to hold, in other words, that exactly the terms in $\mathcal{V}$ should be unusable with respect to $\mathcal{V}$.

Definition 3.3.14.
A set $\mathcal{V}$ of terms is **total** (or **totally monoidic**) if $\mathcal{V} = U(\mathcal{M}^{\leftarrow}(\mathcal{V}), \mathcal{C}^{\mathcal{V}}_{\mathcal{V}})$.

Proposition 3.3.15. For any set $\mathcal{V}$ the following are equivalent

$$\begin{array}{ll} (1) & \mathcal{V} = U(\mathcal{M}^{\leftarrow}(\mathcal{V}), \mathcal{C}^{\mathcal{V}}_{\mathcal{V}}) \\ (2) & \mathcal{V} = U(\mathcal{V}) \\ (3) & \mathcal{V} \; is \; \mathcal{M}^{\leftarrow}(\mathcal{V}) - monoidic \\ (4) & \mathcal{V} \vDash \rightarrow \quad and \quad \forall t \notin \mathcal{V} : \exists C[] \in \mathcal{C}^{\mathcal{V}}_{\mathcal{V}} : C[t] \in \mathcal{M}^{\leftarrow}(\mathcal{V}) \end{array}$$

Proof
"(1) $\iff$ (2)": $U(\mathcal{V}) = U(\mathcal{M}^{\leftarrow}(\mathcal{V}), \mathcal{C}^{\mathcal{V}}_{\mathcal{V}})$ per. def.
"(1) $\iff$ (3)": by Proposition 3.3.12
"(1) $\iff$ (4)":

$\mathcal{V} \supseteq U_{\mathcal{M}^{\leftarrow}(\mathcal{V})}(\mathcal{C}^{\mathcal{V}}_{\mathcal{V}})$ is equivalent to $\forall t \notin \mathcal{V} : \exists C[] \in \mathcal{C}^{\mathcal{V}}_{\mathcal{V}} : C[t] \in \mathcal{M}^{\leftarrow}(\mathcal{V})$, and $\mathcal{V} \vDash \rightarrow$ implies $\mathcal{V} \cap \mathcal{M}^{\leftarrow}(\mathcal{V}) = \emptyset$, hence $\mathcal{V} \subseteq U_{\mathcal{M}^{\leftarrow}(\mathcal{V})}(\mathcal{C}^{\mathcal{V}}_{\mathcal{V}})$. Finally, from $\mathcal{M}^{\leftarrow}(\mathcal{V}) \vDash \leftarrow$ it follows that $U_{\mathcal{M}^{\leftarrow}(\mathcal{V})}(\mathcal{C}) \vDash \rightarrow$ for every set $\mathcal{C}$ of contexts. □

Below we give another characterization and two important classes of total sets as examples.

Lemma 3.3.16. Every subset $\mathcal{V}$ of $S(\Theta, \mathcal{C})$ which is closed under reduction, and closed under taking subterms is also a subset of $\mathcal{TM}(U(\Theta, \mathcal{C}))$

Proof
Let $s \in \mathcal{V}$ reduce to t. Then by assumption t and all of its subterms are in $S(\Theta, \mathcal{C})$, i.e. t has no subterms in $U(\Theta, \mathcal{C})$. Since this is true for every reduct t of s, s is totally meaningful with respect to $U(\Theta, \mathcal{C})$. □

Lemma 3.3.17. Let $\mathcal{C} \in \mathbb{M}$ be arbitrary and $\Theta \vDash \leftarrow$. Then

$$S(\mathcal{M}^{\leftarrow}(U(\Theta, \mathcal{C})), \mathcal{C}) \subseteq S(\Theta, \mathcal{C})$$

Proof
Let $s \in S(\mathcal{M}^{\leftarrow}(U(\Theta, \mathcal{C})), \mathcal{C})$ and $C[] \in \mathcal{C}$ be such that $C[s] \twoheadrightarrow t \in \mathcal{TM}(U(\Theta, \mathcal{C}))$. Then $t \in S(\Theta, \mathcal{C})$ so there is a context $D[] \in \mathcal{C}$ such that $D[C[s]] \twoheadrightarrow D[t] \in \Theta$, which means that $D[C[s]] \in \Theta$ as well, since $\Theta \vDash \leftarrow$ by assumption. Since $\mathcal{C}$ is a monoid $D[C[]] \in \mathcal{C}$, proving $s \in S(\Theta, \mathcal{C})$. □

Example 3.3.18. For every $\mathcal{C} \in \mathbb{M}$

i) $U(\mathcal{WN}, \mathcal{C})$ is total

ii) $U(\mathcal{M}^{\leftarrow}(\mathcal{V}), \mathcal{C})$ is total for every $\mathcal{V} \subseteq \mathfrak{T}$.

Proof

i) $\mathcal{WN} \vDash \leftarrow$, hence by Lemma 3.3.17 $S(\mathcal{M}^{\leftarrow}(U(\mathcal{WN}, \mathcal{C})), \mathcal{C}) \subseteq S(\mathcal{WN}, \mathcal{C})$. On the other hand $\mathcal{NF} \subseteq S(\mathcal{WN}, \mathcal{C})$ (by []) and $\mathcal{NF}$ is closed under reduction and taking subterms, so by Lemma 3.3.16 $\mathcal{NF} \subseteq \mathcal{TM}(U(\mathcal{WN}, \mathcal{C}))$ and by monotonicity of closure $\mathcal{WN} \subseteq \mathcal{M}^{\leftarrow}(U(\mathcal{WN}, \mathcal{C}))$. By monotonicity of S $S(\mathcal{WN}, \mathcal{C}) \subseteq S(\mathcal{M}^{\leftarrow}(U(\mathcal{WN}, \mathcal{C})), \mathcal{C})$.
From the above we conclude that $U(\mathcal{WN}, \mathcal{C}) = U(\mathcal{M}^{\leftarrow}(U(\mathcal{WN}, \mathcal{C})), \mathcal{C})$, i.e. $U(\mathcal{WN}, \mathcal{C})$ is $\mathcal{M}^{\leftarrow}(U(\mathcal{WN}, \mathcal{C}))$-monoidic, hence total by Proposition 3.3.15.

ii) The proof is analogous to that of i). Just note that $\mathcal{M}^{\leftarrow}(\mathcal{V}) \vDash \leftarrow$ just as $\mathcal{WN} \vDash \leftarrow$, and $\mathcal{TM}(\mathcal{V})$ is closed under reduction and taking subterms, just as $\mathcal{NF}$ and $\mathcal{TM}(\mathcal{V}) \subseteq S(\mathcal{M}^{\leftarrow}(\mathcal{V}), \mathcal{C})$ (by []) just as $\mathcal{NF} \subseteq S(\mathcal{WN}, \mathcal{C})$.

□

Corollary 3.3.19. *(Sets of unusables*[7] *are total.)*
For every set $\mathcal{V}$ of terms $U(\mathcal{V})$ is total, that is $U(U(\mathcal{V})) = U(\mathcal{V})$.

Proof
By definition $U(\mathcal{V}) = U(\mathcal{M}^{\leftarrow}(\mathcal{V}), \mathcal{C}^{\mathcal{V}}_{\mathcal{V}})$ which is total as seen in Example 3.3.18 ii).
□

[7] in the sense of Definition 3.3.1, not in the more general $(\Theta, \mathcal{C})$-usability sense

Remark 3.3.20.
In fact, the example showed that $U(\mathcal{M}^{\leftarrow}(\mathcal{V}),\mathcal{C}_{\mathcal{V}}^{\mathcal{V}}) = U(\mathcal{M}^{\leftarrow}(U(\mathcal{M}^{\leftarrow}(\mathcal{V}),\mathcal{C}_{\mathcal{V}}^{\mathcal{V}})),\mathcal{C}_{\mathcal{V}}^{\mathcal{V}})$, but we can only prove $\mathcal{M}^{\leftarrow}(\mathcal{V}) = \mathcal{M}^{\leftarrow}(U(\mathcal{M}^{\leftarrow}(\mathcal{V}),\mathcal{C}_{\mathcal{V}}^{\mathcal{V}}))$ if $\mathcal{V}$ is closed under reduction.

Corollary 3.3.21. *(Exactly the sets of unusables are total)*
Let $\mathcal{V}$ be an arbitrary set of terms. Then

$$\mathcal{V} \textit{ is total} \quad \Longleftrightarrow \quad \exists \mathcal{W} \subseteq \mathfrak{T} : \mathcal{V} = U(\mathcal{W})$$

Proof
If $\mathcal{V}$ is total, then by definition $\mathcal{V} = U(\mathcal{V})$, and if $\mathcal{V} = U(\mathcal{W})$, then by Corollary 3.3.19 $\mathcal{V}$ is total. ☐

Remark 3.3.22. (On the name "total" or "totally monoidic")
At the beginning of this section we observed that if a set $\mathcal{V}$ satisfies axiom U1., i.e. it is closed under reduction, then $\mathcal{V} \subseteq U(\mathcal{V})$. Totally monoidic sets do satisfy axiom U1. as also remarked, hence they are maximal in the sense that they reach this upper bound.
In Lemma 3.1.44 we proved, that if a total set $\mathcal{V}$ is weakly generic, then it is also widely generic, hence indiscernible (see Definition 3.1.32), thus if a total set $\mathcal{V}$ satisfies axiom U2. (Orthogonality) then it also satisfies axiom U5. (Indiscernability) — see Theorem 3.1.47. Axiom U5. was also baptized "Maximality" in accordance with our remark after Definition 3.1.9 that it expresses that identifying elements of a set $\mathcal{V}$ satisfying U5. will not necessitate identifying any other terms with them. So totally monoidic sets, which satisfy U2., are maximal in this respect.
The name is also intended to refer to totally meaningful terms which form the "center" of $\overline{\mathcal{V}}$ of a total set $\mathcal{V}$ in the sense that every term outside of $\mathcal{V}$ is usable to compute a totally meaningful term.

3.3.4 Generator problem

Another general problem is to find a minimal generator system, or better still, a base for any given context monoid in any framework, and in particular to find a base for the maximal non-trivial monoid of separating contexts, giving a characterization of non-generic terms, just as we did with K-strict contexts in $\lambda\mu\mathbf{E}$.

For TRSs the answer is remarkably given by combining the idea of K-strict contexts of [Kup94] and the axioms and results of [KOV99].

Definition 3.3.23. Let t be a term, and π a position in t.
We say that π is a **non-trivial position in** t, if π is not the root and $t|\pi$ is not a variable. The set of non-trivial positions of t will be denoted by $\Pi(t)$.
The context obtained by placing [] at position π in t is called an **associated context of** t, and will be denoted by $C^t_\pi[]$. Thus $C^t_\pi[t|\pi] \equiv t$.

Definition 3.3.24. (Generalized K-strict contexts)
Let $T = (\mathfrak{T}, R)$ be a TRS. The set $\mathcal{K}$ of **K-strict contexts over T** is defined as the monoid generated by the following non-trivial contexts associated to redexes:

$$\mathcal{K} = < \{C^{\rho(l)}_\pi[] \,:\, (l \to r) \in R, \pi \in \Pi(l), \rho \textit{ arbitrary valuation}\} >$$

The similarity of the above definition and axiom U2. is not coincidental. The idea of generalizing K-strict contexts to arbitrary rewriting systems this way is due to Kuper, and it was admittedly motivated by the intuition behind axiom U2. In fact, it is easy to check that K-strict contexts defined this way over $\lambda\mu\mathbf{E}$ are identical to Kuper's original strict contexts, with the exception of β and μ redexes.

To relate Θ-generic terms with K-strict contexts, we employ the axioms and results of [KOV99].

Proposition 3.3.25. For every $\Theta \subseteq \mathfrak{T}$

i) $U_\Theta(\mathcal{K}) \models U2$.

ii) If $\overline{\Theta} \models \rightarrow$ then $U_\Theta(\mathcal{K}) \models U1$.

Proof

i) Let $u \in U_\Theta(\mathcal{K})$ overlap a redex $\rho(l)$ at the non-trivial position π of l. Then, by definition, $\rho(l) \equiv C_\pi^{\rho(l)}[u]$, and $C_\pi^{\rho(l)} \in \mathcal{K}$. Hence by strictness of contexts in $\mathcal{K}$ with respect to $U_\Theta(\mathcal{K})$, $C_\pi^{\rho(l)}[u] \in U_\Theta(\mathcal{K})$ as well, proving U2. for $U_\Theta(\mathcal{K})$

ii) Remember that $U_\Theta(\mathcal{K})$ was defined as $\{t \in \mathfrak{T} : \forall C[] \in \mathcal{K} : C[t] \notin \Theta\}$. Now if $\overline{\Theta} \models \rightarrow$, $C[t] \notin \Theta$, and $t \twoheadrightarrow s$ then $C[t] \twoheadrightarrow C[s] \notin \Theta$ as well, proving U1. for $U_\Theta(\mathcal{K})$

□

Corollary 3.3.26.

$$\Theta \models \leftarrow \quad \Longrightarrow \quad U_\Theta(\mathcal{K}) \subseteq G(U_\Theta(\mathcal{K})) \subseteq \mathcal{G}_{\mathcal{M}^{\leftarrow}(U_\Theta(\mathcal{K}))}$$

Proof
By Proposition 3.3.25 $U_\Theta(\mathcal{K})$ satisfies axioms U1. U2., and by the Genericity Lemma (Theorem 3.1.20) $U_\Theta(\mathcal{K})$ is generic in the sense of [KOV99], i.e. $U_\Theta(\mathcal{K}) \subseteq G(U_\Theta(\mathcal{K}))$. By Remark 3.3.7 we can translate this to genericity in the sense of Definition 3.1.18, since $G(U_\Theta(\mathcal{K})) \subseteq \mathcal{G}_{\mathcal{M}^{\leftarrow}(U_\Theta(\mathcal{K}))}$. □

Our intention is to relate Θ-unusable terms determined by $\mathcal{K}$ with Θ-generic terms, instead of $\mathcal{M}^{\leftarrow}(U_\Theta(\mathcal{K}))$-generic terms as in Corollary 3.3.26 above. To prove $U_\Theta(\mathcal{K}) \subseteq \mathcal{G}_\Theta$ using our existing result, we need to prove $G(\mathcal{M}^{\leftarrow}(U_\Theta(\mathcal{K}))) \subseteq \mathcal{G}_\Theta$. Below we will do just that using different side-conditions, after giving a preliminary proposition.

Proposition 3.3.27. ($\mathcal{G} : (P(\mathfrak{T}), \subseteq) \longrightarrow (P(\mathfrak{T}), \supseteq)$ is continuous)

$$\mathcal{G}_{\bigcup_i \Theta_i} = \bigcap_i \mathcal{G}_{\Theta_i}$$

Corollary 3.3.28.

$$\Theta_1 \subseteq \Theta_2 \quad \Longrightarrow \quad \mathcal{G}_{\Theta_1} \supseteq \mathcal{G}_{\Theta_2}$$

Theorem 3.3.29.

$$\begin{array}{rcll} U_{\mathcal{WN}}(\mathcal{K}) & \subseteq & \mathcal{G}_{\mathcal{WN}} & \\ U_{\mathcal{M}^{\leftarrow}(\mathcal{V})}(\mathcal{K}) & \subseteq & \mathcal{G}_{\mathcal{M}^{\leftarrow}(\mathcal{V})} & (\forall \mathcal{V} \subseteq \mathfrak{T}) \end{array}$$

Proof
Note that $[] \in \mathcal{K} \Longrightarrow \Theta \subseteq S_\Theta(\mathcal{K})$ for any Θ, hence for $\mathcal{WN}$ and $\mathcal{M}^{\leftarrow}(\mathcal{V})$ as well, but then

$$\mathcal{NF} \subseteq S_{\mathcal{WN}}(\mathcal{K}), \quad \mathcal{TM}(\mathcal{V}) \subseteq S_{\mathcal{M}^{\leftarrow}(\mathcal{V})}(\mathcal{K})$$

respectively, hence by Lemma 3.3.16

$$\mathcal{NF} \subseteq \mathcal{TM}(U_{\mathcal{WN}}(\mathcal{K})), \quad \mathcal{TM}(\mathcal{V}) \subseteq \mathcal{TM}(U_{\mathcal{M}^{\leftarrow}(\mathcal{V})}(\mathcal{K}))$$

so by monotonicity of closure

$$\mathcal{WN} \subseteq \mathcal{M}^{\leftarrow}(U_{\mathcal{WN}}(\mathcal{K})), \quad \mathcal{M}^{\leftarrow}(\mathcal{V}) \subseteq \mathcal{M}^{\leftarrow}(U_{\mathcal{M}^{\leftarrow}(\mathcal{V})}(\mathcal{K}))$$

and finally

$$\mathcal{G}_{\mathcal{M}^{\leftarrow}(U_{\mathcal{WN}}(\mathcal{K}))} \subseteq \mathcal{G}_{\mathcal{WN}}, \quad \mathcal{G}_{\mathcal{M}^{\leftarrow}(U_{\mathcal{M}^{\leftarrow}(\mathcal{V})}(\mathcal{K}))} \subseteq \mathcal{G}_{\mathcal{M}^{\leftarrow}(\mathcal{V})}$$

by Corollary 3.3.28, and the result follows from Corollary 3.3.26, since $\mathcal{WN}, \mathcal{TM}(\mathcal{V}) \models \leftarrow$. □

Chapter 4

Conclusion

4.1 Summary

In chapter 2 of this thesis we have tracked the evolution of solvability, the most widely accepted formal notion of meaningfulness, from its roots in the λI-calculus to its generalizations to various (typed) $\lambda\delta$-calculi. In section 2.1 we shed light on the silent role it played in Kleene's representation of the partial recursive functions in the λI-calculus, showed how the same construction can be used to represent λK-terms in λI-calculus locally to a set of uniformly solvable terms, and argued that (a generalization of) this technique is most likely to be the best achievable. In section 2.2 we investigated refinements of solvability arriving at the class of breakable terms and a notion of relative solvability. Alongside these investigations, we have also looked at the concept of strictness from different perspectives. In section 2.3 we briefly presented usability using strict contexts, its relation to solvability and genericity, and with further generalizations in mind we gave two alternative equivalent definitions.

In chapter 3 we took an overview of the works [AKKSV94] and [KOV99] in the world of term rewriting systems, looking at refinements, other characterizations and some connections between their axioms as well as generalized Böhm trees. Finally in section 3.3 we presented a general approach of defining sets of terms using sets of contexts closed under composition, we studied strict contexts and usability for TRSs in this framework and illustrated how these can be put to use to learn more about some of the earlier mentioned axioms. An interesting class of sets of terms, the so called total sets seemed to have an importance in these investigations.

4.2 Further Research

I believe that the following topics could be interesting for further research:

- a syntactical characterization of breakability (such as the equivalence of solvability and the existence of a head normal form or the termination of the head reduction)
- defining information content or some related partial order of terms
- applicability and limits of defining meaninglessness using context monoids
- the role and importance of total sets in such constructions
- a generalization of usability to combinatory reduction systems

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.

[BB79] J.Baeten and B.Boerboom: *Ω Can be anything it should not be*, Indag. Math., vol. 41., pp 111-120, 1979.

[BI93] Alessandro Berarducci and Benedetto Intrigila: *Some new results on easy lambda-terms*, Theoretical Computer Science volume 121 pages 71-88, Elsevier, 1993.

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

[Int91] Benedetto Intrigila: *A problem on easy terms in λ-calculus*, Fund. Inform. **XV** (1), 99-106, 1991.

[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.