

Simona Ronchi Della Rocca
Luca Paolini



The Parametric Lambda Calculus

A Metamodel for Computation



Springer

Texts in Theoretical Computer Science

An EATCS Series

Editors: W. Brauer G. Rozenberg A. Salomaa

On behalf of the European Association
for Theoretical Computer Science (EATCS)

Advisory Board: G. Ausiello M. Broy C. S. Calude
S. Even J. Hartmanis J. Hromkovič N. Jones
T. Leighton M. Nivat C. Papadimitriou D. Scott

Springer-Verlag Berlin Heidelberg GmbH

Simona Ronchi Della Rocca
Luca Paolini

The Parametric Lambda Calculus

A Metamodel for Computation



Springer

Authors

Prof. Simona Ronchi Della Rocca
Università di Torino
Dipartimento di Informatica
corso Svizzera 185
10149 Torino, Italy
ronchi@di.unito.it
www.di.unito.it/~ronchi

Dr. Luca Paolini
Università di Torino
Dipartimento di Informatica
corso Svizzera 185
10149 Torino, Italy
paolini@di.unito.it
www.di.unito.it/~paolini

Series Editors

Prof. Dr. Wilfried Brauer
Institut für Informatik der TUM
Boltzmannstr. 3, 85748 Garching, Germany
Brauer@informatik.tu-muenchen.de

Prof. Dr. Grzegorz Rozenberg
Leiden Institute of Advanced Computer Science
University of Leiden
Niels Bohrweg 1, 2333 CA Leiden, The Netherlands
rozenber@liacs.nl

Prof. Dr. Arto Salomaa
Turku Centre for Computer Science
Lemminkäisenkatu 14 A, 20520 Turku, Finland
asalomaa@utu.fi

Library of Congress Cataloging-in-Publication Data

Ronchi Della Rocca, S. (Simona)
The parametric lambda calculus : A metamodel for computation / Simona Ronchi Della Rocca,
Luca Paolini.
p. cm. – (Texts in theoretical computer science)
Includes bibliographical references and index.
ISBN 978-3-642-05746-5 ISBN 978-3-662-10394-4 (eBook)
DOI 10.1007/978-3-662-10394-4
1. Lambda calculus. I. Paolini, Luca, 1970– II. Title. III. Series.
QA9.5.R66 2004 511.3'5-dc22 2003069100

ACM Computing Classification (1998): F.4, F.3, I.2.3, D.2

ISBN 978-3-642-05746-5

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer-Verlag Berlin Heidelberg GmbH.

Violations are liable for prosecution under the German Copyright Law.

springeronline.com

© Springer-Verlag Berlin Heidelberg 2004

Originally published by Springer-Verlag Berlin Heidelberg New York in 2004

Softcover reprint of the hardcover 1st edition 2004

The use of general descriptive names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and therefore free for general use.

Cover Design: KünkelLopka, Heidelberg

Typesetting: Camera-ready by the authors

Printed on acid-free paper 45/3142/GF - 5 4 3 2 1 0

*To Corrado Böhm, from which Simona and –
by transitivity – Luca learned the pleasure
of research and the interest in λ -calculus*

Preface

The λ -calculus was invented by Church in the 1930s with the purpose of supplying a logical foundation for logic and mathematics [25]. Its use by Kleene as a coding for computable functions makes it the first programming language, in an abstract sense, exactly as the Turing machine can be considered the first computer machine [57]. The λ -calculus has quite a simple syntax (with just three formation rules for terms) and a simple operational semantics (with just one operation, substitution), and so it is a very basic setting for studying computation properties.

The first contact between λ -calculus and real programming languages was in the years 1956-1960, when McCarthy developed the LISP programming language, inspired from λ -calculus, which is the first “functional” programming language, i.e., where functions are first-class citizens [66]. But the use of λ -calculus as an abstract paradigm for programming languages started later as the work of three important scientists: Strachey, Landin and Böhm. Strachey used the λ -notation as a descriptive tool to represent functional features in programming when he posed the basis for a formal semantics of programming languages [92]. Landin formalized the idea that the semantics of a programming language can be given by translating it into a simpler language that is easier to understand. He identified such a target language in λ -calculus and experimented with this idea by giving a complete translation of ALGOL60 into λ -calculus [64]. Moreover, he declared in [65] that a programming language is nothing more than λ -calculus plus some “syntactic sugar”. Böhm was the first to use λ -calculus as an effective programming language, defining, with W. Gross, the CUCH language, which is a mixture of λ -calculus and the Curry combinators language, and showing how to represent in it the most common data structures [19].

But, until the end of the 1960s, λ -calculus suffered from the lack of a formal semantics. In fact, while it was possible to codify in it all the computable functions, the meaning of a generic λ -term not related to this coding was unclear. The attempt to interpret λ -terms as set-theoretic functions failed, since it would have been necessary to interpret it into a set D isomorphic to the set of functions from D to D , which is impossible since the two spaces always have different cardinality. Scott [88, 89] solved the problem by interpreting λ -calculus in a lattice isomorphic to the space of its continuous functions,

thus giving it a clear mathematical interpretation. So the technique of interpretation by translation, first developed by Landin, became a standard tool to study the denotational semantics of programming languages; almost all textbooks in denotational semantics follow this approach [91, 98].

But there was a gap between λ -calculus and the real functional programming languages. The majority of real functional languages have a “call-by-value” parameter passing policy, i.e., parameters are evaluated before being passed to a function, while the reduction rule of λ -calculus reflects a “call-by-name” policy, i.e., a policy where parameters are passed without being evaluated. In the folklore there was the idea that a call-by-value behaviour could be mimicked in λ -calculus just by defining a suitable reduction strategy. Plotkin proved that this intuition was wrong and that λ -calculus is intrinsically call-by-name [78]. So, in order to describe the call-by-value evaluation, he proposed a different calculus, which has the same syntax as λ -calculus, but a different reduction rule.

The aim of this book is to introduce both the call-by-name and the call-by-value λ -calculi and to study their syntactical and semantical properties, on which their status of paradigmatic programming languages is based. In order to study them in a uniform way we present a new calculus, the $\lambda\Delta$ -calculus, whose reduction rule is parametric with respect to a subset Δ of terms (called the set of input values) that enjoy some suitable conditions. Different choices of Δ allow us to define different languages, in particular the two λ -calculus variants we are speaking about. The most interesting feature of $\lambda\Delta$ -calculus is that it is possible to prove important properties (like confluence) for a large class of languages in just one step. We think that $\lambda\Delta$ -calculus can be seen as the foundation of functional programming.

Organization of the Book

The book is divided into four parts, each one composed of different chapters. The first part is devoted to the study of the syntax of $\lambda\Delta$ -calculus. Some syntactical properties, like confluence and standardization, can be studied for the whole Δ class. Other properties, like solvability and separability, cannot be treated in a uniform way, and they are therefore introduced separately for different instances of Δ .

In the second part the operational semantics of $\lambda\Delta$ -calculus is studied. The notion of operational semantics can be given in a parametric way, by supplying not only a set of input values but also a set of output values Θ , enjoying some very natural properties. A universal reduction machine is defined, parametric into both Δ and Θ , enjoying a sort of correctness property in the sense that, if a term can be reduced to an output value, then the machine stops, returning a term operationally equivalent to it. Then four particular reduction machines are presented, three for the call-by-name λ -calculus and one for the call-by-value λ -calculus, thereby presenting four operational behaviours that

are particularly interesting for modeling programming languages. Moreover, the notion of extensionality is revised, giving a new parametric definition that depends on the operational semantics we want to consider.

The third part is devoted to denotational semantics. The general notion of a model of $\lambda\Delta$ -calculus is defined, and then the more restrictive and useful notion of a filter model, based on intersection types, is given. Then four particular filter models are presented, each one correct with respect to one of the operational semantics studied in the previous part. For two of them completeness is also proved. The other two models are incomplete: we prove that there are no filter models enjoying the completeness property with respect to given operational semantics, and we build two complete models by using a technique based on intersection types. Moreover, the relation between the filter models and Scott's models is given.

The fourth part deals with the computational power of $\lambda\Delta$ -calculus. It is well known that λ -calculus is Turing complete, in both its call-by-name and call-by-value variants, i.e. it has the power of the computable functions. Here we prove something more, namely that each one of the reduction machines we present in the third part of this book can be used for computing all the computable functions.

Use of the Book

This book is dedicated to researchers, and it can be used as a textbook for master's or PhD courses in Foundations of Computer Science. Moreover, we wish to advise the reader that its aim is not to cover all possible topics concerning λ -calculus, but just those syntactical and semantics properties which can be used as tools for the foundation of programming languages. The reader interested in studying λ -calculus in itself can use the classical textbook by Barendregt [9], or other more descriptive ones such as [51] or [60]. The reader interested in a typed approach can read Mitchell's text [69] for an introduction, in which two chapters are dedicated to simply typed λ -calculus and its model, and the book of Hindley for a complete development of the topic [49].

Acknowledgement. Both authors would like to thank all the people of the “lambda-group” at the Dipartimento di Informatica of the Università di Torino for their support and collaboration. Moreover they are grateful to Roger Hindley and Elaine Pimentel for pointing out some inaccuracies. Luca Paolini thanks Pino Rosolini for the useful and interesting discussions about the topics of this book. Simona Ronchi Della Rocca did the final revision of the book during a sabbatical period. Some friends offered her hospitality and a stimulating scientific environment: Betti Venneri, Gigi Liquori, Rocco De Nicola, Pierre Lescanne and Philippe De Groote. To all of them she wants to

express her gratitude. Last but not least, both the authors thank the publisher Ingeborg Mayer, whose patient assistance made possible the publication of this book.

Torino, May 2004

Simona Ronchi Della Rocca
Luca Paolini

Contents

Part I. Syntax

1.	The Parametric λ-Calculus	3
1.1	The Language of λ -Terms	3
1.2	The $\lambda\Delta$ -Calculus	6
1.2.1	Proof of Confluence and Standardization Theorems	14
1.3	Δ -Theories	21
2.	The Call-by-Name λ-Calculus	25
2.1	The Syntax of $\lambda\Lambda$ -Calculus	25
2.1.1	Proof of Λ -Solvability Theorem	27
2.1.2	Proof of Böhm's Theorem	28
3.	The Call-by-Value λ-Calculus	35
3.1	The Syntax of the $\lambda\Gamma$ -Calculus	35
3.1.1	$\Xi\ell$ -Confluence and $\Xi\ell$ -Standardization	41
3.1.2	Proof of Potential Γ -Valuability and Γ -Solvability Theorems	43
3.1.3	Proof of Γ -Separability Theorem	49
3.2	Potentially Γ -Valuable Terms and Λ -Reduction	58
4.	Further Reading	61

Part II. Operational Semantics

5.	Parametric Operational Semantics	65
5.1	The Universal Δ -Reduction Machine	70
6.	Call-by-Name Operational Semantics	73
6.1	H-Operational Semantics	73
6.2	N-Operational Semantics	77
6.3	L-Operational Semantics	81
6.3.1	An Example	85

7. Call-by-Value Operational Semantics	89
7.1 V-Operational Semantics	89
7.1.1 An Example	93
8. Operational Extensionality	95
8.1 Operational Semantics and Extensionality	95
8.1.1 Head-Discriminability	99
9. Further Reading	101

Part III. Denotational Semantics

10. $\lambda\Delta$-Models	105
10.1 Filter $\lambda\Delta$ -Models	108
11. Call-by-Name Denotational Semantics	119
11.1 The Model \mathcal{H}	119
11.1.1 The \leq_∞ -Intersection Relation	129
11.1.2 Proof of the \mathcal{H} -Approximation Theorem	132
11.1.3 Proof of Semiseparability, \mathcal{H} -Discriminability and \mathcal{H} -Characterization Theorems	136
11.2 The Model \mathcal{N}	144
11.2.1 The \leq_\bowtie -Intersection Relation	151
11.2.2 Proof of \mathcal{N} -Approximation Theorem	154
11.2.3 Proof of \mathcal{N} -Discriminability and \mathcal{N} -Characterization Theorems	157
11.3 The Model \mathcal{L}	162
11.3.1 Proof of \mathcal{L} -Approximation Theorem	168
11.3.2 Proof of Theorems 11.3.15 and 11.3.16	170
11.4 A Fully Abstract Model for the L-Operational Semantics ..	172
11.5 Crossing Models	178
11.5.1 The Model \mathcal{H}	178
11.5.2 The Model \mathcal{N}	179
11.5.3 The Model \mathcal{L}	179
12. Call-by-Value Denotational Semantics	181
12.1 The Model \mathcal{V}	181
12.1.1 The \leq_\vee -Intersection Relation	190
12.1.2 Proof of Theorem 12.1.6	192
12.1.3 Proof of the \mathcal{V} -Approximation Theorem	195
12.1.4 Proof of Theorems 12.1.24 and 12.1.25	198
12.2 A Fully Abstract Model for the V-Operational Semantics ..	201

13. Filter $\lambda\Delta$-Models and Domains	207
13.1 Domains	207
13.1.1 \mathcal{H} as Domain	214
13.1.2 \mathcal{N} as Domain	216
13.1.3 \mathcal{L} as Domain	217
13.1.4 \mathcal{V} as Domain	218
13.1.5 Another Domain	219
14. Further Reading	221

Part IV. Computational Power

15. Preliminaries	225
15.1 Kleene's Recursive Functions	225
15.2 Representing Data Structures	227
16. Representing Functions	233
16.1 Call-by-Name Computational Completeness	233
16.2 Call-by-Value Computational Completeness	237
16.3 Historical Remarks	239
Bibliography	241
Index	247

Part I

Syntax

1. The Parametric λ -Calculus

A calculus is a language equipped with some reduction rules. All the calculi we consider in this book share the same language, which is the language of λ -calculus, while they differ each other in their reduction rules. In order to treat them in an uniform way we define a parametric calculus, the $\lambda\Delta$ -calculus, which gives rise to different calculi by different instantiations of the parameter Δ . In Part I we study the syntactical properties of the $\lambda\Delta$ -calculus, and in particular those of its two most important instances, the call-by-name and the call-by-value λ -calculi. The $\lambda\Delta$ -calculus has been introduced first in [85] and further studied in [74]. We use the terminology of [9].

1.1 The Language of λ -Terms

Definition 1.1.1 (The language Λ).

Let Var be a countable set of variables. The set Λ of λ -terms is a set of words on the alphabet $\text{Var} \cup \{ (,) , . , \lambda \}$ inductively defined as follows:

- $x \in \text{Var}$ implies $x \in \Lambda$,
- $M \in \Lambda$ and $x \in \text{Var}$ implies $(\lambda x.M) \in \Lambda$ (abstraction),
- $M \in \Lambda$ and $N \in \Lambda$ implies $(MN) \in \Lambda$ (application).

λ -terms will be ranged over by Latin capital letters. Sets of λ -terms will be denoted by Greek capital letters.

Sometimes, we will refer to λ -terms simply as terms. The symbol \equiv will denote syntactical identity of terms. We will use the following abbreviations in order to avoid an excessive number of parenthesis: $\lambda x_1\dots x_n.M$ will stand for $(\lambda x_1.(\dots(\lambda x_n.M)\dots))$, and $MN_1N_2\dots N_n$ will stand for $(\dots((MN_1)N_2)\dots N_n)$. Moreover, \vec{M} will denote a sequence of terms M_1, \dots, M_n for some $n \geq 0$, and $\lambda \vec{x}.M$ and $\vec{M}\vec{N}$ will denote respectively $\lambda x_1\dots x_n.M$ and $M_1\dots M_m N_1\dots N_n$ for some $n, m \geq 0$. The length of the sequence \vec{N} is denoted by $\|\vec{N}\|$. By abusing the notation, $N \in \vec{N}$ denotes that the term N occurs in the sequence \vec{N} .

Example 1.1.2. $\lambda x.xx$, $\lambda x.x(\lambda z.zy)$, $\lambda y.(\lambda x.x)(\lambda u v.u)$ are examples of λ -terms. Some λ -terms have standard names for historical reasons. The names

that will be extensively used in this book are:

$$I \equiv \lambda x.x, K \equiv \lambda xy.x, O \equiv \lambda xy.y, D \equiv \lambda x.xx, E \equiv \lambda xy.xy.$$

Definition 1.1.3 (Subterms).

A term N is a subterm of M if and only if one of the following conditions arises:

- $M \equiv N$,
- $M \equiv \lambda x.M'$ and N is a subterm of M' ,
- $M \equiv PQ$ and N is a subterm either of P or of Q .

A term N occurs in a term M if and only if N is a subterm of M .

Example 1.1.4. The set of subterms of the term $\lambda x.x(\lambda z.zy)$ is

$$\{\lambda x.x(\lambda z.zy), x(\lambda z.zy), \lambda z.zy, zy, x, z, y\}.$$

The symbol “ λ ” plays the role of binder for variables, as formalized in the next definition.

Definition 1.1.5 (Free variables).

- (i) The set of free variables of a term M , denoted by $\text{FV}(M)$, is inductively defined as follows:
 - $M \equiv x$ implies $\text{FV}(M) = \{x\}$,
 - $M \equiv \lambda x.M'$ implies $\text{FV}(M) = \text{FV}(M') - \{x\}$,
 - $M \equiv PQ$ implies $\text{FV}(M) = \text{FV}(P) \cup \text{FV}(Q)$.
 A variable is bound in M when it is not free in M .
- (ii) A term M is closed if and only if $\text{FV}(M) = \emptyset$. A term is open if it is not closed. For every subset of terms $\Theta \subseteq \Lambda$, we will denote with Θ^0 the restriction of Θ to closed terms.

Example 1.1.6. $\text{FV}(\lambda z.(\lambda x.x(\lambda z.zy))(\lambda xyz.yz)) = \{y\}$, $\text{FV}(\lambda z.x(\lambda x.xy)) = \{x, y\}$, and $\text{FV}((\lambda yx.x)y) = \{y\}$.

The replacement of a free variable by a term is the basic syntactical operation on Λ on which the definition of reduction rules will be based. But the replacement must respect the status of the variables: e.g., x can be replaced by $M \equiv \lambda y.zy$ in $\lambda u.xu$, so obtaining the term $\lambda u.(\lambda y.zy)u$, while the same replacement cannot take place in the term $\lambda z.xz$, since in the obtained term $\lambda z.(\lambda y.zy)z$ the free occurrence of z in M would become bound. The notion is formalized in the next definition.

Definition 1.1.7. The statement “ M is free for x in N ” is defined by induction on N as follows:

- M is free for x in x ;
- M is free for x in y ;
- If M is free for x both in P and Q then M is free for x in PQ ;

- If M is free for x in N and $x \not\equiv y$ and $y \notin \text{FV}(M)$
then M is free for x in $\lambda y.N$.

Example 1.1.8. $\lambda xy.xz$ is free for x and y in $(\lambda u.x)(\lambda u.xz)$ but is not free for u in both $\lambda xz.u$ and $\lambda zu.u$.

Let M be free for x in N ; so $N[M/x]$ denotes the *simultaneous replacement* of all free occurrences of x in N by M . Clearly,

$$\text{FV}(N[M/x]) = \begin{cases} \text{FV}(N) & \text{if } x \notin \text{FV}(N), \\ (\text{FV}(N) - \{x\}) \cup \text{FV}(M) & \text{otherwise.} \end{cases}$$

For example, $(\lambda x.u(xy))[xy/u]$ is not defined because xy is not free for u in $\lambda x.u(xy)$, while $(\lambda x.u(xu))[u(\lambda z.z)/u] \equiv \lambda x.u(\lambda z.z)(xu(\lambda z.z))$.

Let $\|\vec{N}\| = \|\vec{x}\|$; both $\vec{M}[N_1/x_1, \dots, N_n/x_n]$ and $\vec{M}[\vec{N}/\vec{x}]$ are abbreviations for the simultaneous replacement of x_i by N_i in every M_j ($0 \leq i \leq \|\vec{x}\| = n$, $0 \leq j \leq \|\vec{M}\|$).

In the standard mathematical notation, the name of a bound variable is meaningless; for example, $\sum_{1 \leq i \leq n} i$ and $\sum_{1 \leq j \leq n} j$ both denote the sum of the first n natural numbers. Also in the language Λ , it is natural to consider the terms modulo names of bound variables. The renaming is formalized in the next definition.

Definition 1.1.9 (α -Reduction).

- $\lambda x.M \rightarrow_\alpha \lambda y.M[y/x]$ if y is free for x in M and $y \notin \text{FV}(M)$.
- $=_\alpha$ is the reflexive, symmetric, transitive and contextual closure of \rightarrow_α .

Example 1.1.10. $\lambda x.x =_\alpha \lambda y.y =_\alpha \lambda z.z$, $\lambda xy.x =_\alpha \lambda xz.x$ and $\lambda xy.x =_\alpha \lambda yx.y$. On the other hand, $\lambda x.y \neq_\alpha \lambda x.x$ and $\lambda x.yx \neq_\alpha \lambda y.yy$.

In the entire book, we will consider terms modulo $=_\alpha$.

Thus we can also safely extend the notation $N[M/x]$ to the case where M is not free for x in N . In this case $N[M/x]$ denotes the result of replacing x by M in a term $N' =_\alpha N$ such that M is free for x in N' . Clearly such an N' always exists and the notation is well posed. So $(\lambda x.u(xy))[xy/u]$ is α -equivalent to the term $\lambda z.xy(zy)$.

An alternative way of denoting a simultaneous replacement is by explicitly using the notion of substitution. A *substitution* is a function from variables to terms. If s is a substitution and $\text{FV}(M) = \{x_1, \dots, x_n\}$, $s(M)$ denotes $M[s(x_1)/x_1, \dots, s(x_n)/x_n]$.

An important syntactical tool that will be extensively used in the following chapters is the notion of *context*. Informally, a context is a term that can contain some occurrences of a *hole* (denoted by the constant $[.]$) that can be filled by a term.

Definition 1.1.11 (Context).

Let Var be a countable set of variables, and $[.]$ be a constant (the hole).

- (i) The set Λ_C of contexts is a set of words on $\text{Var} \cup \{ (,) , . , \lambda , [.] \}$ inductively defined as follows:
 - $[.] \in \Lambda_C$;
 - $x \in \text{Var}$ implies $x \in \Lambda_C$;
 - $C[.] \in \Lambda_C$ and $x \in \text{Var}$ implies $(\lambda x.C[.]) \in \Lambda_C$;
 - $C_1[.] \in \Lambda_C$ and $C_2[.] \in \Lambda_C$ implies $(C_1[.]C_2[.]) \in \Lambda_C$.
 Contexts will be denoted by $C[.]$, $C'[.]$, $C_1[.]$,
- (ii) A context of the shape: $(\lambda \vec{x}.[.])\vec{P}$ is a head context.
- (iii) Let $C[.]$ be a context and M be a term. Then $C[M]$ denotes the term obtained by replacing by M every occurrence of $[.]$ in $C[.]$.

We will use the same abbreviation notation for contexts that we used for terms.

Note that filling a hole in a context is *not* a substitution; in fact, free variables in M can become bound in $C[M]$. For example, filling the hole of $\lambda x.[.]$ with the free variable x results in the term $\lambda x.x$.

1.2 The $\lambda\Delta$ -Calculus

We will present some λ -calculi, all based on the language Λ , defined in the previous section, each one characterized by different reduction rules.

The $\lambda\Delta$ -calculus is the language Λ equipped with a set $\Delta \subseteq \Lambda$ of input values, satisfying some closure conditions. Informally, input values represent partially evaluated terms that can be passed as parameters. Call-by-name and call-by-value parameter passing can be seen as the two most radical choices; parameters are not evaluated in the former policy, while in the latter they are evaluated until an output result is reached.

Most of the known variants of λ -calculus can be obtained from this parametric calculus by instantiating Δ in a suitable way. The set Δ of input values and the reduction \rightarrow_Δ induced by it are defined in Definition 1.2.1.

Definition 1.2.1. Let $\Delta \subseteq \Lambda$.

- (i) The Δ -reduction (\rightarrow_Δ) is the contextual closure of the following rule:

$$(\lambda x.M)N \rightarrow M[N/x] \text{ if and only if } N \in \Delta.$$

$(\lambda x.M)N$ is called a Δ -redex (or simply redex), and $M[N/x]$ is called its Δ -contractum (or simply contractum).

- (ii) \rightarrow_{Δ}^* and $=_{\Delta}$ are respectively the reflexive and transitive closure of \rightarrow_{Δ} and the symmetric, reflexive and transitive closure of \rightarrow_{Δ} .
- (iii) A set $\Delta \subseteq \Lambda$ is said set of input values when the following conditions are satisfied:
 - $\text{Var} \subseteq \Delta$ (Var-closure);
 - $P, Q \in \Delta$ implies $P[Q/x] \in \Delta$, for each $x \in \text{Var}$ (substitution closure);
 - $M \in \Delta$ and $M \rightarrow_{\Delta} N$ imply $N \in \Delta$ (reduction closure).
- (iv) A term is in Δ -normal form ($\Delta\text{-nf}$) if it has not Δ -redexes and it has a Δ -normal form, or it is Δ -normalizing if it reduces to a Δ -normal form; the set of Δ -nf is denoted by $\Delta\text{-NF}$.
- (v) A term is Δ -strongly normalizing if it is Δ -normalizing, and moreover there is not an infinite Δ -reduction sequence starting from it.

The closure conditions on the set of input values need some comment. Since, as already said, input values represent partially evaluated terms, it is natural to ask that this partial evaluation is preserved by reduction, which is the rule on which the evaluation process is based. The substitution closure comes naturally from the fact that variables always belong to the set of input values.

In this book the symbol Δ will denote a generic set of input values. We will omit the prefix Δ in cases where it is clear from the context.

Example 1.2.2. Let I, K, O, D be the terms defined in the Example 1.1.2, and let M, N be input values. Then $IM \rightarrow_{\Delta} M$, so I has the behaviour of the identity function, $KMN \rightarrow_{\Delta}^* M$, $OMN \rightarrow_{\Delta}^* N$, $DM \rightarrow_{\Delta} MM$. If $D \in \Delta$ then $DD \rightarrow_{\Delta}^* DD$.

Now some possible sets of input values will be defined.

- Definition 1.2.3.** (i) $\Gamma = \text{Var} \cup \{\lambda x.M \mid M \in \Lambda\}$.
 (ii) Λ_I is the language obtained from the grammar generating Λ , given in Definition 1.1.1, by modifying the formation rule for abstraction in the following way:

$$(\lambda x.M) \in \Lambda_I \text{ if and only if } M \in \Lambda \text{ and } x \in \text{Var} \text{ and } x \text{ occurs in } M.$$

The next property shows that there exists some set of input values, although not all sets of terms are sets of input values.

- Property 1.2.4.** (i) Λ is a set of input values.
 (ii) Γ is a set of input values.
 (iii) Λ_I is a set of input values.
 (iv) $\Lambda\text{-NF}$ is not a set of input values.
 (v) $\text{Var} \cup \Lambda\text{-NF}^0$ is a set of input values.
 (vi) $\Upsilon = \text{Var} \cup \{\lambda x.P \mid x \in \text{FV}(P)\}$ is not a set of input values.

Proof. The first case is obvious. In cases 2, 3, and 5, it is easy to check that the closure properties of Definition 1.2.1 are satisfied. Λ -NF is not closed under substitution. It is easy to see that Υ is closed under substitution, but it is not closed under reduction. In fact, $\lambda x.KIx \in \Upsilon$, while $\lambda x.KIx \rightarrow_{\Upsilon} \lambda x.I \notin \Upsilon$. \square

The choice $\Delta = \Lambda$ gives rise to the classical call-by-name λ -calculus [25], while $\Delta = \Gamma$ gives rise to a *pure* version (i.e. without constants) of the call-by-value λ -calculus, first defined by Plotkin [78].

The fact that $\text{Var} \cup \Lambda\text{-NF}^0$ is a correct set of input values was first noticed in [39].

It is easy to check that every term M has the following shape:

$$\lambda x_1 \dots x_n. \zeta M_1 \dots M_m \quad (n, m \geq 0),$$

where $M_i \in \Lambda$ are the *arguments* of M ($1 \leq i \leq m$) and ζ is the *head* of M . Here ζ is either a variable (*head variable*) or an application of the shape $(\lambda z.P)Q$, which can be either a redex (*head redex*) or not (*head block*), depending on the fact that Q belongs or not to the set Δ .

The natural interpretation of an abstraction term $\lambda x.M$ is a function whose formal parameter is x . The interpretation of an application $(\lambda x.M)N$, when $N \in \Delta$, is the application of the function $\lambda x.M$ to the actual parameter N , and so the Δ -reduction rule models the replacement of the formal parameter x by the actual parameter N in the body M of the function. Thus the Δ -normal form of a term, if it exists, can be seen as the final result of a computation.

The following fundamental theorem implies that this interpretation is correct, i.e. if the computation process stops, then the result is unique.

Theorem 1.2.5 (Confluence). [26, 74]

Let $M \rightarrow_{\Delta}^{*} N_1$ and $M \rightarrow_{\Delta}^{*} N_2$. There is Q such that both $N_1 \rightarrow_{\Delta}^{*} Q$ and $N_2 \rightarrow_{\Delta}^{*} Q$.

Proof. The proof is in Sect. 1.2.1. \square

Corollary 1.2.6. *The Δ -normal form of a term, if it exists, is unique.*

Proof. Assume by absurdum that a term M has two different normal forms M_1 and M_2 . Then, by the confluence theorem, there is a term N such that both M_1 and M_2 Δ -reduce to N , against the hypothesis that both are normal forms. \square

It is natural to ask if the closure conditions on input values, given in Definition 1.2.1, are necessary in order to assure the confluence of the calculus. It can be observed that they are not strictly necessary, but a weaker version of them is needed.

Let $P \in \Delta$ be such that, for every $Q \not\equiv P$ such that $P \rightarrow_{\Delta}^* Q$, $Q \notin \Delta$. Thus $(\lambda x.M)P$ reduces both to $M[P/x]$ and to $(\lambda x.M)Q$, which do not have a common reduct, since the last term will be never a redex. Thus the weaker version of *reduction closure* that is necessary is the following: $M \in \Delta$ and $M \rightarrow_{\Delta}^* N$ imply that there is $P \in \Delta$ such that $N \rightarrow_{\Delta}^* P$.

On the other hand, let $N, P \in \Delta$ but for all Q such that $N[P/x] \rightarrow_{\Delta}^* Q$, $Q \notin \Delta$. Thus $(\lambda x.(\lambda y.M)N)P$ reduces both to $(\lambda y.M[P/x])N[P/x]$ and to $(M[N/y])[P/x]$, which do not have a common reduct. Thus the weaker version of the *substitution closure* that is necessary is the following: $P, Q \in \Delta$ implies there is $R \in \Delta$ such that $P[Q/x] \rightarrow_{\Delta}^* R$.

Assume $M \rightarrow_{\Delta}^* N$, and assume that there is more than one Δ -reduction sequence from M to N . The standardization theorem says that, in case the set of input values enjoys a particular property, there is a “standard” reduction sequence from M to N , reducing the redexes in a given order.

Let us introduce formally the notion of standard reduction sequence.

Definition 1.2.7. (i) A symbol λ in a term M is active if and only if it is the first symbol of a Δ -redex of M .

(ii) The Δ -sequentialization $(M)^\circ$ of a term M is a function from Λ to Λ defined as follows:

- $(xM_1\dots M_m)^\circ = x(M_1)^\circ\dots(M_m)^\circ$;
- $((\lambda x.P)QM_1\dots M_m)^\circ = (\lambda x.P)^\circ(Q)^\circ(M_1)^\circ\dots(M_m)^\circ$, if $Q \in \Delta$;
- $((\lambda x.P)QM_1\dots M_m)^\circ = (Q)^\circ(\lambda x.P)^\circ(M_1)^\circ\dots(M_m)^\circ$, if $Q \notin \Delta$;
- $(\lambda x.P)^\circ = \lambda x.(P)^\circ$.

(iii) The degree of a redex R in M is the numbers of λ 's that both are active in M and occur on the left of $(R)^\circ$ in $(M)^\circ$.

(iv) The principal redex of M , if it exists, is the redex of M with minimum degree. The principal reduction $M \rightarrow_{\Delta}^p N$ denotes that N is obtained from M by reducing the principal redex of M . Moreover, $\rightarrow_{\Delta}^{*p}$ is the reflexive and transitive closure of \rightarrow_{Δ}^p .

(v) A sequence $M \equiv P_0 \rightarrow_{\Delta} P_1 \rightarrow_{\Delta} \dots \rightarrow_{\Delta} P_n \rightarrow_{\Delta} N$ is standard if and only if the degree of the redex contracted in P_i is less than or equal to the degree of the redex contracted in P_{i+1} , for every $i < n$.

We denote by $M \rightarrow_{\Delta}^{\circ} N$ a standard reduction sequence from M to N .

It is important to notice that the degree of a redex can change during the reduction; in particular, the redex of minimum degree always has degree zero. Moreover, note that the reduction sequences of length 0 and 1 are always standard. It is easy to check that, for every M , the Λ -sequentialization is $(M)^\circ \equiv M$; thus in this case the redex of degree 0 is always the leftmost one.

Example 1.2.8. (i) Let $\Delta = \Lambda$, and let $M \equiv (\lambda x.x(KI))(II)$. Thus M has degree 0, KI has degree 1 and II has degree 2 (in the term M). The following reduction sequence is standard:

$$(\lambda x.x(KI))(II) \rightarrow_{\Lambda} (II)(KI) \rightarrow_{\Lambda} I(KI) \rightarrow_{\Lambda} I(\lambda y.y).$$

- (ii) Let M be as before, and let $\Delta = \Gamma$. Thus II has degree 0, and KI has degree 1. Note that now M is no more a redex. The following reduction sequence is standard:
- $$(\lambda x.x(KI))(II) \rightarrow_{\Gamma} (\lambda x.x(KI))I \rightarrow_{\Gamma} I(KI) \rightarrow_{\Gamma} I(\lambda y.I) \rightarrow_{\Gamma} \lambda y.I.$$
- (iii) Let M be as before, and let $\Delta = \text{Var} \cup \Delta\text{-NF}^0$. Thus KI has degree 0 and II has degree 1. Also in this case M is not a redex. The following reduction sequence is standard:
- $$(\lambda x.x(KI))(II) \rightarrow_{\Delta} (\lambda x.x(KI))I \rightarrow_{\Delta} (\lambda x.x(\lambda y.I))I.$$

The notion of a standard set of input values, which is given in Definition 1.2.9 is key for having the standardization property.

Definition 1.2.9 (Standard input values).

A set Δ of input values is standard if and only if $M \notin \Delta$ and $M \rightarrow_{\Delta}^* N$ by reducing at every step a not principal redex imply $N \notin \Delta$.

Now the standardization property can be stated.

Theorem 1.2.10 (Standardization). [74]

Let Δ be standard. $M \rightarrow_{\Delta}^* N$ implies there is a standard reduction sequence from M to N .

Proof. The proof is in Sect. 1.2.1. □

The next property shows that some sets of input values are standard, while some are not standard.

Property 1.2.11. (i) Λ and Γ are standard.

(ii) $\text{Var} \cup \Delta\text{-NF}^0$ is standard, for every Δ .

(iii) Λ_I is not standard.

Proof. (i) Λ is trivially standard. Let us consider Γ ; we will prove that, if $M \notin \Gamma$, and $M \rightarrow_{\Gamma} N$ through a not principal reduction, then $N \notin \Gamma$.

$M \notin \Gamma$ implies that M has one of the following shapes:

1. $yM_1\dots M_m$ ($m > 1$).

2. $(\lambda x.M_1)M_2\dots M_m$ ($m \geq 2$) and either $(\lambda x.M_1)M_2$ is a redex or it is a head block.

Case 1 is trivial, since M can never be reduced to a term in Γ .

In case 2, if $M_2 \in \Gamma$ then the principal redex is $(\lambda x.M_1)M_2$, while if $M_2 \notin \Gamma$ then if $M_2 \notin \Gamma\text{-NF}$ the principal redex is in M_2 ; if $M_2 \in \Gamma\text{-NF}$ then the principal redex is in some M_j ($j \leq 3$). So the reduction of a not principal redex cannot produce a term belonging to Γ .

(ii) $\text{Var} \cup \Delta\text{-NF}^0$ is standard since not principal reductions preserve the presence of the redex of minimum degree.

(iii) Consider the term, $M \equiv \lambda x.x(DD)((\lambda z.I)I)$. Clearly $M \notin \Lambda_I$ and the principal redex of M is DD . So $M \rightarrow_{\Lambda_I} \lambda x.x(DD)I \in \Lambda_I$ and in this reduction the reduced redex is not principal, while for every sequence of $\rightarrow_{\Lambda_I}^{*p}$ reductions; $M \rightarrow_{\Lambda_I}^{*p} M \notin \Lambda_I$. □

It is easy to see that the substitution closure on input values, given in Definition 1.2.1, is necessary in order to assure the standardization property.

In fact, let $M, N \in \Delta$ and $M[N/x] \notin \Delta$. The following non-standard reduction sequence $(\lambda x.I M)N \rightarrow_{\Delta} (\lambda x.M)N \rightarrow_{\Delta} M[N/x]$ does not have a standard counterpart, in fact $I(M[N/x]) \not\rightarrow_{\Delta} M[N/x]$.

Theorem 1.2.12. *The condition that Δ is standard is necessary and sufficient for the $\lambda\Delta$ -calculus enjoys the standardization property.*

Proof. The sufficiency of the condition is a consequence of the Standardization Theorem. To prove its necessity, assume Δ is not standard; we can find a term $M \notin \Delta$ such that $M \rightarrow_{\Delta}^* N \in \Delta$, without reducing the principal redex. Hence $I M \rightarrow_{\Delta} I N \rightarrow_{\Delta} N$, by reducing first a redex of degree different from 0 and then a redex of degree 0. Clearly, there is no way of commuting the order of reductions. \square

An important consequence of the standardization property is the fact that the reduction sequence reducing, at every step, the principal redex is normalizing, as shown in Corollary 1.2.13.

Corollary 1.2.13. *Let Δ be standard.*

If $M \rightarrow_{\Delta}^ N$ and N is a normal form then $M \rightarrow_{\Delta}^{*p} N$.*

Proof. By Corollary 1.2.6 and by the definition of the standard set of input values. \square

Example 1.2.14. (i) Let $\Delta = \Lambda$. The term $KI(DD)$ has Λ -normal form I . In fact, the principal Λ -reduction sequence is $KI(DD) \rightarrow_{\Lambda} (\lambda y.I)(DD) \rightarrow_{\Lambda} I$, while the Λ -reduction sequence choosing at every step the rightmost Λ -redex never stops. Notice that, if we choose $\Delta = \Gamma$, $KI(DD)$ has not Γ -normal form.

- (ii) The term $II(II(II))$ is Λ -strongly normalizing and Γ -strongly normalizing, while $KI(DD)$ is neither Λ -strongly normalizing nor Γ -strongly normalizing.
- (iii) Let $\Delta = \text{Var} \cup \Lambda\text{-NF}^0$. Thus $I(K(xx))$ is the Δ -normal form of term $I(II)(K(xx))$.

Remark 1.2.15. The first notion of standardization was given, for the $\lambda\Lambda$ -calculus, by Curry and Feys [34, 35]. With respect to their notion, if $M \rightarrow_{\Lambda}^* N$ then there is a standard reduction sequence from M to N , but this reduction sequence is not necessarily unique. For instance, $\lambda x.x(I)(II) \rightarrow_{\Lambda} \lambda x.xI(II) \rightarrow_{\Lambda} \lambda x.xII$ and $\lambda x.x(I)(II) \rightarrow_{\Lambda} \lambda x.x(II)I \rightarrow_{\Lambda} \lambda x.xII$ are both standard reduction sequences. Klop [58] introduced a notion of strong standardization, according to which, if $M \rightarrow_{\Lambda}^* N$, then there is a unique strongly standard reduction sequence from M to N , and he designed an algorithm for transforming a reduction sequence into a strongly standard one. According to his notion, in the example before only the first reduction sequence is

standard. Our definition, when restricted to the $\lambda\Lambda$ -calculus, is quite similar to the strong standardization. In fact, according to our definition, the standard reduction sequence is unique, but in some degenerated case: e.g. for $\Delta = \Lambda$, there are infinite reduction sequences from $x(DD)$ to $x(DD)$, each one performing a different number of Λ -reductions.

Plotkin [78] extended the notion of standardization to the $\lambda\Gamma$ -calculus. His notion of standardization is not strong using Klop's terminology. Our definition, when restricted to $\lambda\Gamma$ -calculus, is similar to a strong version of Plotkin's standardization. The advantage of our notion of standardization is the validity of Corollary 1.2.13, i.e. the fact that the principal reduction is Δ -normalizing.

A notion that will play an important role in what follows is that one of solvability.

Definition 1.2.16. (i) *An head context $(\lambda\vec{x}[\cdot])\vec{P}$ is Δ -valuable if and only if each $P \in \vec{P}$ is such that $P \in \Delta$.*
(ii) *A term M is Δ -solvable if and only if there is a Δ -valuable head context $C[\cdot] \equiv (\lambda\vec{x}[\cdot])\vec{N}$ such that:*

$$C[M] =_{\Delta} I.$$

(iii) *A term is Δ -unsolvable if and only if it is not Δ -solvable.*

Note that $(\lambda\vec{x}[\cdot])\vec{N} =_{\Delta} I$ means $(\lambda\vec{x}[\cdot])\vec{N} \rightarrow_{\Delta}^* I$, since I is in Δ -nf, for every Δ .

We will abbreviate Δ -solvable and Δ -unsolvable respectively as solvable and unsolvable, when the meaning is clear from the context. Informally speaking, a solvable term is a term that is in some sense *computationally meaningful*. In fact, let $M \in \Lambda^0$ be solvable, and let P be an input value; we can always find a sequence \vec{N} of terms such that $M\vec{N}$ reduces to P : just take the sequence \vec{Q} such that $M\vec{Q} =_{\Delta} I$, which exists since M is solvable, and pose $\vec{N} \equiv \vec{Q}P$. So a closed solvable term can mimic the behaviour of any term, if applied to suitable arguments.

It would be interesting to syntactically characterize the solvable terms. Unfortunately, there is not a general characterization for the $\lambda\Delta$ -calculus, so we will study this problem for some particular instances of Δ .

Example 1.2.17. (i) Consider the two sets of input values Λ and Γ . In both calculi, the term I is solvable, while DD is unsolvable. $\lambda x.x(DD)$ is an example of a term that is Λ -solvable and Γ -unsolvable. In fact, $(\lambda x.x(DD))O \rightarrow_{\Lambda}^* I$, while there is no term P such that $P(DD) \rightarrow_{\Gamma}^* I$, since $DD \notin \Gamma$ and $DD \rightarrow_{\Gamma}^* DD$.

(ii) Let Φ be the set of input values $\text{Var} \cup \Lambda\text{-NF}^0$. Then $I(\lambda x.I(xx)) \in \Phi\text{-NF}$ is a Φ -unsolvable term.

In order to understand the behaviour of unsolvable terms, it is important to stress some of their closure properties.

- Property 1.2.18.* (i) The unsolvability is preserved by substitution of variables by input values.
(ii) The unsolvability is preserved by Δ -valuable head contexts.

Proof. Let M be unsolvable.

- (i) By contraposition let us assume $M[P/z]$ to be solvable for some input values P . Then there is a Δ -valuable head context $C[.] \equiv (\lambda\vec{x}[.])\vec{Q}$ such that $C[M[P/z]] \rightarrow_{\Delta}^{*} I$.

Without loss of generality, we can assume $\|\vec{Q}\| > \|\vec{x}\|$. Indeed, in the case $\|\vec{Q}\| \leq \|\vec{x}\|$, we can choose a closed solvable term N such that there is \vec{R} such that $N\vec{R} \rightarrow_{\Delta}^{*} I$ and $\|\vec{R}\| = \|\vec{x}\| - \|\vec{Q}\|$, and then consider the Δ -valuable context $C[.]N\vec{R}$. So let $\vec{Q} \equiv \vec{Q}_1\vec{Q}_2$, where $\|\vec{Q}_1\| = \|\vec{x}\|$.

$(\lambda\vec{x}.M[P/z])\vec{Q}_1\vec{Q}_2 \rightarrow_{\Delta}^{*} I$ implies $(\lambda\vec{x}.(\lambda z.M)P)\vec{Q}_1\vec{Q}_2 \rightarrow_{\Delta}^{*} I$ (since $P \in \Delta$). This in turn implies $(\lambda z.(\lambda\vec{x}.M)\vec{Q}_1)(P[\vec{Q}_1/\vec{x}])\vec{Q}_2 \rightarrow_{\Delta}^{*} I$ and $(\lambda z\vec{x}.M)(P[\vec{Q}_1/\vec{x}])\vec{Q}_1\vec{Q}_2 \rightarrow_{\Delta}^{*} I$, because by α -equivalence we can assume $z \notin \text{FV}(Q_1)$ and $z \notin \vec{x}$. But $P[\vec{Q}_1/\vec{x}] \in \Delta$ (since input values are closed under substitution) which means that the Δ -valuable head context $C'[.] \equiv (\lambda z\vec{x}[.])(P[\vec{Q}_1/\vec{x}])\vec{Q}_1\vec{Q}_2$ is such that $C'[M] \rightarrow_{\Delta}^{*} I$.

- (ii) By contraposition let us assume $C'[M]$ to be solvable for some Δ -valuable head context $C'[.] \equiv (\lambda\vec{z}[.])\vec{P}$. Then there is a Δ -valuable head context $C[.] \equiv (\lambda\vec{x}[.])\vec{Q}$, such that $C[C'[M]] \rightarrow_{\Delta}^{*} I$. If $\vec{z} \equiv \vec{z}_0\vec{z}_1$ and $\|\vec{P}\| = \|\vec{z}_0\|$ then $C[C'[M]] \rightarrow_{\Delta}^{*} C[\lambda\vec{z}_1.M[\vec{P}/\vec{z}_0]] \rightarrow_{\Delta}^{*} I$, thus $M[\vec{P}/\vec{z}_0]$ is solvable, and by the previous part of this property M is also solvable. Otherwise $\vec{P} \equiv \vec{P}_0\vec{P}_1$, $\|\vec{P}_1\| > 1$ and $\|\vec{P}_0\| = \|\vec{z}\|$. Thus

$$C[C'[M]] \rightarrow_{\Delta}^{*} C[M[\vec{P}_0/\vec{z}]\vec{P}_1] \equiv (\lambda\vec{x}.M[\vec{P}_0/\vec{z}]\vec{P}_1)\vec{Q} \rightarrow_{\Delta}^{*} I.$$

Without loss of generality we can assume $\|\vec{Q}\| > \|\vec{x}\|$, $\vec{Q} \equiv \vec{Q}_0\vec{Q}_1$ and $\|\vec{Q}_0\| = \|\vec{x}\|$. So

$$\begin{aligned} (\lambda\vec{x}.M[\vec{P}_0/\vec{z}]\vec{P}_1)\vec{Q} &\rightarrow_{\Delta}^{*} (M[\vec{P}_0/\vec{z}]\vec{P}_1)[\vec{Q}_0/\vec{x}]\vec{Q}_1 \equiv \\ &(M[\vec{P}_0/\vec{z}][\vec{Q}_0/\vec{x}])(\vec{P}_1[\vec{Q}_0/\vec{x}])\vec{Q}_1 \rightarrow_{\Delta}^{*} I, \end{aligned}$$

which implies $(M[\vec{P}_0/\vec{z}][\vec{Q}_0/\vec{x}])$ solvable. Again the proof follows from part (i) of this property. \square

We will see that in all the calculi we will study in the following, the property of solvability is not preserved by either substitution or by head contexts. As an example in the $\lambda\Lambda$ -calculus xD is Λ -solvable, but $xD[D/x]$ is not Λ -solvable.

1.2.1 Proof of Confluence and Standardization Theorems

Both the proofs are based on the notion of parallel reduction.

Definition 1.2.19. Let Δ be a set of input values.

- (i) The deterministic parallel reduction \hookrightarrow_{Δ} is inductively defined as follows:
 - 1. $x \hookrightarrow_{\Delta} x$;
 - 2. $M \hookrightarrow_{\Delta} N$ implies $\lambda x.M \hookrightarrow_{\Delta} \lambda x.N$;
 - 3. $M \hookrightarrow_{\Delta} M'$, $N \hookrightarrow_{\Delta} N'$ and $N \in \Delta$ imply $(\lambda x.M)N \hookrightarrow_{\Delta} M'[N'/x]$;
 - 4. $M \hookrightarrow_{\Delta} M'$, $N \hookrightarrow_{\Delta} N'$ and $N \notin \Delta$ imply $MN \hookrightarrow_{\Delta} M'N'$.
- (ii) The nondeterministic parallel reduction \Rightarrow_{Δ} is inductively defined as follows:
 - 1. $x \Rightarrow_{\Delta} x$;
 - 2. $M \Rightarrow_{\Delta} N$ implies $\lambda x.M \Rightarrow_{\Delta} \lambda x.N$;
 - 3. $M \Rightarrow_{\Delta} M'$, $N \Rightarrow_{\Delta} N'$ and $N \in \Delta$ imply $(\lambda x.M)N \Rightarrow_{\Delta} M'[N'/x]$;
 - 4. $M \Rightarrow_{\Delta} M'$, $N \Rightarrow_{\Delta} N'$ imply $MN \Rightarrow_{\Delta} M'N'$.

Roughly speaking, the deterministic parallel reduction reduces in one step all the redexes present in a term, while the nondeterministic one reduces a subset of them.

Example 1.2.20. Let $M \equiv I(II)$. If $\Delta \equiv \Lambda$ then $M \hookrightarrow_{\Delta} I$, while $M \Rightarrow_{\Delta} M$, $M \Rightarrow_{\Delta} II$ and $M \Rightarrow_{\Delta} I$. If $\Delta \equiv \Gamma$ then $M \hookrightarrow_{\Delta} II$ while $M \Rightarrow_{\Delta} M$ and $M \Rightarrow_{\Delta} II$.

The following lemma shows the relation between the \Rightarrow_{Δ} and \rightarrow_{Δ} reductions.

Lemma 1.2.21. Let Δ be a set of input values.

- (i) $M \rightarrow_{\Delta} N$ implies $M \Rightarrow_{\Delta} N$.
- (ii) $M \Rightarrow_{\Delta} N$ implies $M \rightarrow_{\Delta}^* N$.
- (iii) \rightarrow_{Δ}^* is the transitive closure of \Rightarrow_{Δ} .

Proof. Easy. □

\Rightarrow_{Δ} enjoys a useful substitution property.

Lemma 1.2.22. Let $M \Rightarrow_{\Delta} M'$ and $N \Rightarrow_{\Delta} N'$.

If $N \in \Delta$ then $M[N/x] \Rightarrow_{\Delta} M'[N'/x]$.

Proof. By induction on M . Let us prove just the most difficult case, i.e. the term M is a Δ -redex. Let $M \equiv (\lambda z.P)Q$, $Q \in \Delta$, $P \Rightarrow_{\Delta} P'$, $Q \Rightarrow_{\Delta} Q'$ and $M' \equiv P'[Q'/z]$. By induction $P[N/x] \Rightarrow_{\Delta} P'[N'/x]$ and $Q[N/x] \Rightarrow_{\Delta} Q'[N'/x]$, where $Q'[N'/x] \in \Delta$ for the closure conditions on Δ . Thus

$$\begin{aligned} ((\lambda z.P)Q)[N/x] &\equiv (\lambda z.P[N/x])Q[N/x] \Rightarrow_{\Delta} \\ P'[N'/x][Q'[N'/x]/z] &\equiv (P'[Q'/z])[N'/x] \end{aligned}$$

by point 3 of the definition of \Rightarrow_{Δ} . □

The next property, whose proof is obvious, states that, for every term M , there is a unique term N such that $M \hookrightarrow_{\Delta} N$.

Property 1.2.23. $M \hookrightarrow_{\Delta} P$ and $M \hookrightarrow_{\Delta} Q$ implies $P \equiv Q$.

Proof. Trivial. \square

Let $[M]_{\Delta}$ be the term such $M \hookrightarrow_{\Delta} [M]_{\Delta}$. In the literature $[M]_{\Delta}$ is called the *complete development* of M (see [93]). The following lemma holds.

Lemma 1.2.24. $M \Rightarrow_{\Delta} N$ implies $N \Rightarrow_{\Delta} [M]_{\Delta}$.

Proof. By induction on M .

- If $M \equiv x$, then $N \equiv x$ and $[M]_{\Delta} \equiv x$.
- If $M \equiv \lambda x.P$ then $N \equiv \lambda x.Q$, for some Q such that $P \Rightarrow_{\Delta} Q$. By induction $Q \Rightarrow_{\Delta} [P]_{\Delta}$, and so $N \Rightarrow_{\Delta} \lambda x.[P]_{\Delta} \equiv [M]_{\Delta}$.
- If $M \equiv P_1 P_2$ and it is not a Δ -redex, then $N \equiv Q_1 Q_2$ for some Q_1 and Q_2 such that $P_1 \Rightarrow_{\Delta} Q_1$ and $P_2 \Rightarrow_{\Delta} Q_2$. So, by induction, $Q_1 \Rightarrow_{\Delta} [P_1]_{\Delta}$ and $Q_2 \Rightarrow_{\Delta} [P_2]_{\Delta}$, which implies $N \Rightarrow_{\Delta} [P_1]_{\Delta} [P_2]_{\Delta} \equiv [M]_{\Delta}$.
- If $M \equiv (\lambda x.P_1)P_2$ is a redex (i.e. $P_2 \in \Delta$) then either $N \equiv (\lambda x.Q_1)Q_2$ or $N \equiv Q_1[P_2/x]$, for some Q_i such that $P_i \Rightarrow_{\Delta} Q_i$ ($1 \leq i \leq 2$). By induction, $Q_i \Rightarrow_{\Delta} [P_i]_{\Delta}$ ($1 \leq i \leq 2$). Note that $[P_2]_{\Delta} \in \Delta$ by Lemma 1.2.21.(ii). In both cases, $N \Rightarrow_{\Delta} [P_1]_{\Delta} [[P_2]_{\Delta}/x] \equiv [M]_{\Delta}$, in the former case simply by induction, and in the latter both by induction and by Lemma 1.2.22. \square

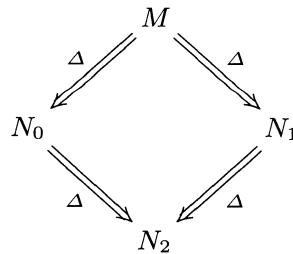


Fig. 1.1. Diamond property.

The proof of confluence follows the Takahashi pattern [93], which is a simplification of the original proof made by Taït and Martin Löf for classical $\lambda\Lambda$ -calculus. It is based on the property that a reduction that is the transitive closure of another one enjoying the Diamond Property is confluent.

Lemma 1.2.25 (Diamond property of \Rightarrow_{Δ}).

If $M \Rightarrow_{\Delta} N_0$ and $M \Rightarrow_{\Delta} N_1$ then there is N_2 such that both $N_0 \Rightarrow_{\Delta} N_2$ and $N_1 \Rightarrow_{\Delta} N_2$.

Proof. By Lemma 1.2.24, $M \Rightarrow_{\Delta} N$ implies $N \Rightarrow_{\Delta} [M]_{\Delta}$. So, if $M \Rightarrow_{\Delta} M_1$ and $M \Rightarrow_{\Delta} M_2$, then both $M_1 \Rightarrow_{\Delta} [M]_{\Delta}$ and $M_2 \Rightarrow_{\Delta} [M]_{\Delta}$, as shown in Fig. 1.1 (pag. 15). \square

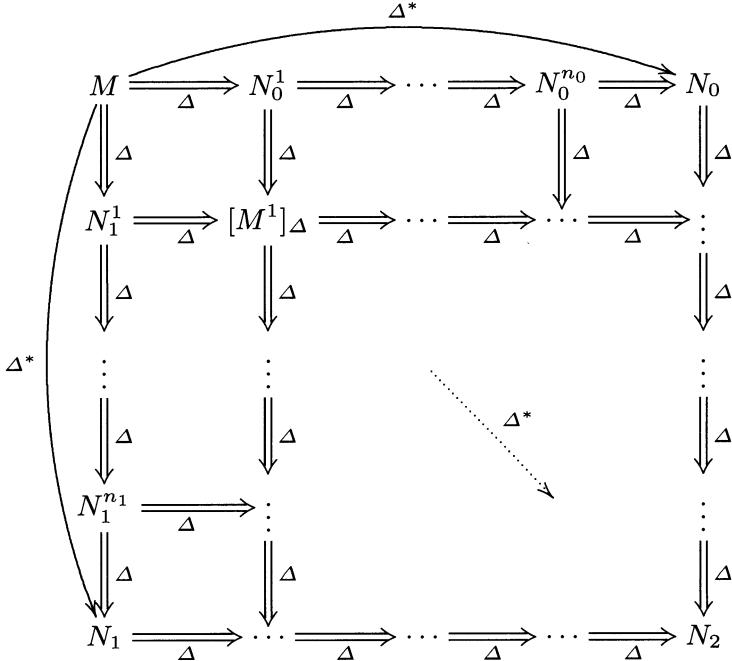


Fig. 1.2. Diamond closure.

▼ Proof of Confluence Theorem (Theorem 1.2.5 pag. 8).

By Property 1.2.21.(iii), \rightarrow_{Δ}^* is the transitive closure of \Rightarrow_{Δ} . This means that there are $N_0^1, \dots, N_0^{n_0}, N_1^1, \dots, N_1^{n_1}$ ($n_0, n_1 \geq 1$) such that $M \Rightarrow_{\Delta} N_0^1 \dots \Rightarrow_{\Delta} N_0^{n_0} \Rightarrow_{\Delta} N_0$ and $M \Rightarrow_{\Delta} N_1^1 \dots \Rightarrow_{\Delta} N_m^{n_1} \Rightarrow_{\Delta} N_1$. Then the proof follows by repeatedly applying the diamond property of \Rightarrow_{Δ} (diamond closure), as shown in Fig. 1.2. \blacksquare

The rest of this subsection is devoted to the proof of the standardization theorem. First, we need to establish some technical results.

Let $M \Rightarrow_{\Delta}^o N$ denote “ $M \rightarrow_{\Delta}^o N$ and $M \Rightarrow_{\Delta} N$ ”.

The following lemma, at the point (ii), shows that a nondeterministic parallel reduction can always be transformed into a standard reduction sequence.

Lemma 1.2.26. Let \vec{P}, \vec{Q} be two sequences of terms such that $\|\vec{P}\| = \|\vec{Q}\|$; moreover, let $P_i \in \Delta$ and $P_i \Rightarrow_{\Delta}^{\circ} Q_i$ for all $i \leq \|\vec{P}\|$.

- (i) If $M \Rightarrow_{\Delta}^{\circ} N$ then $M[\vec{P}/\vec{x}] \Rightarrow_{\Delta}^{\circ} N[\vec{Q}/\vec{x}]$.
- (ii) If $M \Rightarrow_{\Delta} N$ then $M \Rightarrow_{\Delta}^{\circ} N$.

Proof. Parts (i) and (ii) can be proved by mutual induction on M .

- (i) By Lemma 1.2.22, $M[\vec{P}/\vec{x}] \Rightarrow_{\Delta} N[\vec{Q}/\vec{x}]$, hence it suffices to show that $M[\vec{P}/\vec{x}] \rightarrow_{\Delta}^{\circ} N[\vec{Q}/\vec{x}]$.

Let $M \equiv \lambda y_1 \dots y_h . \zeta M_1 \dots M_m$ ($h, m \in \mathbb{N}$), where either ζ is a variable or $\zeta \equiv (\lambda z.T)U$.

If $h > 0$, then the proof follows by induction.

Let $h = 0$, thus $N \equiv \xi N_1 \dots N_m$ such that $\zeta \Rightarrow_{\Delta}^{\circ} \xi$ and $M_i \Rightarrow_{\Delta}^{\circ} N_i$; furthermore, let $M'_i \equiv M_i[\vec{P}/\vec{x}]$ and $N'_i \equiv N_i[\vec{Q}/\vec{x}]$ ($1 \leq i \leq m$).

The proof is organized according to the possible shapes of ζ .

1. Let ζ be a variable. If $m = 0$ then the proof is trivial, so let $m > 0$. There are two cases to be considered.

- 1.1. $\zeta \notin \vec{x}$, so $\xi[\vec{Q}/\vec{x}] \equiv \zeta$. By induction $M_i[\vec{P}/\vec{x}] \rightarrow_{\Delta}^{\circ} N_i[\vec{Q}/\vec{x}]$ and the standard reduction sequence is

$$\zeta M'_1 \dots M'_m \rightarrow_{\Delta}^{\circ} \zeta N'_1 M'_2 \dots M'_m \rightarrow_{\Delta}^{\circ} \dots \rightarrow_{\Delta}^{\circ} \zeta N'_1 \dots N'_m.$$

- 1.2. $\zeta \equiv x_j \in \vec{x}$ ($1 \leq j \leq l$), so $\xi[\vec{Q}/\vec{x}] \equiv Q_j$. But $P_j \Rightarrow_{\Delta}^{\circ} Q_j$ means that there is a standard sequence $P_j \equiv S_0 \rightarrow_{\Delta} \dots \rightarrow_{\Delta} S_n \equiv Q_j$ ($n \in \mathbb{N}$). Two cases can arise.

- 1.2.1. $\forall i \leq n, S_i \not\equiv \lambda z.S'$. Then the following reduction sequence

$$\sigma : S_0 M'_1 \dots M'_m \rightarrow_{\Delta} \dots \rightarrow_{\Delta} S_n M'_1 \dots M'_m$$

is standard. Since by induction $M_i[\vec{P}/\vec{x}] \rightarrow_{\Delta}^{\circ} N_i[\vec{Q}/\vec{x}]$, there is a standard reduction sequence

$$\tau : S_n M'_1 \dots M'_m \rightarrow_{\Delta}^{\circ} S_n N'_1 M'_2 \dots M'_m \rightarrow_{\Delta}^{\circ} \dots \rightarrow_{\Delta}^{\circ} S_n N'_1 \dots N'_m.$$

Note that $S_0 M'_1 \dots M'_m \equiv M[\vec{P}/\vec{x}]$ and $S_n N'_1 \dots N'_m \equiv N[\vec{Q}/\vec{x}]$, so σ followed by τ is the desired standard reduction sequence.

- 1.2.2. There is a minimum $k \leq n$ such that $S_k \equiv \lambda z.S'$.

By induction on (ii), $M_1 \Rightarrow_{\Delta}^{\circ} N_1$. Therefore, by induction $M_1[\vec{P}/\vec{x}] \rightarrow_{\Delta}^{\circ} N_1[\vec{Q}/\vec{x}]$, where $M_1[\vec{P}/\vec{x}] \rightarrow_{\Delta}^{\circ} N_1[\vec{Q}/\vec{x}]$ is $M_1[\vec{P}/\vec{x}] \equiv R_0 \rightarrow_{\Delta} \dots \rightarrow_{\Delta} R_p \equiv N_1[\vec{Q}/\vec{x}]$ ($p \in \mathbb{N}$). There are two subcases:

1.2.2.1. $\forall i \leq p$, $R_i \notin \Delta$. Then the following reduction sequence:

$$\begin{aligned}\sigma' : M[\vec{P}/\vec{x}] &\equiv S_0 R_0 M'_2 \dots M'_m \rightarrow_{\Delta} \dots \rightarrow_{\Delta} S_k R_0 M'_2 \dots M'_m \\ &\rightarrow_{\Delta} \dots \rightarrow_{\Delta} S_k R_p M'_2 \dots M'_m \\ &\rightarrow_{\Delta} S_{k+1} R_p M'_2 \dots M'_m \rightarrow_{\Delta} \dots \rightarrow_{\Delta} S_n R_p M'_2 \dots M'_m\end{aligned}$$

is also standard. Moreover, since $M_i[\vec{P}/\vec{x}] \rightarrow_{\Delta}^{\circ} N_i[\vec{P}/\vec{x}]$, the following reduction sequence:

$$\begin{aligned}\tau' : S_n R_p M'_2 \dots M'_m &\rightarrow_{\Delta}^{\circ} \\ S_n R_p N'_2 M'_3 \dots M'_m &\rightarrow_{\Delta}^{\circ} \dots \rightarrow_{\Delta}^{\circ} S_n R_p N'_2 \dots N'_m\end{aligned}$$

is also standard. Clearly σ' followed by τ' is the desired standard reduction sequence.

1.2.2.2. There is a minimum $q \leq p$ such that $R_q \in \Delta$. So

$$\begin{aligned}\sigma'' : M[\vec{P}/\vec{x}] &\equiv S_0 R_0 M'_2 \dots M'_m \rightarrow_{\Delta} \dots \rightarrow_{\Delta} S_k R_0 M'_2 \dots M'_m \\ &\rightarrow_{\Delta} \dots \rightarrow_{\Delta} S_k R_q M'_2 \dots M'_m \rightarrow_{\Delta} S_{k+1} R_q M'_2 \dots M'_m \\ &\rightarrow_{\Delta} \dots \rightarrow_{\Delta} S_n R_q M'_2 \dots M'_m \rightarrow_{\Delta} \dots \rightarrow_{\Delta} S_n R_p M'_2 \dots M'_m\end{aligned}$$

is a standard reduction sequence. The desired standard reduction sequence is σ'' followed by τ' .

2. Let $\zeta \equiv (\lambda z.T)U$. Thus $N \equiv (\lambda z.\bar{T})\bar{U}N_1 \dots N_m$ or $N \equiv \bar{T}[\bar{U}/z]N_1 \dots N_m$, where $T \Rightarrow_{\Delta} \bar{T}$, $U \Rightarrow_{\Delta} \bar{U}$ and $M_i \Rightarrow_{\Delta} N_i$ ($1 \leq i \leq m$).

By induction, $U' \equiv U[\vec{P}/\vec{x}] \Rightarrow_{\Delta}^{\circ} \bar{U}[\vec{Q}/\vec{x}] \equiv U''$, $T' \equiv T[\vec{P}/\vec{x}] \Rightarrow_{\Delta}^{\circ} \bar{T}[\vec{Q}/\vec{x}] \equiv T''$ and $M'_i \equiv M_i[\vec{P}/\vec{x}] \Rightarrow_{\Delta}^{\circ} N_i[\vec{Q}/\vec{x}] \equiv N'_i$ ($1 \leq i \leq m$).

Let $U' \equiv R_0 \rightarrow_{\Delta} \dots \rightarrow_{\Delta} R_p \equiv U''$ ($p \in \mathbb{N}$) be the standard sequence $U' \rightarrow_{\Delta}^{\circ} U''$. Without loss of generality let us assume $z \notin \vec{x}$.

2.1. Let $N \equiv (\lambda z.\bar{T})\bar{U}N_1 \dots N_m$. There are two cases.

2.1.1. $\forall i \leq p$ $R_i \notin \Delta$. Then the standard reduction sequence $M[\vec{P}/\vec{x}] \rightarrow_{\Delta}^{\circ} N[\vec{Q}/\vec{x}]$ is

$$\begin{aligned}(\lambda z.T')R_0 M'_1 \dots M'_m &\rightarrow_{\Delta} \dots \rightarrow_{\Delta} (\lambda z.T')R_p M'_1 \dots M'_m \\ &\rightarrow_{\Delta}^{\circ} (\lambda z.T'')R_p M'_1 \dots M'_m \rightarrow_{\Delta}^{\circ} (\lambda z.T'')R_p N'_1 M'_2 \dots M'_m \\ &\quad \rightarrow_{\Delta}^{\circ} \dots \rightarrow_{\Delta}^{\circ} (\lambda z.T'')R_p N'_1 \dots N'_m.\end{aligned}$$

2.1.2. There is a minimum $q \leq p$ such that $R_q \in \Delta$. Thus the desired standard reduction sequence is:

$$\begin{aligned}(\lambda z.T')R_0 M'_1 \dots M'_m &\rightarrow_{\Delta} \dots \rightarrow_{\Delta} (\lambda z.T')R_q M'_1 \dots M'_m \\ &\rightarrow_{\Delta}^{\circ} (\lambda z.T'')R_q M'_1 \dots M'_m \rightarrow_{\Delta} \dots \rightarrow_{\Delta} (\lambda z.T'')R_p M'_1 \dots M'_m \\ &\rightarrow_{\Delta}^{\circ} (\lambda z.T'')R_p N'_1 M'_2 \dots M'_m \rightarrow_{\Delta}^{\circ} \dots \rightarrow_{\Delta}^{\circ} (\lambda z.T'')R_p N'_1 \dots N'_m.\end{aligned}$$

2.2. Let $N \equiv \bar{T}[\bar{U}/z]N_1\dots N_m$. So, there is a minimum $q \leq p$ such that $R_q \in \Delta$; let μ be the standard reduction sequence:

$$\begin{aligned} M[\vec{P}/\vec{x}] &\equiv (\lambda z.T')R_0M'_1\dots M'_m \rightarrow_{\Delta} \dots \rightarrow_{\Delta} (\lambda z.T')R_qM'_1\dots M'_m \\ &\quad \rightarrow_{\Delta} T'[R_q/z]M'_1\dots M'_m. \end{aligned}$$

$T \Rightarrow_{\Delta}^o \bar{T}$, by induction on (ii). Furthermore, since $R_q \Rightarrow_{\Delta}^o U''$, it follows by induction that $T[\vec{P}/\vec{x}][R_q/z] \Rightarrow_{\Delta}^o \bar{T}[\vec{Q}/\vec{x}][U''/z]$.

Let $T[\vec{P}/\vec{x}][R_q/z] \equiv T_0 \rightarrow_{\Delta} \dots \rightarrow_{\Delta} T_t \equiv \bar{T}[\vec{Q}/\vec{x}][U''/z]$ be the corresponding standard reduction sequence. Two subcases can arise:

2.2.1. $\forall i \leq t, T_i \not\equiv \lambda z.S'$. The desired standard reduction sequence is μ followed by:

$$\begin{aligned} T'[R_p/z]M'_1\dots M'_m &\equiv T[\vec{P}/\vec{x}][R_p/z]M'_1\dots M'_m \rightarrow_{\Delta} T_1M'_1\dots M'_m \\ &\rightarrow_{\Delta} \dots \rightarrow_{\Delta} T_tM'_1\dots M'_m \rightarrow_{\Delta}^o \dots \rightarrow_{\Delta}^o T_tN'_1\dots N'_m \equiv [\vec{Q}/\vec{x}] \end{aligned}$$

2.2.2. Let $k \leq t$ be the minimum index such that $T_k \equiv \lambda y.T'_k$.

The construction of the standard reduction sequence depends on the fact that M_2 may or may not become an input value, but, in every case, it can be easily built as in the previous cases.

(ii) The cases $M \equiv x$ and $M \equiv \lambda z.M'$ are easy.

1. Let $M \equiv PQ \Rightarrow_{\Delta} P'Q' \equiv N, P \Rightarrow_{\Delta} P'$ and $Q \Rightarrow_{\Delta} Q'$.

By induction, there are standard sequences $P \equiv P_0 \rightarrow_{\Delta} \dots \rightarrow_{\Delta} P_p \equiv P'$ and $Q \equiv Q_0 \rightarrow_{\Delta} \dots \rightarrow_{\Delta} Q_q \equiv Q'$.

If $\forall i \leq p, P_i \not\equiv \lambda z.P'_i$, then $M \rightarrow_{\Delta}^o N$ is $P_0Q_0 \rightarrow_{\Delta}^o P_pQ_0 \rightarrow_{\Delta}^o P_pQ_q$. Otherwise, let k be the minimum index such that $P_k \equiv \lambda z.P'_k$.

- If $\forall j \leq q, Q_j \notin \Delta$, then $M \rightarrow_{\Delta}^o N$ is

$$P_0Q_0 \rightarrow_{\Delta} \dots \rightarrow_{\Delta} P_kQ_0 \rightarrow_{\Delta}^o P_kQ_q \rightarrow_{\Delta} P_{k+1}Q_q \rightarrow_{\Delta} \dots \rightarrow_{\Delta} P_pQ_q.$$

- If there is a minimum h such that $Q_h \in \Delta$, the standard sequence is $P_0Q_0 \rightarrow_{\Delta}^o P_kQ_0 \rightarrow_{\Delta}^o P_kQ_h \rightarrow_{\Delta} P_{k+1}Q_h \rightarrow_{\Delta}^o P_pQ_h \rightarrow_{\Delta}^o P_pQ_q$.

2. Let $M \equiv (\lambda x.P)Q \Rightarrow_{\Delta} P'[Q'/x] \equiv N$ where $P \Rightarrow_{\Delta} P', Q \Rightarrow_{\Delta} Q'$ and $Q \in \Delta$. Hence $P \Rightarrow_{\Delta}^o P'$ and $Q \Rightarrow_{\Delta}^o Q'$ follow by induction, so $P[Q/x] \Rightarrow_{\Delta}^o P'[Q'/x]$, by induction on (i). Thus, the desired standard reduction sequence is $(\lambda x.P)Q \rightarrow_{\Delta} P[Q/x] \rightarrow_{\Delta}^o P'[Q'/x]$. \square

In order to prove the standardization theorem some auxiliary definitions are necessary.

Definition 1.2.27. Let $M, N \in \Lambda$.

- (i) $M \rightarrow_{\Delta}^i N$ denotes that N is obtained from M by reducing a redex that is not the principal redex.
- (ii) $M \Rightarrow_{\Delta}^i N$ denotes $M \Rightarrow_{\Delta} N$ and $M \rightarrow_{\Delta}^{*i} N$.

According to this new terminology, a set of input values is standard, in the sense of Definition 1.2.9 (pag. 10), if and only if $M \notin \Delta$ and $M \rightarrow_{\Delta}^{*i} N$ imply $N \notin \Delta$.

Lemma 1.2.28. $M \Rightarrow_{\Delta} N$ implies there is P such that $M \rightarrow_{\Delta}^{*p} P \Rightarrow_{\Delta}^i N$.

Proof. Trivial, by Lemma 1.2.26.(ii).

Notice that it can be $M \equiv P$, by definition of $\rightarrow_{\Delta}^{*p}$. \square

Example 1.2.29. Let $M \equiv (\lambda xy.I(\lambda z.IK(II)))I \Rightarrow_{\Gamma} \lambda yz.IKI$.

Therefore $M \rightarrow_{\Gamma}^p \lambda y.I(\lambda z.IK(II)) \rightarrow_{\Gamma}^p \lambda yz.IK(II) \Rightarrow_{\Gamma}^i \lambda yz.IKI$ and clearly $\lambda yz.IK(II) \in \Gamma$.

Note that if Δ is standard and R is the principal redex of M and $M \rightarrow_{\Delta}^{*i} N$, then R is the principal redex of N .

Lemma 1.2.30. Let Δ be standard.

$M \Rightarrow_{\Delta}^i P \rightarrow_{\Delta}^p N$ implies $M \rightarrow_{\Delta}^{*p} Q \Rightarrow_{\Delta}^i N$, for some Q .

Proof. By induction on M . If either $M \equiv \lambda x.M'$, or the head of M is a variable, then the proof follows by induction. Otherwise, let $M \equiv (\lambda y.M_0)M_1...M_m$; thus it must be $P \equiv (\lambda y.P_0)P_1...P_m$. Note that $M \Rightarrow_{\Delta}^i P$ implies $M_i \Rightarrow_{\Delta} P_i$ ($1 \leq i \leq m$). Now there are two cases, according to whether $P_1 \in \Delta$ or not.

- Let $P_1 \in \Delta$; it follows that P_1 is the argument of the principal redex of P , thus $N \equiv P_0[P_1/y]P_2...P_m$.

Let $M_1 \in \Delta$. Then we can build the following reduction sequence:

$M \equiv (\lambda y.M_0)M_1...M_m \rightarrow_{\Delta}^p M_0[M_1/y]...M_m \Rightarrow_{\Delta} P_0[P_1/y]P_2...P_m$, which can be transformed into a standard one by Lemma 1.2.28.

Let $M_1 \notin \Delta$ and $P_1 \in \Delta$; since the set Δ is standard, $M_1 \Rightarrow_{\Delta} P_1 \in \Delta$ if and only if $M_1 \rightarrow_{\Delta}^{*p} P'_1 \Rightarrow_{\Delta}^i P_1$, where $P'_1 \in \Delta$. But this would imply that in the reduction $M \Rightarrow_{\Delta}^i P$ the principal redex of M_1 has been reduced; but by definition the principal redex of M_1 coincides with the principal redex of M , against the hypothesis that $M \Rightarrow_{\Delta}^i P$. So this case is not possible.

- Let $P_1 \notin \Delta$. Then there is $j \geq 0$ such that the principal redex of P_j is the principal redex of P . Let $j \geq 2$; so $\forall k \leq j$ P_k is a normal form. So $N \equiv (\lambda y.P_0)P_1...P'_j...P_m$, where $P_j \rightarrow_{\Delta}^p P'_j$. From the hypothesis that $M \Rightarrow_{\Delta}^i P$, it follows that $M_i \equiv P_i$ ($0 \leq i \leq j-1$), and $M_i \Rightarrow_{\Delta} P_i$ ($j < i \leq m$). Then by induction there is P_j^* such that $M_j \rightarrow_{\Delta}^{*p} P_j^* \Rightarrow_{\Delta}^i P'_j$, and we can build the following reduction sequence:

$$(\lambda y.M_0)M_1...M_m \rightarrow_{\Delta}^{*p} (\lambda y.M_0)M_1...P_j^*P_{j+1}...P_m \Rightarrow_{\Delta} (\lambda y.M_0)M_1...P'_j...P_m$$

which can be transformed into a standard one by Lemma 1.2.28.

The case $j < 2$ is similar. \square

This Lemma has a key corollary.

Corollary 1.2.31. Let Δ be standard.

If $M \rightarrow_{\Delta}^{*} N$ then $M \rightarrow_{\Delta}^{*p} Q \xrightarrow{\underbrace{\Rightarrow_{\Delta}^i \dots \Rightarrow_{\Delta}^i}_{k}} N$, for some Q and some k .

Proof. Note that if $P \rightarrow_{\Delta} P'$ then $P \Rightarrow_{\Delta} P'$. So $M \rightarrow_{\Delta}^{*} N$ implies $M \Rightarrow_{\Delta} N_1 \Rightarrow_{\Delta} \dots \Rightarrow_{\Delta} N_n \Rightarrow_{\Delta} N$. So, by repeatedly applying Lemma 1.2.28 and Lemma 1.2.30 we reach the proof. \square

Now we are able to prove the standardization theorem.

▼ Proof of Standardization Theorem (Theorem 1.2.10 pag. 10).

The proof is given by induction on N . From Corollary 1.2.31, $M \rightarrow_{\Delta}^{*} N$ implies $M \rightarrow_{\Delta}^{*p} Q \rightarrow_{\Delta}^{*i} N$ for some Q . Obviously, the reduction sequence $\sigma : M \rightarrow_{\Delta}^{*p} Q$ is standard by definition of $\rightarrow_{\Delta}^{*p}$. Note that, by definition of $\rightarrow_{\Delta}^{*i}$, $Q \rightarrow_{\Delta}^{*i} N$ implies that Q and N have the same structure, i.e. $Q \equiv \lambda x_1 \dots x_n. \zeta Q_1 \dots Q_n$ and $N \equiv \lambda x_1 \dots x_n. \zeta' N_1 \dots N_n$, where $Q_i \rightarrow_{\Delta}^{*} N_i$ ($i \leq n$) and either ζ and ζ' are the same variable, or $\zeta \equiv (\lambda x.R)S$, $\zeta' \equiv (\lambda x.R')S'$, $R \rightarrow_{\Delta}^{*} R'$ and $S \rightarrow_{\Delta}^{*} S'$.

The case when ζ is a variable follows by induction. Otherwise, by induction there are standard reduction sequences $\sigma_i : Q_i \rightarrow_{\Delta}^{*} N_i$ ($1 \leq i \leq n$), $\tau_R : R \rightarrow_{\Delta}^{*} R'$ and $\tau_S : S \rightarrow_{\Delta}^{*} S'$. Let $S \equiv S_0 \rightarrow_{\Delta} \dots \rightarrow_{\Delta} S_k \equiv S'$ ($k \in \mathbb{N}$).

If $\forall i \leq k \ S_i \notin \Delta$ then the desired standard reduction sequence is σ followed by $\tau_S, \tau_R, \sigma_1, \dots, \sigma_n$.

Otherwise, there is $S_h \in \Delta$ ($h \leq k$). In this case, let $\tau_S^0 : S_0 \rightarrow_{\Delta} \dots \rightarrow_{\Delta} S_h$ and $\tau_S^1 : S_{h+1} \rightarrow_{\Delta} \dots \rightarrow_{\Delta} S_k$; the desired standard reduction sequence is σ followed by $\tau_S^0, \tau_R, \tau_S^1, \sigma_1, \dots, \sigma_n$. \blacksquare

1.3 Δ -Theories

In order to model computation, Δ -equality is too weak. As an example, let Δ be either Λ or Γ . If we want to model the termination property, both the terms DD and $(\lambda x.xxx)(\lambda x.xxx)$ represent programs that run forever, while the two terms are \neq_{Δ} each other. Indeed $DD \rightarrow_{\Delta} DD$ and $(\lambda x.xxx)(\lambda x.xxx) \rightarrow_{\Delta} (\lambda x.xxx)(\lambda x.xxx)(\lambda x.xxx)$. So it would be natural to consider them equal in this particular setting. But if we want to take into account not only termination but also the size of terms, they need to be different; in fact, the first one reduces to itself while the second increases its size during the reduction. As we will see in the following, for all instances of Δ we will consider, all interesting interpretations of the calculus also equate terms that are not $=_{\Delta}$.

Let us introduce the notion of Δ -theory.

Definition 1.3.1. (i) $T \subseteq \Lambda \times \Lambda$ is a congruence whenever:

- $(M, M) \in T$ for each $M \in \Lambda$,

- $(M, N) \in \mathcal{T}$ implies $(N, M) \in \mathcal{T}$,
 - $(M, P) \in \mathcal{T}$ and $(P, N) \in \mathcal{T}$ imply $(M, N) \in \mathcal{T}$,
 - $(M, N) \in \mathcal{T}$ implies $(C[M], C[N]) \in \mathcal{T}$, for all contexts $C[.]$.
- (ii) $\mathcal{T} \subseteq \Lambda \times \Lambda$ is a Δ -theory if and only if it is a congruence and $M =_{\Delta} N$ implies $(M, N) \in \mathcal{T}$.

We will denote $(M, N) \in \mathcal{T}$ also by $M =_{\mathcal{T}} N$.

Clearly a Δ -theory equating all terms would be completely uninteresting. So we will ask for consistency.

- Definition 1.3.2.** (i) A Δ -theory \mathcal{T} is consistent if and only if there are $M, N \in \Lambda$ such that $M \neq_{\mathcal{T}} N$. Otherwise \mathcal{T} is inconsistent.
- (ii) A Δ -theory \mathcal{T} is input consistent if and only if there are $M, N \in \Delta$ such that $M \neq_{\mathcal{T}} N$. Otherwise \mathcal{T} is input inconsistent.
- (iii) A Δ -theory \mathcal{T} is maximal if and only if it has no consistent extension, i.e. for all $M, N \in \Lambda$ such that $M \neq_{\mathcal{T}} N$, any Δ -theory \mathcal{T}' containing \mathcal{T} and such that $M =_{\mathcal{T}'} N$ is inconsistent.

Property 1.3.3. Let \mathcal{T} be a Δ -theory.

If \mathcal{T} is input consistent then it is consistent.

Proof. Obvious. □

In the last section of this book, we will see that in order to use a $\lambda\Delta$ -calculus for computing, we need to work inside theories that are both consistent and input consistent.

Δ -theories can be classified according to their behaviour with respect to the Δ -solvable terms.

- Definition 1.3.4.** (i) A Δ -theory is sensible if it equates all Δ -unsolvable terms.
- (ii) A Δ -theory is semisensible if it never equates a Δ -solvable term and a Δ -unsolvable term.

Another important notion for Δ -theories is that of separability. In fact, this property help us to understand what equalities cannot be induced by a theory.

Definition 1.3.5. Let Δ be a set of input values.

Two terms M, N are Δ -separable if and only if there is a context $C[.]$ such that $C[M] =_{\Delta} x$ and $C[N] =_{\Delta} y$ for two different variables x and y .

Property 1.3.6. Let M, N be Δ -separable.

If \mathcal{T} is a Δ -theory such that $M =_{\mathcal{T}} N$ then \mathcal{T} is input inconsistent.

Proof. Let $C[.]$ be the context separating M and N , i.e. $C[M] =_{\Delta} x$ and $C[N] =_{\Delta} y$ for two different variables x and y . Since $=_{\mathcal{T}}$ is a congruence, $M =_{\mathcal{T}} N$ implies $C[M] =_{\mathcal{T}} C[N]$, and so, since \mathcal{T} is closed under $=_{\Delta}$, $x =_{\mathcal{T}} y$. But this implies $\lambda xy.x =_{\mathcal{T}} \lambda xy.y$, i.e. $K =_{\mathcal{T}} O$. But, since $=_{\mathcal{T}}$ is a congruence, this implies $KMN =_{\mathcal{T}} OMN$ for all terms M, N . In particular, if $M, N \in \Delta$ then $M =_{\mathcal{T}} N$ by Δ -reduction. \square

A theory is *fully extensional* if all terms in it (not only abstractions) have a functional behaviour. So, in a fully extensional theory, the equality between terms must be extensional (in the usual sense), i.e., it must satisfy the property:

$$(\text{EXT}) \quad Mx = Nx \Rightarrow M = N \quad x \notin \text{FV}(M) \cup \text{FV}(N).$$

Clearly $=_{\Delta}$ does not satisfy (EXT). In fact, (EXT) holds for $=_{\Delta}$ only if it is restricted to terms that reduce to an abstraction: indeed, $xy =_{\Delta} (\lambda z.xz)y$, but $x \neq_{\Delta} \lambda z.xz$.

The least extensional extension of $=_{\Delta}$ is induced by the η -reduction rule, defined as follows:

Definition 1.3.7 (η -Reduction).

- (i) *The η -reduction (\rightarrow_{η}) is the contextual closure of the following rule:
 $\lambda x.Mx \rightarrow_{\eta} M$ if and only if $x \notin \text{FV}(M)$;
 $\lambda x.Mx$ is a η -redex and M is its contractum;*
- (ii) *$M \rightarrow_{\Delta\eta} N$ if N is obtained from M by reducing either a Δ or a η redex in M ;*
- (iii) *$\rightarrow_{\Delta\eta}^*$ and $=_{\Delta\eta}$ are respectively the reflexive and transitive closure of $\rightarrow_{\Delta\eta}$ and the symmetric, reflexive and transitive closure of $\rightarrow_{\Delta\eta}$.*

The next theorem shows an interesting result for η -reduction.

Theorem 1.3.8. $=_{\Delta\eta}$ is the least extensional extension of $=_{\Delta}$.

Proof. It is immediate to check that $=_{\Delta\eta}$ is extensional. In fact, for $x \notin \text{FV}(M)$, $Mx =_{\Delta\eta} Nx$ implies $\lambda x.Mx =_{\Delta\eta} \lambda x.Nx$ (since $=_{\Delta\eta}$ is a congruence), and this implies, $M =_{\Delta\eta} N$ by $=_{\eta}$.

On the other hand, let \mathcal{T} be a fully extensional Δ -theory, i.e. $Mx =_{\mathcal{T}} Nx$ implies $M =_{\mathcal{T}} N$. For $x \notin \text{FV}(M)$, $(\lambda x.Mx)x =_{\mathcal{T}} Mx$, since $(\lambda x.Mx)x \rightarrow_{\Delta} Mx$, and thus by (EXT), $\lambda x.Mx =_{\mathcal{T}} M$. So \mathcal{T} is closed under $=_{\eta}$. \square

In the literature, full extensionality is called simply extensionality. We use this name to stress the fact that it is also possible to define weaker notions of extensionality. We will develop this topic in Sect. 8.1.

2. The Call-by-Name λ -Calculus

A parameter passing policy is said to be *call-by-name* if the parameters need not be evaluated in order to be supplied to the function. In our setting, this means that all terms can be considered as input values. So, in order to mimic this policy with the parametric $\lambda\Delta$ -calculus, it is sufficient to define $\Delta = \Lambda$. Then all terms are input values, and every application of the shape $(\lambda x.M)N$ is a redex. The $\lambda\Lambda$ -calculus coincides with the standard λ -calculus, defined by Church [25], and the reduction \rightarrow_Λ is the well known β -reduction.

2.1 The Syntax of $\lambda\Lambda$ -Calculus

By the definition of \rightarrow_Λ , in the $\lambda\Lambda$ -calculus the head of a term is either a variable or a redex. If the head of M is a variable then M is in Λ -head normal form (Λ -hnf), namely M is of the shape $\lambda x_1\dots x_n.zM_1\dots M_m$ ($n, m \in \mathbb{N}$).

M has a Λ -head normal form if it reduces to a term in Λ -hnf. Λ -HNF denotes the set of all Λ -head normal forms.

It is easy to see that M is in Λ -normal form (Λ -nf) if and only if both its head is a variable and its arguments are in Λ -normal form too. So the set of terms having Λ -hnf strictly includes the set of terms having Λ -nf. Consider, for example, the term $\lambda x.x(DD)$; it is in Λ -hnf, but it does not have Λ -nf. An example of a term having neither Λ -hnf nor Λ -nf is DD .

A term is in Λ -lazy head normal form (Λ -lhnf) if and only if it is either an abstraction or a head normal form. A term has a Λ -lazy head normal form if and only if it reduces to a lazy head normal form. Λ -LHNF denotes the set of all Λ -lazy head normal forms. Clearly $\lambda x.DD$ is a Λ -lhnf, but it has neither Λ -hnf nor Λ -nf. In the literature, a Λ -lazy head normal form is called weak-head normal form. We changed this terminology to stress the fact that to reach a Λ -lazy head normal form it is not necessary to reduce the Λ -redexes that do not occur under the scope of a λ -abstraction. Both Λ -head normal forms and Λ -lazy head normal forms are important classes of terms from the computational point of view.

The general definition of Δ -solvability is given in Definition 1.2.16. In the $\lambda\Lambda$ -calculus, solvable terms have a very nice syntactical characterization.

Theorem 2.1.1 (Λ -Solvability).

A term is Λ -solvable if and only if it has a Λ -head normal form.

Proof. The proof is in Sect. 2.1.1. \square

Let us notice that the Λ -head normal form of a term is not unique. Consider $\lambda x.(\lambda uv.u)x(DD)(II)$. It reduces both to $\lambda x.x(II)$ and to $\lambda x.xI$, which are both Λ -head normal forms. But it is easy to show that all the Λ -hnf's obtained by Λ -reduction from the same term share some structural properties.

First we need to introduce some naming. If $M \equiv \lambda x_1 \dots x_n.zM_1 \dots M_m$, then n is the Λ -order of M and m is its Λ -degree.

Property 2.1.2. Let M be Λ -solvable. Then there are unique $n, m \in \mathbb{N}$ such that $M \rightarrow_{\Lambda}^* N$ and N in Λ -hnf, imply that the Λ -order and the Λ -degree of N are respectively n and m .

Proof. By contraposition, let M have two Λ -head normal forms with different Λ -order and Λ -degree, i.e. $M \rightarrow_{\Lambda}^* P_1 \equiv \lambda x_1 \dots x_n.xM_1 \dots M_m$ and $M \rightarrow_{\Lambda}^* P_2 \equiv \lambda x_1 \dots x_p.xN_1 \dots N_q$, where $n \neq p$ and/or $m \neq q$. By the confluence theorem, it must be a term Q such that both $P_1 \rightarrow_{\Lambda}^* Q$ and $P_2 \rightarrow_{\Lambda}^* Q$. But this is impossible, since the only redexes can occur in M_i or in N_j ($1 \leq i \leq m$, $1 \leq j \leq q$), and their reduction cannot change any of n, m, p, q . \square

The notion of Λ -order of a term can be easily extended to terms not in Λ -head normal form.

Definition 2.1.3. A term M has Λ -order n if and only if n is the largest i such that $M =_{\Lambda} \lambda x_1 \dots x_i.N$. If such an n does not exist M has Λ -order ∞ .

Example 2.1.4. DD and $xM_1 \dots M_m$ ($m \geq 0$) have Λ -order 0; while both $\lambda x_1 \dots x_n.DD$ and $\lambda x_1 \dots x_n.z$ have Λ -order n and $(\lambda xy.xx)(\lambda xy.xx)$ has Λ -order ∞ , since $\forall k \in \mathbb{N} (\lambda xy.xx)(\lambda xy.xx) \rightarrow_{\Lambda}^* \lambda x_0 \dots x_k.(\lambda xy.xx)(\lambda xy.xx)$.

A particularly interesting Λ -theory is the theory $\Lambda\eta$. The Λ -normal forms play an important role in this theory, as shown in the next theorem.

Theorem 2.1.5 (Böhm's theorem). [15]

Let $M, N \in \Lambda\text{-NF}$. If $M \neq_{\Lambda\eta} N$ then M and N are Λ -separable.

Proof. The proof is in Sect. 2.1.2. \square

Böhm's theorem has an interesting semantical consequence, namely that two Λ -nf's that are $\neq_{\Lambda\eta}$ cannot be equated in any consistent or input consistent Λ -theory (note that, for the $\lambda\Lambda$ -calculus, consistency and input consistency coincide).

Corollary 2.1.6. Let M, N be two Λ -normal forms and let $M \neq_{\Lambda\eta} N$. For every Λ -theory \mathcal{T} , if $M =_{\mathcal{T}} N$ then \mathcal{T} is (input) inconsistent.

Proof. The proof is identical to the proof of Property 1.3.6, just putting $\Delta = \Lambda$. \square

Given a Λ -theory, there is an easy way of proving its full extensionality, as shown in the following property.

Property 2.1.7. Let $E \equiv \lambda xy.xy$, and let \mathcal{T} be a Λ -theory.
 $I =_{\mathcal{T}} E$ if and only if \mathcal{T} is fully extensional.

Proof. (\Rightarrow) $I =_{\mathcal{T}} E$ implies $IM =_{\mathcal{T}} EM$, which implies, by Λ -reduction $M =_{\mathcal{T}} \lambda x.Mx$, where $x \notin \text{FV}(M)$. The proof follows, by Theorem 1.3.8.
(\Leftarrow) By Theorem 1.3.8 $x =_{\mathcal{T}} \lambda y.xy$ where $y \notin \text{FV}(M)$; so $\lambda x.x =_{\mathcal{T}} \lambda xy.xy$, since $=_{\mathcal{T}}$ is a congruence. \square

We will prove in Chap. 16 that the $\lambda\Lambda$ -calculus can be considered as a programming language, in the sense that it is possible to define some evaluation machines performing the Λ -reduction, and the $\lambda\Lambda$ -calculus equipped by each of this machines, has the computational power of all the partial computable functions. The key property on which this result is based is the fact that every term in the $\lambda\Lambda$ -calculus has a fixed point.

Theorem 2.1.8 (Call-by-name fixed point).

Every term $M \in \Lambda$ has a fixed point, i.e. for every term M there is a term N such that $MN =_{\Lambda} N$.

Proof. Let $Y \equiv \lambda x.(\lambda y.x(yy))(\lambda y.x(yy))$. It is readily checked that, for every M , $YM =_{\Lambda} M(YM)$. Hence YM is a fixed point of M . \square

The term Y in the proof of the previous theorem is called a *call-by-name fixed point operator* since, when applied to a term M , it produces one of its fixed points.

2.1.1 Proof of Λ -Solvability Theorem

First we need to prove a property.

Property 2.1.9. (i) The lack of Λ -hnf is preserved by substitution, i.e. if M does not have Λ -hnf then $M[N/y]$ does not have Λ -hnf either, for all $x \in \text{Var}$ and $N \in \Lambda$.
(ii) The lack of Λ -hnf is preserved by head contexts, i.e. if M does not have Λ -hnf then $(\lambda \vec{x}.M)\vec{N}$ does not have Λ -hnf either, for all \vec{x} and \vec{N} .

Proof. (i) By contraposition assume that $M[N/y]$ has Λ -hnf. We will prove that this implies that M has Λ -hnf too. The proof is given by induction on the length p of standard Λ -reduction sequence from $M[N/y]$ to its Λ -hnf. The cases $p = 0, 1$ are trivial. Let $p > 1$ and $M \equiv \lambda \vec{x}.(\lambda z.P)QM$, otherwise M is already in Λ -hnf, and let $R \equiv \lambda \vec{x}.P[Q/z]\vec{M}$. Then

$$M[N/y] \equiv \lambda \vec{x}.(\lambda z.P')Q'\vec{M}' \rightarrow_{\Lambda} \lambda \vec{x}.P'[Q'/z]\vec{M}' \equiv R[N/y],$$

where $P' \equiv P[N/y]$, $Q' \equiv Q[N/y]$ and $\vec{M}' \equiv \vec{M}[N/y]$. Thus $R[N/y]$ has Λ -hnf in less than p steps, so by induction R has Λ -hnf, and by the Church Rosser theorem M has Λ -hnf too.

- (ii) We assume that $(\lambda \vec{x}.M)\vec{N}$ has Λ -hnf and we prove that this implies that M has Λ -hnf too. The proof is given by induction on the length p of a standard Λ -reduction sequence from $(\lambda \vec{x}.M)\vec{N}$ to its Λ -hnf. The cases $p = 0, 1$ are trivial. Let $p > 1$ and $M \equiv \lambda \vec{y}.(\lambda z.P)Q\vec{M}$. If $\|\vec{y}\| + \|\vec{x}\| \geq \|\vec{N}\|$ then the proof follows from part (i) of this property and from the confluence property of the Λ -reduction. Otherwise $\exists \vec{N}_1$ such that $\|\vec{N}_1\| = \|\vec{y}\| + \|\vec{x}\|$, $\vec{N} \equiv \vec{N}_1 \vec{N}_2$ and $\|\vec{N}_2\| > 1$. In this case, it must be

$$(\lambda \vec{x}.(\lambda \vec{y}.(\lambda z.P)Q\vec{M}))\vec{N}_1 \vec{N}_2 \rightarrow_{\Lambda}^* (\lambda z.P')Q'\vec{M}'\vec{N}_2 \rightarrow_{\Lambda} P'[Q'/z]\vec{M}'\vec{N}_2,$$

where $P' \equiv P[\vec{N}_1/\vec{x}\vec{y}]$, $Q' \equiv Q[\vec{N}_1/\vec{x}\vec{y}]$ and $\forall i$, $M'_i \equiv M_i[\vec{N}_1/\vec{x}\vec{y}]$.

$P'[Q'/z]\vec{M}'\vec{N}_2$ has Λ -hnf in fewer steps than $(\lambda \vec{x}.M)\vec{N}$, so by induction $\lambda \vec{y}.P[Q/z]\vec{M}$ has Λ -hnf and by confluence $(\lambda \vec{x}.M)\vec{N}$ has Λ -hnf too. \square

Note that analogous properties have been proved for the Λ -unsolvable terms (Property 1.2.18).

Now we are able to prove the theorem.

▼ Proof of Λ -Solvability Theorem (Theorem 2.1.1 pag. 26).

- (\Leftarrow) Without loss of generality, we can assume that M is closed. Let $M \equiv \lambda x_1 \dots x_n.x_i M_1 \dots M_m$ ($1 \leq i \leq n$). Let $P_i \equiv \lambda x_1 \dots x_{m+1}.x_{m+1}$. Then for every sequence $P_1 \dots P_i \dots P_n$, where P_j is any term, for $i \neq j$,

$$MP_1 \dots P_i \dots P_n =_{\Lambda} I.$$

- (\Rightarrow) If M does not have Λ -hnf, then by Property 2.1.9, for all head contexts $C[.]$, $C[M]$ does not have Λ -hnf; in particular, $C[M]$ cannot be reduced to I . \blacksquare

2.1.2 Proof of Böhm's Theorem

The proof will be given in a constructive way, by showing a *separability algorithm*. The algorithm is defined as a formal system, proving statements of the shape:

$$M, N \Rightarrow_{\Lambda} C[.],$$

where M, N are Λ -normal forms such that $M \neq_{\Lambda} N$ and $C[.]$ is a context. (A very general presentation of formal systems can be found at the beginning of Chap. 5).

The rules of the system are defined by induction on the fact that M, N are Λ -normal forms that are η -different.

Definition 2.1.10. Let c be a sequence of $n \geq 0$ natural numbers (ϵ denotes the empty sequence) and M, N be Λ -normal forms. $M \not\sim_c N$ if and only if one of following cases arises:

- (i) if $c \equiv \epsilon$ then either $|p - m| \neq |q - n|$ or $x \not\equiv y$;
- (ii) if $c \equiv i, c'$ then $M =_\eta \lambda x_1 \dots x_p. x M_1 \dots M_m$ and $N =_\eta \lambda x_1 \dots x_p. y N_1 \dots N_m$ where $M_i \not\sim_{c'} N_i$ ($1 \leq i \leq m$).

Property 2.1.11. Let M and N be Λ -nf's such that $M \neq_{\Lambda\eta} N$. Then there is a sequence c of natural numbers such that $M \not\sim_c N$.

Proof. Easy. \square

Some terms will be used extensively in the rest of this section, in particular

$$\begin{aligned} B^n &\equiv \lambda x_1 \dots x_{n+1}. x_{n+1} x_1 \dots x_n \\ O^n &\equiv \lambda x_1 \dots x_{n+1}. x_{n+1} \\ U_n^i &\equiv \lambda x_1 \dots x_i. x_n \end{aligned} \quad (i \leq n, n \in \mathbb{N}).$$

A useful structural measure of a term M in Λ -nf is the maximum Λ -degree of its subterms.

Definition 2.1.12. Let $M \in \Lambda\text{-NF}$; $\text{args}(M) \in \mathbb{N}$ is defined inductively as:

- $\text{args}(x M_1 \dots M_m) = \max\{m, \text{args}(M_1), \dots, \text{args}(M_n)\}$;
- $\text{args}(\lambda x. M) = \text{args}(M)$.

Example 2.1.13. Let $M \equiv \lambda x. x(\lambda xy. x)x(xu)$; so

$$\text{args}(M) = \max\{3, \text{args}(\lambda xy. x), \text{args}(x), \text{args}(xu)\} = 3.$$

It is easy to check that if N is a subterm of M then $\text{args}(N) \leq \text{args}(M)$.

Definition 2.1.14. Let M be a term having Λ -normal form. The Λ -normal form of M will be denoted by $\text{nf}_\Lambda(M)$.

The separability algorithm is presented in Fig. 2.1 (pag. 32). For the sake of simplicity, we assume that all bound and free variables have different names.

The following lemma proves a property on which both the termination and the correctness proofs of the algorithm are based. In fact, rule (A7) of the algorithm is based on it.

Lemma 2.1.15.

Let $M, N \in \Lambda\text{-NF}$, $r \geq \max\{\text{args}(M), \text{args}(N)\}$ and $C_x^r[.] \equiv (\lambda x.[.])B^r$.

- (i) $\exists \bar{M} \in \Lambda\text{-NF}$ such that $C_x^r[M] \rightarrow_{\Lambda}^{*} \bar{M}$ and $r \geq \text{args}(\bar{M})$.
(ii) If $M \not\simeq_c N$ then $\text{nf}_{\Lambda}(C_x^r[M]) \not\simeq_c \text{nf}_{\Lambda}(C_x^r[N])$.

Proof. (i) By induction on M .

If $M \equiv \lambda z.P$ or $M \equiv z M_1 \dots M_m$ (where $z \not\equiv x$ and $m \leq r$) then the proof follows by induction. Let $M \equiv x M_1 \dots M_m$ ($m \leq r$); so by induction $\forall i \leq m$ there is $\bar{M}_i \in \Lambda\text{-NF}$ such that $C_x^r[M_i] \rightarrow_{\Lambda}^{*} \bar{M}_i$ and $r \geq \text{args}(\bar{M}_i)$. Clearly $(\lambda x.M)B^r \rightarrow_{\Lambda}^{*} \lambda x_{m+1} \dots x_{r+1}.x_{r+1}\bar{M}_1 \dots \bar{M}_m x_{m+1} \dots x_r$; hence

$$r \geq \max\{r, \text{args}(\bar{M}_1), \dots, \text{args}(\bar{M}_m), \underbrace{0, \dots, 0}_{r-m}\} = r.$$

Note that $\text{nf}_{\Lambda}(C_x^r[M])$ is well defined.

- (ii) Let $M \equiv \lambda z_1 \dots z_p.z M_1 \dots M_m$ and $N \equiv \lambda y_1 \dots y_q.y N_1 \dots N_n$; we reason by induction on c . Let $c \equiv \epsilon$. Let $z \equiv y$.

If x is different from y, z then the proof is trivial.

In case $|p - m| \neq |q - n|$, let $\bar{M}_i \equiv \text{nf}_{\Lambda}(C_x^r[M_i])$ and $\bar{N}_i \equiv \text{nf}_{\Lambda}(C_x^r[N_i])$ for each i ; thus

$$\begin{aligned} \text{nf}_{\Lambda}(C_x^r[M]) &\equiv \lambda z_1 \dots z_p x_{m+1} \dots x_{r+1}.x_{r+1}\bar{M}_1 \dots \bar{M}_m x_{m+1} \dots x_r, \\ \text{nf}_{\Lambda}(C_x^r[N]) &\equiv \lambda z_1 \dots z_q x_{n+1} \dots x_{r+1}.x_{r+1}\bar{N}_1 \dots \bar{N}_n x_{n+1} \dots x_r. \end{aligned}$$

Since $|p - m| \neq |q - n|$,

$$|(p + (r + 1) - m) - r| = |p - m + 1| \neq |q - n + 1| = |(q + (r + 1) - n) - r|.$$

If $z \not\equiv y$ then the proof is simpler.

If $c \equiv i, c'$ (where $i \geq 1$) then the proof follows by induction. \square

Example 2.1.16. Let $M \equiv \lambda x y u.x(u(x(yy))(vv))$ and $N \equiv \lambda x y u.x(u(yy)(vv))$.

Thus $\text{args}(M) = \text{args}(N) = 2$, so let us pose $r = 2$.

The derivation proving the statement $M, N \Rightarrow_{\Lambda} C[.]$ follows:

$$\begin{array}{c} \frac{x_3 \not\equiv y \quad C_5[.] \equiv (\lambda x_3 y[.]).(\lambda x_1 x_2 \tilde{x})(\lambda x_1 x_2 x_3 \tilde{y})}{x_3(yy)x_2, yyx_2x_3 \Rightarrow_{\Lambda} C_5[.]} \quad (A5) \\ \frac{}{\frac{\lambda x_2 x_3 x_3(yy)x_2, yy \Rightarrow_{\Lambda} C_2[.] \equiv C_5[.]x_2x_3}{\frac{u(\lambda x_2 x_3 x_3(yy)x_2)(vv), u(yy)(vv) \Rightarrow_{\Lambda} C_6[.] \equiv C_2[(\lambda u[.]).(\lambda z_1 z_2.z_1)]}{\frac{x(u(x(yy))(vv)), x(u(yy)(vv)) \Rightarrow_{\Lambda} C_7[.] \equiv C_6[(\lambda x[.]).(\lambda x_1 x_2 x_3 x_3 x_1 x_2)I(\lambda z_1 z_2.z_1)]}{\frac{}{\lambda x y u.x(u(x(yy))(vv)), \lambda x y u.x(u(yy)(vv)) \Rightarrow_{\Lambda} C_7[.]x y u}}}}} \quad (A2) \\ \quad (A6) \\ \quad (A7) \\ \quad (A1) \end{array}$$

where:

$$\begin{aligned} C_5[.] &\equiv (\lambda x_3 y[.]) & (\lambda x_1 x_2 \tilde{x})(\lambda x_1 x_2 x_3 \tilde{y}) \\ C_2[.] &\equiv (\lambda x_3 y[.])x_2x_3 & (\lambda x_1 x_2 \tilde{x})(\lambda x_1 x_2 x_3 \tilde{y}) \\ C_6[.] &\equiv (\lambda x_3 y[.])U_1^1 x_2 x_3) & (\lambda x_1 x_2 \tilde{x})(\lambda x_1 x_2 x_3 \tilde{y}) \\ C_7[.] &\equiv (\lambda x_3 y[.])U_2^1 x_2 x_3) & (\lambda x_1 x_2 \tilde{x})(\lambda x_1 x_2 x_3 \tilde{y}) \\ C[.] &\equiv (\lambda x_3 y[.])U_2^1 x_2 x_3) & (\lambda x_1 x_2 \tilde{x})(\lambda x_1 x_2 x_3 \tilde{y}). \end{aligned}$$

So

$$\begin{aligned}
 C[M] \equiv & (\lambda x_3 y. ((\lambda u. (\lambda x. M x y u) B^2 I U_2^1) U_2^1 x_2 x_3)) (\lambda x_1 x_2. \tilde{x}) (\lambda x_1 x_2 x_3. \tilde{y}) \rightarrow_A^* \\
 & (\lambda x_3 y. ((\lambda u. M B^2 y u I U_2^1) U_2^1 x_2 x_3)) (\lambda x_1 x_2. \tilde{x}) (\lambda x_1 x_2 x_3. \tilde{y}) \rightarrow_A^* \\
 & (\lambda x_3 y. (M B^2 y U_2^1 I U_2^1 x_2 x_3)) (\lambda x_1 x_2. \tilde{x}) (\lambda x_1 x_2 x_3. \tilde{y}) \rightarrow_A^* \\
 & (\lambda x y u. x(u(x(yy))(vv))) B^2 (\lambda x_1 x_2 x_3. \tilde{y}) U_2^1 I U_2^1 x_2 (\lambda x_1 x_2. \tilde{x}) \rightarrow_A^* \\
 & B^2 (U_2^1 (B^2 ((\lambda x_1 x_2 x_3. \tilde{y}) (\lambda x_1 x_2 x_3. \tilde{y}))) (vv)) I U_2^1 x_2 (\lambda x_1 x_2. \tilde{x}) \rightarrow_A^* \\
 & U_2^1 (U_2^1 (B^2 ((\lambda x_1 x_2 x_3. \tilde{y}) (\lambda x_1 x_2 x_3. \tilde{y}))) (vv)) I x_2 (\lambda x_1 x_2. \tilde{x}) \rightarrow_A^* \\
 & U_2^1 (B^2 ((\lambda x_1 x_2 x_3. \tilde{y}) (\lambda x_1 x_2 x_3. \tilde{y}))) (vv) x_2 (\lambda x_1 x_2. \tilde{x}) \rightarrow_A^* \\
 & B^2 ((\lambda x_1 x_2 x_3. \tilde{y}) (\lambda x_1 x_2 x_3. \tilde{y})) x_2 (\lambda x_1 x_2. \tilde{x}) \rightarrow_A^* \\
 & (\lambda x_1 x_2. \tilde{x}) ((\lambda x_1 x_2 x_3. \tilde{y}) (\lambda x_1 x_2 x_3. \tilde{y})) x_2 \rightarrow_A^* \tilde{x},
 \end{aligned}$$

while on the other hand

$$\begin{aligned}
 C[N] \equiv & (\lambda x_3 y. ((\lambda u. (\lambda x. N x y u) B^2 I U_2^1) U_2^1 x_2 x_3)) (\lambda x_1 x_2. \tilde{x}) (\lambda x_1 x_2 x_3. \tilde{y}) \rightarrow_A^* \\
 & (\lambda x_3 y. ((\lambda u. N B^2 y u I U_2^1) U_2^1 x_2 x_3)) (\lambda x_1 x_2. \tilde{x}) (\lambda x_1 x_2 x_3. \tilde{y}) \rightarrow_A^* \\
 & (\lambda x_3 y. (N B^2 y U_2^1 I U_2^1 x_2 x_3)) (\lambda x_1 x_2. \tilde{x}) (\lambda x_1 x_2 x_3. \tilde{y}) \rightarrow_A^* \\
 & (\lambda x y u. x(u(yy)(vv))) B^2 (\lambda x_1 x_2 x_3. \tilde{y}) U_2^1 I U_2^1 x_2 (\lambda x_1 x_2. \tilde{x}) \rightarrow_A^* \\
 & B^2 (U_2^1 ((\lambda x_1 x_2 x_3. \tilde{y}) (\lambda x_1 x_2 x_3. \tilde{y}))) (vv) I U_2^1 x_2 (\lambda x_1 x_2. \tilde{x}) \rightarrow_A^* \\
 & U_2^1 (U_2^1 ((\lambda x_1 x_2 x_3. \tilde{y}) (\lambda x_1 x_2 x_3. \tilde{y}))) (vv) I x_2 (\lambda x_1 x_2. \tilde{x}) \rightarrow_A^* \\
 & U_2^1 ((\lambda x_1 x_2 x_3. \tilde{y}) (\lambda x_1 x_2 x_3. \tilde{y})) (vv) x_2 (\lambda x_1 x_2. \tilde{x}) \rightarrow_A^* \\
 & (\lambda x_1 x_2 x_3. \tilde{y}) (\lambda x_1 x_2 x_3. \tilde{y}) x_2 (\lambda x_1 x_2. \tilde{x}) \rightarrow_A^* \tilde{y}.
 \end{aligned}$$

Now we will prove that the algorithm is correct and complete.

Lemma 2.1.17 (Termination).

If $M, N \in \Lambda\text{-NF}$ and $M \not\sim_c N$ then $M, N \Rightarrow_A C[.]$.

Proof. By induction on c .

Let $c = \epsilon$. Let us consider first the case when M and N have no initial abstractions. If they have different head variables, then axiom (A5) must be applied, otherwise either axiom (A3) or axiom (A4), and then the algorithm stops. If they have initial abstractions, then either rule (A1) or (A2) must be applied, and the previous situation is reached.

If $c \neq \epsilon$, either rule (A6) or (A7) must be used, and then the result follows, in the first case by induction, in the second one by induction and Lemma 2.1.15. \square

Lemma 2.1.18 (Correctness).

Let $M, N \in \Lambda\text{-NF}$ be such that $M \not\sim_c N$. If $M, N \Rightarrow_A C[.]$ then $C[M] =_A \tilde{x}$ and $C[N] =_A \tilde{y}$.

Proof. By induction on the derivation of $M, N \Rightarrow_A C[.]$, i.e. by cases on the last applied rule.

Let $M, N \in \Lambda$ -normal form, $M \not\sim_c N$, $r \geq \max\{\text{args}(M), \text{args}(N)\}$ and \tilde{x}, \tilde{y} be fresh variables such that $\tilde{x} \not\equiv \tilde{y}$.

The rules of the system proving statements $M, N \Rightarrow_\Lambda C[.]$, are the following:

$$\frac{p \leq q \quad xM_1\dots M_m x_{p+1}\dots x_q, yN_1\dots N_n \Rightarrow_\Lambda C[.]}{\lambda x_1\dots x_p. xM_1\dots M_m, \lambda x_1\dots x_q. yN_1\dots N_n \Rightarrow_\Lambda C[[.]]x_1\dots x_q} \quad (\Lambda 1)$$

$$\frac{q < p \quad xM_1\dots M_m, yN_1\dots N_n x_{q+1}\dots x_p \Rightarrow_\Lambda C[.]}{\lambda x_1\dots x_p. xM_1\dots M_m, \lambda x_1\dots x_q. yN_1\dots N_n \Rightarrow_\Lambda C[[.]]x_1\dots x_p} \quad (\Lambda 2)$$

$$\frac{n < m}{xM_1\dots M_m, xN_1\dots N_n \Rightarrow_\Lambda (\lambda x.[.])O^m \underbrace{I\dots I}_{m-n-2} K\tilde{x}\tilde{y}} \quad (\Lambda 3)$$

$$\frac{m < n}{xM_1\dots M_m, xN_1\dots N_n \Rightarrow_\Lambda (\lambda x.[.])O^n \underbrace{I\dots I}_{n-m-2} K\tilde{y}\tilde{x}} \quad (\Lambda 4)$$

$$\frac{x \not\equiv y}{xM_1\dots M_m, yN_1\dots N_n \Rightarrow_\Lambda (\lambda xy.[.])(\lambda x_1\dots x_m.\tilde{x})(\lambda x_1\dots x_n.\tilde{y})} \quad (\Lambda 5)$$

$$\frac{x \notin \text{FV}(M_k) \cup \text{FV}(N_k) \quad M_k \neq_{\Lambda\eta} N_k \quad M_k, N_k \Rightarrow_\Lambda C[.]}{xM_1\dots M_m, xN_1\dots N_m \Rightarrow_\Lambda C[(\lambda x.[.])U_m^k]} \quad (\Lambda 6)$$

$$\frac{x \in \text{FV}(M_k) \cup \text{FV}(N_k) \quad M_k \neq_{\Lambda\eta} N_k \quad C_x^r[.] \equiv (\lambda x.[.])B^r \quad \text{nf}_\Lambda(C_x^r[M_k]), \text{nf}_\Lambda(C_x^r[N_k]) \Rightarrow_\Lambda C[.]}{xM_1\dots M_m, xN_1\dots N_m \Rightarrow_\Lambda C[C_x^r[.] \underbrace{I\dots I}_{r-m} U_r^k]} \quad (\Lambda 7)$$

Fig. 2.1. Call-by-name separability algorithm.

(A1) By induction $C[xM_1\dots M_mx_{p+1}\dots x_q] \rightarrow_\Lambda^* \tilde{x}$ and $C[yN_1\dots N_n] \rightarrow_\Lambda^* \tilde{y}$; so

$$\begin{aligned} C[(\lambda x_1\dots x_p.xM_1\dots M_m)x_1\dots x_q] &\rightarrow_\Lambda^* C[xM_1\dots M_mx_{p+1}\dots x_q] \rightarrow_\Lambda^* \tilde{x} \\ C[(\lambda x_1\dots x_q.yN_1\dots N_n)x_1\dots x_q] &\rightarrow_\Lambda^* C[yN_1\dots N_n] \rightarrow_\Lambda^* \tilde{y}. \end{aligned}$$

(A2) Similar to (A1).

(A3) Clearly

$$(\lambda x.xM_1\dots M_m)O^m \underbrace{I\dots\dots\dots I}_{m-n-2} K\tilde{x}\tilde{y} \rightarrow_\Lambda^*$$

$$O^m \underbrace{M_1[O^m/x]\dots M_m[O^m/x]}_m \underbrace{I\dots\dots\dots I}_{m-n-2} K\tilde{x}\tilde{y} \rightarrow_\Lambda^* K\tilde{x}\tilde{y} \rightarrow_\Lambda^* \tilde{x}$$

while on the other hand,

$$(\lambda x.xN_1\dots N_n)O^m \underbrace{I\dots\dots\dots I}_{m-n-2} K\tilde{x}\tilde{y} \rightarrow_\Lambda^*$$

$$O^m \underbrace{N_1[O^m/x]\dots N_n[O^m/x]}_m \underbrace{I\dots\dots\dots I}_{m-n-2} K\tilde{x}\tilde{y} \rightarrow_\Lambda^* \tilde{y}.$$

(A4) Similar to (A3).

(A5) Easy.

(A6) By induction.

(A7) By induction $C[\text{nf}_\Lambda(C_x^r[M_k])] \rightarrow_\Lambda^* \tilde{x}$ and $C[\text{nf}_\Lambda(C_x^r[N_k])] \rightarrow_\Lambda^* \tilde{y}$, where $C_x^r[.] \equiv (\lambda x.[.]B^r$; thus $C[M_k[B^r/x]] \rightarrow_\Lambda^* \tilde{x}$ and $C[N_k[B^r/x]] \rightarrow_\Lambda^* \tilde{y}$ too.

Hence

$$C[((\lambda x.xM_1\dots M_m)B^r)\underbrace{I\dots\dots\dots I}_{r-m} U_r^k] \rightarrow_\Lambda^*$$

$$C[B^rM_1[B^r/x]\dots M_m[B^r/x]\underbrace{I\dots\dots\dots I}_{r-m} U_r^k] \rightarrow_\Lambda^*$$

$$C[U_r^k M_1[B^r/x]\dots M_m[B^r/x]\underbrace{I\dots\dots\dots I}_{r-m}] \rightarrow_\Lambda^* C[M_k[B^r/x]].$$

The proof of $C[((\lambda x.xN_1\dots N_m)B^r)\underbrace{I\dots\dots\dots I}_{r-m} U_r^k] \rightarrow_\Lambda^* \tilde{y}$ is similar. □

▼ Proof of Böhm's Theorem (Theorem 2.1.5 pag. 26).

The proof follows directly from Lemmas 2.1.17 and 2.1.18. ■

Note that $M, N \Rightarrow_\Lambda C[.]$ does not imply that $C[.]$ is a head context. The original algorithm designed by Böhm produces head contexts. However, the proof of correctness for our version is simpler than that of Böhm. It can be an useful exercise for the reader to modify the algorithm of Fig. 2.1 in such a way that it produces as output a head context.

3. The Call-by-Value λ -Calculus

The more usual programming languages are such that parameters must be evaluated in order to be supplied to a function, and moreover the body of a function is evaluated only when parameters are supplied. The first policy is the so called *call-by-value* parameter passing, and the second policy is called *lazy*-evaluation. In order to mimic this kind of computation with the parametric $\lambda\Delta$ -calculus, it is necessary that Δ be a proper subset of Λ , and moreover it contain all the abstraction terms.

So we choose $\Delta = \Gamma$, where $\Gamma = \text{Var} \cup \{\lambda x.M \mid M \in \Lambda\}$ was proved to be a set of input values in Property 1.2.4. The $\lambda\Gamma$ -calculus coincides with the $\lambda\beta_v$ -calculus, first introduced by Plotkin in [78].

3.1 The Syntax of the $\lambda\Gamma$ -Calculus

A term of the $\lambda\Gamma$ -calculus is always of the shape: $\lambda x_1 \dots x_n. \zeta M_1 \dots M_m$, where the head ζ is either a variable or a Γ -redex or a head block (see pag. 8).

A term is in Γ -normal form (Γ -nf) if it is of the shape $\lambda x_1 \dots x_n. \zeta M_1 \dots M_m$, where M_i is in Γ -normal form ($1 \leq i \leq m$) and ζ is either a variable or a head block $(\lambda x.P)Q$, where both P and Q are in Γ -normal form. Γ -NF denotes the set of all Γ -normal forms.

Example 3.1.1. Both xID and $(\lambda x.xI)(yz)w$ are terms in Γ -normal form. DD is a term without Γ -normal form.

Note that, differently from the $\lambda\Lambda$ -calculus, here if we want to manipulate some subterms, we need first to transform them into input values. So the notions of Γ -valuable and potentially Γ -valuable terms are important for studying such a calculus.

- Definition 3.1.2.** (i) A term M is Γ -valuable if and only if there is $N \in \Gamma$ such that $M \rightarrow_{\Gamma}^{*} N$.
- (ii) A term M is potentially Γ -valuable if and only if there is a substitution s , replacing variables by closed terms belonging to Γ , such that $s(M)$ is Γ -valuable.

It is readily verified that a closed term is potentially Γ -valuable if and only if it is valuable. Note that a term can be in Γ -normal form and not potentially Γ -valuable; consider, for example, the term $M \equiv (\lambda z.D)(yI)D$, which is in Γ -normal form. For each term $Q \in \Lambda^0$, the term $M[Q/y] \equiv (\lambda z.D)(QI)D$ is not Γ -valuable; indeed, there are two possible cases:

1. QI is Γ -valuable. Then $M[Q/y] \rightarrow_{\Gamma}^{*} DD$ and DD is not Γ -valuable, being closed and such that $DD \rightarrow_{\Gamma} DD \notin \Gamma$.
2. QI is not Γ -valuable. Then $(\lambda z.D)Q'D$ is not Γ -valuable, for every Q' such that $QI \rightarrow_{\Gamma}^{*} Q'$, since $(\lambda z.D)Q'$ is not a Γ -redex.

So to be potentially Γ -valuable is a stronger and more interesting property than to have Γ -normal form.

The class of potentially Γ -valuable terms cannot be characterized through the \rightarrow_{Γ} reduction; a new kind of reduction must be defined.

Definition 3.1.3. Let $\Psi \subseteq \Lambda$.

- (i) The lazy Ψ -reduction ($\rightarrow_{\Psi\ell}$) is the closure under application of the following rule:

$$(\lambda x.M)N \rightarrow M[N/x] \text{ if and only if } N \in \Psi;$$

$(\lambda x.M)N$, when it does not occur under the scope of a λ -abstraction, and when $N \in \Psi$, is called a $\Psi\ell$ -redex (or lazy Ψ -redex) and $M[N/x]$ is called its $\Psi\ell$ -contractum (or lazy Ψ -contractum).

- (ii) $\rightarrow_{\Psi\ell}^{*}$ and $=_{\Psi\ell}$ are respectively the reflexive and transitive closure of $\rightarrow_{\Psi\ell}$ and the symmetric, reflexive and transitive closure of $\rightarrow_{\Psi\ell}$.
 (iii) A term is in $\Psi\ell$ -normal form ($\Psi\ell$ -nf) if it has not $\Psi\ell$ -redexes and it has a $\Psi\ell$ -normal form, or it is $\Psi\ell$ -normalizing if it reduces to a $\Psi\ell$ -normal form; the set of $\Psi\ell$ -nf is denoted by $\Psi\ell$ -NF.
 (iv) A term is $\Psi\ell$ -strongly normalizing if it is $\Psi\ell$ -normalizing and moreover there is not an infinite $\Psi\ell$ -reduction sequence starting from it.

Let us notice that, in the previous definition, Ψ is not asked to be a set of input values. Moreover, the definition of $\Psi\ell$ -reduction, in point (i), does not agree with Definition 1.2.1. In fact, the reduction is defined by closing the reduction rule only under application, while in the standard case the closure is under abstraction too. This allows us to formalize the notion of lazy reduction, where no reduction can be made under the scope of a λ -abstraction.

Potentially Γ -valuable terms will be characterized by the lazy reduction induced by the following subset of Λ .

Definition 3.1.4. $\Xi \subseteq \Lambda$ is defined as follows:

$$\Xi = \Gamma \cup \{xM_1 \dots M_m \mid \forall i \leq m \quad M_i \in \Xi\}.$$

Example 3.1.5. $\lambda x.DD \in \Xi$, $xy(\lambda x.II) \in \Xi$, $I(xy) \notin \Xi$. Note that the last term is in Γ -normal form, while the first two are not.

We will show that terms having $\Xi\ell$ -normal forms are all and only the potentially Γ -valuable terms.

Property 3.1.6. Let $M \in \Lambda$.

A term M has $\Xi\ell$ -normal form if and only if $M \rightarrow_{\Xi\ell}^* P$ for some $P \in \Xi$.

Proof. It is easy to see that $M \in \Xi$ if and only if M is a $\Xi\ell$ -normal form. \square

Note that Ξ is not a set of input values. In fact, it is easy to see that the contextual reduction \rightarrow_Ξ would not be confluent. Let $P \equiv (\lambda x.(\lambda yz.z)(xD))D$. Clearly $P \rightarrow_\Xi P_1 \equiv (\lambda yz.z)(DD)$ and $P \rightarrow_\Xi P_2 \equiv (\lambda xz.z)D$, but there does not exist a $P_3 \in \Lambda$ such that $P_1 \rightarrow_\Xi^* P_3$ and $P_2 \rightarrow_\Xi^* P_3$.

Thanks to its “lazy” definition, the $\rightarrow_{\Xi\ell}$ reduction enjoys all the good properties we expect.

Theorem 3.1.7. *The $\rightarrow_{\Xi\ell}$ reduction enjoys both the confluence and the standardization properties.*

Proof. The proof is in Sect. 3.1.1. \square

Moreover, $\rightarrow_{\Xi\ell}$ and \rightarrow_Γ reductions commute as proved by Property 3.1.8.

Property 3.1.8. Let $M \rightarrow_{\Xi\ell} P$ and $M \rightarrow_\Gamma Q$. Then there is N such that both $Q \rightarrow_{\Xi\ell}^* N$ and $P \rightarrow_\Gamma^* N$.

Proof. $M \rightarrow_{\Xi\ell} P$ implies M is of the shape $\zeta \vec{M}$, where ζ is either a variable, or a Γ -redex, or a head block. Let $M \equiv (\lambda x.R)(z\vec{S})\vec{M}$, since the variable’s case is simpler. The proof is given by cases.

1. Let $R \rightarrow_\Gamma R'$. It is easy to see that the following diagram commutes:

$$\begin{array}{ccc} (\lambda x.R)(z\vec{S})\vec{M} & \xrightarrow{\Gamma} & (\lambda x.R')(z\vec{S})\vec{M} \\ \downarrow \Xi\ell & & \downarrow \Xi\ell \\ R[z\vec{S}/x]\vec{M} & \xrightarrow{\Gamma} & R'[z\vec{S}/x]\vec{M} \end{array}$$

2. Let $\vec{S} \equiv S_1 \dots S_j \dots S_m$ and let $S_j \rightarrow_\Gamma S'_j$ ($1 \leq j \leq m$). \vec{S}' will denote the sequence $S_1 \dots S'_j \dots S_m$. It is easy to see that the following diagram commutes:

$$\begin{array}{ccc} (\lambda x.R)(z\vec{S})\vec{M} & \xrightarrow{\Gamma} & (\lambda x.R)(z\vec{S}')\vec{M} \\ \downarrow \Xi\ell & & \downarrow \Xi\ell \\ R[z\vec{S}/x]\vec{M} & \xrightarrow[\Gamma]{}^* & R[z\vec{S}'/x]\vec{M} \end{array}$$

when a number ≥ 0 of Γ -reductions is needed in order to deal with the copies of $z\vec{S}$ generated by the $\Xi\ell$ -reduction.

3. Let $\vec{M} \equiv M_1 \dots M_j \dots M_m$ and let $M_j \rightarrow_{\Gamma} M'_j$ ($1 \leq j \leq m$) and let \vec{M}' denote the sequence $M_1 \dots M'_j \dots M_m$. It is easy to see that the following diagram commutes:

$$\begin{array}{ccc} (\lambda x.R)(z\vec{S})\vec{M} & \xrightarrow{\Gamma} & (\lambda x.R)(z\vec{S})\vec{M}' \\ \downarrow \Xi\ell & & \downarrow \Xi\ell \\ R[z\vec{S}/x]\vec{M} & \xrightarrow{\Gamma} & R[z\vec{S}'/x]\vec{M}' \end{array}$$

4. The cases when the $\Xi\ell$ and Γ -reductions are made in disjoint subterms of either \vec{S} or \vec{M} are immediate.
 5. The cases when the $\Xi\ell$ and Γ -reductions are made in the same subterm of either \vec{S} or \vec{M} can be treated in a similar way as the previous ones.
 6. Let $M \equiv (\lambda x.R)S\vec{M}$, where $S \in \Gamma$. Then either $P \equiv Q$, or one of the previous cases applies. \square

The $\rightarrow_{\Xi\ell}$ -reduction allows a complete characterization of the potentially Γ -valuable terms.

Theorem 3.1.9 (Potential Γ -valuability). [74]

M is potentially Γ -valuable if and only if there is $N \in \Xi$ such that $M \rightarrow_{\Xi\ell}^ N$.*

Proof. The proof is in Sect. 3.1.2. \square

As an example, let us consider the term $M \equiv (\lambda z.D)(yI)D$, which we proved before to be not potentially Γ -valuable. In fact, $(\lambda z.D)(yI)D \rightarrow_{\Xi\ell} DD$, and clearly DD does not have $\Xi\ell$ -normal form, since $DD \rightarrow_{\Xi\ell} DD$.

Now let us study the problem of characterizing the Γ -solvable terms. The next lemma shows us the relationship between the potentially Γ -valuable terms and the Γ -solvable ones.

Lemma 3.1.10. *The class of Γ -solvable terms is properly included in the class of potentially Γ -valuable terms.*

Proof. Let us first prove the inclusion. Let M be Γ -solvable, so there is a head context $(\lambda \vec{x}.[.])\vec{N}$ such that $(\lambda \vec{x}.M)\vec{N} \rightarrow_{\Gamma}^* I$ (since I is in normal form). Assume $\|\vec{x}\| \leq \|\vec{N}\|$ (otherwise consider the context $(\lambda \vec{x}.[.])\vec{N}\underbrace{I \dots I}_{p}$, where $p = \|\vec{x}\| - \|\vec{N}\|$) and $\vec{N} \equiv \vec{N}_1 \vec{N}_2$ such that $\|\vec{x}\| = \|\vec{N}_1\|$. So $M[\vec{N}_1/\vec{x}]\vec{N}_2 \rightarrow_{\Gamma}^* I$. Let s be a substitution such that $s(x) \in \Gamma^0$, for each $x \in \text{Var}$. Therefore $s(M[\vec{N}_1/\vec{x}])\vec{N}_2 \rightarrow_{\Gamma}^* s(I) \equiv I$, by Remark 3.1.30 pag. 43, hence $s(M[\vec{N}/\vec{x}]) \equiv s(M)[s(\vec{N})/\vec{x}]$ is Γ -valuable.

The inclusion is proper, since $\lambda x.DD$ is valuable, and so potentially valuable, but clearly Γ -unsolvable. \square

In order to characterize the Γ -solvable terms, we need to define a relation between terms, based on the $\rightarrow_{\Xi\ell}$ -reduction.

Definition 3.1.11. (i) *The relation $\searrow \subseteq \Lambda \times \Lambda$ is defined inductively in the following way:*

- $\lambda x.P \searrow \lambda x.Q$ if and only if $P \searrow Q$,
- $xM_1\dots M_m \searrow xN_1\dots N_m$ if and only if $M_i \rightarrow_{\Xi\ell}^* N_i \in \Xi$ ($1 \leq i \leq m$),
- $(\lambda x.P)QM_1\dots M_m \searrow R$ if and only if $Q \rightarrow_{\Xi\ell}^* \bar{Q} \in \Xi$ and
 $P[\bar{Q}/x]M_1\dots M_m \searrow R$.

- (ii) M is in Γ -head normal form (Γ -hnf) if and only if $M \equiv \lambda \vec{x}.xM_1\dots M_m$, and for all $1 \leq i \leq m$, $M_i \in \Xi$; Γ -HNF denotes the set of all Γ -head normal forms.
- (iii) M has Γ -head-normal form if and only if $M \searrow \lambda \vec{x}.xM_1\dots M_m$ and $M_i \in \Xi$, for all $1 \leq i \leq m$. $\|\vec{x}\|$ is the Γ -order and m is the Γ -degree of M .

Note that Γ -HNF is a proper subclass of Λ -HNF. In fact, $\lambda x.x(DD) \in \Lambda$ -HNF, but $\lambda x.x(DD) \notin \Gamma$ -HNF since $DD \notin \Xi$.

The notion of Γ -order (or simply order, when the set of input values is clear from the context) can be extended to terms not having Γ -hnf in the following way:

Definition 3.1.12. (i) M is of Γ -order 0 if and only if there is no P such that $M \rightarrow_{\Xi\ell}^* \lambda x.P$;

(ii) M is of Γ -order $n \geq 1$ if and only if n is the maximum integer such that $M \rightarrow_{\Xi\ell}^* \lambda x_1.M_1$, $M_i \rightarrow_{\Xi\ell}^* \lambda x_{i+1}.M_{i+1}$ ($1 \leq i \leq n$) and M_n is Γ -unsolvable of order 0. If such an n does not exist M is of Γ -order ∞ .

Example 3.1.13. DD and $(\lambda zx.xD)(yI)D$ are Γ -unsolvable of order 0, and xy is Γ -solvable of order 0. $(\lambda xy.xx)(\lambda xy.xx)$ is Γ -unsolvable of order ∞ .

Theorem 3.1.14 (Γ -Solvability). [74]

A term is Γ -solvable if and only if it has Γ -head-normal form.

Proof. The proof is in Sect. 3.1.2. □

It is also possible to give an operational characterization of the Γ -solvable terms through the notion of $\Xi\ell$ -reduction.

Property 3.1.15. M is Γ -solvable if and only if there are terms M_1, \dots, M_n , for some $n \in \mathbb{N}$, such that $M \rightarrow_{\Xi\ell}^* \lambda x_1.M_1$, $M_i \rightarrow_{\Xi\ell}^* \lambda x_{i+1}.M_{i+1}$ ($1 \leq i \leq n$) and $M_n \equiv xP_1\dots P_m$ where $P_i \in \Xi$ for some $m \in \mathbb{N}$.

Proof. (\Rightarrow) By induction on Definition 3.1.11.(i). If M is in Γ -hnf then the proof is trivial. Otherwise, the only not obvious case is when $M \equiv (\lambda x.P)QM$. In this case M is Γ -solvable if and only if $Q \rightarrow_{\Xi\ell}^* Q'$ and $P[Q'/x]\vec{M} \searrow R$, and R is in Γ -hnf. By induction there are terms

$M'_1, \dots, M'_{n'},$ for some $n \in \mathbb{N}$, such that $P[Q'/x]\vec{M} \rightarrow_{\Xi\ell}^* \lambda x_1.M'_1$, $M'_i \rightarrow_{\Xi\ell}^* \lambda x_{i+1}.M'_{i+1}$ ($1 \leq i \leq n'$) and $M'_n \equiv xP'_1\dots P'_{m'}$, for some m' .

Let $M'_0 \equiv P[Q'/x]\vec{M}$. Since $M \rightarrow_{\Xi\ell}^* M'_0$, the proof is given.

(\Leftarrow) By induction on n . If $n = 1$, then $M \rightarrow_{\Xi\ell}^* xM_1\dots M_m$, and so M is Γ -solvable. In all other cases the proof follows easily by induction. \square

Differently from the call-by-name case, in the $\lambda\Gamma$ -calculus the notion of Γ -nf is not semantically meaningful; in fact, we have seen that a term in Γ -nf can be not potentially valueable, and so is Γ -unsolvable. Moreover, consider the two terms $(\lambda z.D)(yI)D$ and $(\lambda z.D)(yK)D$: they are Γ -normal forms, and they are \neq_Γ , but they are both Γ -unsolvable of Γ -order 0. We will see that all the Γ -unsolvable terms of Γ -order 0 can be consistently equated.

Nevertheless, Λ -normal forms maintain a semantic importance also in this calculus, as the next theorem shows. Note that a Λ -normal form is a particular case of a $\Xi\ell$ -normal form.

Theorem 3.1.16 (Γ -Separability). [72]

Let $M, N \in \Lambda\text{-NF}$. If $M \neq_{\Lambda\eta} N$ then M and N are Γ -separable.

Proof. The proof is in Sect. 3.1.3. \square

The Γ -separability theorem has an interesting semantical consequence.

Corollary 3.1.17. Let M, N be two Λ -normal forms, and let $M \neq_{\Lambda\eta} N$. For every Γ -theory \mathcal{T} , if $M =_{\mathcal{T}} N$ then \mathcal{T} is input inconsistent.

Proof. The proof is identical to the proof of Property 1.3.6, just putting $\Delta = \Gamma$. \square

We proved that, for every $\lambda\Lambda$ -theory \mathcal{T} , $I =_{\mathcal{T}} E$ if and only if \mathcal{T} is fully extensional. In the case of the $\lambda\Gamma$ -calculus, we can prove only a weaker property.

Property 3.1.18. Let $M \in \Lambda$ and $x \notin \text{FV}(M)$.

$I =_{\mathcal{T}} E$ if and only if $M =_{\mathcal{T}} \lambda x.Mx$, for every $M \in \Gamma$.

Proof. (\Rightarrow) $I =_{\mathcal{T}} E$ implies $IM =_{\mathcal{T}} EM$, for all M . Thus, if $M \in \Gamma$ then $M =_{\mathcal{T}} \lambda x.Mx$ ($x \notin \text{FV}(M)$).

(\Leftarrow) If $M =_{\mathcal{T}} \lambda x.Mx$, for each $M \in \Gamma$, then $x =_{\mathcal{T}} \lambda y.xy$, so $\lambda x.x =_{\mathcal{T}} \lambda xy.xy$. \square

The notion of fixed point can be easily extended in the call-by-value setting, in the sense that N is a call-by-value fixed point of M if and only if $MN =_{\Gamma} N$.

Theorem 3.1.19. M is Γ -valueable implies that M has a call-by-value fixed point, i.e. there is N such that $MN =_{\Gamma} N$.

Proof. Let $M \rightarrow_{\Gamma}^* M' \in \Gamma$ and let Y be defined as in the proof of Theorem 2.1.8. Then $YM \rightarrow_{\Gamma}^* YM' \rightarrow_{\Gamma}^* (\lambda y.M'(yy))(\lambda y.M'(yy)) =_{\Gamma} M'(YM') =_{\Gamma} M(YM)$. \square

Let us call *call-by-value fixed point operator* terms Z such that if M is Γ -valuable then ZM is a call-by-value fixed point of M .

We will see, using denotational tools, that in the call-by-value setting, the notion of fixed point is in some sense meaningless, since every fixed point operator Z is such that ZM is not potentially Γ -valuable, for every M .

A more useful notion related to this one is the notion of *call-by-value recursion operator*. In fact, it will be used in Sect. 16.2, for expressing the recursive functions in a call-by-value setting. A call-by-value recursion operator is a term Z such that $ZM =_{\Gamma} M(\lambda z.ZMz)$, for all Γ -valuable terms M . The following theorem holds.

Theorem 3.1.20. *A call-by-value recursion operator exists.*

Proof. The term $\lambda x.(\lambda y.x(\lambda z.yyz))(\lambda y.x(\lambda z.yyz))$ has the desired behaviour. \square

3.1.1 $\Xi\ell$ -Confluence and $\Xi\ell$ -Standardization

The confluence property for the reduction $\rightarrow_{\Xi\ell}$ follows directly from the fact that it enjoys the diamond property, as proved in the next lemma.

Lemma 3.1.21 ($\Xi\ell$ -Diamond property).

Let $M, N_0, N_1 \in \Lambda$ and $N_0 \not\equiv N_1$. If $M \rightarrow_{\Xi\ell} N_0$ and $M \rightarrow_{\Xi\ell} N_1$ then there is $Q \in \Lambda$ such that $N_0 \rightarrow_{\Xi\ell} Q$ and $N_1 \rightarrow_{\Xi\ell} Q$.

Proof. We will prove only the most difficult case, i.e. $M \equiv (\lambda x.P)M_1\dots M_m$, by induction on M .

- If $N_0 \equiv (\lambda x.P)M_1\dots M'_k\dots M_m$ such that $M_k \rightarrow_{\Xi\ell} M'_k$, for some k , and $N_1 \equiv (\lambda x.P)M_1\dots M'_h\dots M_m$ such that $M_h \rightarrow_{\Xi\ell} M'_h$, for some $k \neq h$, then $Q \equiv (\lambda x.P)M_1\dots M'_h\dots M'_k\dots M_m$.
- Let $N_0 \equiv (\lambda x.P)M_1\dots M'_k\dots M_m$ and $N_1 \equiv (\lambda x.P)M_1\dots M''_k\dots M_m$ such that $M'_k \not\equiv M''_k$, $M_k \rightarrow_{\Xi\ell} M'_k$ and $M_k \rightarrow_{\Xi\ell} M''_k$, for some k . By induction on M_k there is Q' such that $M'_k \rightarrow_{\Xi\ell} Q'$ and $M''_k \rightarrow_{\Xi\ell} Q'$, thus $Q \equiv (\lambda x.P)M_1\dots Q'\dots M_m$.
- Let $M_1 \in \Xi$, so both $(\lambda x.P)$ and M_1 are $\Xi\ell$ -normal forms. Let $N_0 \equiv P[M_1/x]M_2\dots M_m$ and $N_1 \equiv (\lambda x.P)M_1\dots M'_k\dots M_m$ such that $M_k \rightarrow_{\Xi\ell} M'_k$, for some k . Clearly, $Q \equiv P[M_1/x]M_2\dots M'_k\dots M_m$. \square

Theorem 3.1.22 ($\Xi\ell$ -Confluence).

Let $M, N_0, N_1 \in \Lambda$. If $M \rightarrow_{\Xi\ell}^* N_0$ and $M \rightarrow_{\Xi\ell}^* N_1$ then there is $P \in \Lambda$ such that $N_0 \rightarrow_{\Xi\ell}^* P$ and $N_1 \rightarrow_{\Xi\ell}^* P$.

Proof. By Lemma 3.1.21, following the same reasoning as in Theorem 1.2.5. \square

Let $M \rightarrow_{\Xi\ell}^* N$; by the $\Xi\ell$ -confluence theorem, M has $\Xi\ell$ -normal form if and only if N has $\Xi\ell$ -normal form.

Corollary 3.1.23. *The $\Xi\ell$ -normal form of a term, if it exists, is unique.*

In order to state a standardization theorem for $\rightarrow_{\Xi\ell}$, we need to redefine some notions already stated for the $\lambda\Delta$ -calculus. The fact that Ξ is not a set of input values forces this redefinition.

- Definition 3.1.24.** (i) A symbol λ in a term M is $\Xi\ell$ -active if and only if it is the first symbol of a $\Xi\ell$ -redex of M .
(ii) The $\Xi\ell$ -degree of a $\Xi\ell$ -redex R in M is the numbers of λ 's that both are active in M and occur on the left of R .
(iii) The principal $\Xi\ell$ -redex of M , if it exists, is the redex of M with minimum degree.
(iv) A sequence $M \equiv P_0 \rightarrow_{\Xi\ell} P_1 \rightarrow_{\Xi\ell} \dots \rightarrow_{\Xi\ell} P_n \rightarrow_{\Xi\ell} N$ is standard if and only if the $\Xi\ell$ -degree of the redex contracted in P_i is less than or equal to the degree of the redex contracted in P_{i+1} , for every $i < n$. We denote by $M \xrightarrow{\circ}_{\Xi\ell} N$ a standard reduction sequence from M to N .

It can be easily checked that the definition of $\Xi\ell$ -degree of a redex, given in the definition before, can be obtained by specializing the general notion of sequentialization given in Definition 1.2.7: its simplification is due to the laziness of the reduction.

If $M \rightarrow_{\Xi\ell} N$ by reducing a $\Xi\ell$ -redex of degree $k \in \mathbb{N}$, then we use the notation $M \xrightarrow{k}_{\Xi\ell} N$.

Lemma 3.1.25. Let $P_0 \xrightarrow{k}_{\Xi\ell} P_1 \xrightarrow{h}_{\Xi\ell} P_2$ and $k > h$.

There is $n \in \mathbb{N}$ and $P'_1 \in \Lambda$ such that $P_0 \xrightarrow{h}_{\Xi\ell} P'_1 \xrightarrow{n}_{\Xi\ell} P_2$ and $n \geq h$.

Proof. By induction on P_0 .

We will prove only the most difficult case, when $P_0 \equiv (\lambda x.P)QM_1\dots M_m$ ($m \in \mathbb{N}$). Note that $k > h$ implies $k \geq 1$, so the principal redex cannot be reduced in $P_0 \xrightarrow{k}_{\Xi\ell} P_1$; thus either $P_1 \equiv (\lambda x.P)Q'M_1\dots M_m$ where $Q \xrightarrow{k}_{\Xi\ell} Q'$ or $P_1 \equiv (\lambda x.P)QM_1\dots M'_j\dots M_m$ where $M_j \rightarrow_{\Xi\ell} M'_j$ ($1 \leq j \leq m$).

- In the first case, $k > h$ implies there is $Q'' \in \Lambda$ such that $Q' \rightarrow_{\Xi\ell} Q''$, $P_2 \equiv (\lambda x.P)Q''M_1\dots M_m$. The proof follows by induction on Q .

Note that $Q \xrightarrow{k}_{\Xi\ell} Q'$ implies $Q \notin \Xi$; moreover, $Q \xrightarrow{k}_{\Xi\ell} Q'$ and $k > h$ imply $Q' \notin \Xi$, since the reduction is not principal.

- In the last case:

1. either $P_2 \equiv (\lambda x.P)QM_1\dots M''_j\dots M_m$ where $M \xrightarrow{k'}_{\Xi\ell} M' \xrightarrow{h'}_{\Xi\ell} M''$ and $k' \geq h'$;
2. or $P_2 \equiv (\lambda x.P)QM_1\dots M'_j\dots M'_r\dots M_m$ where $r > j$.

In case 1 the proof follows by induction on M_j , in case 2 we take the reduction sequence $P_0 \rightarrow_{\Xi\ell} (\lambda x.P)QM_1\dots M'_j\dots M_m \rightarrow_{\Xi\ell} P_2$.

□

Corollary 3.1.26. If $P_0 \xrightarrow{k} P_1 \xrightarrow{0} P_2$ and $k \geq 1$, then there are $P'_1 \in \Lambda$ and $h \in \mathbb{N}$ such that $P_0 \xrightarrow{0} P'_1 \xrightarrow{h} P_2$.

Proof. By the Lemma 3.1.25, just putting $h = 0$. \square

Now we can state the standardization theorem.

Theorem 3.1.27 ($\Xi\ell$ -Standardization).

If $M \rightarrow_{\Xi\ell}^* N$ then there is a standard reduction sequence from M to N .

Proof. By induction on M . Let $M \equiv xM_1\dots M_m$, thus N must be of the shape $xP_1\dots P_m$, where $M_i \rightarrow_{\Xi\ell}^* P_i$. By induction there is a standard reduction sequence $M_i \rightarrow_{\Xi\ell}^* P_i$, and so the desired standard sequence is: $M \rightarrow_{\Xi\ell}^* xP_1M_2\dots M_m \rightarrow_{\Xi\ell}^* xP_1P_2\dots M_m \rightarrow_{\Xi\ell}^* xP_1\dots P_m$ ($1 \leq i \leq m$).

If $M \equiv \lambda x.M'$ then it must be that $M \equiv N$, and the empty reduction sequence is trivially standard.

Let $M \equiv (\lambda x.P)M_1\dots M_m$ ($m \geq 1$). The proof follows by induction on the length of the reduction $M \rightarrow_{\Xi\ell}^* N$, by using the previous corollary. \square

The principal reduction is normalizing.

Corollary 3.1.28. $M \rightarrow_{\Xi\ell}^* N \in \Xi$ if and only if $M \rightarrow_{\Xi\ell}^* N \in \Xi$.

Proof. Trivial. \square

3.1.2 Proof of Potential Γ -Valuability and Γ -Solvability Theorems

In order to prove the theorems, we need to introduce a measure for carrying out some inductive proofs.

Definition 3.1.29. The weight $\langle \cdot \rangle : \Lambda \longrightarrow \mathbb{N}$ is the partial function defined as follows:

- $\langle \lambda x.M' \rangle = 0$.
- $\langle xM_1\dots M_m \rangle = 1 + \langle M_1 \rangle + \dots + \langle M_m \rangle$.
- $\langle (\lambda x.M_0)M_1\dots M_m \rangle = 1 + \langle M_1 \rangle + \langle M_0[M_1/x]M_2\dots M_m \rangle$.

In Sect. 3.2, we will show that the weight of a term M is defined if and only if M has $\Xi\ell$ -normal form.

The following remark will be extensively used in what follows.

Remark 3.1.30. Let $M, N, P, Q \in \Delta$, where Δ is a set of input values. If $M \rightarrow_{\Delta} N$ and $P \rightarrow_{\Delta} Q$ then $M[P/z] \rightarrow_{\Delta} N[Q/z]$.

Proof. Easy, by induction on M . \square

Lemma 3.1.31. *Let $Q \in \Xi$, $P \in \Lambda$ and $C[.]$ be a context.*

- (i) *If $Q \in \Xi$ then $\langle Q \rangle$ is defined.*
- (ii) *If $M \rightarrow_{\Xi\ell} N$ and $\langle N \rangle$ is defined then $\langle M \rangle$ is defined.*
- (iii) *If M has $\Xi\ell$ -normal form then $\langle M \rangle$ is defined.*

Proof. (i) By induction on Q .

(ii) By induction on $\langle N \rangle$.

If $\langle N \rangle = 0$ then $N \equiv \lambda x.N'$, so $M \equiv (\lambda z.P)Q$ and $Q \in \Xi$; hence, $\langle M \rangle = 1 + \langle Q \rangle + \langle N \rangle$ where $\langle Q \rangle$ is defined by the previous point of this lemma.

Let $\langle N \rangle \geq 1$; there are many cases.

1. Let $M \equiv xM_1\dots M_m \rightarrow_{\Xi\ell} xN_1\dots N_m \equiv N$ ($m \geq 1$), where there is a unique $k \leq m$ such that $M_k \rightarrow_{\Xi\ell} N_k$ while $M_h \equiv N_h$ if $h \neq k$. The proof follows easily by induction.
2. Let $M \equiv (\lambda z.P)QM_1\dots M_m \rightarrow_{\Xi\ell} RM_1\dots M_m \equiv N$ ($m \geq 1$), where $Q \in \Xi$ and $(\lambda z.P)Q \rightarrow_{\Xi\ell} R$; hence, $\langle M \rangle = 1 + \langle Q \rangle + \langle N \rangle$ where $\langle Q \rangle$ is defined by the point (i) of this lemma.
3. The case $M \equiv \lambda x.P$ is not possible, since $\rightarrow_{\Xi\ell}$ is lazy.

- (iii) By induction on the length of the sequence to $\Xi\ell$ -normal form, by using the previous points of this lemma. \square

The weight of a term allows us to induce on the length of reduction sequences with respect to different notions of reduction.

Remark 3.1.32. If $M \in \Lambda^0$ is Γ -valuable then $M \equiv (\lambda z.P)QM_1\dots M_m$, for some $m \in \mathbb{N}$; moreover, M_i is closed and also Γ -valuable ($1 \leq i \leq m$).

Property 3.1.33. Let $M, N \in \Lambda^0$.

- (i) If M is Γ -valuable then $\langle M \rangle$ is defined.
- (ii) $M \rightarrow_A^* N$ and $\langle M \rangle$ is defined imply $\langle N \rangle$ is defined and $\langle M \rangle \geq \langle N \rangle$.
- (iii) Let either $M \rightarrow_\Gamma^* N$ or $M \rightarrow_{\Xi\ell}^* N$.
If $\langle M \rangle$ is defined then $\langle N \rangle$ is defined and $\langle M \rangle \geq \langle N \rangle$.

Proof. (i) Γ is a standard set of input values and $M \rightarrow_\Gamma^* N \in \Gamma$ imply that there is $N' \in \Gamma$ such that $M \rightarrow_\Gamma^{*p} N' \in \Gamma$; moreover, since M is closed there is M' such that $M \rightarrow_\Gamma^p M_1 \rightarrow_\Gamma^p M_2 \dots \rightarrow_\Gamma^p M_r \rightarrow_\Gamma^p \lambda z.M' \rightarrow_\Gamma^{*p} N' \in \Gamma$, where M_i is not an abstraction, for all i ($1 \leq i \leq r$).

Then $M \rightarrow_{\Gamma\ell} \lambda z.M'$ by definition of principal reduction. Clearly $M \rightarrow_{\Gamma\ell} \lambda z.M' \in \Gamma$ implies $M \rightarrow_{\Xi\ell} \lambda z.M' \in \Xi$, so the proof follows by Lemma 3.1.31.(iii).

- (ii) Let $\langle M \rangle = k$ and let p be the number of steps of the standard reduction sequence $M \rightarrow_A^* N$. The proof is given by induction on the pair (k, p) , ordered according to the lexicographical order.

The cases where either $\langle M \rangle = 0$ or $p = 0$ are trivial. $M \equiv xM_1\dots M_m$ is not possible, since $M \in \Lambda^0$ by hypothesis. Let $M \equiv (\lambda x.M_0)M_1\dots M_m$, $h' = \langle M_1 \rangle$ and $h'' = \langle M_0[M_1/x]M_2\dots M_m \rangle$, thus $k = 1 + h' + h''$.

Let the reduction path be: $M \rightarrow_{\Lambda} R_1 \rightarrow_{\Lambda} \dots \rightarrow_{\Lambda} R_p \equiv N$ ($p > 0$). There are three cases:

1. If $R_1 \equiv M_0[M_1/x]M_2\dots M_m$ then $\langle R_1 \rangle = h'' < k$, so the proof follows by induction.
2. Let $R_1 \equiv (\lambda x.N_0)M_1N_2\dots N_m$ where $\exists! j \in \mathbb{N}$ such that $M_j \rightarrow_{\Lambda} N_j$, while $\forall i \neq j$, $M_i \equiv N_i$ ($0 \leq i \leq m$ and $i \neq 1$). Hence $M_0[M_1/x]M_2\dots M_m \rightarrow_{\Lambda} N_0[M_1/x]N_2\dots N_m$ and $h'' < k$ imply $\langle N_0[M_1/x]N_2\dots N_m \rangle \leq h''$, by induction.
Thus $\langle R_1 \rangle = 1 + \langle M_1 \rangle + \langle N_0[M_1/x]N_2\dots N_m \rangle \leq k$ and the proof follows by induction.
3. Let $R_1 \equiv (\lambda x.M_0)N_1M_2\dots M_m$, where $M_1 \rightarrow_{\Lambda} N_1$. Thus by induction on $M_0[M_1/x]M_2\dots M_m \rightarrow_{\Lambda}^* M_0[N_1/x]M_2\dots M_m$ and $h'' < k$, $\langle M_0[N_1/x]M_2\dots M_m \rangle \leq h''$. Again, by induction $\langle M_1 \rangle \geq \langle N_1 \rangle$. Thus the conclusion follows by definition of weight and by induction.

- (iii) By the previous point of this Property, since $M \rightarrow_{\Psi} N$ implies $M \rightarrow_{\Lambda} N$, for each $\Psi \subseteq \Lambda$. \square

The Lemma 3.1.34 proves that if a term is potentially Γ -valuable, then it has $\Xi\ell$ -normal form.

Lemma 3.1.34. *Let $M \in \Lambda$, $\text{FV}(M) \subseteq \{x_1\dots x_n\}$ and let \mathbf{s} be a substitution such that $\mathbf{s}(x_i) = P_i \in \Gamma^0$. If $\mathbf{s}(M) \rightarrow_{\Gamma}^* \bar{M} \in \Gamma$ then there is $N \in \Xi$ such that both $M \rightarrow_{\Xi\ell}^* N$ and $\mathbf{s}(N) \rightarrow_{\Gamma}^* \bar{M}$.*

Proof. The proof is carried out by induction on $k = \langle \mathbf{s}(M) \rangle$, where we assume $\mathbf{s}(x_i) = P_i$ ($1 \leq i \leq n$) and $\mathbf{s}(M) \equiv M[P_1/x_1, \dots, P_n/x_n]$.

- $k = 0$. Thus $\mathbf{s}(M)$ is an abstraction; there are two cases:
 1. $M \equiv x_j$ and $P_j \equiv \lambda z.P \in \Lambda^0$, so $N \equiv x_j$.
 2. $M \equiv \lambda z.P$, so $N \equiv \lambda z.P$.
 In both cases the proof is immediate.
- $k > 0$. $\mathbf{s}(M) \in \Lambda^0$, so $\mathbf{s}(M) \equiv (\lambda u.R_0)R_1\dots R_r$ ($r \geq 1$). Two cases are possible, according to the shape of M :

1. $M \equiv x_j M_1\dots M_m$ ($j \leq n$, $1 \leq m$). Assume $P_j \equiv (\lambda z.P')$ (indeed $P_j \in \Gamma^0$); then $\mathbf{s}(M) \equiv P_j \mathbf{s}(M_1)\dots \mathbf{s}(M_m) \rightarrow_{\Gamma}^* \bar{M} \in \Gamma$.

Since $\mathbf{s}(M)$ is Γ -valuable, there are \bar{M}_i such that $\mathbf{s}(M_i) \rightarrow_{\Gamma}^* \bar{M}_i \in \Gamma$ and $\langle \mathbf{s}(M_i) \rangle < \langle \mathbf{s}(M) \rangle$; hence, by induction there are $N_i \in \Xi$ such that $M_i \rightarrow_{\Xi\ell} N_i$ and $\mathbf{s}(N_i) \rightarrow_{\Gamma}^* \bar{M}_i$ ($1 \leq i \leq m$).

Let $N \equiv x_j N_1\dots N_m \in \Xi$, thus $x_j M_1\dots M_m \rightarrow_{\Xi\ell}^* x_j N_1\dots N_m$ and $P_j \mathbf{s}(N_1)\dots \mathbf{s}(N_m) \rightarrow_{\Gamma}^* P_j \bar{M}_1\dots \bar{M}_m \rightarrow_{\Gamma}^* M$ ($1 \leq i \leq m$).

2. $M \equiv (\lambda z.P)QM_1\dots M_m$ ($m \geq 0$). Since $\mathbf{s}(M)$ is Γ -valuable, there is \bar{Q} such that $\mathbf{s}(Q) \rightarrow_{\Gamma}^* \bar{Q} \in \Gamma$ and $\mathbf{s}(P[\bar{Q}/z])\mathbf{s}(M_1)\dots \mathbf{s}(M_m) \rightarrow_{\Gamma}^* \bar{M}$. Moreover, $\langle \mathbf{s}(Q) \rangle < \langle \mathbf{s}(M) \rangle$, so by induction there is $R \in \Xi$ such that $Q \rightarrow_{\Xi\ell}^* R$ and $\mathbf{s}(R) \rightarrow_{\Gamma}^* \bar{Q}$.

But from $\mathbf{s}(P[R/z])\mathbf{s}(M_1)\dots \mathbf{s}(M_m) \rightarrow_{\Gamma}^* \mathbf{s}(P[\bar{Q}/z])\mathbf{s}(M_1)\dots \mathbf{s}(M_m)$ together with $\mathbf{s}(P[\bar{Q}/z])\mathbf{s}(M_1)\dots \mathbf{s}(M_m) \rightarrow_{\Gamma}^* \bar{M}$ it follows that:

$\mathbf{s}(P[R/z])\mathbf{s}(M_1)\dots \mathbf{s}(M_m) \rightarrow_{\Gamma}^* \bar{M}$ and, by Property 3.1.33.(iii)

$$\langle \mathbf{s}(P[\mathbf{s}(R)/z])\mathbf{s}(M_1)\dots\mathbf{s}(M_m) \rangle \leq \langle \mathbf{s}(P[\bar{Q}/z])\mathbf{s}(M_1)\dots\mathbf{s}(M_m) \rangle < \langle \mathbf{s}(M) \rangle.$$

Then, by induction, there is $T \in \Xi$ such that $P[R/z]M_1\dots M_m \rightarrow_{\Xi\ell}^* T$ and $\mathbf{s}(T) \rightarrow_{\Gamma}^* \bar{M}$. Let $N \equiv T$; clearly $M \equiv (\lambda z.P)QM_1\dots M_m \rightarrow_{\Xi\ell}^* (\lambda z.P)RM_1\dots M_m \rightarrow_{\Xi\ell} P[R/z]M_1\dots M_m \rightarrow_{\Xi\ell}^* N$, so the proof is given.

□

The Lemma 3.1.35 proves that if a term has $\Xi\ell$ -normal form then it is potentially Γ -valuable.

Lemma 3.1.35. *Let $M \in \Lambda$, $\text{FV}(M) \subseteq \{x_1, \dots, x_n\}$.*

$M \rightarrow_{\Xi\ell}^ N \in \Xi$ implies that $\exists h \in \mathbb{N}$ such that $\forall r \geq h$, $\exists \bar{M}^r \in \Gamma$*

$$\begin{aligned} M[O^r/x_1, \dots, O^r/x_n] &\rightarrow_{\Gamma}^* \bar{M}^r, \\ N[O^r/x_1, \dots, O^r/x_n] &\rightarrow_{\Gamma}^* \bar{M}^r, \end{aligned}$$

where $O^r \equiv \lambda x_1 \dots x_{r+1}.x_{r+1}$.

Proof. Let \mathbf{p}_r be the substitution such that $\mathbf{p}_r(y) = O^r$, for all $y \in \text{Var}$ and $r \geq 0$; so $\mathbf{p}_r(M) = M[O^r/x_1, \dots, O^r/x_n]$.

The proof will be given by induction on $\langle M \rangle$.

- Let $\langle M \rangle = 0$, so M is an abstraction and the proof is trivial.
- Let $\langle M \rangle \geq 1$. If $M \equiv xM_1\dots M_m$ ($m \in \mathbb{N}$) then by induction, $\forall i \leq m$ there are $h_i \in \mathbb{N}$ such that $\forall r \geq \max\{m, h_1, \dots, h_m\}$, $\mathbf{p}_r(M_i) \rightarrow_{\Gamma}^* \bar{M}_i^r \in \Gamma$ and the proof is immediate.

Otherwise, let $M \equiv (\lambda z.P)QM_1\dots M_m$ ($m \in \mathbb{N}$); M has $\Xi\ell$ -normal form implies that there is $R \in \Xi$ such that $Q \rightarrow_{\Xi\ell}^* R$. Hence $\langle Q \rangle < \langle M \rangle$ and this implies by induction, that there is $h_0 \in \mathbb{N}$ such that $\forall r \geq h_0 \exists \bar{Q}^r \in \Gamma$, $\mathbf{p}_r(Q) \rightarrow_{\Gamma}^* \bar{Q}^r$ and $\mathbf{p}_r(R) \rightarrow_{\Gamma}^* \bar{Q}^r$. Clearly $P[R/z]M_1\dots M_m \rightarrow_{\Xi\ell}^* N$ too. By Property 3.1.33.(iii) $\langle P[R/z]M_1\dots M_m \rangle \leq \langle P[Q/z]M_1\dots M_m \rangle < \langle M \rangle$ then, by induction, there is $h_1 \in \mathbb{N}$ such that $\forall r \geq h_1 \exists \bar{P}^r \in \Gamma$ satisfying $\mathbf{p}_r(P[R/z]M_1\dots M_m) \rightarrow_{\Gamma}^* \bar{P}^r$ and $\mathbf{p}_r(N) \rightarrow_{\Gamma}^* \bar{P}^r$.

$\forall r \geq \max\{h_0, h_1\}$, $\exists \bar{Q}^r \in \Lambda^0$ $\mathbf{p}_r(R) \rightarrow_{\Gamma}^* \bar{Q}^r$ implies, by the confluence theorem,

$$\mathbf{p}_r(P[R/z])\mathbf{p}_r(M_1)\dots\mathbf{p}_r(M_m) \rightarrow_{\Gamma}^* \mathbf{p}_r(P[\bar{Q}^r/z])\mathbf{p}_r(M_1)\dots\mathbf{p}_r(M_m) \rightarrow_{\Gamma}^* \bar{P}^r.$$

Since $\mathbf{p}_r(M) \rightarrow_{\Gamma}^* \mathbf{p}_r(P[\bar{Q}^r/z])\mathbf{p}_r(M_1)\dots\mathbf{p}_r(M_m)$, the proof is done.

□

▼ Proof of Potential Γ -Valuability Theorem (Theorem 3.1.9 pag. 38).

The proof of the (only if) part follows directly from Lemma 3.1.34, while the proof of the (if) part follows directly from Lemma 3.1.35. ■

The Lemma 3.1.36 implies as an immediate corollary that, if $M \in \Lambda$ has Γ -head normal form, then M is Γ -solvable.

Lemma 3.1.36. *If M has Γ -head normal form and $\text{FV}(M) = \{x_1, \dots, x_n\}$ then $\exists s \in \mathbb{N}, \forall r \geq s, \exists k \in \mathbb{N}$ such that $(\lambda x_1 \dots x_n. M) \underbrace{O^r \dots O^r}_r \rightarrow_{\Gamma}^{*} O^k$.*

Proof. Let \mathbf{p}_r (where $r \in \mathbb{N}$) be the substitution such that $\mathbf{p}_r(y) = O^r$, for each $y \in \text{Var}$. The proof is done by induction on the minimum number q of steps necessary to prove that $M \searrow N$, for some N in Γ -head normal form.

If M is an abstraction, the proof follows directly by induction. Let $M \equiv xM_1 \dots M_m$, where M_i have $\Xi\ell$ -normal forms ($1 \leq i \leq m$). By Lemma 3.1.35, $\exists s_i \in \mathbb{N}$ such that $\forall r \geq s_i, M_i[O^r/x_1, \dots, O^r/x_n] \rightarrow_{\Gamma}^{*} \bar{M}_i \in \Gamma$ ($1 \leq i \leq m$). Let $r \geq \max\{m, n, s_1, \dots, s_m\}$; thus for some $k \in \mathbb{N}$,

$$(\lambda x_1 \dots x_n. M) \underbrace{O^r \dots O^r}_r \rightarrow_{\Gamma}^{*} O^r \bar{M}_1 \dots \bar{M}_m \underbrace{O^r \dots O^r}_{r-n} \rightarrow_{\Gamma}^{*} O^{r-m} \underbrace{O^r \dots O^r}_{r-n} \rightarrow_{\Gamma}^{*} O^k.$$

If $r - m \geq r - n$ then $n \geq m$ and $k = (r - m) - (r - n) = n - m$; otherwise $r - m < r - n$ and $O^{r-m} \underbrace{O^r \dots O^r}_{r-n} \rightarrow_{\Gamma}^{*} \underbrace{O^r \dots O^r}_{r-n-(r-m)}$, thus $r - n - (r - m) = m - n$

and $k = r - (m - n - 1) = r + 1 + n - m$.

Let $M \equiv (\lambda x.P)QM_1 \dots M_m$ ($m \geq 1$). By definition $Q \rightarrow_{\Xi\ell}^{*} R \in \Xi$ and $M \searrow P[R/x]M_1 \dots M_m$, which has Γ -head normal form. $Q \rightarrow_{\Xi\ell}^{*} R \in \Xi$ implies that $\exists h_0 \in \mathbb{N}$ such that $\forall r \geq h_0, \exists \bar{Q}^r \in \Gamma, \mathbf{p}_r(Q) \rightarrow_{\Gamma}^{*} \bar{Q}^r$ and $\mathbf{p}_r(R) \rightarrow_{\Gamma}^{*} \bar{Q}^r$, by Lemma 3.1.35. By induction $\exists h \in \mathbb{N}$ such that $\forall r \geq h, (\lambda x_1 \dots x_n.P[R/x]M_1 \dots M_m) \underbrace{O^r \dots O^r}_r \rightarrow_{\Gamma}^{*} O^k$, for some $k \in \mathbb{N}$.

Let $r \geq \max\{h_0, h\}$, so for some $v \in \mathbb{N}$,

$$\begin{aligned} (\lambda x_1 \dots x_n. M) \underbrace{O^r \dots O^r}_r &\rightarrow_{\Gamma}^{*} (\mathbf{p}_r(\lambda x.P)\mathbf{p}_r(Q)\mathbf{p}_r(M_1) \dots \mathbf{p}_r(M_m)) \underbrace{O^r \dots O^r}_{r-n} \\ &\rightarrow_{\Gamma}^{*} (\mathbf{p}_r(\lambda x.P)\bar{Q}^r\mathbf{p}_r(M_1) \dots \mathbf{p}_r(M_m)) \underbrace{O^r \dots O^r}_{r-n} \\ &\rightarrow_{\Gamma} (\mathbf{p}_r(P[\bar{Q}^r/x])\mathbf{p}_r(M_1) \dots \mathbf{p}_r(M_m)) \underbrace{O^r \dots O^r}_{r-n} \rightarrow_{\Gamma}^{*} O^v. \end{aligned}$$

But $\mathbf{p}_r(P[\mathbf{p}_r(R)/x])\mathbf{p}_r(M_1) \dots \mathbf{p}_r(M_m) \rightarrow_{\Gamma}^{*} \mathbf{p}_r(P[\bar{Q}^r/x])\mathbf{p}_r(M_1) \dots \mathbf{p}_r(M_m)$, thus by the confluence theorem, it follows that $(\lambda x_1 \dots x_n. M) \underbrace{O^r \dots O^r}_r \rightarrow_{\Gamma}^{*} O^v$ too. \square

The following lemma implies as an immediate corollary that, if $M \in \Lambda$ is Γ -solvable, then M has Γ -head normal form.

Lemma 3.1.37. *Let $M \in \Lambda$, $\text{FV}(M) \subseteq \{x_1, \dots, x_n\}$ and $P_1, \dots, P_k \in \Gamma^0$. If $(\lambda x_1 \dots x_n. M)P_1 \dots P_k \rightarrow_{\Gamma}^{*} I$ then there is N in Γ -head-normal form such that $M \searrow N$ and $(\lambda x_1 \dots x_n. N)P_1 \dots P_k \rightarrow_{\Gamma}^{*} I$.*

Proof. Let \mathbf{s} be a substitution such that $\mathbf{s}(x_i) = P_i \in \Gamma^0$.

Let $M \equiv \lambda x_{n+1} \dots x_r. \xi M_1 \dots M_m$ ($m, r \in \mathbb{N}, n \leq r$) where either $\xi \equiv x_j$ ($j \leq r$) or $\xi \equiv (\lambda x.P)Q$, for some $P, Q \in \Lambda$, and let $C[.] \equiv (\lambda x_1 \dots x_n.[.])P_1 \dots P_k$, so

$$C[M] \equiv (\lambda x_1 \dots x_r. \xi M_1 \dots M_m) P_1 \dots P_k.$$

Note that $r \leq k + 1$, otherwise $C[M] \rightarrow_{\Gamma}^{*} \lambda x_{k+1} \dots x_r. S' \neq_{\Gamma} I$. The proof is given for induction on $\langle C[M] \rangle_e$ by taking into account all possible shapes of the term M .

- $M \equiv \lambda x_{n+1} \dots x_r. x_j M_1 \dots M_m$. If $m = 0$ then the proof is trivial by putting $N \equiv M$, so let $m \geq 1$. There are 2 cases.

1. If $r \leq k$ then

$$\begin{aligned} C[M] &\equiv (\lambda x_1 \dots x_r. x_j M_1 \dots M_m) P_1 \dots P_k \\ &\rightarrow_{\Gamma}^{*} P_j \mathbf{s}(M_1) \dots \mathbf{s}(M_m) P_r + 1 \dots P_k \rightarrow_{\Gamma}^{*} I. \end{aligned}$$

By Remark 3.1.32, let $\mathbf{s}(M_i) \rightarrow_{\Gamma}^{*} \bar{M}_i \in \Gamma$ ($1 \leq i \leq m$); so by Lemma 3.1.34 $M_i \rightarrow_{\Xi\ell}^{*} N_i \in \Xi$ and $\mathbf{s}(N_i) \equiv N_i[P_1/x_1, \dots, P_n/x_n] \rightarrow_{\Gamma}^{*} \bar{M}_i$.

Let $N \equiv \lambda x_{n+1} \dots x_r. x_j N_1 \dots N_m$; so $\lambda x_{n+1} \dots x_r. x_j M_1 \dots M_m \searrow N$ and

$$\begin{aligned} (\lambda x_1 \dots x_r. x_j N_1 \dots N_m) P_1 \dots P_k &\rightarrow_{\Gamma}^{*} P_j \mathbf{s}(N_1) \dots \mathbf{s}(N_m) P_{r+1} \dots P_k \\ &=_{\Gamma} (\lambda x_1 \dots x_r. x_j M_1 \dots M_m) P_1 \dots P_k \\ &\rightarrow_{\Gamma}^{*} P_j \bar{M}_1 \dots \bar{M}_m P_{r+1} \dots P_k \\ &=_{\Gamma} (\lambda x_1 \dots x_n. M) P_1 \dots P_k \rightarrow_{\Gamma}^{*} I. \end{aligned}$$

2. If $r = k + 1$ then the proof is similar to that of the previous case, since

$$\begin{aligned} C[M] &\equiv (\lambda x_1 \dots x_r. x_j M_1 \dots M_m) P_1 \dots P_k \\ &\rightarrow_{\Gamma}^{*} \lambda x_r. P_j \mathbf{s}(M_1) \dots \mathbf{s}(M_m) \rightarrow_{\Gamma}^{*} \lambda x_r. x_r. \end{aligned}$$

- $M \equiv \lambda x_{n+1} \dots x_r. (\lambda z. P) Q M_1 \dots M_m$ ($m \geq 0$, $r \geq n$); there are 2 cases.

1. If $r \leq k$ then

$$\begin{aligned} C[M] &\equiv (\lambda x_1 \dots x_r. (\lambda z. P) Q M_1 \dots M_m) P_1 \dots P_k \\ &\rightarrow_{\Gamma}^{*} \mathbf{s}(\lambda z. P) \mathbf{s}(Q) \mathbf{s}(M_1) \dots \mathbf{s}(M_m) P_{r+1} \dots P_k \\ &\rightarrow_{\Gamma}^{*} \mathbf{s}(\lambda z. P) \bar{Q} \mathbf{s}(M_1) \dots \mathbf{s}(M_m) P_{r+1} \dots P_k \rightarrow_{\Gamma}^{*} I, \end{aligned}$$

where $\mathbf{s}(Q) \rightarrow_{\Gamma}^{*} \bar{Q} \in \Gamma$. Hence, by Lemma 3.1.34 $Q \rightarrow_{\Xi\ell}^{*} R \in \Xi$ and $\mathbf{s}(R) \rightarrow_{\Gamma}^{*} \bar{Q} \in \Gamma$. Moreover

$$\begin{aligned} \mathbf{s}(P[R/z]) \mathbf{s}(M_1) \dots \mathbf{s}(M_m) P_{r+1} \dots P_k &\rightarrow_{\Gamma}^{*} \\ \mathbf{s}(P[\bar{Q}/z]) \mathbf{s}(M_1) \dots \mathbf{s}(M_m) P_{r+1} \dots P_k &\rightarrow_{\Gamma}^{*} I. \end{aligned}$$

Let $U \equiv \lambda x_{n+1} \dots x_r. P[R/z] M_1 \dots M_m$, thus $C[U] \rightarrow_{\Gamma}^{*} I$. Remember that $P_1, \dots, P_k \in \Gamma^0$; then

$$\begin{aligned} \langle C[M] \rangle_e &= \\ &= r + \langle P_1 \rangle + \dots + \langle P_r \rangle + \langle \mathbf{s}(\lambda z. P) \mathbf{s}(Q) \mathbf{s}(M_1) \dots \mathbf{s}(M_m) P_{r+1} \dots P_k \rangle = \\ &= r + \underbrace{0 + \dots + 0}_{r} + 1 + \langle \mathbf{s}(Q) \rangle + \langle \mathbf{s}(P[\mathbf{s}(Q)/z]) \mathbf{s}(M_1) \dots \mathbf{s}(M_m) P_{r+1} \dots P_k \rangle, \end{aligned}$$

and by Property 3.1.33.(ii)

$$\begin{aligned} \langle s(P[s(Q)/z])s(M_1) \dots s(M_m)P_{r+1} \dots P_k \rangle \geq \\ \langle s(P[s(R)/z])s(M_1) \dots s(M_m)P_{r+1} \dots P_k \rangle. \end{aligned}$$

Hence $\langle C[U] \rangle < \langle C[M] \rangle$, and by induction, we can state $U \searrow T$ and $C[U] \rightarrow_\Gamma^* I$, for some T in Γ -head normal form.

Let N be T ; so $C[N] \rightarrow_\Gamma^* I$ and

$$M \equiv \lambda x_{n+1} \dots x_r. (\lambda z.P) Q M_1 \dots M_m \searrow \lambda x_{n+1} \dots x_r. P[R/z] M_1 \dots M_m \searrow T.$$

2. The case $r = k + 1$ is similar to the previous one. \square

▼ Proof of Γ -Solvability Theorem (Theorem 3.1.14 pag. 39).

Let $\text{FV}(M) = \{x_1, \dots, x_n\}$.

(\Rightarrow) By Lemma 3.1.36 for some r and h :

$$(\lambda x_1 \dots x_n. M) \underbrace{O^r \dots O^r}_r \rightarrow_\Gamma^* O^h \quad (h \geq 0).$$

Let $R_1, \dots, R_h \in \Gamma$, thus $(\lambda x_1 \dots x_n. M) \underbrace{O^r \dots O^r}_r R_1 \dots R_h \rightarrow_\Gamma^* I$.

(\Leftarrow) By Lemma 3.1.37. \blacksquare

3.1.3 Proof of Γ -Separability Theorem

The proof will be given in a constructive way, by showing a *separability algorithm*. The algorithm is defined as a formal system, proving statements of the shape

$$M, N \Rightarrow_\Gamma C[.],$$

where M, N are Λ -normal forms such that $M \neq_{\Lambda\eta} N$.

Differently from the call-by-name case, the context $C[.]$ generated from the algorithm is not yet the separating one. More precisely, it is a separating context if $M, N \in \Lambda^0$, but for open terms some additional work must be done.

Let $B^n \equiv \lambda x_1 \dots x_{n+1}. x_{n+1} x_1 \dots x_n$, $O^n \equiv \lambda x_1 \dots x_{n+1}. x_{n+1}$ and $U_i^n \equiv \lambda x_1 \dots x_n. x_i$ ($i \leq n, n \in \mathbb{N}$).

Furthermore, if $S \subseteq \text{Var}$ then let $X_{x,S}^n \equiv \begin{cases} \lambda z_1 \dots z_n. x_i z_1 \dots z_n & \text{if } x \notin S, \\ x & \text{otherwise.} \end{cases}$

The notions of `args`, nf_Λ and \simeq_c are defined respectively in Definitions 2.1.12, 2.1.14 and 2.1.10.

Remark 3.1.38. It is easy to check that every Λ -normal form M is a potentially Γ -valuable term; so every subterm of a Γ -nf, being in turn a Γ -nf, is potentially Γ -valuable.

A first problem in the development of a separability theorem for the Γ -calculus is the transformation of potentially valuable terms (subterms) in valuable ones. As proved in the next Lemma, the solution is to substitute to free variables some values with a suitable number of initial abstractions.

Lemma 3.1.39. *Let $M \in \Lambda\text{-NF}$, $\text{FV}(M) = \{x_1, \dots, x_n\}$ and $r \geq \text{args}(M)$. If $\forall j \leq n$, $Q_j^r \equiv \lambda x_1 \dots x_r. Q_j$ and $Q_j \in \Gamma$ then*

$$M[Q_1^r/x_1, \dots, Q_n^r/x_n] \rightarrow_{\Gamma}^{*} \bar{M} \in \Gamma.$$

Proof. By induction on M . □

Lemma 3.1.40 proves an important result on which the inductive rule ($\Gamma 7$) of the algorithm is based. The relation $\not\sim_c$ was defined in Definition 2.1.10.

Lemma 3.1.40. *Let M, N be Λ -NF, $r \geq \max\{\text{args}(M), \text{args}(N)\}$, $y, z \in \text{Var}$ and $C_k^r[\cdot] \equiv (\lambda x_1 \dots x_k. [\cdot]) X_{x_1, \{y, z\}}^r \dots X_{x_k, \{y, z\}}^r$, for some $k \in \mathbb{N}$.*

- (i) $\exists \bar{M} \in \Lambda\text{-NF}$ such that $C_k^r[M] \rightarrow_{\Lambda}^{*} \bar{M}$ and $r \geq \text{args}(\bar{M})$.
- (ii) If $M \not\sim_c N$ then $\text{nf}_{\Lambda}(C_k^r[M]) \not\sim_c \text{nf}_{\Lambda}(C_k^r[N])$.

Proof. (i) By induction on M .

If $M \equiv \lambda u.P$ or $M \equiv uM_1 \dots M_m$ (where $u \notin \{x_1, \dots, x_k\}$ and $m \leq r$) then the proof follows by induction. Let $M \equiv x_j M_1 \dots M_m$ ($1 \leq j \leq k$, $m \leq r$); so by induction $\forall i \leq m$ there is $\bar{M}_i \in \Lambda\text{-NF}$ such that $C_k^r[M_i] \rightarrow_{\Lambda}^{*} \bar{M}_i$ and $r \geq \text{args}(\bar{M}_i)$. If $x_j \in \{y, z\}$ the proof is immediate, since $X_{x_j, \{y, z\}}^r \equiv x_j$. Let $x_j \notin \{y, z\}$; clearly

$$\begin{aligned} C_k^r[M] &\equiv (\lambda x_1 \dots x_k. x_j M_1 \dots M_m) X_{x_1, \{y, z\}}^r \dots X_{x_k, \{y, z\}}^r \rightarrow_{\Lambda}^{*} \\ &(\lambda u_1 \dots u_r. x_j u_1 \dots u_r) \bar{M}_1 \dots \bar{M}_m \rightarrow_{\Lambda}^{*} \lambda u_{m+1} \dots u_r. x_j \bar{M}_1 \dots \bar{M}_m u_{m+1} \dots u_r; \end{aligned}$$

so

$$r \geq \max\{r, \text{args}(\bar{M}_1), \dots, \text{args}(\bar{M}_m), \underbrace{0, \dots, 0}_{r-m}\} = r.$$

Note that $\text{nf}_{\Lambda}(C_k^r[M])$ is well defined.

- (ii) Let $M \equiv \lambda u_1 \dots u_p. u M_1 \dots M_m$ and $N \equiv \lambda v_1 \dots v_q. v N_1 \dots N_n$; we reason by induction on c . Let $c \equiv \epsilon$, $|p - m| \neq |q - n|$ and $v \equiv u$. $\forall i \bar{M}_i \equiv \text{nf}_{\Lambda}(C_k^r[M_i])$ and $\bar{N}_i \equiv \text{nf}_{\Lambda}(C_k^r[N_i])$. Let $u \equiv v \equiv x_j$, for some $j \leq k$ and $u, v \notin \{y, z\}$; otherwise the proof is simpler. Thus

$$\begin{aligned} \text{nf}_{\Lambda}(C_k^r[M]) &\equiv \lambda u_1 \dots u_p w_{m+1} \dots w_r. x_j \bar{M}_1 \dots \bar{M}_m w_{m+1} \dots w_r, \\ \text{nf}_{\Lambda}(C_k^r[N]) &\equiv \lambda v_1 \dots v_q w_{n+1} \dots w_r. x_j \bar{N}_1 \dots \bar{N}_n w_{n+1} \dots w_r. \end{aligned}$$

$|((p + (r - m)) - (m + (r - m)))| \neq |((q + (r - n)) - (n + (r - n)))|$, since $|p - m| \neq |q - n|$.

If $u \not\equiv v$ the proof is similar.

If $c \equiv i, c'$ (where $i \geq 1$) then the proof follows by induction. \square

The call-by-value separability algorithm is presented in fig. 3.1 (pag. 52). For sake of simplicity, in the algorithm description we assume that different bound variables have different names. The algorithm follows essentially the same pattern as the call-by-name separability algorithm, but the context in the conclusion replaces variables by terms having enough initial abstractions to assure that subterms become Γ -valuable, by using the result of Lemma 3.1.39. Note that, in rules $(\Gamma 3)$ – $(\Gamma 6)$, every occurrence of the term B^r in the context could be safely replaced by I , thereby following an approach similar to the call-by-name case. In fact such terms are erased when the context, filled by one of the two input terms, is Γ -reduced. Using B^r allows for an easier correctness proof.

Example 3.1.41. Let us consider the same terms as in Example 2.1.16, i.e. let $M \equiv \lambda x y u. x(u(x(yy))(vv))$ and $N \equiv \lambda x y u. x(u(yy)(vv))$.

Clearly $\text{args}(M) = \text{args}(N) = 2$, so let $r = 2$. The derivation proving the statement $M, N \Rightarrow_{\Gamma} C[\cdot]$ is the following:

$$\begin{array}{c}
 \frac{x_3 \not\equiv y \quad C_5[\cdot] \equiv (\lambda x_3 y [\cdot])(\lambda x_1 x_2 x_3 x_4 \tilde{x})(\lambda x_1 x_2 x_3 x_4 x_5 \tilde{y}) B^2 B^2}{x_3(yy)(\lambda v_1 v_2. x_2 v_1 v_2), yy(\lambda v_1 v_2. x_2 v_1 v_2)x_3 \Rightarrow_{\Gamma} C_5[\cdot]} \quad (\Gamma 5) \\
 \frac{}{\lambda x_2 x_3. x_3(yy)x_2, yy \Rightarrow_{\Gamma} C_2[\cdot] \equiv C_5[\cdot](\lambda v_1 v_2. x_2 v_1 v_2)x_3} \quad (\Gamma 2) \\
 \frac{}{u(\lambda x_2 x_3. x_3(yy)x_2)(vv), u(yy)(vv) \Rightarrow_{\Gamma} C_6[\cdot] \equiv C_2[(\lambda u [\cdot])(\lambda z_1 z_2. z_1)]} \quad (\Gamma 6) \\
 \frac{x(u(x(yy))(vv)), x(u(yy)(vv)) \Rightarrow_{\Gamma} C_7[\cdot] \equiv C_6[(\lambda x [\cdot])(\lambda x_1 x_2 x_3. x_3 x_1 x_2)B^2(\lambda z_1 z_2. z_1)]}{\lambda x y u. x(u(x(yy))(vv)), \lambda x y u. x(u(yy)(vv)) \Rightarrow_{\Gamma} C_7[\cdot] xy(\lambda u_1 u_2. uu_1 u_2)} \quad (\Gamma 7) \\
 \end{array}$$

where, if $\tilde{X}_4 \equiv (\lambda x_1 x_2 x_3 x_4 \tilde{x})$ and $\tilde{Y}_5 \equiv (\lambda x_1 x_2 x_3 x_4 x_5 \tilde{y})$:

$$\begin{array}{lll}
 C_5[\cdot] \equiv & (\lambda x_3 y [\cdot]) & \tilde{X}_4 \tilde{Y}_5 B^2 B^2 \\
 C_2[\cdot] \equiv & (\lambda x_3 y [\cdot](\lambda v_1 v_2. x_2 v_1 v_2)x_3) & \tilde{X}_4 \tilde{Y}_5 B^2 B^2 \\
 C_6[\cdot] \equiv & (\lambda x_3 y. ((\lambda u [\cdot]) U_2^1(\lambda v_1 v_2. x_2 v_1 v_2)x_3)) & \tilde{X}_4 \tilde{Y}_5 B^2 B^2 \\
 C_7[\cdot] \equiv & (\lambda x_3 y. ((\lambda u. (\lambda x [\cdot]) B^2 B^2 U_2^1) U_2^1(\lambda v_1 v_2. x_2 v_1 v_2)x_3)) & \tilde{X}_4 \tilde{Y}_5 B^2 B^2 \\
 C[\cdot] \equiv & (\lambda x_3 y. ((\lambda u. (\lambda x [\cdot]) xy(\lambda u_1 u_2. uu_1 u_2)) B^2 B^2 U_2^1) U_2^1(\lambda v_1 v_2. x_2 v_1 v_2)x_3)) & \tilde{X}_4 \tilde{Y}_5 B^2 B^2
 \end{array}$$

We can check that $(C[M])[O^2/v] \rightarrow_{\Gamma}^* \tilde{x}$; in fact,

Let $M, N \in \Lambda\text{-NF}$, $M \not\sim_c N$, $r \geq \max\{\text{args}(M), \text{args}(N)\}$ and \tilde{x}, \tilde{y} be fresh variables such that $\tilde{x} \not\equiv \tilde{y}$.

The rules of the system proving statements $M, N \Rightarrow_{\Gamma} C[.]$, are the following:

$$\frac{\begin{array}{c} p \leq q \\ C_k^r[.] \equiv (\lambda x_1 \dots x_k[.]) X_{x_1, \{x, y\}}^r \dots X_{x_k, \{x, y\}}^r \quad (k \in \{p, q\}) \\ x \mathsf{nf}_\Lambda(C_p^r[M_1]) \dots \mathsf{nf}_\Lambda(C_p^r[M_m]) X_{x_{p+1}, \{x, y\}}^r \dots X_{x_q, \{x, y\}}^r, \\ y \mathsf{nf}_\Lambda(C_q^r[N_1]) \dots \mathsf{nf}_\Lambda(C_q^r[N_n]) \end{array}}{\lambda x_1 \dots x_p.x M_1 \dots M_m, \lambda x_1 \dots x_q.y N_1 \dots N_n \Rightarrow_{\Gamma} C[.] X_{x_1, \{x, y\}}^r \dots X_{x_q, \{x, y\}}^r] \quad (\Gamma 1)$$

$$\frac{\begin{array}{c} p > q \\ C_k^r[.] \equiv (\lambda x_1 \dots x_k[.]) X_{x_1, \{x, y\}}^r \dots X_{x_k, \{x, y\}}^r \quad (k \in \{p, q\}) \\ y \mathsf{nf}_\Lambda(C_p^r[M_1]) \dots \mathsf{nf}_\Lambda(C_p^r[M_m]), \\ y \mathsf{nf}_\Lambda(C_q^r[N_1]) \dots \mathsf{nf}_\Lambda(C_q^r[N_n]) X_{x_{q+1}, \{x, y\}}^r \dots X_{x_p, \{x, y\}}^r \end{array}}{\lambda x_1 \dots x_p.x M_1 \dots M_m, \lambda x_1 \dots x_q.y N_1 \dots N_n \Rightarrow_{\Gamma} C[.] X_{x_1, \{x, y\}}^r \dots X_{x_p, \{x, y\}}^r] \quad (\Gamma 2)$$

$$\frac{n < m}{x M_1 \dots M_m, x N_1 \dots N_n \Rightarrow_{\Gamma} (\lambda x[.]) O^{r+n} \underbrace{B^r \dots B^r}_{r+n-m} (\lambda x_1 \dots x_{m-n}.\tilde{x}) \underbrace{\tilde{y} \dots \tilde{y}}_{m-n}} \quad (\Gamma 3)$$

$$\frac{m < n}{x M_1 \dots M_m, x N_1 \dots N_n \Rightarrow_{\Gamma} (\lambda x[.]) O^{r+m} \underbrace{B^r \dots B^r}_{r+m-n} (\lambda x_1 \dots x_{n-m}.\tilde{y}) \underbrace{\tilde{x} \dots \tilde{x}}_{n-m}} \quad (\Gamma 4)$$

$$\frac{x \not\equiv y}{x M_1 \dots M_m, y N_1 \dots N_n \Rightarrow_{\Gamma} (\lambda xy[.]) (\lambda x_1 \dots x_{r+m}.\tilde{x}) (\lambda x_1 \dots x_{r+n}.\tilde{y}) \underbrace{B^r \dots B^r}_r} \quad (\Gamma 5)$$

$$\frac{M_k \neq_{\Lambda\eta} N_k \quad x \notin \text{FV}(M_k) \cup \text{FV}(N_k) \quad M_k, N_k \Rightarrow_{\Gamma} C[.]}{x M_1 \dots M_m, x N_1 \dots N_m \Rightarrow_{\Gamma} C[(\lambda x[.]) U_k^r \underbrace{B^r \dots B^r}_{r-m}]} \quad (\Gamma 6)$$

$$\frac{\begin{array}{c} M_k \neq_{\Lambda\eta} N_k \quad x \in \text{FV}(M_k) \cup \text{FV}(N_k) \\ C_x^r[.] \equiv (\lambda x[.]) B^r \quad \mathsf{nf}_\Lambda(C_x^r[M_k]), \mathsf{nf}_\Lambda(C_x^r[N_k]) \Rightarrow_{\Gamma} C[.] \end{array}}{x M_1 \dots M_m, x N_1 \dots N_m \Rightarrow_{\Gamma} C[C_x^r[.] \underbrace{B^r \dots B^r}_{r-m} U_k^r]} \quad (\Gamma 7)$$

Fig. 3.1. Call-by-value separability algorithm

$$\begin{aligned}
& (\lambda x_3 y. ((\lambda u. (\lambda x. M[O^2/v]xy(\lambda u_1 u_2.uu_1 u_2))B^2 B^2 U_2^1)U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)x_3)) \tilde{X}_4 \tilde{Y}_5 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& (\lambda x_3 y. ((\lambda u. M[O^2/v]B^2 y(\lambda u_1 u_2.uu_1 u_2)B^2 U_2^1)U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)x_3)) \tilde{X}_4 \tilde{Y}_5 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& (\lambda x_3 y. (M[O^2/v]B^2 y(\lambda u_1 u_2.U_2^1 u_1 u_2)B^2 U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)x_3)) \tilde{X}_4 \tilde{Y}_5 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& (\lambda x y u. x(u(x(yy))(vv))) [O^2/v]B^2 \tilde{Y}_5(\lambda u_1 u_2.U_2^1 u_1 u_2)B^2 U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)\tilde{X}_4 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& (\lambda x y u. x(u(x(yy))(O^2 O^2))) B^2 \tilde{Y}_5(\lambda u_1 u_2.u_1)B^2 U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)\tilde{X}_4 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& B^2((\lambda u_1 u_2.u_1)(B^2(\tilde{Y}_5 \tilde{Y}_5))(O^2 O^2)) B^2 U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)\tilde{X}_4 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& B^2((\lambda u_1 u_2.u_1)(B^2(\lambda x_2 x_3 x_4 x_5.\tilde{y}))O^1) B^2 U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)\tilde{X}_4 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& B^2((\lambda u_1 u_2.u_1)(\lambda x_2 x_3.x_3(\lambda x_2 x_3 x_4 x_5.\tilde{y})x_2)O^1) B^2 U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)\tilde{X}_4 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& B^2(\lambda x_2 x_3.x_3(\lambda x_2 x_3 x_4 x_5.\tilde{y})x_2) B^2 U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)\tilde{X}_4 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& U_2^1(\lambda x_2 x_3.x_3(\lambda x_2 x_3 x_4 x_5.\tilde{y})x_2) B^2(\lambda v_1 v_2.x_2 v_1 v_2)\tilde{X}_4 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& (\lambda x_2 x_3.x_3(\lambda x_2 x_3 x_4 x_5.\tilde{y})x_2)(\lambda v_1 v_2.x_2 v_1 v_2)\tilde{X}_4 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& \tilde{X}_4(\lambda x_2 x_3 x_4 x_5.\tilde{y})(\lambda v_1 v_2.x_2 v_1 v_2)B^2 B^2 \rightarrow_{\Gamma}^{*} \tilde{x}.
\end{aligned}$$

While $(C[N])[O^2/v] \rightarrow_{\Gamma}^{*} \tilde{y}$; in fact,

$$\begin{aligned}
& (\lambda x_3 y. ((\lambda u. (\lambda x. N[O^2/v]xy(\lambda u_1 u_2.uu_1 u_2))B^2 B^2 U_2^1)U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)x_3)) \tilde{X}_4 \tilde{Y}_5 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& (\lambda x_3 y. ((\lambda u. N[O^2/v]B^2 y(\lambda u_1 u_2.uu_1 u_2)B^2 U_2^1)U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)x_3)) \tilde{X}_4 \tilde{Y}_5 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& (\lambda x_3 y. N[O^2/v]B^2 y(\lambda u_1 u_2.U_2^1 u_1 u_2)B^2 U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)x_3) \tilde{X}_4 \tilde{Y}_5 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& (\lambda x y u. x(u(yy))(O^2 O^2)) B^2 \tilde{Y}_5(\lambda u_1 u_2.U_2^1 u_1 u_2)B^2 U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)\tilde{X}_4 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& B^2((\lambda u_1 u_2.U_2^1 u_1 u_2)(\tilde{Y}_5 \tilde{Y}_5))(O^2 O^2)) B^2 U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)\tilde{X}_4 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& B^2((\lambda u_1 u_2.u_1)(\lambda x_2 x_3 x_4 x_5.\tilde{y})O^1) B^2 U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)\tilde{X}_4 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& B^2(\lambda x_2 x_3 x_4 x_5.\tilde{y}) B^2 U_2^1(\lambda v_1 v_2.x_2 v_1 v_2)\tilde{X}_4 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& U_2^1(\lambda x_2 x_3 x_4 x_5.\tilde{y}) B^2(\lambda v_1 v_2.x_2 v_1 v_2)\tilde{X}_4 B^2 B^2 \rightarrow_{\Gamma}^{*} \\
& (\lambda x_2 x_3 x_4 x_5.\tilde{y})(\lambda v_1 v_2.x_2 v_1 v_2)\tilde{X}_4 B^2 B^2 \rightarrow_{\Gamma}^{*} \tilde{y}.
\end{aligned}$$

Lemma 3.1.42 (Termination).

If $M, N \in \Lambda\text{-NF}$ and $M \not\leq_c N$ then $M, N \Rightarrow_{\Gamma} C[\cdot]$.

Proof. By induction on c . Similar to the termination proof (Lemma 2.1.17) of the Böhm Theorem, by using Lemma 3.1.40. \square

The next Lemma is necessary for proving the correctness.

Lemma 3.1.43. Let $z P_1 \dots P_m, T \in \Lambda\text{-NF}$, $\text{args}(z P_1 \dots P_m) \leq r$, $\text{args}(T) \leq r$. Let $D[\cdot] \equiv (\lambda u_1 \dots u_k [\cdot]) R_1 \dots R_h$ be a context ($k \leq h$) such that both $R_j \equiv \lambda x_1 \dots x_r. \bar{R}_j$ ($1 \leq j \leq h$) and $\text{nf}_{\Lambda}(D[T]) \equiv z P_1 \dots P_m$.

Let \mathbf{b} denote a substitution such that, if $y \in \text{FV}(D[T])$ then there is $Q^y \in \Lambda$ and $\mathbf{b}(y) = \lambda x_1 \dots x_r. Q^y$. In particular, $\mathbf{b}(z) = \lambda x_1 \dots x_r. Q^z$.

If $x_1, \dots, x_r \notin \text{FV}(Q^z)$ then $\mathbf{b}(D[T]) \rightarrow_{\Gamma}^{*} \lambda x_{r-m} \dots x_r. Q^z$.

Proof. The proof is given by induction on h .

$h = 0$. Clearly $D[\cdot] \equiv [\cdot]$, so T must be $z P_1 \dots P_m$.

$\mathbf{b}(P_j) \rightarrow_{\Gamma}^{*} \bar{P}_j \in \Gamma$ ($1 \leq j \leq m$), by Lemma 3.1.39; so, $\mathbf{b}(z P_1 \dots P_m) \rightarrow_{\Gamma}^{*} (\lambda x_1 \dots x_r. Q^z) \bar{P}_1 \dots \bar{P}_m \rightarrow_{\Gamma}^{*} \lambda x_{r-m} \dots x_r. Q^z$.

$h \geq 1$. If $k = 0$ then $TR_1...R_h \equiv zP_1...P_m$, so the proof is similar to the previous case.

Let $k \geq 1$, let $D'[\cdot]$ be $[\cdot]R_{k+1}...R_h$ and \mathbf{b}' denote a substitution such that $\mathbf{b}'(u_j) = R_j$ ($1 \leq j \leq k$), while $\mathbf{b}'(y) = \mathbf{b}(y)$ for all other variables. Hence, $\mathbf{b}'(D'[T]) \xrightarrow{\Gamma} \lambda x_{r-m}...x_r.Q^z$ by case $k = 0$; the proof follows, since $\mathbf{b}(D[zP_1...P_m]) \xrightarrow{\Gamma} \mathbf{b}'(D'[T])$. \square

In the next lemma, \tilde{x}, \tilde{y} are the pair of fresh variables considered by the algorithm.

Lemma 3.1.44 (Correctness).

Let M, N be different Λ -normals form such that $r \geq \max\{\text{args}(M), \text{args}(N)\}$. If $M, N \Rightarrow_{\Gamma} C[\cdot]$ and $\text{FV}(C[M]) \cup \text{FV}(C[N]) = \{u_1, \dots, u_n\} - \{\tilde{x}, \tilde{y}\}$ then $(\lambda u_1...u_n.C[M]) \underbrace{O^r...O^r}_n \xrightarrow{\Gamma} \tilde{x}$ and $(\lambda u_1...u_n.C[N]) \underbrace{O^r...O^r}_n \xrightarrow{\Gamma} \tilde{y}$.

Proof. Let \mathbf{e} denote a substitution such that $\mathbf{e}(\tilde{x}) = \tilde{x}$, $\mathbf{e}(\tilde{y}) = \tilde{y}$, while $\forall z \in \text{Var} - \{\tilde{x}, \tilde{y}\}$, $\mathbf{e}(z) = O^r$.

We will prove that $\mathbf{e}(C[M]) \xrightarrow{\Gamma} \tilde{x}$ and $\mathbf{e}(C[N]) \xrightarrow{\Gamma} \tilde{y}$;

moreover, let $T \in \Lambda\text{-NF}$, let $\text{args}(T) \leq r$, let $D[\cdot] \equiv (\lambda u_1...u_k[\cdot]).R_1...R_h$ be a context ($k \leq h$) such that either $R_j = B^r$ or $R_j = U_t^r$ or $R_j = X_{u,S}^r$ (where $1 \leq j \leq h$, $t \leq r$, $u \in \text{Var}$ and $S \subseteq \text{Var}$) and let $\tilde{x}, \tilde{y} \notin \text{FV}(D[T])$, so:

- if $\text{nf}_{\Lambda}(D[T]) \equiv M$ then $\mathbf{e}(C[D[T]]) \xrightarrow{\Gamma} \tilde{x}$,
- if $\text{nf}_{\Lambda}(D[T]) \equiv N$ then $\mathbf{e}(C[D[T]]) \xrightarrow{\Gamma} \tilde{y}$.

The proof is given by induction on the derivation proving $M, N \Rightarrow_{\Gamma} C[\cdot]$.

(Γ 1) Let $C_p^r[\cdot] \equiv (\lambda x_1...x_p[\cdot]).X_{x_1,\{x,y\}}^r...X_{x_p,\{x,y\}}^r$ and

$$\left. \begin{array}{l} x\text{nf}_{\Lambda}(C_p^r[M_1])...x\text{nf}_{\Lambda}(C_p^r[M_m])X_{x_{p+1},\{x,y\}}^r...X_{x_q,\{x,y\}}^r, \\ y\text{nf}_{\Lambda}(C_q^r[N_1])...y\text{nf}_{\Lambda}(C_q^r[N_n]) \end{array} \right\} \Rightarrow_{\Gamma} C[\cdot];$$

so the two inductive hypothesis follow.

- $\mathbf{e}(C[x\text{nf}_{\Lambda}(C_p^r[M_1])...x\text{nf}_{\Lambda}(C_p^r[M_m])X_{x_{p+1},\{x,y\}}^r...X_{x_q,\{x,y\}}^r]) \xrightarrow{\Gamma} \tilde{x}$.
- Moreover, if $T \in \Lambda\text{-NF}$, $\text{args}(T) \leq r$, $D[\cdot] \equiv (\lambda u_1...u_k[\cdot]).R_1...R_h$ is a context ($k \leq h$), where $R_j \in \{B^r, U_t^r, X_{u,S}^r \mid t \in \mathbb{N} \wedge u \in \text{Var} \wedge S \subseteq \text{Var}\}$ ($1 \leq j \leq h$),

$$\text{nf}_{\Lambda}(D[T]) \equiv x\text{nf}_{\Lambda}(C_p^r[M_1])...x\text{nf}_{\Lambda}(C_p^r[M_m])X_{x_{p+1},\{x,y\}}^r...X_{x_q,\{x,y\}}^r$$

and $\tilde{x}, \tilde{y} \notin \text{FV}(D[T])$ then $\mathbf{e}(C[D[T]]) \xrightarrow{\Gamma} \tilde{x}$.

Let $T \equiv xM_1...M_m$ and $D[\cdot] \equiv C_p^r[\cdot]$; so, $\text{args}(T) \leq r$, $\tilde{x}, \tilde{y} \notin \text{FV}(D[T])$ and $\text{nf}_{\Lambda}(C_p^r[T]) \equiv x\text{nf}_{\Lambda}(C_p^r[M_1])...x\text{nf}_{\Lambda}(C_p^r[M_m])X_{x_{p+1},\{x,y\}}^r...X_{x_q,\{x,y\}}^r$ imply, by induction, $\mathbf{e}(C[D[xM_1...M_m]]) \xrightarrow{\Gamma} \tilde{x}$, so

$$\mathbf{e}\left(C[(\lambda x_1 \dots x_p.xM_1 \dots M_m)X_{x_1, \{x,y\}}^r \dots X_{x_q, \{x,y\}}^r]\right) \rightarrow_{\Gamma}^{*} \tilde{x}.$$

We must yet prove that, if $T^* \in \Lambda\text{-NF}$, $D^*[.] \equiv (\lambda u_1^* \dots u_{k^*}^*.[.])R_1^* \dots R_{h^*}^*$ is a context ($k^* \leq h^*$) such that either $R_j^* = B^r$ or $R_j^* = U_t^r$ or $R_j^* = X_{u,S}^r$ (where $1 \leq j \leq h^*$, $t \in \mathbb{N}$, $u \in \text{Var}$ and $S \subseteq \text{Var}$), $\text{args}(T^*) \leq r$, $\tilde{x}, \tilde{y} \notin \text{FV}(D^*[T^*])$ and $\text{nf}_\Lambda(D^*[T^*]) \equiv \lambda x_1 \dots x_p.xM_1 \dots M_m$ then

$$\mathbf{e}(C^*[D^*[T^*]]) \rightarrow_{\Gamma}^{*} \tilde{x}, \text{ where } C^*[.] \equiv C[.]X_{x_1, \{x,y\}}^r \dots X_{x_q, \{x,y\}}^r.$$

Let $T \equiv T^*$ and $D[.] \equiv D^*[.]X_{x_1, \{x,y\}}^r \dots X_{x_q, \{x,y\}}^r$; thus, both

$$\begin{aligned} \text{nf}_\Lambda(D^*[T^*]X_{x_1, \{x,y\}}^r \dots X_{x_q, \{x,y\}}^r) \\ \equiv x\text{nf}_\Lambda(C_p^r[M_1]) \dots \text{nf}_\Lambda(C_p^r[M_m])X_{x_{p+1}, \{x,y\}}^r \dots X_{x_q, \{x,y\}}^r, \end{aligned}$$

$\text{args}(T) \leq r$ and $\tilde{x}, \tilde{y} \notin \text{FV}(D[T^*])$. By induction,

$$\mathbf{e}\left(C[D[T^*]]\right) \equiv \mathbf{e}\left(C^*[D^*[T^*]]\right) \rightarrow_{\Gamma}^{*} \tilde{x}.$$

The proof for the term on the right is similar.

($\Gamma 2$) Similar to ($\Gamma 1$).

($\Gamma 3$) Let $n < m$ and let $xM_1 \dots M_m, xN_1 \dots N_n \Rightarrow_{\Gamma} C[.]$ where

$$C[.] \equiv (\lambda x.[.])O^{r+n} \underbrace{B^r \dots B^r}_{r+n-m} (\lambda x_1 \dots x_{m-n}.\tilde{x}) \underbrace{\tilde{y} \dots \tilde{y}}_{m-n}.$$

We will prove that, if $T \in \Lambda\text{-NF}$, $\text{args}(T) \leq r$, $D[.] \equiv (\lambda u_1 \dots u_k.[.])R_1 \dots R_h$ is a context ($h, k \in \mathbb{N}$), where

$$R_j \in \{B^r, U_t^r, X_{u,S}^r \mid t \in \mathbb{N} \wedge u \in \text{Var} \wedge S \subseteq \text{Var}\} \quad (1 \leq j \leq h),$$

$\tilde{x}, \tilde{y} \notin \text{FV}(D[T])$ and $\text{nf}_\Lambda(D[T]) \equiv xM_1 \dots M_m$ then $\mathbf{e}(C[D[T]]) \rightarrow_{\Gamma}^{*} \tilde{x}$.

Let $\mathbf{b}(x) = O^{r+n}$ and $\mathbf{b}(y) = \mathbf{e}(y)$ for all other variables; therefore, by Lemma 3.1.43, $\mathbf{b}(D[T])) \rightarrow_{\Gamma}^{*} O^{r+n-m}$. Hence

$$\begin{aligned} \mathbf{e}\left((\lambda x.D[T])O^{r+n} \underbrace{B^r \dots B^r}_{r+n-m} (\lambda x_1 \dots x_{m-n}.\tilde{x}) \underbrace{\tilde{y} \dots \tilde{y}}_{m-n}\right) \rightarrow_{\Gamma}^{*} \\ O^{r+n-m} \underbrace{B^r \dots B^r}_{r+n-m} (\lambda x_1 \dots x_{m-n}.\tilde{x}) \underbrace{\tilde{y} \dots \tilde{y}}_{m-n} \rightarrow_{\Gamma}^{*} \tilde{x}. \end{aligned}$$

On the other hand, we will prove that, if $T \in \Lambda\text{-NF}$, $\text{args}(T) \leq r$, $D[.] \equiv (\lambda u_1 \dots u_k.[.])R_1 \dots R_h$ is a context ($h, k \in \mathbb{N}$), where

$$R_j \in \{B^r, U_t^r, X_{u,S}^r \mid t \in \mathbb{N} \wedge u \in \text{Var} \wedge S \subseteq \text{Var}\} \quad (1 \leq j \leq h),$$

$\tilde{x}, \tilde{y} \notin \text{FV}(D[T])$ and $\text{nf}_\Lambda(D[T]) \equiv xN_1 \dots N_n$ then $\mathbf{e}(C[D[T]]) \rightarrow_{\Gamma}^{*} \tilde{y}$.

Let $\mathbf{b}(x) = O^{r+n}$ and $\mathbf{b}(y) = \mathbf{e}(y)$ for all other variables; therefore, by Lemma 3.1.43, $\mathbf{b}(D[T])) \rightarrow_{\Gamma}^{*} O^r$. Hence

$$\begin{aligned} & \mathbf{e}\left((\lambda x.D[T])O^{r+n} \underbrace{B^r \dots B^r}_{r+n-m} (\lambda x_1 \dots x_{m-n}.\tilde{x}) \underbrace{\tilde{y} \dots \tilde{y}}_{m-n}\right) \rightarrow_{\Gamma}^* \\ & O^r \underbrace{B^r \dots B^r}_{r+1} (\lambda x_1 \dots x_{m-n}.\tilde{x}) \underbrace{\tilde{y} \dots \tilde{y}}_{m-n} \rightarrow_{\Gamma}^* \tilde{y}. \end{aligned}$$

($\Gamma 4$) Symmetric to ($\Gamma 3$).

($\Gamma 5$) Let $xM_1 \dots M_m, yN_1 \dots N_n \Rightarrow_{\Gamma} C[\cdot]$ where

$$C[\cdot] \equiv (\lambda xy.[\cdot])(\lambda x_1 \dots x_{r+m}.\tilde{x})(\lambda x_1 \dots x_{r+n}.\tilde{y}) \underbrace{B^r \dots B^r}_r.$$

We will prove that, if $T \in \Lambda\text{-NF}$, $\text{args}(T) \leq r$, $D[\cdot] \equiv (\lambda u_1 \dots u_k.[\cdot])R_1 \dots R_h$ is a context ($h, k \in \mathbb{N}$), where

$$R_j \in \{B^r, U_t^r, X_{u,S}^r \mid t \in \mathbb{N} \wedge u \in \text{Var} \wedge S \subseteq \text{Var}\} \quad (1 \leq j \leq h),$$

$\tilde{x}, \tilde{y} \notin \text{FV}(D[T])$ and $\text{nf}_{\Lambda}(D[T]) \equiv xM_1 \dots M_m$ then $\mathbf{e}(C[D[T]]) \rightarrow_{\Gamma}^* \tilde{x}$.
Let $\mathbf{b}(x) = \lambda x_1 \dots x_{r+m}.\tilde{x}$, $\mathbf{b}(y) = \lambda x_1 \dots x_{r+n}.\tilde{y}$ and $\mathbf{b}(z) = \mathbf{e}(z)$ for each other variable z ; thus, by Lemma 3.1.43, $\mathbf{b}(D[T]) \rightarrow_{\Gamma}^* \lambda x_1 \dots x_r.\tilde{x}$. Hence
 $\mathbf{e}\left((\lambda xy.xM_1 \dots M_m)(\lambda x_1 \dots x_{r+m}.\tilde{x})(\lambda x_1 \dots x_{r+n}.\tilde{y}) \underbrace{B^r \dots B^r}_r\right) \rightarrow_{\Gamma}^* \tilde{x}.$

On the other hand, we will prove that, if $T \in \Lambda\text{-NF}$, $\text{args}(T) \leq r$, and $D[\cdot] \equiv (\lambda u_1 \dots u_k.[\cdot])R_1 \dots R_h$ is a context ($h, k \in \mathbb{N}$) where

$$R_j \in \{B^r, U_t^r, X_{u,S}^r \mid t \in \mathbb{N} \wedge u \in \text{Var} \wedge S \subseteq \text{Var}\} \quad (1 \leq j \leq h),$$

$\tilde{x}, \tilde{y} \notin \text{FV}(D[T])$ and $\text{nf}_{\Lambda}(D[T]) \equiv yN_1 \dots N_n$ then $\mathbf{e}(C[D[T]]) \rightarrow_{\Gamma}^* \tilde{y}$.
Let $\mathbf{b}(x) = \lambda x_1 \dots x_{r+m}.\tilde{x}$, $\mathbf{b}(y) = \lambda x_1 \dots x_{r+n}.\tilde{y}$ and $\mathbf{b}(z) = \mathbf{e}(z)$ for each other variable z ; thus, by Lemma 3.1.43, $\mathbf{b}(D[T]) \rightarrow_{\Gamma}^* \lambda x_1 \dots x_r.\tilde{y}$. Hence
 $\mathbf{e}\left((\lambda xy.yN_1 \dots N_n)(\lambda x_1 \dots x_{r+m}.\tilde{x})(\lambda x_1 \dots x_{r+n}.\tilde{y}) \underbrace{B^r \dots B^r}_r\right) \rightarrow_{\Gamma}^* \tilde{y}.$

($\Gamma 6$) Let $M_k \neq_{\Lambda\eta} N_k$, $x \notin \text{FV}(M_k) \cup \text{FV}(N_k)$ and $M_k, N_k \Rightarrow_{\Gamma} C[\cdot]$.

The two inductive hypothesis follow.

- $\mathbf{e}(C[M_k]) \rightarrow_{\Gamma}^* \tilde{x}$.
- If $T \in \Lambda\text{-NF}$, $\text{args}(T) \leq r$, $D[\cdot] \equiv (\lambda u_1 \dots u_k.[\cdot])R_1 \dots R_h$ is a context ($k \leq h$), where

$$R_j \in \{B^r, U_t^r, X_{u,S}^r \mid t \in \mathbb{N} \wedge u \in \text{Var} \wedge S \subseteq \text{Var}\} \quad (1 \leq j \leq h),$$

$\text{nf}_{\Lambda}(D[T]) \equiv M_k$ and $\tilde{x}, \tilde{y} \notin \text{FV}(D[T])$ then $\mathbf{e}(C[D[T]]) \rightarrow_{\Gamma}^* \tilde{x}$.

Let $T \equiv xM_1 \dots M_m$ and $D[\cdot] \equiv (\lambda x.[\cdot])U_k^r \underbrace{B^r \dots B^r}_{r-m}$; thus, $\text{args}(T) \leq r$,
 $\tilde{x}, \tilde{y} \notin \text{FV}(D[T])$ and $\text{nf}_{\Lambda}(D[T]) \equiv M_k$ imply, by induction,

$$\mathbf{e}(C[D[T]]) \equiv \mathbf{e}\left(C[(\lambda x.xM_1 \dots M_m)U_k^r \underbrace{B^r \dots B^r}_{r-m}]\right) \rightarrow_{\Gamma}^* \tilde{x}.$$

We must yet prove that, if $T^* \in \Lambda\text{-NF}$, $D^*[.] \equiv (\lambda u_1^* \dots u_{k^*}^* . [.]) R_1^* \dots R_{h^*}^*$ is a context ($k^* \leq h^*$) such that either $R_j^* = B^r$ or $R_j^* = U_t^r$ or $R_j^* = X_{u,S}^r$ (where $1 \leq j \leq h^*$, $t \in \mathbb{N}$, $u \in \text{Var}$ and $S \subseteq \text{Var}$), $\text{args}(T^*) \leq r$, $\tilde{x}, \tilde{y} \notin \text{FV}(D^*[T^*])$ and $\text{nf}_\Lambda(D^*[T^*]) \equiv xM_1 \dots M_m$, then

$$\mathbf{e}(C^*[D^*[T^*]]) \rightarrow_\Gamma^* \tilde{x}, \text{ where } C^*[.] \equiv C[(\lambda x . [.]) U_k^r \underbrace{B^r \dots B^r}_{r-m}].$$

Let $D[.] \equiv (\lambda x u_1^* \dots u_{k^*}^* . [.]) U_k^r R_1^* [U_k^r / x] \dots R_{h^*}^* [U_k^r / x] \underbrace{B^r \dots B^r}_{r-m}$ and $T \equiv T^*$;

therefore $R_j^* [U_k^r / x] \in \{B^r, U_t^r, X_{u,S}^r \mid t \in \mathbb{N} \wedge u \in \text{Var} \wedge S \subseteq \text{Var}\}$ ($1 \leq j \leq h^*$), $\text{nf}_\Lambda(D[T]) \equiv M_k$ (since $x \notin \text{FV}(M_k)$), $\text{args}(T) \leq r$ and $\tilde{x}, \tilde{y} \notin \text{FV}(D[T])$. So, by induction,

$$\mathbf{e}\left(C[(\lambda x . D^*[T^*]) U_k^r \underbrace{B^r \dots B^r}_{r-m}]\right) =_\Gamma \mathbf{e}\left(C[D[T]]\right) \rightarrow_\Gamma^* \tilde{x}.$$

The proof for the term on the right is similar.

- ($\Gamma 7$) Let $M_k \neq_{\Lambda\eta} N_k$ and $x \in \text{FV}(M_k) \cup \text{FV}(N_k)$; furthermore, let $C_x^r[.] \equiv (\lambda x . [.]) B^r$ and $\text{nf}_\Lambda(C_x^r[M_k]), \text{nf}_\Lambda(C_x^r[N_k]) \Rightarrow_\Gamma C[.]$.

The two inductive hypothesis follow.

- $\mathbf{e}(C[\text{nf}_\Lambda(C_x^r[M_k])]) \rightarrow_\Gamma^* \tilde{x}$.
- moreover, if $T \in \Lambda\text{-NF}$, $\text{args}(T) \leq r$, and $D[.] \equiv (\lambda u_1 \dots u_k . [.]) R_1 \dots R_h$ is a context ($k \leq h$), where

$$R_j \in \{B^r, U_t^r, X_{u,S}^r \mid t \in \mathbb{N} \wedge u \in \text{Var} \wedge S \subseteq \text{Var}\} \quad (1 \leq j \leq h),$$

$\text{nf}_\Lambda(D[T]) \equiv \text{nf}_\Lambda(C_x^r[M_k])$ and $\tilde{x}, \tilde{y} \notin \text{FV}(D[T])$, then $\mathbf{e}(C[D[T]]) \rightarrow_\Gamma^* \tilde{x}$.

Let $T \equiv xM_1 \dots M_m$ and $D[.] \equiv (\lambda x . [.]) B^r \underbrace{B^r \dots B^r}_{r-m} U_k^r$; so, $\text{args}(T) \leq r$, $\tilde{x}, \tilde{y} \notin \text{FV}(D[T])$ and $\text{nf}_\Lambda(D[T]) \equiv \text{nf}_\Lambda(C_x^r[M_k])$ imply, by induction,

$$\mathbf{e}\left(C[D[T]]\right) \equiv \mathbf{e}\left(C[(\lambda x . xM_1 \dots M_m) B^r \underbrace{B^r \dots B^r}_{r-m} U_k^r]\right) \rightarrow_\Gamma^* \tilde{x}.$$

We must yet prove that, if $T^* \in \Lambda\text{-NF}$, $D^*[.] \equiv (\lambda u_1^* \dots u_{k^*}^* . [.]) R_1^* \dots R_{h^*}^*$ is a context ($k^* \leq h^*$) such that either $R_j^* = B^r$ or $R_j^* = U_t^r$ or $R_j^* = X_{u,S}^r$ (where $1 \leq j \leq h^*$, $t \in \mathbb{N}$, $u \in \text{Var}$ and $S \subseteq \text{Var}$), $\text{args}(T^*) \leq r$, $\tilde{x}, \tilde{y} \notin \text{FV}(D^*[T^*])$ and $\text{nf}_\Lambda(D^*[T^*]) \equiv xM_1 \dots M_m$, then

$$\mathbf{e}(C^*[D^*[T^*]]) \rightarrow_\Gamma^* \tilde{x}, \text{ where } C^*[.] \equiv C[(\lambda x . [.]) B^r \underbrace{B^r \dots B^r}_{r-m} U_k^r].$$

Let $D[.] \equiv (\lambda x u_1^* \dots u_{k^*}^* . [.]) B^r R_1^* [B^r / x] \dots R_{h^*}^* [B^r / x] \underbrace{B^r \dots B^r}_{r-m} U_k^r$ and let

$T \equiv T^*$. Thus $R_j^* [B^r / x] \in \{B^r, U_t^r, X_{u,S}^r \mid t \in \mathbb{N} \wedge u \in \text{Var} \wedge S \subseteq \text{Var}\}$ where $1 \leq j \leq h^*$, $\text{nf}_\Lambda(D[T]) \equiv \text{nf}_\Lambda(C_x^r[M_k])$, $\text{args}(T) \leq r$ and $\tilde{x}, \tilde{y} \notin \text{FV}(D[T])$. By induction,

$$\mathbf{e}\left(C[(\lambda x.D^*[T^*])B^r \underbrace{B^r \dots B^r}_{r-m} U_k^r]\right) =_R \mathbf{e}\left(C[D[T]]\right) \rightarrow_R^* \tilde{x}.$$

The proof for the term on the right is similar. \square

▼ Proof of Γ -Separability Theorem (Theorem 3.1.16 pag. 40).

The proof follows directly from Lemmas 3.1.42 and 3.1.44. \blacksquare

3.2 Potentially Γ -Valuable Terms and Λ -Reduction

In this section the relation between the call-by-value $\lambda\Gamma$ -calculus and the call-by-name $\lambda\Lambda$ -calculus is explored. In particular, we show that the potentially Γ -valuable terms, which were characterized through the notion of $\Xi\ell$ -reduction, introduced on purpose, coincide with the strongly normalizing terms with respect to the $\Lambda\ell$ -reduction. The notion of $\Lambda\ell$ -reduction is a particular case of Definition 3.1.3. In order to prove the result, the notion of weight of a term, introduced in Definition 3.1.29, will be used.

Lemma 3.2.1. *If $\langle M \rangle$ is defined then M has $\Xi\ell$ -normal form.*

Proof. By induction on $\langle M \rangle$.

$\langle M \rangle = 0$. Trivial, since M is an abstraction.

$\langle M \rangle \geq 1$. If $M \equiv xM_1\dots M_m$ ($m \in \mathbb{N}$) the proof follows by induction. Let $M \equiv (\lambda x.P)QM_1\dots M_m$ ($m \in \mathbb{N}$). By induction $P[Q/x]M_1\dots M_m$ and Q have $\Xi\ell$ -normal forms, so let $Q \rightarrow_{\Xi\ell}^* R \in \Xi$; by Property 3.1.33.(ii)

$$\langle P[R/x]M_1\dots M_m \rangle \leq \langle P[Q/x]M_1\dots M_m \rangle < \langle M \rangle,$$

so the proof follows by induction. \square

Corollary 3.2.2. *M has $\Xi\ell$ -normal form if and only if $\langle M \rangle$ is defined.*

Proof. By the previous lemma and Lemma 3.1.31.(iii). \square

The next lemma proves that the notion of weight also works well for the $\Lambda\ell$ -reduction.

Lemma 3.2.3. *Let $M \in \Lambda$ and $\langle M \rangle$ be defined.*

If $M \rightarrow_{\Lambda\ell} N$ then $\langle N \rangle$ is defined and $\langle N \rangle < \langle M \rangle$.

Proof. The proof is given by induction on $k = \langle M \rangle$.

The case where $k = 0$ is not possible, since M is an abstraction and it cannot be $\Lambda\ell$ -reduced; so let $k \geq 1$.

If $M \equiv xM_1\dots M_m$ then the proof follows by induction.

Let $M \equiv (\lambda x.M_0)M_1\dots M_m$, $h' = \langle M_1 \rangle$ and $h'' = \langle M_0[M_1/x]M_2\dots M_m \rangle$, thus $k = 1 + h' + h''$. There are only three cases, by the laziness of $\rightarrow_{\Lambda\ell}$.

- (i) If $N \equiv M_0[M_1/x]M_2\dots M_m$ then the proof follows from the definition of weight.
- (ii) Let $N \equiv (\lambda x.M_0)M_1N_2\dots N_m$ where there is a unique $j \geq 2$ such that $M_j \rightarrow_{\Lambda\ell} N_j$, while $\forall i \neq j \quad M_i \equiv N_i$ ($0 \leq i \leq m$ and $i \neq 1$).
 $M_0[M_1/x]M_2\dots M_m \rightarrow_{\Lambda\ell} M_0[M_1/x]N_2\dots N_m$ and $h'' < k$ imply, by induction, $\langle N_0[M_1/x]N_2\dots N_m \rangle < h''$.
Thus $\langle N \rangle = 1 + \langle M_1 \rangle + \langle N_0[M_1/x]N_2\dots N_m \rangle < k$.
- (iii) Let $N \equiv (\lambda x.M_0)N_1M_2\dots M_m$, where $M_1 \rightarrow_{\Lambda\ell} N_1$. By Property 3.1.33.(ii) we can state $\langle M_0[N_1/x]M_2\dots M_m \rangle \leq h''$, since $M_0[M_1/x]M_2\dots M_m \xrightarrow{\Lambda}^* M_0[N_1/x]M_2\dots M_m$. Again, by induction $\langle M_1 \rangle < \langle N_1 \rangle$; so the proof follows by the definition of weight. \square

Theorem 3.2.4.

M has $\Xi\ell$ -normal form if and only if M is $\Lambda\ell$ -strongly normalizing.

Proof. (\Leftarrow) Trivial, since $M \rightarrow_{\Xi\ell} N$ implies $M \rightarrow_{\Lambda\ell} N$.

(\Rightarrow) By Corollary 3.2.2, $\langle M \rangle$ is defined. Let N be such that $M \rightarrow_{\Lambda\ell} N$; thus by Lemma 3.2.3, both $\langle N \rangle$ is defined and $\langle N \rangle < \langle M \rangle$. This implies that there is not an infinite sequence of $\Lambda\ell$ -reductions starting from M , and so M is $\Lambda\ell$ -strongly normalizing, by Definition 3.1.3.(iv). \square

Corollary 3.2.5 shows the desired result.

Corollary 3.2.5. Let $M \in \Lambda^0$.

M is Γ -valuable if and only if M is $\Lambda\ell$ -strongly normalizing.

Proof. From Theorems 3.1.9 and 3.2.4. \square

4. Further Reading

Λ -separability. The separability property of Λ -normal forms was extended to finite sets of different $\Lambda\eta$ -normal forms in [18], and the separability of infinite sets of Λ -normal forms was studied in [84]. An algebraic analysis of the technique used by Böhm for proving his theorem was developed in [76]. More refined notions of separability were studied in [20] and [22].

Call-by-value λ -calculus. Extensions of the $\lambda\Gamma$ -calculus for studying imperative and control features in the call-by-value setting were introduced respectively in [45] and [46]. Moggi [70], starting from the $\lambda\Gamma$ -calculus, developed a further paradigmatic language for reasoning about the call-by-value computation, called the partial λ -calculus.

Call-by-value versus call-by-name. Some interesting observations on the relationship between call-by-value and call-by-name computation can be found in [33], in a typed setting, where it was shown that call-by-value is the De Morgan dual of call-by-name. This idea was further developed in [96].

Part II

Operational Semantics

5. Parametric Operational Semantics

In this part we will study the evaluation of terms and the induced operational semantics. Our notion of operational semantics is inspired by the structured operational semantics (SOS) developed by Plotkin [80] and by Kahn [55].

In Sect. 1.2, we introduced in an informal way the notion of evaluation, by saying that a possible way of evaluating a term is to apply the reduction rule to it until a normal form is reached. Clearly, such evaluation can never stop, for example, in the case when $D \in \Delta$ and the Δ -reduction is applied to the term DD , which do not have Δ -normal form.

But the normal forms are not the only terms we can reasonably consider as output results. For example, we defined the notion of head normal form, both in the $\lambda\Lambda$ and in the $\lambda\Gamma$ setting. It is natural to ask if such terms can be considered as output values, and so if it is possible to check, through an evaluation, whether or not a term possesses head normal form.

Hence, in order to study the evaluation of terms, we need to introduce behind the notion of input values, that of output values. The definition of a set of output results is parametric with respect to the set of input values.

Definition 5.0.1. *Let Δ be a set of input values.*

A set of output values with respect to Δ is any set $\Theta \subseteq \Lambda$ such that:

- (i) *Θ contains all the Δ -normal forms,*
- (ii) *if $M =_{\Delta} N$ and $N \in \Theta$ then there is $P \in \Theta$ such that $M \rightarrow_{\Delta}^{*p} P$*
(principality condition).

The first condition of the previous definition takes into account the fact that the set of normal forms is in some sense the most “natural” set of output values, corresponding to the complete evaluation of terms. Remember that Corollary 1.2.13 assures us that, to reach the normal form of a term, if it exists, it is sufficient to perform at every step the principal redex. So the second condition simply says that we are interested in those evaluations that are an initial step of the complete one. As we show in the following, each evaluation of interest is of this kind.

Lemma 5.0.2. *Let $\Theta \subseteq \Lambda$ be such that $\Delta\text{-NF} \subseteq \Theta$.*

If Θ is closed under \rightarrow_{Δ} and the set $\{M \in \Lambda \mid M \notin \Theta\}$ is closed under \rightarrow_{Δ}^i , then Θ is a set of output values with respect to Δ .

Proof. We must prove that Θ satisfies the principality condition.

If $M =_{\Delta} N \in \Theta$ then there is a term $M' \in \Lambda$ such that $M \rightarrow_{\Delta}^* M'$ and $N \rightarrow_{\Delta}^* M'$ by the confluence theorem; so $M' \in \Theta$ by the fact that output values are closed under \rightarrow_{Δ} . By the standardization theorem, there is a standard reduction sequence $M \rightarrow_{\Delta}^{*p} M'' \rightarrow_{\Delta}^* M'$; hence $M' \in \Theta$ implies $M'' \in \Theta$, by the fact that $\{M \in \Lambda \mid M \notin \Theta\}$ is closed under \rightarrow_{Δ}^i . \square

The next property shows some examples of sets of output values.

Property 5.0.3. 1. Λ , Λ -NF, Λ -HNF and Λ -LHNF are sets of output values with respect to Λ .

2. Λ and Γ -NF are sets of output values with respect to Γ .
3. The set of Γ -lazy blocked normal forms (Γ -lbnf's), namely Γ -LBNF = $\{\lambda x.M \mid M \in \Lambda\} \cup \{xM_1 \dots M_m \mid M_i \in \Lambda, m \in \mathbb{N}\} \cup \{(\lambda x.P)QM_1 \dots M_m \mid P, M_i \in \Lambda, Q \notin \Gamma, Q \in \Gamma$ -LBNF, $m \in \mathbb{N}\}$, is a set of output values with respect to Γ .
4. Γ is not a set of output values with respect to either Λ nor Γ .
5. Ξ is a set of output values with respect to Λ , but not with respect to Γ .
6. $\Gamma\ell$ -NF is not a set of output values with respect to either Λ nor Γ .

Proof. 1. The case for Λ is trivial. In case $\Theta \in \{\Lambda\text{-NF}, \Lambda\text{-HNF}, \Lambda\text{-LHNF}\}$, the proof follows by Lemma 5.0.2. In fact,

- Λ -NF $\subseteq \Lambda$ -HNF $\subseteq \Lambda$ -LHNF,
- if $P \in \Theta$ and $P \rightarrow_{\Lambda} Q$ then $Q \in \Theta$,
- if $P \rightarrow_{\Lambda}^i Q$ and $P \notin \Theta$ then $Q \notin \Theta$.

2. The case for Λ is trivial.

Γ -NF is a set of output values with respect to Γ by Corollary 1.2.13.

3. The proof follows by Lemma 5.0.2. In fact,

- Γ -NF $\subseteq \Gamma$ -LBNF,
- if $P \in \Gamma$ -LBNF and $P \rightarrow_{\Gamma} Q$ then $Q \in \Gamma$ -LBNF,
- if $P \rightarrow_{\Gamma}^i Q$ and $P \notin \Gamma$ -LBNF then $Q \notin \Gamma$ -LBNF.

4. In fact, $xI \in \Lambda$ -NF and $xI \in \Gamma$ -NF, but $xI \notin \Gamma$.

5. It is easy to see that $\Xi \equiv \Lambda\ell$ -NF (see Definition 3.1.3 and Property 3.1.6), thus Λ -NF $\subseteq \Xi$; so Ξ is a set of output values with respect to Λ , by Lemma 5.0.2. But it is not a set of output values with respect to Γ ; in fact, $I(xI) \in \Gamma$ -NF, but $I(xI) \notin \Xi$.

6. Let $\Delta \in \{\Lambda, \Gamma\}$; thus $(\lambda x.DD)(xI)(II) =_{\Delta\ell} (\lambda x.DD)(xI)I \in \Gamma\ell$ -NF; nevertheless, there is no $P \in \Delta\ell$ -NF such that $(\lambda x.DD)(xI)(II) \rightarrow_{\Delta}^{*p} P$, against the principality condition. \square

In what follows, Θ always denotes a generic set of output values.

Definition 5.0.4. Let Θ be a set of output values with respect to Δ .

- (i) An evaluation relation \mathbf{O} on the $\lambda\Delta$ -calculus with respect to Θ is any subset of $\Lambda \times \Theta$, such that $(M, N) \in \mathbf{O}$ implies $M \rightarrow_{\Delta}^* N$.

- (ii) $\mathcal{E}(\Delta, \Theta)$ denotes the class of all evaluation relations \mathbf{O} on the $\lambda\Delta$ -calculus with respect to Θ .

Evaluation relations are denoted by bold capital letters.

Example 5.0.5. It is easy to see that the following evaluation relations are well defined.

1. Let $\mathbf{N}_{\text{nd}} \in \mathcal{E}(\Lambda, \Lambda\text{-NF})$ be $\{(M, N) \in \Lambda \times \Lambda\text{-NF} \mid M \rightarrow_{\Lambda}^{*} N\}$.
2. Let $\mathbf{N} \in \mathcal{E}(\Lambda, \Lambda\text{-NF})$ be $\{(M, N) \in \Lambda \times \Lambda\text{-NF} \mid M \rightarrow_{\Lambda}^{*p} N\}$.
3. Let $\mathbf{H}_{\text{nd}} \in \mathcal{E}(\Lambda, \Lambda\text{-HNF})$ be $\{(M, N) \in \Lambda \times \Lambda\text{-HNF} \mid M \rightarrow_{\Lambda}^{*} N\}$.
4. Let $\mathbf{H} \in \mathcal{E}(\Lambda, \Lambda\text{-HNF})$ be $\{(M, N) \in \Lambda \times \Lambda\text{-HNF} \mid M \rightarrow_{\Lambda}^{p} M_1 \rightarrow_{\Lambda}^{p} \dots \rightarrow_{\Lambda}^{p} M_r \rightarrow_{\Lambda}^{p} N, \text{ and } M_i \notin \Lambda\text{-HNF} (1 \leq i \leq r)\}$.
5. Let $\mathbf{L}_{\text{nd}} \in \mathcal{E}(\Lambda, \Lambda\text{-LHNF})$ be $\{(M, N) \in \Lambda \times \Lambda\text{-LHNF} \mid M \rightarrow_{\Lambda}^{*} N\}$.
6. Let $\mathbf{L} \in \mathcal{E}(\Lambda, \Lambda\text{-LHNF})$ be $\{(M, N) \in \Lambda \times \Lambda\text{-LHNF} \mid M \rightarrow_{\Lambda}^{p} M_1 \rightarrow_{\Lambda}^{p} \dots \rightarrow_{\Lambda}^{p} M_r \rightarrow_{\Lambda}^{p} N, \text{ and } M_i \notin \Lambda\text{-LHNF} (1 \leq i \leq r)\}$.
7. Let $\mathbf{G}_{\text{nd}} \in \mathcal{E}(\Gamma, \Gamma\text{-NF})$ be $\{(M, N) \in \Lambda \times \Gamma\text{-NF} \mid M \rightarrow_{\Gamma}^{*} N\}$.
8. Let $\mathbf{G} \in \mathcal{E}(\Gamma, \Gamma\text{-NF})$ be $\{(M, N) \in \Lambda \times \Gamma\text{-NF} \mid M \rightarrow_{\Gamma}^{*p} N\}$.
9. Let $\mathbf{V}_{\text{nd}} \in \mathcal{E}(\Gamma, \Gamma\text{-LBNF})$ be $\{(M, N) \in \Lambda \times \Gamma\text{-LBNF} \mid M \rightarrow_{\Gamma}^{*} N\}$.
10. Let $\mathbf{V} \in \mathcal{E}(\Gamma, \Gamma\text{-LBNF})$ be $\{(M, N) \in \Lambda \times \Gamma\text{-LBNF} \mid M \rightarrow_{\Gamma}^{p} M_1 \rightarrow_{\Gamma}^{p} \dots \rightarrow_{\Gamma}^{p} M_r \rightarrow_{\Gamma}^{p} N, \text{ and } M_i \notin \Gamma\text{-LBNF} (1 \leq i \leq r)\}$.

An evaluation relation can be presented by using a formal system.

A *logical rule*, or briefly rule, has the following shape:

$$\frac{\mathfrak{P}_1 \dots \mathfrak{P}_m}{\mathfrak{C}} \text{ name}$$

where the *premises* \mathfrak{P}_i ($1 \leq i \leq m$) and the *conclusion* \mathfrak{C} are logical judgments (written using metavariables); while *name* is the name of the rule.

The intended meaning of a rule is that, for every instance s of the metavariables in the rule, $s(\mathfrak{C})$ is implied by the logical AND of $s(\mathfrak{P}_i)$ ($1 \leq i \leq m$).

For sake of simplicity, we will use the syntax of terms for denoting the metaterms in the logical rules.

A *derivation* is a finite tree of logical rules, such that each leaf is an axiom, each intermediate node has as premises the consequences of its son nodes and its consequence is one of the premises of its father node. The conclusion of the root node is the proved judgment.

The *size* of a derivation is the number of nodes in it.

A formal system defining an evaluation relation $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ is a set of logical rules for establishing judgments of the shape $M \Downarrow_{\mathbf{O}} N$, whose meaning is $(M, N) \in \mathbf{O}$. We will denote with $M \Downarrow_{\mathbf{O}}$ the fact that the judgment $M \Downarrow_{\mathbf{O}} N$

can be proved in the system for some N , i.e. $(M, N) \in \mathbf{O}$. We will denote with $M \uparrow_{\mathbf{O}}$ the fact that there is no $N \in \Theta$ such that $M \Downarrow_{\mathbf{O}} N$.

The evaluation relation $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ is *deterministic* if, in case $M \Downarrow_{\mathbf{O}}$, there is a unique term N such that $M \Downarrow_{\mathbf{O}} N$ (i.e. the evaluation relation is a partial function). All the evaluation relations we show in this part of the book are deterministic.

Example 5.0.6. The evaluation relations \mathbf{N}_{nd} and \mathbf{N} , defined in Example 5.0.5, coincide and are both deterministic by Corollaries 1.2.6 and 1.2.13. The same is true for the evaluation relations \mathbf{G}_{nd} and \mathbf{G} .

The evaluation relations \mathbf{H} , \mathbf{L} and \mathbf{V} are deterministic, while \mathbf{H}_{nd} , \mathbf{L}_{nd} and \mathbf{V}_{nd} are not deterministic. For example, both $(\lambda x.x(II), \lambda x.x(II)) \in \mathbf{H}_{\text{nd}}$ and $(\lambda x.x(II), \lambda x.xI) \in \mathbf{H}_{\text{nd}}$; the same two pairs of terms are in \mathbf{L}_{nd} and \mathbf{V}_{nd} .

A formal system establishing judgments of the shape $M \Downarrow_{\mathbf{O}} N$ can be viewed as a logical representation of a *reduction machine*. In particular the evaluation process of the machine is simulated by a derivation in the logical system. In the terminology of reduction machines, $M \Downarrow_{\mathbf{O}} N$ means that “on input M , the reduction machine \mathbf{O} stops and gives as output N ”; $M \Downarrow_{\mathbf{O}}$ means that “on input M , the reduction machine \mathbf{O} stops”; while $M \uparrow_{\mathbf{O}}$ means “on input M , the reduction machine never stops”.

In the rest of the book, we will use the metavariable \mathbf{O} to denote an evaluation relation actually defined by a formal system.

Definition 5.0.7. An evaluation relation $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ induces naturally an operational semantics, i.e. a preorder relation on terms denoted by $\preceq_{\mathbf{O}}$. The operational preorder induced by \mathbf{O} is defined as:

$$\begin{gathered} M \preceq_{\mathbf{O}} N \\ \text{if and only if} \\ \forall C[\cdot] \text{ such that } C[M], C[N] \in \Lambda^0 \quad (C[M] \Downarrow_{\mathbf{O}} \text{ implies } C[N] \Downarrow_{\mathbf{O}}). \end{gathered}$$

$\prec_{\mathbf{O}}$ denotes the strict version of $\preceq_{\mathbf{O}}$, while $\approx_{\mathbf{O}}$ is the equivalence relation on terms induced by $\preceq_{\mathbf{O}}$.

If $M \approx_{\mathbf{O}} N$ then M and N are \mathbf{O} -operationally equivalent.

This operational equivalence amounts to the Leibniz Equality Principle for programs, i.e. a criterion for establishing equivalence on the basis of the behaviour of programs regarded as black boxes. It is natural to model a program by a closed term. So a context can be viewed as a *partially specified* program, where every occurrence of the hole denotes a place that must be filled by a subprogram, while a generic term can be viewed as a subprogram. So two terms are equivalent if they can be replaced by each other in the same program without changing its behaviour (with respect to an evaluation relation \mathbf{O}).

Since we are considering “pure” calculi, i.e. calculi without constants, the only behaviour we can observe on terms is the termination, and this justifies the previous definition of operational semantics.

In the presence of constants, a subset of them (the “basic constants”) will be the possible results of a computation, and the definition would change in the following way:

$$\begin{aligned} M \preceq N \\ \text{if and only if} \\ \forall C[\cdot] \text{ such that } C[M], C[N] \in \Lambda^0, \text{ for all basic constants } a, \\ (C[M] \Downarrow a \text{ implies } C[N] \Downarrow a). \end{aligned}$$

Definition 5.0.8. Let $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ be an evaluation relation.

- (i) The $\lambda\Delta$ -calculus is correct with respect to the \mathbf{O} -operational semantics if and only if $M =_{\Delta} N$ implies $M \approx_{\mathbf{O}} N$.
- (ii) The $\lambda\Delta$ -calculus is complete with respect to the \mathbf{O} -operational semantics if and only if $M \approx_{\mathbf{O}} N$ implies $M =_{\Delta} N$.

The $\lambda\Delta$ -calculus is correct with respect to the \mathbf{O} -operational semantics if and only if $\approx_{\mathbf{O}}$ is a Δ -theory. In fact, it is easy to check that $\approx_{\mathbf{O}}$ is always a congruence relation (see Definition 1.3.1).

Example 5.0.9. The evaluation relations of Example 5.0.5 are correct with respect to their respective set of input values. Some counterexamples to the correctness follow:

- (i) Let $\mathbf{J} \in \mathcal{E}(\Lambda, \Lambda\text{-NF})$ be $\{(M, N) \in \Lambda \times \Lambda\text{-NF} \mid M \rightarrow_{\Lambda}^{*i} N\}$.
The $\lambda\Lambda$ -calculus is not correct with respect to \mathbf{J} . In fact, $KI(DD) =_{\Lambda} I$ but $KI(DD) \not\approx_{\mathbf{J}} I$, since $KI(DD) \uparrow_{\mathbf{J}}$ while $I \Downarrow_{\mathbf{J}}$.
- (ii) Let $\mathbf{W} \in \mathcal{E}(\Gamma, \Gamma\text{-LBNF})$ be $\{(M, N) \in \Gamma \times \Gamma\text{-LBNF} \mid M \rightarrow_{\Gamma}^{*i} N\}$. The $\lambda\Gamma$ -calculus is not correct with respect to \mathbf{W} . In fact, $KI(\lambda x.DD) =_{\Gamma} I$ but $KI(\lambda x.DD) \not\approx_{\mathbf{W}} I$, since $KI(\lambda x.DD) \uparrow_{\mathbf{W}}$ while $I \Downarrow_{\mathbf{W}}$.

The notion of \mathbf{O} -relevant context, introduced in the next definition, is a technical tool that is useful for proving operational equivalences.

Definition 5.0.10. Let $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$.

- (i) A context $C[\cdot]$ is \mathbf{O} -relevant if and only if there are $M, N \in \Lambda^0$ such that $C[M] \Downarrow_{\mathbf{O}}$ and $C[N] \uparrow_{\mathbf{O}}$.
- (ii) Let $M, N \in \Lambda$. A context $C[\cdot]$ is said a discriminating context for M and N if and only if $C[M] \Downarrow_{\mathbf{O}}$ and $C[N] \uparrow_{\mathbf{O}}$, or vice versa.

This notion of relevance is inspired by the corresponding one of relevant context, introduced by Plotkin in order to study the operational behaviour of the paradigmatic programming language PCF (see [79]).

5.1 The Universal Δ -Reduction Machine

The fact that the set of output values satisfies the principality condition allows us to define a universal evaluation relation, parametric both in the set of input and output values, from which many interesting evaluation relations can be derived by suitable instantiations. Such an evaluation relation is based on a formal system, defining the principal evaluation of a term of the $\lambda\Delta$ -calculus.

Definition 5.1.1. *A formal system proving statements of the kind*

$$M \rightarrow_{\Delta}^p N$$

where $M, N \in \Lambda$, is formalized in Fig. 5.1.

The notation $M \rightarrow_{\Delta}^p N$ is defined in Definition 1.2.7 (N is obtained from M by reducing its principal redex).

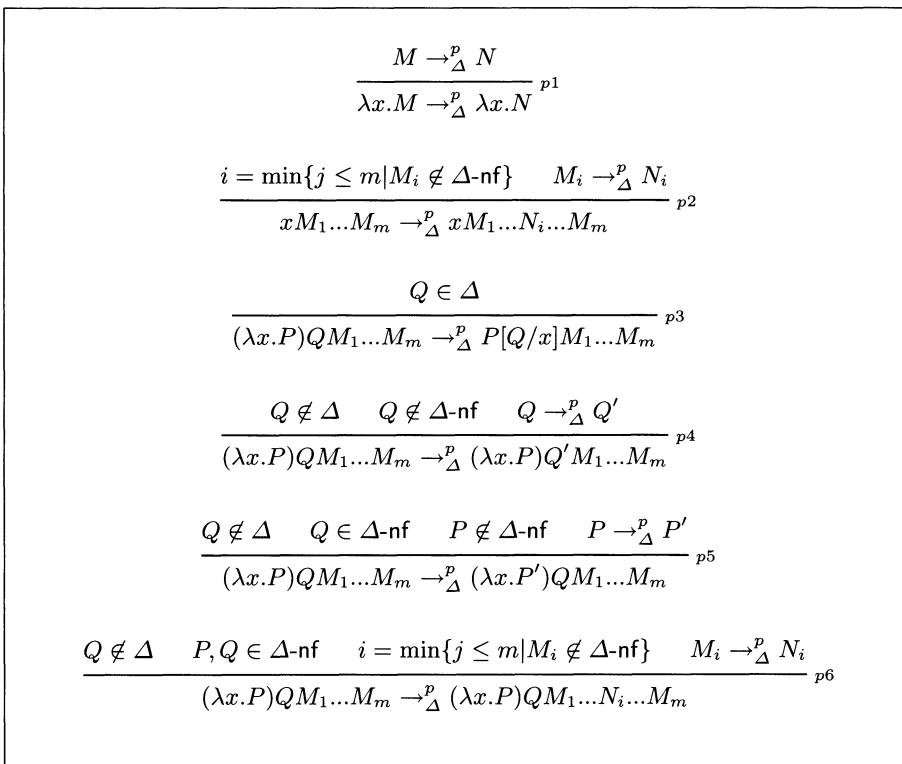


Fig. 5.1. Principal reduction machine

The machine described in Fig. 5.1 is “step-by-step”, since each of its rules describes just one application of the reduction rule.

Definition 5.1.2 (Universal evaluation relation).

Let Θ be a set of output values with respect to the set of input value Δ .

- (i) $\mathcal{U}_\Theta^\Delta \in \mathcal{E}(\Delta, \Theta)$ is the evaluation relation defined through the following rules:

$$\frac{M \in \Theta}{M \Downarrow_{\mathcal{U}_\Theta^\Delta} M} \text{ (axiom)} \qquad \frac{M \rightarrow_\Delta^p P \quad P \Downarrow_{\mathcal{U}_\Theta^\Delta} N}{M \Downarrow_{\mathcal{U}_\Theta^\Delta} N} \text{ (eval)}$$

- (ii) $M \preceq_{\mathcal{U}_\Theta^\Delta} N$ if and only if, for all contexts $C[.]$ such that $C[M], C[N] \in \Lambda^0$,
 $(C[M] \Downarrow_{\mathcal{U}_\Theta^\Delta} C[N] \Downarrow_{\mathcal{U}_\Theta^\Delta})$.
- (iii) $M \approx_{\mathcal{U}_\Theta^\Delta} N$ if and only if $M \preceq_{\mathcal{U}_\Theta^\Delta} N$ and $N \preceq_{\mathcal{U}_\Theta^\Delta} M$.

It is easy to check that the previous definition is well posed, i.e. $M \Downarrow_{\mathcal{U}_\Theta^\Delta} N$ implies $M \rightarrow_\Delta^* N$. Furthermore, the evaluation relation $\mathcal{U}_\Theta^\Delta$ is deterministic for all Δ, Θ .

Theorem 5.1.3 proves that the evaluation relation $\mathcal{U}_\Theta^\Delta$ is universal, in the sense that it subsumes all deterministic evaluation relations obtained by instantiating Δ and Θ in a correct way.

Theorem 5.1.3. If $M \rightarrow_\Delta^* N \in \Theta$ then $M \Downarrow_{\mathcal{U}_\Theta^\Delta} N$.

Proof. Since Θ satisfies the principality condition, $M \rightarrow_\Delta^* N \in \Theta$ implies there is $N' \in \Theta$ such that $M \rightarrow_\Delta^{*p} N'$. Then the proof follows by induction on the length of the reduction sequence $M \rightarrow_\Delta^{*p} N' \in \Theta$. If $M \in \Theta$, then the proof follows by rule (axiom) of the formal system defining $\mathcal{U}_\Theta^\Delta$. Otherwise, $M \rightarrow_\Delta^{*p} N'$ means $M \rightarrow_\Delta^p N'' \rightarrow_\Delta^{*p} N'$, so the proof follows by induction. \square

For each choice of the sets of the input and output values, the $\lambda\Delta$ -calculus is *correct* with respect to the $\mathcal{U}_\Theta^\Delta$ operational semantics, as proved in Theorem 5.1.4.

Theorem 5.1.4 ($\mathcal{U}_\Theta^\Delta$ -Correctness).

The $\lambda\Delta$ -calculus is correct with respect to the $\mathcal{U}_\Theta^\Delta$ -operational semantics.

Proof. $M =_\Delta N$ implies $C[M] =_\Delta C[N]$, for all contexts $C[.]$.

If there is $P \in \Theta$ such that $C[M] \rightarrow_\Delta^* P$, then $C[M] \Downarrow_{\mathcal{U}_\Theta^\Delta}$, by Theorem 5.1.3. Clearly $P =_\Delta C[N]$; thus, by principality, there is $P' \in \Theta$ such that $C[N] \rightarrow_\Delta^{*p} P'$, so $C[N] \Downarrow_{\mathcal{U}_\Theta^\Delta}$.

In case there is not such a P , both $C[M] \uparrow_{\mathcal{U}_\Theta^\Delta}$ and $C[N] \uparrow_{\mathcal{U}_\Theta^\Delta}$. \square

So, $\mathcal{U}_\Theta^\Delta$ -operational semantics induce a Δ -theory; as far as completeness is concerned, it depends on the choice of the set of output values. But all operational semantics of interest are not complete, as we will see in the following.

Example 5.1.5. Let $\mathcal{U}_{\Lambda\text{-HNF}}^{\Lambda}$ be the universal evaluation relation, where Λ is the set of input values and $\Lambda\text{-HNF}$ is the set of output values.

- (i) Let $M_0 \equiv \lambda x.(\lambda uv.xuv)I(DD)$, $M_1 \equiv \lambda x.(\lambda v.xIv)(DD)$ and $M_2 \equiv \lambda x.xI(DD)$; note that $M_0 \rightarrow_{\Lambda}^p M_1 \rightarrow_{\Lambda}^p M_2 \in \Lambda\text{-HNF}$.
 $\lambda x.(\lambda uv.xuv)I(DD) \Downarrow_{\mathcal{U}_{\Lambda\text{-HNF}}^{\Lambda}} \lambda x.xI(DD)$; in fact, we can build the following derivation:

$$\frac{\frac{\frac{M_1 \rightarrow_{\Lambda}^p M_2}{M_2 \Downarrow_{\mathcal{U}_{\Lambda\text{-HNF}}^{\Lambda}} M_2} \text{(axiom)}}{M_1 \Downarrow_{\mathcal{U}_{\Lambda\text{-HNF}}^{\Lambda}} M_2} \text{(eval)}}{M_0 \rightarrow_{\Lambda}^p M_1} \text{(eval)}$$

$$\lambda x.(\lambda uv.xuv)I(DD) \Downarrow_{\mathcal{U}_{\Lambda\text{-HNF}}^{\Lambda}} \lambda x.xI(DD)$$

- (ii) It is possible to check that there is not a derivation proving $\lambda x.DD \Downarrow_{\mathcal{U}_{\Lambda\text{-HNF}}^{\Lambda}}$, i.e. $\lambda x.DD \uparrow_{\mathcal{U}_{\Lambda\text{-HNF}}^{\Lambda}}$. Every derivation proving $\lambda x.DD \Downarrow_{\mathcal{U}_{\Lambda\text{-HNF}}^{\Lambda}}$ must be of the following shape:

$$\frac{\frac{\lambda x.DD \rightarrow_{\Lambda}^p \lambda x.DD \quad d}{\lambda x.DD \Downarrow_{\mathcal{U}_{\Lambda\text{-HNF}}^{\Lambda}} R} \text{(eval)}}{\lambda x.DD \Downarrow_{\mathcal{U}_{\Lambda\text{-HNF}}^{\Lambda}} R} \text{(eval)}$$

for some $R \in \Lambda$ and some derivation d proving $\lambda x.DD \Downarrow_{\mathcal{U}_{\Lambda\text{-HNF}}^{\Lambda}} R$. Since all derivations are applications of a finite number of rules, d cannot exist, and so also the whole derivation.

In the remainder of this part of the book we will present four different operational semantics: three for the call-by-name λ -calculus and one for the call-by-value calculus. They formalize the deterministic evaluation relations given in Example 5.0.5, except for **G**. We will not develop such a semantics, since the notion of Γ -normal form is semantically meaningless, as already noted.

Each one of the operational semantics we are interested in can be derived from the “universal Δ -reduction machine” by instantiating the sets of input and output values in a suitable way. But we choose to present the various operational semantics independently, both for clarity and for technical reasons. In fact, while the universal reduction machine is based on a step-by-step description of the evaluation relation, the reduction machines we will present supply an input-output description of it, and this makes the proofs easier.

6. Call-by-Name Operational Semantics

6.1 H-Operational Semantics

$\mathbf{H} \in \mathcal{E}(\Lambda, \Lambda\text{-HNF})$ is the first evaluation relation that we will study; it is the universal evaluation relation $\mathcal{U}_{\Lambda\text{-HNF}}^{\Lambda}$ (see Example 5.1.5).

In this setting, the converging terms represent computations that can always produce a given output value when applied to suitable arguments. In fact, the set of terms having Λ -HNF coincides with the set of Λ -solvable terms.

Definition 6.1.1 (H-Operational semantics).

- (i) $\mathbf{H} \in \mathcal{E}(\Lambda, \Lambda\text{-HNF})$ is the evaluation relation induced by the formal system proving judgments of the shape

$$M \Downarrow_{\mathbf{H}} N$$

where $M \in \Lambda$ and $N \in \Lambda\text{-HNF}$. It consists of the following rules:

$$\frac{m \geq 0}{xM_1 \dots M_m \Downarrow_{\mathbf{H}} xM_1 \dots M_m} \text{ (var)}$$

$$\frac{M \Downarrow_{\mathbf{H}} N}{\lambda x.M \Downarrow_{\mathbf{H}} \lambda x.N} \text{ (abs)}$$

$$\frac{P[Q/x]M_1 \dots M_m \Downarrow_{\mathbf{H}} N}{(\lambda x.P)QM_1 \dots M_m \Downarrow_{\mathbf{H}} N} \text{ (head)}$$

- (ii) $M \preceq_{\mathbf{H}} N$ if and only if, for all contexts $C[.]$ such that $C[M], C[N] \in \Lambda^0$, $(C[M] \Downarrow_{\mathbf{H}} \text{ implies } C[N] \Downarrow_{\mathbf{H}})$.
- (iii) $M \approx_{\mathbf{H}} N$ if and only if $M \preceq_{\mathbf{H}} N$ and $N \preceq_{\mathbf{H}} M$.

As we already noticed, \mathbf{H} is deterministic.

Example 6.1.2. (i) $\lambda x.(\lambda u v. x u v) I(DD) \Downarrow_{\mathbf{H}} \lambda x. x I(DD)$. In fact, we can build the following derivation:

$$\frac{\frac{\frac{xI(DD) \Downarrow_{\mathbf{H}} xI(DD)}{(var)}}{(\lambda v. xIv)(DD) \Downarrow_{\mathbf{H}} xI(DD)}{(head)}}{(\lambda u v. x u v) I(DD) \Downarrow_{\mathbf{H}} xI(DD)}{(head)} \\ \frac{}{\lambda x.(\lambda u v. x u v) I(DD) \Downarrow_{\mathbf{H}} \lambda x. x I(DD)}{(abs)}$$

where the unique leaf is the axiom (var) and the conclusion of the root node is the judgment $\lambda x.(\lambda u v. x u v) I(DD) \Downarrow_{\mathbf{H}} \lambda x. x I(DD)$. Note that, in the particular case of the system $\Downarrow_{\mathbf{H}}$, every derivation is such that each node has a unique son.

- (ii) It is possible to check that there is no derivation proving $\lambda x. DD \Downarrow_{\mathbf{H}}$. In fact, if a such derivation exists then it must be of the following shape:

$$\frac{d}{\frac{DD \Downarrow_{\mathbf{H}} R}{\lambda x. DD \Downarrow_{\mathbf{H}} \lambda x. R}}{(head)}{(abs)}$$

for some R , and some derivation d . But the rule (head) implies that the derivation d must be in its turn of the shape

$$\frac{d}{DD \Downarrow_{\mathbf{H}} R} \quad (head)$$

Since all derivations are the application of a finite number of rules, d cannot exist, and so also the whole derivation.

The system $\Downarrow_{\mathbf{H}}$ characterizes completely the class of terms having Λ -head normal forms, as shown in Theorem 6.1.3.

Theorem 6.1.3. (i) $M \Downarrow_{\mathbf{H}} N$ implies $M \rightarrow_{\Lambda}^{*p} N$ and N is in Λ -hnf.
(ii) $M \Downarrow_{\mathbf{H}}$ if and only if M has a Λ -hnf.

Proof. (i) By induction on the definition of $\Downarrow_{\mathbf{H}}$.

(ii) (\Rightarrow) The proof is a consequence of (i).

(\Leftarrow) M has Λ -hnf means that there is $N \in \Lambda$ -HNF such that $M =_{\Lambda} N$. But Λ -HNF is a set of output values with respect to Λ , by Property 5.0.3; so there is a reduction sequence $M \rightarrow_{\Lambda}^{*p} M' \in \Lambda$ -HNF.

The proof is done by induction on the length of the reduction sequence $M \rightarrow_{\Lambda}^{*p} M'$. Let $M \equiv \lambda x_1 \dots x_n. \zeta M_1 \dots M_m$ ($n, m \in \mathbb{N}$).

If ζ is a variable then M is already in Λ -hnf. In fact $M \Downarrow_{\mathbf{H}} M$, by n applications of rule (abs) and one application of the rule (var).

If $\zeta \equiv (\lambda x.P)Q$ then by induction, $P[Q/x]M_1\dots M_m \Downarrow_{\mathbf{H}} N$, for some N ; thus $M \Downarrow_{\mathbf{H}} \lambda x_1\dots x_n.N$, by n applications of rule (abs) and one application of the rule (head). \square

The following property will be quite useful in the Sect. 15.2.

Property 6.1.4. Let $M, N, T, U \in \Lambda$ and $M \Downarrow_{\mathbf{H}} N$.

- (i) $M[T/z] \Downarrow_{\mathbf{H}} U$ if and only if $N[T/z] \Downarrow_{\mathbf{H}} U$.
- (ii) $MT \Downarrow_{\mathbf{H}} U$ if and only if $NT \Downarrow_{\mathbf{H}} U$.

Proof. (i) Clearly $M[T/z] =_{\Lambda} N[T/z]$, so by the confluence theorem $M[T/z]$ has Λ -hnf if and only if $N[T/z]$ has Λ -hnf; hence let $M[T/z] \Downarrow_{\mathbf{H}} U_0$ and $N[T/z] \Downarrow_{\mathbf{H}} U_1$, for some $U_0, U_1 \in \Lambda$.

We show that $U_0 \equiv U_1$ by induction on the derivation of $M \Downarrow_{\mathbf{H}} N$. The case (var) is trivial. The case (abs) follows by induction. The more complex case is (head); if $P[Q/x]M_1\dots M_m \Downarrow_{\mathbf{O}} N$ then by induction, $(P[Q/x])[T/z]M_1[T/z]\dots M_m[T/z] \Downarrow_{\mathbf{H}} V$ and $N[T/z] \Downarrow_{\mathbf{H}} V$, thus $(\lambda x.P[T/z])Q[T/z]M_1[T/z]\dots M_m[T/z] \Downarrow_{\mathbf{H}} V$ too and the proof is done.

- (ii) Since $M =_{\Lambda} N$ implies $MT =_{\Lambda} NT$, we can assume $MT \Downarrow_{\mathbf{H}} U_0$ if and only if $NT \Downarrow_{\mathbf{H}} U_1$. We show that $U_0 = U_1$ by induction on the derivation of $M \Downarrow_{\mathbf{H}} N$. The case (var) is trivial. The case (abs) follows by using the previous point. The case (head) follows by induction. \square

$\approx_{\mathbf{H}}$ is a Λ -theory, as proved by Theorem 6.1.5.

Theorem 6.1.5 (H-Correctness).

The $\lambda\Lambda$ -calculus is correct with respect to the \mathbf{H} -operational semantics.

Proof. We must prove that $M =_{\Lambda} N$ implies $M \approx_{\mathbf{H}} N$, by definition of correctness. Let $M =_{\Lambda} N$ and let $C[.]$ be a context such that $C[M], C[N] \in \Lambda^0$. By definition of $=_{\Lambda}$, $C[M] =_{\Lambda} C[N]$. So the proof follows from Theorem 6.1.3.(ii), since by the confluence theorem the property of having Λ -hnf is closed under $=_{\Lambda}$. \square

The $\lambda\Lambda$ -calculus, nevertheless, is not complete with respect to the \mathbf{H} -operational semantics. To show the incompleteness, the notion of \mathbf{H} -relevant context is used, which is the specialization to \mathbf{H} of the general notion presented in Definition 5.0.10. Lemma 6.1.6 shows a syntactical characterization of \mathbf{H} -relevant context.

Lemma 6.1.6 (H-Relevance).

A context $C[.]$ is \mathbf{H} -relevant whenever there is a context $C'[.] \equiv [.]C_1[.] \dots C_m[.]$ ($m \in \mathbb{N}$) such that for all $M \in \Lambda^0$, $C[M] \Downarrow_{\mathbf{H}}$ if and only if $C'[M] \Downarrow_{\mathbf{H}}$.

Proof. (\Rightarrow) Assume that $C[.]$ is \mathbf{H} -relevant, namely there are $M, N \in \Lambda^0$ such that $C[M] \Downarrow_{\mathbf{H}}$ and $C[N] \uparrow_{\mathbf{H}}$. By induction on $C[M] \Downarrow_{\mathbf{H}}$ we will prove that there is a context $C'[.]$ satisfying the statement.

If the last applied rule is (var) then either $C[.] \equiv xC_1[.]...C_m[.]$ ($m \in \mathbb{N}$) or $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$). In the first case the context is not relevant, while the second case is not possible, since $M \in \Lambda^0$.

If the last applied rule is (abs) then either $C[.] \equiv [.]$ or $C[.] \equiv \lambda z.C''[.]$. The first case is immediate, while the second follows by induction.

If the last applied rule is (head) then either $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$) or $C[.] \equiv (\lambda z.C_0[.])C_1[.]...C_m[.]$ ($m \geq 1$). The first case is trivial, while the second follows by induction; in fact, the context $C_0[.][C_1[.]/z]C_2[.]...C_m[.]$ is discriminating M and N and so is \mathbf{H} -relevant too.

(\Leftarrow) Let $C'[.]$ be a context satisfying the statement of this Lemma, so $C'[M] \Downarrow_{\mathbf{H}}$ if and only if $C[M] \Downarrow_{\mathbf{H}}$, for each $M \in \Lambda$. Thus $M \equiv DD$ and $N \equiv \lambda x_1...x_m z. z$ are witnesses of the \mathbf{H} -relevance of $C[.]$. \square

By observing the details of the proof, it is easy to see that actually, for all $M \in \Lambda^0$, if $C[M] \Downarrow_{\mathbf{H}}$ then in the derivation of $C[M] \Downarrow_{\mathbf{H}}$ there are contexts $C_1[.], ..., C_m[.]$ ($m \in \mathbb{N}$) and there is a subderivation proving $MC_1[M]...C_m[M] \Downarrow_{\mathbf{H}}$.

Lemma 6.1.7. *Let $C[.]$ be \mathbf{H} -relevant. If $M \in \Lambda^0$ and $C[M] \Downarrow_{\mathbf{H}}$ then $M \Downarrow_{\mathbf{H}}$.*

Proof. By induction on the derivation of $C[M] \Downarrow_{\mathbf{H}}$.

If the last applied rule is (var) then either $C[.] \equiv xC_1[.]...C_m[.]$ ($m \in \mathbb{N}$) or $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$). In the first case the context is not relevant, while the second case is not possible, since $M \in \Lambda^0$.

If the last applied rule is (abs) then either $C[.] \equiv [.]$ or $C[.] \equiv \lambda z.C''[.]$. The first case is immediate, while the second follows by induction.

If the last applied rule is (head) then either $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$) or $C[.] \equiv (\lambda z.C_0[.])C_1[.]...C_m[.]$ ($m \geq 1$). The first case follows by Property 2.1.9.(ii), while the second follows by induction on the derivation proving $C_0[M][C_1[M]/z]C_2[M]...C_m[M]$. \square

We can check that all closed Λ -unsolvable terms are equated in the \mathbf{H} -operational semantics.

Theorem 6.1.8 (H-Incompleteness).

The λ -calculus is incomplete with respect to the \mathbf{H} -operational semantics.

Proof. Let P and Q be two closed Λ -unsolvable terms such that $M \neq_{\Lambda} N$. A non- \mathbf{H} -relevant context cannot discriminate P and Q . Let $C[.]$ be a \mathbf{H} -relevant context: P, Q have not Λ -hnf (by Theorem 2.1.1), thus both $P \uparrow_{\mathbf{H}}$ and $Q \uparrow_{\mathbf{H}}$. Thus $C[P] \uparrow_{\mathbf{H}}$ and $C[Q] \uparrow_{\mathbf{H}}$ by Lemma 6.1.7. Hence, $P \approx_{\mathbf{H}} Q$. \square

The proof of the following property is an example of a useful technique for proving operational equality between terms.

Property 6.1.9. $I \approx_{\mathbf{H}} E$.

Proof. By absurdum assume that the two terms can be discriminated. This means that there is a context $C[.]$ discriminating them. Let $C[.]$ be such that $C[I] \Downarrow_{\mathbf{H}}$ while $C[E] \Uparrow_{\mathbf{H}}$. Clearly $C[.]$ must be \mathbf{H} -relevant.

Let $C[.]$ be a minimal discriminating context for I and E , in the sense that the derivation of $C[I] \Downarrow_{\mathbf{H}}$ has a minimal size between all the proofs of $C'[I] \Downarrow_{\mathbf{H}}$, for every $C'[.]$ discriminating between I and E in such a way that $C'[I] \Downarrow_{\mathbf{H}}$ while $C'[E] \Uparrow_{\mathbf{H}}$. The proof is done by considering the last applied rule in the derivation proving $C[I] \Downarrow_{\mathbf{H}}$.

The last used rule cannot be (var), since $C[.] \equiv xC_1[.]...C_m[.]$ ($m \in \mathbb{N}$) is not \mathbf{H} -relevant. If the last used rule is (abs) then either $C[.] \equiv \lambda x.C'[.]$ or $C[.] \equiv [.]$. In the former case, $C'[.]$ would be a discriminating context such that the derivation of $C'[I] \Downarrow_{\mathbf{H}}$ has smaller size than the derivation of $C[I] \Downarrow_{\mathbf{H}}$, against the hypothesis. The latter case is not possible, since clearly $[.]$ is not a discriminating context for I and E .

Let the last used rule be (head); thus either $C[.] \equiv (\lambda x.C_0[.])C_1[.]...C_m[.]$ ($m \geq 1$) or $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$). Let consider the former case. By the rule (head), $C[I] \Downarrow_{\mathbf{H}}$ if and only if $C_0[I][C_1[I]/x]C_2[I]...C_m[I] \Downarrow_{\mathbf{H}}$. But in this case $C_0[.][C_1[.]/x]C_2[.].C_m[.]$ would be a discriminating context for M and N with a derivation having smaller size than $C[.]$, against the hypothesis that $C[.]$ is minimum.

The case $C[.] \equiv [.]C_1[.]...C_m[.]$ leads to a similar contradiction. In fact, in this case $C_1[.]...C_m[.]$ would be a discriminating context “smaller” than $C[.]$.

The case when $C[I] \Uparrow_{\mathbf{H}}$ and $C[E] \Downarrow_{\mathbf{H}}$ is symmetric. \square

Theorem 6.1.10. *The theory \mathbf{H} is fully extensional.*

Proof. By Properties 2.1.7 and 6.1.9. \square

6.2 N-Operational Semantics

$\mathbf{N} \in \mathcal{E}(\Lambda, \Lambda\text{-NF})$ is the evaluation relation studied in this section; it is the universal evaluation relation $\mathcal{U}_{\Lambda\text{-NF}}^{\Lambda}$.

In some sense, \mathbf{N} induces the most natural operational semantics for the $\lambda\Lambda$ -calculus; in fact, converging terms represent the completely terminating computations.

Definition 6.2.1 (N-Operational semantics).

- (i) $\mathbf{N} \in \mathcal{E}(\Lambda, \Lambda\text{-NF})$ is the evaluation relation induced by the formal system proving judgments of the shape

$$M \Downarrow_{\mathbf{N}} N$$

where $M \in \Lambda$ and $N \in \Lambda\text{-NF}$. It consists of the following rules:

$$\frac{(M_i \Downarrow_{\mathbf{N}} N_i)_{(i \leq m)}}{xM_1 \dots M_m \Downarrow_{\mathbf{N}} xN_1 \dots N_m} \text{ (var)}$$

$$\frac{M \Downarrow_{\mathbf{N}} N}{\lambda x. M \Downarrow_{\mathbf{N}} \lambda x. N} \text{ (abs)}$$

$$\frac{P[Q/x]M_1 \dots M_m \Downarrow_{\mathbf{N}} N}{(\lambda x. P)QM_1 \dots M_m \Downarrow_{\mathbf{N}} N} \text{ (head)}$$

(ii) $M \preceq_{\mathbf{N}} N$ if and only if, for all contexts $C[.]$ such that $C[M], C[N] \in \Lambda^0$,
 $(C[M] \Downarrow_{\mathbf{N}} \text{ implies } C[N] \Downarrow_{\mathbf{N}})$.

(iii) $M \approx_{\mathbf{N}} N$ if and only if $M \preceq_{\mathbf{N}} N$ and $N \preceq_{\mathbf{N}} M$.

As is true for \mathbf{H} , the relation \mathbf{N} is also deterministic.

Example 6.2.2. $\lambda x_1 x_2. x_1(ID)((\lambda u v. u)(II)x_2) \Downarrow_{\mathbf{N}} \lambda x_1 x_2. x_1 DI$, as shown by the following derivation.

$$\begin{array}{c} \frac{}{x \Downarrow_{\mathbf{N}} x} \text{ (var)} \quad \frac{}{x \Downarrow_{\mathbf{N}} x} \text{ (var)} \\ \hline \frac{x x \Downarrow_{\mathbf{N}} x x}{\lambda x. x x \Downarrow_{\mathbf{N}} \lambda x. x x} \text{ (abs)} \quad \frac{x \Downarrow_{\mathbf{N}} x}{\lambda x. x \Downarrow_{\mathbf{N}} \lambda x. x} \text{ (var)} \\ \hline \frac{\lambda x. x x \Downarrow_{\mathbf{N}} \lambda x. x x}{ID \Downarrow_{\mathbf{N}} D} \text{ (head)} \quad \frac{\lambda x. x \Downarrow_{\mathbf{N}} \lambda x. x}{II \Downarrow_{\mathbf{N}} I} \text{ (head)} \\ \hline \frac{x_1(ID)((\lambda u v. u)(II)) \Downarrow_{\mathbf{N}} x_1 D(\lambda v. I)}{\lambda x_2. x_1(ID)((\lambda u v. u)(II)) \Downarrow_{\mathbf{N}} \lambda x_2. x_1 D(\lambda v. I)} \text{ (var)} \\ \hline \frac{x_1(ID)((\lambda u v. u)(II)) \Downarrow_{\mathbf{N}} x_1 D(\lambda v. I)}{\lambda x_1 x_2. x_1(ID)((\lambda u v. u)(II)) \Downarrow_{\mathbf{N}} \lambda x_1 x_2. x_1 D(\lambda v. I)} \text{ (abs)} \\ \hline \lambda x_1 x_2. x_1(ID)((\lambda u v. u)(II)) \Downarrow_{\mathbf{N}} \lambda x_1 x_2. x_1 D(\lambda v. I) \end{array}$$

The system $\Downarrow_{\mathbf{N}}$ characterizes completely, from an operational point of view, the class of Λ -normal forms.

Theorem 6.2.3. (i) $M \Downarrow_{\mathbf{N}} N$ implies $M \rightarrow_{\Lambda}^{*p} N$ and N is in Λ -nf.
(ii) $M \Downarrow_{\mathbf{N}}$ if and only if M has Λ -nf.

Proof. (i) By induction on the definition of $\Downarrow_{\mathbf{N}}$.

(ii) (\Rightarrow) Directly from (i).

(\Leftarrow) If $M \rightarrow_{\Lambda}^* N \in \Lambda$ -NF then $M \rightarrow_{\Lambda}^{*p} N$, by Corollary 1.2.13. The proof follows by induction on the pair (M, p) , where p is the length of the reduction sequence $M \rightarrow_{\Lambda}^{*p} N$ ordered in a lexicographic way.

Let $M \equiv \lambda x_1 \dots x_n. \zeta M_1 \dots M_m$.

If ζ is a variable then $N \equiv \lambda x_1 \dots x_n. \zeta \text{nf}_\Lambda(M_1) \dots \text{nf}_\Lambda(M_m)$. By induction $M_i \Downarrow_N$ ($1 \leq i \leq m$), thus $M \Downarrow_N$ by rule (var) having as premises the derivation proving $M_i \Downarrow_N$ and n instances of (abs).

If $\zeta \equiv (\lambda x.P)Q$ then $\text{nf}_\Lambda(M) \equiv \lambda x_1 \dots x_n. \text{nf}_\Lambda(P[Q/x]M_1 \dots M_m)$; so, by induction, $P[Q/x]M_1 \dots M_m \Downarrow_N R$, for some R ; hence $(\lambda x.P)QM_1 \dots M_m \Downarrow_N N$, by applying rule (head) and $M \Downarrow_N \lambda x_1 \dots x_n. N$ by n instances of (abs). \square

The following property will be quite useful in Sect. 15.2.

Property 6.2.4. Let $M \Downarrow_N N$. $MT \Downarrow_N U$ if and only if $NT \Downarrow_N U$.

Proof. Clearly $MT =_\Lambda NT$, so the by confluence theorem MT has Λ -nf if and only if NT has Λ -nf; hence let $MT \Downarrow_N U_0$ and $NT \Downarrow_N U_1$. By Corollary 1.2.6, it is easy to show that $U_0 \equiv U_1$. \square

An immediate consequence of the Theorem 6.2.3.(ii) is that $M \Downarrow_N$ implies $M \Downarrow_H$. Moreover, \approx_N is a Λ -theory as proved in Theorem 6.2.5.

Theorem 6.2.5 (N-Correctness).

The $\lambda\Lambda$ -calculus is correct with respect to the N-operational semantics.

Proof. By definition of correctness, we must prove that $M =_\Lambda N$ implies $M \approx_N N$. Let $M =_\Lambda N$ and let $C[.]$ be a context such that $C[M], C[N] \in \Lambda^0$. By definition of $=_\Lambda$, $C[M] =_\Lambda C[N]$. So the proof follows from Theorem 6.2.3, since by the confluence theorem the property of having Λ -nf is closed under $=_\Lambda$. \square

We will prove that the $\lambda\Lambda$ -calculus is not complete with respect to the N-operational semantics by using a syntactical characterization of N-relevant context.

Lemma 6.2.6 (N-Relevance).

A context $C[.]$ is N-relevant whenever there are $n \geq 1$ contexts $C^i[.] \equiv [.]C_1^i[.] \dots C_{m_i}^i[.]$ ($m_i \in \mathbb{N}$, $1 \leq i \leq n$) such that for all $M \in \Lambda^0$, $C[M] \Downarrow_N$ if and only if $\forall i \leq n$, $C^i[M] \Downarrow_N$.

Proof. (\Rightarrow) Assume that $C[.]$ is N-relevant; namely, there are $M, N \in \Lambda^0$ such that $C[M] \Downarrow_N$ and $C[N] \uparrow_N$. By induction on $C[M] \Downarrow_N$ we will prove that there is at least one context satisfying the statement.

If the last applied rule is (var) then either $C[.] \equiv xC_1[.] \dots C_m[.]$ ($m \in \mathbb{N}$) or $C[.] \equiv [.]C_1[.] \dots C_m[.]$ ($m \in \mathbb{N}$). In the first case the N-relevance implies $m \geq 1$, and $C[M] \Downarrow_N$ implies that $C_j[M] \Downarrow_N$, for each $1 \leq j \leq m$. Let $\{D_1[.], \dots, D_h[.\]\}$ ($h \leq m$) be the subset of all relevant contexts in $\{C_1[.], \dots, C_m[.\]\}$; it is not empty by the hypothesis that $C[.]$ is N-relevant. So the proof follows by induction on contexts $D_i[.]$. The second case is not possible, since $M \in \Lambda^0$.

If the last applied rule is (abs) then either $C[.] \equiv [.]$ or $C[.] \equiv \lambda z.C''[.]$. The first case is immediate, while the second follows by induction.

If the last applied rule is (head) then either $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$) or $C[.] \equiv (\lambda z.C_0[.])C_1[.]...C_m[.]$ ($m \geq 1$). The first case is trivial, while the second follows by induction; in fact, the context $C_0[.][C_1[.]/z]C_2[.]...C_m[.]$ is discriminating M and N and so is \mathbf{N} -relevant too.

(\Leftarrow) Let $k = \max\{m_1, \dots, m_n\}$; it is easy to see that $M \equiv DD$ and $N \equiv \lambda x_1 \dots x_k z. z$ make $C[.]$ relevant. \square

By observing the details of the proof, it is easy to see that actually, for all $M \in \Lambda^0$, if $C[M] \Downarrow_{\mathbf{N}}$ then in the derivation of $C[M] \Downarrow_{\mathbf{N}}$ there are contexts $C_1^i[.], \dots, C_{m_i}^i[.]$ ($m_i \in \mathbb{N}$) and there are n subderivations proving $MC_1^i[M]...C_{m_i}^i[M] \Downarrow_{\mathbf{N}}$ ($1 \leq i \leq n$).

The notion of \mathbf{N} -relevant context is weaker than that of \mathbf{H} -relevant context. In particular, it does not enjoy a property similar to that proved in Lemma 6.1.7. Let $C[.] \equiv [.](\lambda x.I)(DD)$, so $C[\lambda xy.x(DD)] \Downarrow_{\mathbf{N}} I$ but $\lambda xy.y(DD) \uparrow_{\mathbf{N}}$; moreover, $C[\lambda yx.x] \uparrow_{\mathbf{N}}$ while $\lambda yx.x \Downarrow_{\mathbf{N}}$.

So, in order to prove the operational equality between closed unsolvable terms, we will use a less general property of relevant contexts, but sufficient for our purpose.

Lemma 6.2.7. *Let $C[.]$ be \mathbf{N} -relevant.*

If $M \in \Lambda^0$ and $C[M] \Downarrow_{\mathbf{N}}$ then M is solvable.

Proof. By induction on the derivation proving $C[M] \Downarrow_{\mathbf{N}}$.

If the last applied rule is (var) then either $C[.] \equiv xC_1[.]...C_m[.]$ ($m \in \mathbb{N}$) or $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$). In the first case the relevance implies $m \geq 1$, so the proof follows by induction; the second case is not possible, since $M \in \Lambda^0$. If the last applied rule is (abs) then either $C[.] \equiv [.]$ or $C[.] \equiv \lambda z.C'[.]$. The first case is trivial, since M has Λ -nf; the second case follows by induction. If the last applied rule is (head) then either $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$) or $C[.] \equiv (\lambda z.C_0[.])C_1[.]...C_m[.]$ ($m \geq 1$).

The first case follows by Property 2.1.9.(ii), while the second follows by induction on $C_0[M][C_1[M]/z]C_2[M]...C_m[M]$. \square

Also the \mathbf{N} -operational semantics, like the \mathbf{H} one, equates all closed Λ -unsolvable terms.

Theorem 6.2.8 (N-Incompleteness).

The $\lambda\Lambda$ -calculus is incomplete with respect to the \mathbf{N} -operational semantics.

Proof. Let P and Q be two closed Λ -unsolvable terms such that $P \neq_{\Lambda} Q$. A non- \mathbf{N} -relevant context cannot discriminate P and Q . Let $C[.]$ be an \mathbf{N} -relevant context; P, Q do not have hnf (by Theorem 2.1.1), thus both $P \uparrow_{\mathbf{N}}$ and $Q \uparrow_{\mathbf{N}}$; so $C[P] \uparrow_{\mathbf{N}}$ and $C[Q] \uparrow_{\mathbf{N}}$ by Lemma 6.2.7. Hence, $P \approx_{\mathbf{N}} Q$. \square

The following property holds.

Property 6.2.9. $I \approx_{\mathbf{N}} E$.

Proof. By absurdum assume $I \not\approx_{\mathbf{N}} E$. This means that there is a context $C[.]$ discriminating them. Let $C[.]$ be such that $C[I] \Downarrow_{\mathbf{N}}$ while $C[E] \Uparrow_{\mathbf{N}}$.

Let $C[.]$ be a minimal discriminating context for I and E , in the sense that the derivation of $C[I] \Downarrow_{\mathbf{N}}$ has a minimal size between all the proofs of $C'[I] \Downarrow_{\mathbf{N}}$, for every $C'[.]$ such that $C'[I] \Downarrow_{\mathbf{N}}$ and $C'[E] \Uparrow_{\mathbf{N}}$. The proof is done by considering the last applied rule in the derivation proving $C[I] \Downarrow_{\mathbf{N}}$.

If the last applied rule is (var) then $C[.] \equiv xC_1[.]...C_m[.]$ ($m \in \mathbb{N}$), so there is a $C_k[.]$ ($1 \leq k \leq m$) discriminating I and E with a derivation having smaller size, against the hypothesis that $C[.]$ is minimum.

If the last used rule is (abs) then, either $C[.] \equiv \lambda x.C'[.]$ or $C[.] \equiv [.]$. In the former case, $C'[.]$ would be a discriminating context such that the derivation of $C'[I] \Downarrow_{\mathbf{N}}$ has smaller size than the derivation of $C[I] \Downarrow_{\mathbf{N}}$, against the hypothesis. The latter case is not possible, since clearly $[.]$ is not a discriminating context for I and E .

Let the last used rule be (head), thus either $C[.] \equiv (\lambda x.C_0[.])C_1[.]...C_m[.]$ ($m \geq 1$) or $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$). In the former case, the context $C'[.] \equiv C_0[.][C_1[.]/x]C_2[.]...C_m[.]$ would be a discriminating context, such that the derivation $C'[I]$ has smaller size than $C[.]$, against the hypothesis that $C[.]$ is minimum. The case $C[.] \equiv [.]C_1[.]...C_m[.]$ leads to a similar contradiction. In fact, in this case $C_1[.]...C_m[.]$ would be a discriminating context “smaller” than $C[.]$.

The case $C[I] \Uparrow_{\mathbf{N}}$ and $C[E] \Downarrow_{\mathbf{N}}$ is symmetric. \square

Theorem 6.2.10 shows that the \mathbf{N} -operational semantics is fully extensional (see Sect. 1.3).

Theorem 6.2.10. *The theory $\approx_{\mathbf{N}}$ is fully extensional.*

Proof. By Properties 2.1.7 and 6.2.9. \square

6.3 L-Operational Semantics

$\mathbf{L} \in \mathcal{E}(\Lambda, \Lambda\text{-LHNF})$ is the evaluation relation studied in this section; it is the universal evaluation relation $\mathcal{U}_{\Lambda\text{-LHNF}}^{\Lambda}$.

The \mathbf{L} -operational semantics models the so-called lazy evaluation in a call-by-name parameter passing environment. It is characterized by the fact that a Λ -redex is never reduced in case it occurs under the scope of an abstraction. This behaviour is similar to that of the real (call-by-name) programming languages, where the body of a procedure is evaluated only when its parameters are supplied.

Definition 6.3.1 (L-Operational semantics).

- (i) $\mathbf{L} \in \mathcal{E}(\Lambda, \Lambda\text{-LHNF})$ is the evaluation relation induced by the formal system proving judgments of the shape

$$M \Downarrow_{\mathbf{L}} N$$

where $M \in \Lambda$ and $N \in \Lambda\text{-LHNF}$. It consists of the following rules:

$$\frac{m \geq 0}{x M_1 \dots M_m \Downarrow_{\mathbf{L}} x M_1 \dots M_m} \text{ (var)}$$

$$\frac{}{\lambda x. M \Downarrow_{\mathbf{L}} \lambda x. M} \text{ (lazy)}$$

$$\frac{P[Q/x] M_1 \dots M_m \Downarrow_{\mathbf{L}} N}{(\lambda x. P) Q M_1 \dots M_m \Downarrow_{\mathbf{L}} N} \text{ (head)}$$

- (ii) $M \preceq_{\mathbf{L}} N$ if and only if, for all contexts $C[.]$ such that $C[M], C[N] \in \Lambda^0$,
 $(C[M] \Downarrow_{\mathbf{L}} \text{ implies } C[N] \Downarrow_{\mathbf{L}})$.
- (iii) $M \approx_{\mathbf{L}} N$ if and only if $M \preceq_{\mathbf{L}} N$ and $N \preceq_{\mathbf{L}} M$.

The formal system described before, when restricted to closed terms, corresponds to the call-by-name lazy evaluation machine introduced by Plotkin [78]. It is easy to check that \mathbf{L} is deterministic.

Example 6.3.2. $(\lambda xy.x)(DD) \Downarrow_{\mathbf{L}} \lambda y. DD$. In fact, we can build the following derivation:

$$\frac{}{\lambda y. DD \Downarrow_{\mathbf{L}} \lambda y. DD} \text{ (lazy)}$$

$$\frac{}{(\lambda xy.x)(DD) \Downarrow_{\mathbf{L}} \lambda y. DD} \text{ (head)}$$

The following theorem proves that the system \mathbf{L} characterizes completely the class of Λ -lazy head normal forms.

- Theorem 6.3.3.** (i) $M \Downarrow_{\mathbf{L}} N$ implies $M \rightarrow_{\Lambda}^{*p} N$ and N is in Λ -lhnf.
(ii) $M \Downarrow_{\mathbf{L}}$ if and only if M has a Λ -lhnf.

Proof. (i) By induction on the definition of $\Downarrow_{\mathbf{L}}$.

(ii) (\Rightarrow) The proof is a consequence of (i).

(\Leftarrow) M has a Λ -lhnf means that there is $N \in \Lambda\text{-LHF}$ such that $M =_{\Lambda} N$. But $\Lambda\text{-LHF}$ is a set of output values with respect to Λ , by Property 5.0.3; so there is a reduction sequence $M \xrightarrow{\Lambda}^{*p} M' \in \Lambda\text{-LHF}$. The proof is done by induction on the length of the reduction sequence $M \xrightarrow{\Lambda}^{*p} M'$. Let $M \equiv \lambda x_1 \dots x_n. \zeta M_1 \dots M_m$.

If either $n \geq 1$ or ζ is a variable, then a Λ -lhnf of M is M itself, so $M \Downarrow_{\mathbf{L}} M$, by an application of rule (lazy) or an application of rule (var). If $n = 0$ and $\zeta \equiv (\lambda x.P)Q$ then the Λ -lhnf of M is the Λ -lhnf of $P[Q/x]M_1 \dots M_m$. By induction $P[Q/x]M_1 \dots M_m \Downarrow_{\mathbf{L}} N$, for some N , so $M \Downarrow_{\mathbf{L}} N$, by applying the rule (head). \square

By Theorem 6.3.3, it follows that both $M \Downarrow_{\mathbf{H}}$ and $M \Downarrow_{\mathbf{N}}$ imply $M \Downarrow_{\mathbf{L}}$. The following property will be quite useful in Sect. 15.2.

Property 6.3.4. Let $M, N, T, U \in \Lambda^0$ and $M \Downarrow_{\mathbf{L}} N$.

$$MT \Downarrow_{\mathbf{L}} U \text{ if and only if } NT \Downarrow_{\mathbf{L}} U.$$

Proof. By induction on $M \Downarrow_{\mathbf{L}} N$. The last applied rule cannot be (var), since $M \in \Lambda^0$. If the last applied rule is (lazy) then the proof is trivial. If the last applied rule is (head) then the proof follows by induction. \square

Theorem 6.3.5 proves that $\approx_{\mathbf{L}}$ is a Λ -theory.

Theorem 6.3.5 (L-Correctness).

The $\lambda\Lambda$ -calculus is correct with respect to the L-operational semantics.

Proof. By definition of correctness, we must prove that $M =_{\Lambda} N$ implies $M \approx_{\mathbf{L}} N$. Let $M =_{\Lambda} N$ and let $C[.]$ be a context such that $C[M], C[N] \in \Lambda^0$. By definition of $=_{\Lambda}$, $C[M] =_{\Lambda} C[N]$. So the proof follows by Theorem 6.3.3, since by the confluence theorem the property of having Λ -lhnf is closed under $=_{\Lambda}$. \square

We will prove that the $\lambda\Lambda$ -calculus is not complete with respect to the L-operational semantics by using a syntactical characterization of **L-relevant** context.

Lemma 6.3.6 (L-Relevance).

A context $C[.]$ is L-relevant whenever there is a context $C'[.] \equiv [.]C_1[.] \dots C_m[.]$ ($m \in \mathbb{N}$) such that for all $M \in \Lambda^0$, $C[M] \Downarrow_{\mathbf{L}}$ if and only if $C'[M] \Downarrow_{\mathbf{L}}$.

Proof. (\Rightarrow) Assume that $C[.]$ is L-relevant; namely, there are $M, N \in \Lambda^0$ such that $C[M] \Downarrow_{\mathbf{L}}$ and $C[N] \uparrow_{\mathbf{L}}$. By induction on $C[M] \Downarrow_{\mathbf{L}}$ we will prove that there is a context $C'[.]$ satisfying the statement.

The last applied rule cannot be (var), since $C[.] \equiv xC_1[.] \dots C_m[.]$ ($m \in \mathbb{N}$) is not relevant. If the last applied rule is (lazy) then either $C[.] \equiv [.]$ or

$C[.] \equiv \lambda z.C''[.]$. The first case is immediate; while the second is not possible, since $\lambda z.C''[.]$ is not relevant.

If the last applied rule is (head) then either $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$) or $C[.] \equiv (\lambda z.C_0[.])C_1[.]...C_m[.]$ ($m \geq 1$). The first case is trivial, while the second follows by induction; in fact, the context $C_0[.][C_1[.]/z]C_2[.]...C_m[.]$ is discriminating M and N and so is \mathbf{L} -relevant too.

(\Leftarrow) Let $C'[.]$ be a context satisfying the statement of this lemma, so $C'[M] \Downarrow_{\mathbf{L}}$ if and only if $C[M] \Downarrow_{\mathbf{L}}$, for each $M \in \Lambda$. Thus $M \equiv DD$ and $N \equiv \lambda x_1...x_m z.z$ are witnesses of the \mathbf{L} -relevance of $C[.]$. \square

By observing the details of the proof, it is easy to see that, for all $M \in \Lambda$, if $C[M] \Downarrow_{\mathbf{L}}$ then in the derivation of $C[M] \Downarrow_{\mathbf{L}}$ there are contexts $C_1[.], \dots, C_m[.]$ ($m \in \mathbb{N}$) and there is a subderivation proving $MC_1[M]...C_m[M] \Downarrow_{\mathbf{L}}$. Note that the context $\lambda z.[.]$ is \mathbf{H} -relevant nevertheless it is not \mathbf{L} -relevant.

Lemma 6.3.7. *Let $C[.]$ be \mathbf{L} -relevant. If $M \in \Lambda^0$ and $C[M] \Downarrow_{\mathbf{L}}$ then $M \Downarrow_{\mathbf{L}}$.*

Proof. By induction on the derivation proving $C[M] \Downarrow_{\mathbf{L}}$.

The last applied rule cannot be (var), since $C[.] \equiv xC_1[.]...C_m[.]$ ($m \in \mathbb{N}$) is not relevant. If the last applied rule is (abs) then $C[.] \equiv \lambda z.C'[.]$ is not relevant, while the case $C[.] \equiv [.]$ is trivial.

If the last applied rule is (head) then either $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$) or $C[.] \equiv (\lambda z.C_0[.])C_1[.]...C_m[.]$ ($m \geq 1$). The last case follows by induction on $C_0[M][C_1[M]/z]C_2[M]...C_m[M]$; so let $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$). If $M \equiv \lambda z.M'$ then immediately $M \Downarrow_{\mathbf{L}}$, so let $M \equiv (\lambda z.P)QM_1...M_n$ ($n \in \mathbb{N}$). $C[M] \Downarrow_{\mathbf{L}}$ implies, by rule (head) that $P[Q/z]M_1...M_nC_1[M]...C_m[M] \Downarrow_{\mathbf{L}}$; since $[.]C_1[M]...C_m[M]$ is a relevant context (it discriminates DD and M), by induction $P[Q/z]M_1...M_n \Downarrow_{\mathbf{L}}$. Thus by applying the rule (head), $M \Downarrow_{\mathbf{L}}$ follows. \square

An incompleteness result holds.

Theorem 6.3.8 (L-Incompleteness).

The $\lambda\Lambda$ -calculus is incomplete with respect to the \mathbf{L} -operational semantics.

Proof. Let P and Q be two closed Λ -unsolvable terms of order zero such that $P \neq_{\Lambda} Q$. A non- \mathbf{L} -relevant context cannot discriminate them. By Definition 2.1.3, if either $P \rightarrow_{\Lambda}^* R$ or $Q \rightarrow_{\Lambda}^* R$, for some R , then R cannot be an abstraction, hence $P \uparrow_{\mathbf{L}}$ and $Q \uparrow_{\mathbf{L}}$. Let $C[.]$ be any \mathbf{L} -relevant context; so, by Lemma 6.3.7, $C[P] \uparrow_{\mathbf{L}}$ and $C[Q] \uparrow_{\mathbf{L}}$. Hence, $P \approx_{\mathbf{L}} Q$. \square

Property 6.3.9. (i) $m \neq n$ implies $\lambda x_1...x_n.DD \not\approx_{\mathbf{L}} \lambda x_1...x_m.DD$.

(ii) $I \not\approx_{\mathbf{L}} E$.

Proof. (i) It is an exercise, by using the fact that $DD \uparrow_{\mathbf{L}}$.

(ii) The context $[.](DD)$ discriminates the two given terms. \square

From the previous property and by Property 2.1.7, it follows that the operational semantics $\approx_{\mathbf{L}}$, being a Λ -theory, is not fully extensional.

6.3.1 An Example

We will show now that $L_0 \approx_L L_1$, where

$$\begin{aligned} L_0 &\equiv \lambda x.x(x(\lambda x.DD)(DD))(\lambda x.DD), \\ L_1 &\equiv \lambda x.x(\lambda y.x(\lambda x.DD)(DD)y)(\lambda x.DD). \end{aligned}$$

This equivalence was first stated in [2]. The interest of such a result will be clear when we will study the denotational semantics. First, let us prove a general property.

Lemma 6.3.10. *Let $M \rightarrow_\eta^* N$. If $N \Downarrow_L$ then $M \Downarrow_L$.*

Proof. Let s be the size of the derivation proving $N \Downarrow_L$, and let l be the length of the reduction sequence from M to N . The proof is given by induction on the pair (s, l) ; the pairs are ordered according to the lexicographical order. If $l = 0$ then the proof is immediate, so let $l \geq 1$.

If $N \equiv xN_1\dots N_n$ ($n \in \mathbb{N}$) then there are three cases.

1. $M \equiv \lambda y.M_0y$, where $M_0 \rightarrow_\eta^* N$ and $y \notin \text{FV}(M_0)$. The proof follows immediately by rule (lazy).
2. $M \equiv (\lambda y.M_0y)M_1\dots M_m$ ($1 \leq m \leq n$), where $M_0M_1\dots M_m \rightarrow_\eta^* N$ and $y \notin \text{FV}(M_0)$. By induction $M_0M_1\dots M_m \Downarrow_L$, hence $M \Downarrow_L$ by rule (head).
3. $M \equiv xM_1\dots M_n$, where $M_i \rightarrow_\eta^* N_i$ ($1 \leq i \leq n$). The proof follows by rule (var).

If $N \equiv \lambda x.N_0$ then $M \equiv \lambda y.M_0$ and the proof follows by rule (lazy). If $N \equiv (\lambda x.N_0)N_1\dots N_n$ ($n \geq 1$), then there are three cases.

1. $M \equiv \lambda y.M_0y$, where $M_0 \rightarrow_\eta^* N$ and $y \notin \text{FV}(M_0)$. The proof follows by rule (lazy).
2. $M \equiv (\lambda y.M_0y)M_1\dots M_m$ ($1 \leq m \leq n$), where $M_0M_1\dots M_m \rightarrow_\eta^* N$ and $y \notin \text{FV}(M_0)$. By induction $M_0M_1\dots M_m \Downarrow_L$, hence $M \Downarrow_L$ by rule (head).
3. $M \equiv (\lambda x.M_0)M_1\dots M_n$, where $M_i \rightarrow_\eta^* N_i$ ($1 \leq i \leq n$). It is easy to see that $M_0[M_1/x]M_2\dots M_n \rightarrow_\eta^* N_0[N_1/x]N_2\dots N_n$. But $N \Downarrow_L$ implies $N_0[N_1/x]N_2\dots N_n \Downarrow_L$, and there is a derivation having size less than s proving it; hence by induction $M_0[M_1/x]M_2\dots M_n \Downarrow_L$. Then the proof follows by rule (head). \square

Lemma 6.3.10 implies the Corollary 6.3.11.

Corollary 6.3.11. *If $M \rightarrow_\eta^* N$ then $N \preceq_L M$.*

Proof. Clearly $C[M] \rightarrow_\eta^* C[N]$, for all contexts $C[.]$. Then by Lemma 6.3.10, $C[N] \Downarrow_L$ implies $C[M] \Downarrow_L$. \square

In particular, it follows that $L_0 \preceq_L L_1$. The next goal is to prove the reverse relation, namely $L_1 \preceq_L L_0$.

Let $M \in \Lambda^0$; it is easy to check that $M(\lambda y.M(\lambda x.DD)(DD)y)(\lambda x.DD) \Downarrow_L$ if and only if $L_1 M \Downarrow_L$, by rule (head).

Lemma 6.3.12. Let $M, N \in \Lambda^0$ be such that $N =_A M$.

If $M(\lambda y.N(\lambda x.DD)(DD)y)(\lambda x.DD) \Downarrow_L$, then either $N \rightarrow_A^* \lambda x_0x_1.x_0$ or $N \rightarrow_A^* \lambda x_0x_1.x_1$ or $N \rightarrow_A^* \lambda x_0x_1x_2.M'''$, for some $M''' \in \Lambda$.

Proof. Let $R \equiv M(\lambda y.N(\lambda x.DD)(DD)y)(\lambda x.DD)$. The proof is given by induction on the size of the derivation proving $R \Downarrow_L$.

The last applied rule cannot be (var), since $R \in \Lambda^0$. The last applied rule cannot be (lazy), since R is not an abstraction. Hence the last applied rule is (head); we consider all the possible shapes of M .

- $M \equiv xM_1\dots M_m$ ($m \in \mathbb{N}$) is not possible, since $M \in \Lambda^0$.
- $M \equiv (\lambda x_0.M')M_1\dots M_m$ ($m \geq 1$). The proof follows by induction on

$$M'[M_1/x_0]M_2\dots M_m(\lambda y.N(\lambda x.DD)(DD)y)(\lambda x.DD) \Downarrow_L .$$

- Let $M \equiv \lambda x_0.M'$, so $\text{FV}(M') \subseteq \{x_0\}$.
 - $M' \equiv x_0M_1\dots M_m$ ($m \in \mathbb{N}$) is not possible; in fact, \approx_L is a Λ -theory, so $N(\lambda x.DD)(DD) \approx_L M(\lambda x.DD)(DD) \approx_L (\lambda x.DD)M'_1\dots M'_m(DD)$, where $M'_i \equiv M_i[\lambda x.DD/x_0]$ ($1 \leq i \leq m$). This fact implies $M(\lambda y.N(\lambda x.DD)(DD)y)(\lambda x.DD) \uparrow_L$.
 - Let $M' \equiv (\lambda x_1.M_0)M_1\dots M_m$ ($m \geq 1$) and $T \equiv \lambda y.N(\lambda x.DD)(DD)y$, so there is a derivation d and a term R' such that

$$\frac{\frac{\frac{d}{(M_0[M_1/x_1])[T/x_0]M_1[T/x_0]\dots M_m[T/x_0](\lambda x.DD) \Downarrow_L R'}(\dots)}{(x_1.M_0[T/x_0])M_1[T/x_0]\dots M_m[T/x_0](\lambda x.DD) \Downarrow_L R'}(\text{head})}}{(\lambda x_0.(\lambda x_1.M_0)M_1\dots M_m)(\lambda y.N(\lambda x.DD)(DD)y)(\lambda x.DD) \Downarrow_L R'}(\text{head})}$$

hence the proof follows by induction on the following derivation

$$\frac{\frac{d}{(M_0[M_1/x_1])[T/x_0]M_1[T/x_0]\dots M_m[T/x_0](\lambda x.DD) \Downarrow_L R'}(\dots)}}{(\lambda x_0.(M_0[M_1/x_1])M_2\dots M_m)(\lambda y.N(\lambda x.DD)(DD)y)(\lambda x.DD) \Downarrow_L R'}(\text{head})}$$

- Let $M' \equiv \lambda x_1.M_0$. Since $M \in \Lambda^0$, there are only three further cases.

- (i) $M \equiv \lambda x_0x_1.x_0M_1\dots M_m$ ($m \in \mathbb{N}$). If $m \geq 1$, then $N(\lambda x.DD)(DD) \approx_L (\lambda x_0x_1.x_0M_1\dots M_m)(\lambda x.DD)(DD) \approx_L (\lambda x.DD)M'_1\dots M'_m$, where $M'_i \equiv M_i[\lambda x.DD/x_0, DD/x_1]$ ($1 \leq i \leq m$), because \approx_L is a Λ -theory. Hence $R \uparrow_L$ since

$$R \approx_L (\lambda x_0x_1.x_0M_1\dots M_m)(\lambda y.(\lambda x.DD)M'_1\dots M'_m y)(\lambda x.DD) .$$

So let $m = 0$ and the proof is done.

- (ii) $M \equiv \lambda x_0x_1.x_1M_1\dots M_m$ ($m \in \mathbb{N}$); it is easy to see that $m \geq 1$ implies $R \uparrow_L$. Thus $m = 0$.
- (iii) The case $M \equiv \lambda x_0x_1x_2.M'''$ is immediate. □

By using the previous characterization we can prove that $L_1 \preceq_L L_0$.

Lemma 6.3.13. *Let $C[.]$ be such that $C[L_0], C[L_1] \in \Lambda^0$.*

If $C[L_1] \Downarrow_L$ then $C[L_0] \Downarrow_L$.

Proof. The proof is given by induction on the size of $C[L_1] \Downarrow_L$, by considering the last applied rule.

- (var) It is not possible, since $C[L_1] \in \Lambda^0$.
- (lazy) If $C[.] \equiv [.]$ then the proof is trivial, since $L_0 \Downarrow_L$. If $C[.] \equiv \lambda z.C'[.]$ then the proof is trivial, since $C[.]$ is not relevant.
- (head) Either $C[.] \equiv (\lambda z.C_0[.])C_1[.]...C_m[.]$ or $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \geq 1$).

In the first case, the proof follows by induction on

$$C_0[L_1][C_1[L_1]/z]C_2[L_1]...C_m[L_1].$$

In the second case, the hypothesis that the last applied rule is (head) implies $m \geq 1$, and there is a derivation d and a term R such that

$$\frac{d}{\begin{array}{c} C_1[L_1](\lambda y.C_1[L_1](\lambda x.DD)(DD)y)(\lambda x.DD)C_2[L_1]...C_m[L_1] \Downarrow_L R \\ \hline L_1C_1[L_1]...C_m[L_1] \Downarrow_L R \end{array}} \text{(head)}$$

By inductive hypothesis,

$$C_1[L_0](\lambda y.C_1[L_0](\lambda x.DD)(DD)y)(\lambda x.DD)C_2[L_0]...C_m[L_0] \Downarrow_L$$

so by Lemma 6.3.12, there are three possible cases.

- (i) If $C_1[L_0] \rightarrow_A^* \lambda x_0x_1.x_0$ then it is easy to see that

$$C_1[L_0](\lambda y.C_1[L_0](\lambda x.DD)(DD)y)(\lambda x.DD) =_A \lambda y.DD;$$

so $m \geq 2$ is not possible. Let $m = 1$; so it is easy to see that

$$\lambda y.C_1[L_0](\lambda x.DD)(DD)y =_A \lambda x.DD =_A C_1[L_0](\lambda x.DD)(DD).$$

Hence $C_1[L_0](C_1[L_0](\lambda x.DD)(DD))(\lambda x.DD) \Downarrow_L$, since \approx_L is a Λ -theory. By rule (head), $C[L_0] \Downarrow_L$.

- (ii) If $C_1[L_1] \rightarrow_A^* \lambda x_0x_1.x_1$ then it is easy to see that

$$C_1[L_0](\lambda y.C_1[L_0](\lambda x.DD)(DD)y)(\lambda x.DD) =_A \lambda x.DD;$$

so $m \geq 2$ is not possible and we can assume $m = 1$.

But $C_1[L_1] \rightarrow_A^* \lambda x_0x_1.x_1$ implies $C_1[L_0]Q(\lambda x.DD) =_A \lambda x.DD$, for each $Q \in \Lambda$. Hence $C_1[L_0](C_1[L_0](\lambda x.DD)(DD))(\lambda x.DD) \Downarrow_L$, since \approx_L is a Λ -theory. By rule (head), $C[L_0] \Downarrow_L$.

(iii) If $C_1[L_0] \rightarrow_A^* \lambda x_0 x_1 x_2. M'''$, for some $M''' \in \Lambda$, then

$$\begin{aligned} C_1[L_0](\lambda y.C_1[L_0](\lambda x.DD)(DD)y)(\lambda x.DD)C_2[L_0]\dots C_m[L_0] &=_{\Lambda} \\ C_1[L_0](\lambda y.M'''[\lambda x.DD/x_0, DD/x_1, y/x_2])(\lambda x.DD)C_2[L_0]\dots C_m[L_0] &=_{\Lambda} \\ C_1[L_0](C_1[L_0](\lambda x.DD)(DD))(\lambda x.DD)C_2[L_0]\dots C_m[L_0] \end{aligned}$$

Since \approx_L is a $\lambda\Lambda$ -theory,

$$C_1[L_0](C_1[L_0](\lambda x.DD)(DD))(\lambda x.DD)C_2[L_0]\dots C_m[L_0] \Downarrow_L$$

and the proof follows by applying the rule (head). \square

Theorem 6.3.14. $L_1 \approx_L L_0$.

Proof. By Corollary 6.3.11 and Lemma 6.3.13. \square

7. Call-by-Value Operational Semantics

7.1 V-Operational Semantics

As proved in Property 5.0.3, the set of Γ -lazy blocked normal forms (Γ -lbnf's), namely $\Gamma\text{-LBNF} = \{\lambda x.M \mid M \in \Lambda\} \cup \{xM_1\dots M_m \mid M_i \in \Lambda, m \in \mathbb{N}\} \cup \{(\lambda x.P)QM_1\dots M_m \mid P, M_i \in \Lambda, Q \notin \Gamma, Q \in \Gamma\text{-LBNF}, m \in \mathbb{N}\}$, is a set of output values with respect to Γ . Notice that $\Gamma\text{-LBNF}^0 = \Gamma^0$.

$\mathbf{V} \in \mathcal{E}(\Gamma, \Gamma\text{-LBNF})$ is the evaluation relation studied in this section; it is the universal evaluation relation $\mathcal{U}_{\Gamma\text{-LBNF}}^\Gamma$. This operational semantics models the call-by-value parameter passing together with lazy evaluation.

Definition 7.1.1 (V-Operational semantics).

- (i) $\mathbf{V} \in \mathcal{E}(\Gamma, \Gamma\text{-LBNF})$ is the evaluation relation induced by the formal system proving judgments of the shape

$$M \Downarrow_{\mathbf{V}} N$$

where $M \in \Lambda$ and $N \in \Gamma\text{-LBNF}$. It consists of the following rules:

$$\frac{}{xM_1\dots M_m \Downarrow_{\mathbf{V}} xM_1\dots M_m} \text{(var)}$$

$$\frac{}{\lambda x.M \Downarrow_{\mathbf{V}} \lambda x.M} \text{(lazy)}$$

$$\frac{Q \Downarrow_{\mathbf{V}} Q' \quad Q' \in \Gamma \quad P[Q'/x]M_1\dots M_m \Downarrow_{\mathbf{V}} N}{(\lambda x.P)QM_1\dots M_m \Downarrow_{\mathbf{V}} N} \text{(head)}$$

$$\frac{Q \Downarrow_{\mathbf{V}} Q' \quad Q' \notin \Gamma}{(\lambda x.P)QM_1\dots M_m \Downarrow_{\mathbf{V}} (\lambda x.P)Q'M_1\dots M_m} \text{(block)}$$

- (ii) $M \preceq_{\mathbf{V}} N$ if and only if, for all context $C[.]$ such that $C[M], C[N] \in \Lambda^0$,
 $(C[M] \Downarrow_{\mathbf{V}} \text{implies } C[N] \Downarrow_{\mathbf{V}})$.
- (iii) $M \approx_{\mathbf{V}} N$ if and only if $M \preceq_{\mathbf{V}} N$ and $N \preceq_{\mathbf{V}} M$.

The formal system described before, when restricted to closed terms, corresponds to the SECD machine introduced by Landin [63], and further studied by Plotkin [78].

Example 7.1.2. $(\lambda x.yx)(Ky) \Downarrow_{\mathbf{V}} y(\lambda z.y)$. In fact, we can build the following derivation:

$$\frac{\overline{y \Downarrow_{\mathbf{V}} y} \text{ (var)} \quad \overline{\lambda z.y \Downarrow_{\mathbf{V}} \lambda z.y} \text{ (lazy)}}{\overline{Ky \Downarrow_{\mathbf{V}} \lambda z.y} \text{ (head)}} \quad \frac{\overline{y(\lambda z.y) \Downarrow_{\mathbf{V}} y(\lambda z.y)} \text{ (var)}}{\overline{(\lambda x.yx)(Ky) \Downarrow_{\mathbf{V}} y(\lambda z.y)} \text{ (head)}}$$

\mathbf{V} is deterministic and it characterizes the set Γ -LBNF.

Theorem 7.1.3. (i) $M \Downarrow_{\mathbf{V}} N$ implies $M \rightarrow_{\Gamma}^{*p} N$ and N is in Γ -lbnf.
(ii) $M \Downarrow_{\mathbf{V}}$ if and only if M has a Γ -lbnf.

Proof. (i) By induction on the rules of $\Downarrow_{\mathbf{V}}$.
(ii) (\Rightarrow) The proof is a consequence of (i).

(\Leftarrow) M has a Γ -lbnf means that there is $N \in \Gamma$ -LBNF such that $M \rightarrow_{\Gamma}^{*} N$. But Γ -LBNF is a set of output values with respect to Γ , by Property 5.0.3; so there is a reduction sequence $M \rightarrow_{\Gamma}^{*p} M' \in \Gamma$ -LBNF.

Let $M \equiv \lambda x_1 \dots x_n. \zeta M_1 \dots M_m$; we reason by induction on the length of $M \rightarrow_{\Gamma}^{*p} M'$. If $n \neq 0$ then the proof follows by rule (lazy), so let $n = 0$. If $\zeta \in \text{Var}$ the proof follows by rule (var). Otherwise $\zeta \equiv (\lambda z.P)Q$; hence, if Q is Γ -valuable then the proof follows by induction and rule (head), otherwise it follows by induction and rule (block). \square

Corollary 7.1.4. Let $M \in \Lambda^0$. Then $M \Downarrow_{\mathbf{V}}$ if and only if M is Γ -valuable.

Proof. Since $\Gamma^0 = \Gamma$ -LBNF⁰, the proof follows by Theorem 7.1.3.(ii). \square

It follows by Theorem 7.1.5 that $\approx_{\mathbf{V}}$ is a Γ -theory.

Theorem 7.1.5 (V-Correctness).

The $\lambda\Gamma$ -calculus is correct with respect to the \mathbf{V} -operational semantics.

Proof. By definition of correctness, we must prove that $M =_{\Gamma} N$ implies $M \approx_{\mathbf{V}} N$. Let $M =_{\Gamma} N$ and let $C[.]$ be a context such that $C[M], C[N] \in \Lambda^0$. By definition of $=_{\Gamma}$, $C[M] =_{\Gamma} C[N]$. So the proof follows from Theorem 7.1.3, since by the confluence theorem the property of having Γ -lbnf is closed under $=_{\Gamma}$. \square

The **V**-relevant contexts cannot be characterized by using contexts of the shape $[.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$), as in the call-by-name operational settings; in fact, $[.](DD[.])I$ is not **V**-relevant, while $(\lambda xy.y)([.]I)$ is **V**-relevant. However, the following lemma establishes a negative characterization taking into account not **V**-relevant contexts.

Lemma 7.1.6 (V-Relevance).

Let $C[.]$ be not **V**-relevant and $M \in \Lambda^0$. If $C[M] \Downarrow_{\mathbf{V}}$ then there is a context $C'[.]$ such that, $\forall P \in \Lambda^0 \ C[P] \Downarrow_{\mathbf{V}} C'[P]$; moreover, $C'[P] \in \Gamma$ if and only if $C'[M] \in \Gamma$.

Proof. By induction on $C[M] \Downarrow_{\mathbf{V}}$, so by cases on the last applied rule.

(var) If $C[.] \equiv xC_1[.]...C_m[.]$ ($m \in \mathbb{N}$) then the proof is trivial and $C'[.] \equiv C[.]$.

The case $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$) is not possible, since $M \in \Lambda^0$.

(lazy) The case $C[.] \equiv [.]$ is not possible, since it is trivially **V**-relevant; while in case $C[.] \equiv \lambda z.C''[.]$, the proof is trivial.

(head) $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$) is not possible, since $C[M] \Downarrow_{\mathbf{V}}$ while $C[DD] \uparrow_{\mathbf{V}}$ thus $C[.]$ is **V**-relevant, against the hypothesis; so, let $C[.] \equiv (\lambda z.C_0[.])C_1[.]...C_m[.]$ ($m \geq 1$). If there is $N \in \Lambda^0$ such that $C_1[N] \uparrow_{\mathbf{V}}$ then $C[N] \uparrow_{\mathbf{V}}$, by implying that $C[.]$ is **V**-relevant, against the hypothesis. Hence $C_1[.]$ is not relevant and $C_1[M] \Downarrow_{\mathbf{V}}$, thus by induction there is $D_1[.]$ such that $\forall P \in \Lambda^0 \ C_1[P] \Downarrow_{\mathbf{V}} D_1[P]$ and $D_1[P] \in \Gamma$ if and only if $D_1[M] \in \Gamma$. By induction on $C_0[M][D_1[M]/x]C_2[M]...C_m[M]$ there is a context $C'[.]$ such that $C_0[N][D_1[N]/x]C_2[N]...C_m[N] \Downarrow_{\mathbf{V}} C'[N]$ and $C'[N] \in \Gamma$ if and only if $C'[M] \in \Gamma$, for all $N \in \Lambda^0$. The proof follows by rule (head).

(block) The case $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$) is not possible, since it is **V**-relevant (as in the previous case); so, let $C[.] \equiv (\lambda z.C_0[.])C_1[.]...C_m[.]$ ($m \geq 1$). Clearly $C_1[.]$ is not relevant and $C_1[M] \Downarrow_{\mathbf{V}}$, so by induction there is $D_1[.]$ such that $\forall P \in \Lambda^0 \ C_1[P] \Downarrow_{\mathbf{V}} D_1[P]$ and $D_1[P] \notin \Gamma$. Let $C'[.] \equiv (\lambda z.C_0[.])D_1[.]C_2[.]...C_m[.]$ and the proof follows easily. \square

We will prove that the $\lambda\Gamma$ -calculus is not complete with respect to the **V**-operational semantics by using the notion of **V**-relevant context.

Lemma 7.1.7. Let $C[.]$ be **V**-relevant. If $M \in \Lambda^0$ and $C[M] \Downarrow_{\mathbf{V}}$ then $M \Downarrow_{\mathbf{V}}$.

Proof. By induction on $C[M] \Downarrow_{\mathbf{V}}$, so by cases on the last applied rule.

(var) $C[.] \equiv xC_1[.]...C_m[.]$ ($m \in \mathbb{N}$), is not **V**-relevant.

(lazy) $C[.] \equiv \lambda z.C'[.]$ is not **V**-relevant, while $C[.] \equiv [.]$ is trivial.

(head) Let $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$), so there are two cases.

1. If $M \equiv \lambda z.M'$ then $M \Downarrow_{\mathbf{V}}$.

2. Let $M \equiv (\lambda z.P)QM_1...M_n$ ($n \in \mathbb{N}$). $C[M] \Downarrow_{\mathbf{V}}$ implies, by rule (head) that $Q \Downarrow_{\mathbf{V}} Q'$ and $P[Q'/z]M_1...M_nC_1[M]...C_m[M] \Downarrow_{\mathbf{V}}$; since $[.]C_1[M]...C_m[M]$ is a relevant context (it discriminates M and DD),

by induction $P[Q'/z]M_1\dots M_n \Downarrow_{\mathbf{V}}$. Thus by rule (head), it follows that $M \Downarrow_{\mathbf{V}}$.

Let $C[.] \equiv (\lambda z.C_0[.])C_1[.]\dots C_m[.]$ ($m \geq 1$) so there two cases again.

1. If $C_1[.]$ is not relevant then there exists $C'_1[.]$ satisfying the Lemma 7.1.6, thus by induction on $C_0[M][C'_1[M]/z]C_2[M]\dots C_m[M] \Downarrow_{\mathbf{L}}$, the proof follows.
2. Otherwise, $C_1[.]$ is relevant and the proof follows by induction on $C_1[M] \Downarrow_{\mathbf{L}}$.

(block) Let $C[.] \equiv [.]C_1[.]\dots C_m[.]$ ($m \in \mathbb{N}$); if $M \equiv (\lambda x.P)QN_1\dots N_n$ ($n \in \mathbb{N}$) then $Q \Downarrow_{\mathbf{V}} Q'$, but $Q \in \Lambda^0$ and Q' is a closed Γ -lbnf imply that Q' is an abstraction, against the hypothesis that the last applied rule is (block); thus $M \equiv \lambda x.M'$ and $M \Downarrow_{\mathbf{V}}$ by rule (lazy).

Let $C[.] \equiv (\lambda z.C_0[.])C_1[.]\dots C_m[.]$ ($m \geq 1$), so $C_1[.]$ is relevant and the proof follows by induction. \square

The following result holds.

Theorem 7.1.8 (V-Incompleteness).

The $\lambda\Gamma$ -calculus is incomplete with respect to the \mathbf{V} -operational semantics.

Proof. Let P and Q be two closed Γ -unsolvable terms of order zero such that $P \neq_{\Lambda} Q$. A non- \mathbf{V} -relevant context cannot discriminate them. By Definition 3.1.12, if $P \rightarrow_{\Gamma}^{*} R$ or $Q \rightarrow_{\Gamma}^{*} R$, for some R , then R cannot be an abstraction, hence $P \uparrow_{\mathbf{V}}$ and $Q \uparrow_{\mathbf{V}}$. Let $C[.]$ be a \mathbf{V} -relevant context. So, by Lemma 7.1.7, $C[P] \uparrow_{\mathbf{V}}$ and $C[Q] \uparrow_{\mathbf{V}}$. Hence, $P \approx_{\mathbf{V}} Q$. \square

As a corollary, we obtain that the \mathbf{V} -operational equivalence equates all closed Γ -unsolvable terms of the same order.

Corollary 7.1.9. *Let P and Q be closed Γ -unsolvable terms of the same order n . Then $P \approx_{\mathbf{V}} Q$.*

Proof. By induction on n . If $n = 0$, then the proof follows from the proof of the incompleteness result. Otherwise, it follows by induction. \square

The next property shows an interesting characterization of the \mathbf{V} -operational semantics.

Property 7.1.10. *Let M and N be such that M is potentially Γ -valuable while N is not potentially Γ -valuable. Then $M \not\approx_{\mathbf{V}} N$.*

Proof. According to the definition of potentially Γ -valuable terms, we will consider only substitutions whose codomain is Γ . M potentially Γ -valuable means that there is a substitution s such that $s(M) \in \Lambda^0$ is Γ -valuable, while $s'(N)$ is not Γ -valuable, for each substitution s' such that $s'(N) \in \Lambda^0$. So let s be such that $s(M)$ is Γ -valuable; we can easily extend s to a substitution s' such that $s'(M) \in \Lambda^0$ is Γ -valuable while $s'(N) \in \Lambda^0$ is not Γ -valuable. Let $C[.] \equiv (\lambda x_1\dots x_n[.])s'(x_1)\dots s'(x_n)$ where $\text{FV}(M) \cup \text{FV}(N) \subseteq \{x_1, \dots, x_n\}$; therefore, by Corollary 7.1.4, $C[.]$ is discriminating for M and N . \square

The \mathbf{V} -theory is not fully extensional. In fact, $DD \not\approx_{\mathbf{V}} \lambda x.DDx$.

7.1.1 An Example

We will show now that $V_0 \approx_V V_1$, where

$$V_0 \equiv \lambda x.(\lambda x_1 x_2.DD)(x(\lambda x_1.DD)(\lambda x_1.DD)),$$

$$V_1 \equiv \lambda x.(\lambda x_1 x_2 x_3.DD)(x(\lambda x_1.DD)(\lambda x_1 x_2.DD))(x(\lambda x_1 x_2.DD)(\lambda x_1.DD)).$$

This equivalence was first proved in [44]. The interest of such a result will be clear when we will study denotational semantics.

Lemma 7.1.11. *Let $M \in \Lambda^0$.*

- (i) *If $V_0 M \Downarrow_V$ then $V_0 M \Downarrow_V \lambda x.DD$.*
- (ii) *If $V_1 M \Downarrow_V$ then $V_1 M \Downarrow_V \lambda x.DD$.*

Proof. Clearly, $P \in \Lambda^0$ and $P \Downarrow_V P'$ imply P' is an abstraction.

- (i) Let $D^1 \equiv (\lambda x_1.DD)$ and $D^2 \equiv (\lambda x_1 x_2.DD)$; thus

$$\frac{\frac{d_0}{M \Downarrow_V M_0} \dots \frac{\frac{d_1}{M_0 D^1 D^1 \Downarrow_V M_1} \dots \frac{\frac{d_{21}}{\lambda x_2.DD \Downarrow_V \lambda x_2.DD} \dots}{(\lambda x_1 x_2.DD)(M_0 D^1 D^1)} \Downarrow_V \lambda x_2.DD}{(\lambda x_1 x_2.DD)(M_0 D^1 D^1)} \Downarrow_V \lambda x_2.DD}{(\lambda x_1 x_2.DD)(M_0 D^1 D^1)} \Downarrow_V \lambda x_2.DD$$

- (ii) Let $D^1 \equiv (\lambda x_1.DD)$, $D^2 \equiv (\lambda x_1 x_2.DD)$ and $D^3 \equiv (\lambda x_1 x_2 x_3.DD)$; thus

$$\frac{\frac{d_0}{M \Downarrow_V M_0} \dots \frac{\frac{d_{12}}{M_0 D^1 D^2 \Downarrow_V M_{12}} \dots \frac{\frac{d_{21}}{M_0 D^2 D^1 \Downarrow_V M_{21}} \dots \frac{\frac{d_{31}}{D^2((M_0 D^2 D^1)) \Downarrow_V \lambda x.DD} \dots}{\lambda x.DD \Downarrow_V \lambda x.DD}}{D^2((M_0 D^2 D^1)) \Downarrow_V \lambda x.DD}}{D^3(M_0 D^1 D^2)(M_0 D^2 D^1)} \Downarrow_V \lambda x.DD}{(\lambda x_1 x_2 x_3.DD)(D^3(M_0 D^1 D^2)(M_0 D^2 D^1)) \Downarrow_V \lambda x.DD}$$

□

Lemma 7.1.12. *Let $C[.]$ be a context such that $C[V_0], C[V_1] \in \Lambda^0$.*

- (i) *If $C[V_0] \Downarrow_V$ then $\exists C'[.]$ such that $C[V_0] \Downarrow_V C'[V_0]$ and $C[V_1] \Downarrow_V C'[V_1]$.*
- (ii) *If $C[V_1] \Downarrow_V$ then $\exists C'[.]$ such that $C[V_0] \Downarrow_V C'[V_0]$ and $C[V_1] \Downarrow_V C'[V_1]$.*

Proof. (i) By induction on the derivation proving $C[V_0] \Downarrow_V$.

(var) This case is not possible, since by hypothesis $C[V_0] \in \Lambda^0$.

(lazy) $C[.] \equiv \lambda z.C_0[.]$ and $C[.] \equiv [.]$ are both trivial.

(head) Let $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \geq 1$); if $m \geq 2$ then by Lemma 7.1.11.(i) it is easy to see that $C[V_0] \uparrow_V$. In case $m = 1$, the proof follows by Lemma 7.1.11. Let $C[.] \equiv (\lambda z.C_0[.])C_1[.]...C_m[.]$ ($m \geq 1$), so by induction on $C_1[V_0] \Downarrow_V$ there is $C'_1[.]$ such that $C_1[V_0] \Downarrow_V C'_1[V_0]$ and $C_1[V_1] \Downarrow_V C'_1[V_1]$. The proof follows by induction on $C_0[V_0][C'_1[V_0]/z]C_2[V_0]...C_m[V_0] \Downarrow_V$.

(block) This case is not possible, since by hypothesis $C[V_0] \in \Lambda^0$.

- (ii) By induction on the derivation proving $C[V_1] \Downarrow_{\mathbf{V}}$.
- (var) This case is not possible, since by hypothesis $C[V_0] \in \Lambda^0$.
 - (lazy) $C[.] \equiv \lambda z.C_0[.]$ and $C[.] \equiv [.]$ are both trivial.
 - (head) Let $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \geq 1$), if $m \geq 2$ then by Lemma 7.1.11.(ii) it is easy to see that $C[V_0] \uparrow_{\mathbf{V}}$. In case $m = 1$, the proof follows by Lemma 7.1.11. Let $C[.] \equiv (\lambda z.C_0[.])C_1[.]...C_m[.]$ ($m \geq 1$) so by induction on $C_1[V_1] \Downarrow_{\mathbf{V}}$ there is $C'_1[.]$ such that $C_1[V_0] \Downarrow_{\mathbf{V}} C'_1[V_0]$ and $C_1[V_1] \Downarrow_{\mathbf{V}} C'_1[V_1]$. The proof follows by induction on $C_0[V_1][C'_1[V_1]/z]C_2[V_1]...C_m[V_1] \Downarrow_{\mathbf{V}}$.
 - (block) This case is not possible, since by hypothesis $C[V_0] \in \Lambda^0$. \square

Theorem 7.1.13. $V_0 \approx_{\mathbf{V}} V_1$.

Proof. The proof follows from Lemma 7.1.12. \square

8. Operational Extensionality

8.1 Operational Semantics and Extensionality

In Sect. 1.3, the notion of full extensionality was introduced. A Δ -theory \mathcal{T} is fully extensional when all terms in it can be interpreted as functions, i.e. if and only if the full extensionality principle holds:

$$(\text{EXT}) \quad Mx =_{\mathcal{T}} Nx \Rightarrow M =_{\mathcal{T}} N \quad x \notin \text{FV}(M) \cup \text{FV}(N).$$

Moreover, we proved that a Δ -theory \mathcal{T} is fully extensional if and only if it is closed under η -equality, which is the congruence relation induced by the η -reduction rule:

$$(\eta) \quad \lambda x. Mx \rightarrow_{\eta} M \text{ if and only if } x \notin \text{FV}(M).$$

Now the notion of extensionality will be considered in the particular setting of the Δ -theories that arise from operational semantics, namely the Δ -*operational theories*. In the rest of this section, we will restrict our discussion to Δ -operational theories induced from a formal system such that the Δ -calculus is correct with respect to them. Sometimes, for sake of simplicity, we will skip the prefix Δ .

Let $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$; intuitively a term M has a functional behaviour in \mathbf{O} , or equivalently it can be interpreted as a function, when $M \approx_{\mathbf{O}} \lambda x. Mx$ ($x \notin \text{FV}(M)$). If all terms have a functional behaviour in \mathbf{O} , then it is natural to expect that $\approx_{\mathbf{O}}$ in its turn behaves like the extensional equivalence on functions, i.e. if $\forall P \in \Lambda, MP \approx_{\mathbf{O}} NP$ then $M \approx_{\mathbf{O}} N$. But not all operational theories give a functional interpretation to all terms.

Let us consider, for example, the \mathbf{L} -operational theory: DD and $\lambda x. DD$ have the same applicative behaviour (since, for all $y \in \text{Var}$, both $DDy \uparrow_{\mathbf{L}}$ and $(\lambda x. DDx)y \uparrow_{\mathbf{L}}$), nevertheless they cannot be equated, since the context $[.]$ separates them. In some sense DD , in the \mathbf{L} -operational theory, can be seen as a function too, but of arity 0. But the extensionality principle becomes vacuous if extended to 0-arity functions. So we will introduce the notion of *operational extensionality*.

In order to formalize such a notion, let us introduce the key notion of \mathbf{O} -comparable terms, with respect to an operational theory \mathbf{O} .

Definition 8.1.1. Let $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ be defined by a formal system.

M and N are \mathbf{O} -comparable (notation $M \sim_{\mathbf{O}} N$) when, for each substitution $\mathbf{s} : \text{Var} \rightarrow \Delta$ such that $\mathbf{s}(M), \mathbf{s}(N) \in \Lambda^0$

$$\mathbf{s}(M) \Downarrow_{\mathbf{O}} \text{ if and only if } \mathbf{s}(N) \Downarrow_{\mathbf{O}}.$$

Otherwise, M and N are said to be \mathbf{O} -incomparable (notation $M \not\sim_{\mathbf{O}} N$).

We will speak simply about comparable terms if the involved operational semantics is clear by the context.

Note that being \mathbf{O} -comparable does not imply equivalent. In fact, if $\mathbf{O} \in \{\mathbf{H}, \mathbf{N}, \mathbf{L}, \mathbf{V}\}$ then $\lambda x.xzI \sim_{\mathbf{O}} \lambda x.xzO$ but $\lambda x.xzI \not\approx_{\mathbf{O}} \lambda x.xzO$. Note that $x(DD) \sim_{\mathbf{V}} (\lambda y.xy)(DD)$ are \mathbf{V} -comparable, although $x(DD) \Downarrow_{\mathbf{V}}$ and $(\lambda y.xy)(DD) \Uparrow_{\mathbf{V}}$.

To be comparable in a given operational semantics is a *necessary* condition for two terms being equivalent.

In the rest of this section, $\lambda^*x.Mx$ will denote the fact that $x \notin \text{FV}(M)$. Now we can state formally the *operational functionality principle*.

(OP-FUN) $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ is *op-functional* if and only if,
for all $M \in \Lambda$, $M \sim_{\mathbf{O}} \lambda^*x.Mx$ implies $M \approx_{\mathbf{O}} \lambda^*x.Mx$.

It is easy to see that $M \approx_{\mathbf{O}} \lambda^*x.Mx$ implies $M \sim_{\mathbf{O}} \lambda^*x.Mx$.

Informally, an operational theory \mathbf{O} is *op-extensional* when, for all terms M and N , if they can be interpreted as functions and they have the same applicative behaviour, then $M \approx_{\mathbf{O}} N$.

(OP-EXT) $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ is *op-extensional* if and only if,
for all $M, N \in \Lambda$, for all $x \notin \text{FV}(M) \cup \text{FV}(N)$,
if $M \sim_{\mathbf{O}} \lambda^*x.Mx$, $N \sim_{\mathbf{O}} \lambda^*x.Nx$ and $Mx \approx_{\mathbf{O}} Nx$ then $M \approx_{\mathbf{O}} N$.

It is easy to check that the two principles are equivalent when the operational semantics are correct.

Property 8.1.2. OP-FUN if and only if OP-EXT.

Proof. (\Rightarrow) Let $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ satisfy OP-FUN. For all $x \notin \text{FV}(M) \cup \text{FV}(N)$, let $M \sim_{\mathbf{O}} \lambda^*x.Mx$, $N \sim_{\mathbf{O}} \lambda^*x.Nx$ and $Mx \approx_{\mathbf{O}} Nx$.

OP-FUN implies both $M \approx_{\mathbf{O}} \lambda^*x.Mx$ and $N \approx_{\mathbf{O}} \lambda^*x.Nx$. Moreover, $\lambda x^*.Mx \approx_{\mathbf{O}} \lambda^*x.Nx$, since $Mx \approx_{\mathbf{O}} Nx$ and $\approx_{\mathbf{O}}$ is a congruence; thus

$$M \approx_{\mathbf{O}} \lambda^*x.Mx \approx_{\mathbf{O}} \lambda^*x.Nx \approx_{\mathbf{O}} N.$$

(\Leftarrow) Let $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ satisfy OP-EXT; let $M \sim_{\mathbf{O}} \lambda^*x.Mx$.

$Mz \approx_{\mathbf{O}} (\lambda^*x.Mx)z$ and $\lambda^*x.Mx \approx_{\mathbf{O}} \lambda^*u.(\lambda^*x.Mx)u$, since $\approx_{\mathbf{O}}$ is a Δ -theory; thus, $\lambda^*x.Mx \sim_{\mathbf{O}} \lambda^*u.(\lambda^*x.Mx)u$ and by OP-EXT, $M \approx_{\mathbf{O}} \lambda^*x.Mx$. \square

The notion of op-extensionality can be captured by a suitable reduction rule, parameterized with respect to the considered operational semantics.

Definition 8.1.3. Let $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ be an evaluation relation.

- (i) The $\mathbf{O}\eta$ -reduction $\rightarrow_{\mathbf{O}\eta}$ is the contextual closure of the following rule:
 $\lambda^*x.Mx \rightarrow_{\mathbf{O}\eta} M$ if and only if $M \sim_{\mathbf{O}} \lambda^*x.Mx$.
 $\lambda^*x.Mx$ is a $\mathbf{O}\eta$ -redex and M is its contractum.
- (ii) $M \rightarrow_{\mathbf{O}\eta}^* N$ and $=_{\mathbf{O}\eta}$ are respectively the reflexive and transitive closure of $\rightarrow_{\mathbf{O}\eta}$ and the symmetric, reflexive and transitive closure of $\rightarrow_{\mathbf{O}\eta}$.
- (iii) $M \rightarrow_{\Delta\mathbf{O}\eta} N$ when either $M \rightarrow_{\mathbf{O}\eta} N$ or $M \rightarrow_{\Delta} N$.
- (iv) $M \rightarrow_{\Delta\mathbf{O}\eta}^* N$ and $=_{\Delta\mathbf{O}\eta}$ are respectively the reflexive, symmetric and transitive closure of $\rightarrow_{\Delta\mathbf{O}\eta}$ and the symmetric, reflexive and transitive closure of $\rightarrow_{\Delta\mathbf{O}\eta}$.

A Δ -theory $\approx_{\mathbf{O}}$ is a $\Delta\mathbf{O}\eta$ -theory, when $\approx_{\mathbf{O}}$ is closed under $=_{\mathbf{O}\eta}$, namely $P =_{\mathbf{O}\eta} Q$ implies $P \approx_{\mathbf{O}} Q$.

The relationship between op-extensionality and $\mathbf{O}\eta$ -reduction rule is clarified in the next theorem.

Theorem 8.1.4. Let $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ be correct with respect to the $\lambda\Delta$ -calculus. \mathbf{O} is op-extensional if and only if $\approx_{\mathbf{O}}$ is closed under $=_{\mathbf{O}\eta}$.

Proof. (\Rightarrow) Assume $C[M] =_{\mathbf{O}\eta} C[\lambda^*x.Mx]$, so $M \sim_{\mathbf{O}} \lambda^*x.Mx$ by definition of $\rightarrow_{\mathbf{O}\eta}$. Clearly $\lambda^*u.(\lambda^*x.Mx)u =_{\Delta} \lambda^*u.Mu$, so $\lambda^*u.(\lambda^*x.Mx)u \approx_{\mathbf{O}} \lambda^*u.Mu$, since \mathbf{O} is a Δ -theory, thus $\lambda^*x.Mx \sim_{\mathbf{O}} \lambda^*u.(\lambda^*x.Mx)u$ too. $Mz \approx_{\mathbf{O}} (\lambda^*x.Mx)z$, since \mathbf{O} is a Δ -theory. Hence $M \approx_{\mathbf{O}} \lambda^*x.Mx$ by op-extensionality. Thus, $C[M] \approx_{\mathbf{O}} C[\lambda^*x.Mx]$.

(\Leftarrow) Let $M \sim_{\mathbf{O}} \lambda^*x.Mx$, $N \sim_{\mathbf{O}} \lambda^*x.Nx$ and $Mx \approx_{\mathbf{O}} Nx$, for all $x \notin \text{FV}(M) \cup \text{FV}(N)$. Since \mathbf{O} is a $\Delta\mathbf{O}\eta$ -theory, both $M \approx_{\mathbf{O}} \lambda^*x.Mx$ and $N \approx_{\mathbf{O}} \lambda^*x.Nx$. Moreover, $Mx \approx_{\mathbf{O}} Nx$ implies $(\lambda^*x.Mx) \approx_{\mathbf{O}} (\lambda^*x.Nx)$, so the proof follows by transitivity of $\approx_{\mathbf{O}}$. \square

We will prove that \mathbf{H} , \mathbf{N} , \mathbf{L} and \mathbf{V} are operationally extensional.

First we need to characterize the class of terms M such that M and $\lambda^*x.Mx$ are \mathbf{O} -comparable, when $\mathbf{O} \in \{\mathbf{H}, \mathbf{N}, \mathbf{L}, \mathbf{V}\}$.

Lemma 8.1.5. Both $M \sim_{\mathbf{H}} \lambda^*x.Mx$ and $M \sim_{\mathbf{N}} \lambda^*x.Mx$, for all $M \in \Lambda$.

Proof. This proof is easy. \square

In \mathbf{H} and \mathbf{N} , the operational extensionality corresponds to full extensionality (see Theorems 6.1.10 and 6.2.10).

Theorem 8.1.6. \mathbf{H} and \mathbf{N} are operational extensional Λ -theories.

Proof. Obvious, since full extensionality implies operational extensionality. \square

In \mathbf{L} , it is no longer true that $M \sim_{\mathbf{L}} \lambda^*z.Mz$ holds for all $M \in \Lambda$. In fact, y and $\lambda z.yz$ are not \mathbf{L} -comparable (take the substitution s such that $s(y) = DD$).

Lemma 8.1.7. $M \sim_{\mathbf{L}} \lambda^* z. Mz$ if and only if $M \rightarrow_{\Lambda}^* \lambda x. N$, for some $N \in \Lambda$.

Proof. (\Rightarrow) Assume M does not reduce to an abstraction.

This means that either $M \rightarrow_{\Lambda}^* x\vec{Q}$ or $M \rightarrow_{\Lambda}^* U$, where U is a Λ -unsolvable term of order 0. By correctness, this implies either $M \approx_{\mathbf{L}} x\vec{Q}$ or $M \approx_{\mathbf{L}} U$. Let \mathbf{s} be a substitution such that $\mathbf{s}(x) = DD$, for all x .

In both cases, $\mathbf{s}(x\vec{Q})$ and $\mathbf{s}(U)$ are Λ -unsolvable of order 0, therefore $\mathbf{s}(x\vec{Q}) \uparrow_{\mathbf{L}}$ and $\mathbf{s}(U) \uparrow_{\mathbf{L}}$. On the other hand, $\mathbf{s}(\lambda^* z. Mz) =_{\Lambda} \lambda^* z. \mathbf{s}(M)z$, so $\mathbf{s}(\lambda^* z. Mz) \downarrow_{\mathbf{L}}$, against the hypothesis that $M \sim_{\mathbf{L}} \lambda^* z. Mz$.

(\Leftarrow) Let $M \rightarrow_{\Lambda}^* \lambda x. N$; so $\lambda^* z. Mz \rightarrow_{\Lambda}^* \lambda z. N[z/x] =_{\alpha} \lambda x. N$, therefore by correctness $M \approx_{\mathbf{L}} \lambda^* z. Mz$, which implies $M \sim_{\mathbf{L}} \lambda^* z. Mz$. \square

By the previous lemma, the $\mathbf{L}\eta$ -reduction ($\rightarrow_{\mathbf{L}\eta}$) can be restated, without any explicit reference to the comparability relation \mathbf{L} , as follows:

$$\lambda^* x. Mx \rightarrow_{\mathbf{L}\eta} M \text{ if and only if there is } N \in \Lambda \text{ such that } M \rightarrow_{\Lambda}^* \lambda x. N.$$

Theorem 8.1.8. \mathbf{L} is an operational extensional Λ -theory.

Proof. By Lemma 8.1.7, if $M =_{\mathbf{L}\eta} N$ then $M =_{\Lambda} N$ so $M \approx_{\mathbf{L}} N$ by correctness and the proof follows by Theorem 8.1.4. \square

Now let us consider the call-by-value operational semantics $\approx_{\mathbf{V}}$.

Lemma 8.1.9. $M \sim_{\mathbf{V}} \lambda^* z. Mz$ if and only if M is Γ -valuable.

Proof. (\Rightarrow) Assume that $M \rightarrow_{\Gamma}^* N$ implies $N \notin \Gamma$. This means that either $N \equiv x\vec{Q}$ for some sequence $\|\vec{Q}\| > 0$, or $N \equiv (\lambda x. P)Q\vec{R}$ where $Q \rightarrow_{\Gamma}^* Q'$ and $Q' \notin \Gamma$, or $M \rightarrow_{\Gamma}^* U$ where U is a Γ -unsolvable term of order 0. We prove by induction on N that there is a substitution \mathbf{s} such that $\mathbf{s}(N) \uparrow_{\mathbf{V}}$. Let \mathbf{s} be a substitution such that $\forall x \in \text{Var}, \mathbf{s}(x) = \lambda x. DD \in \Gamma$. The first and the third cases are obvious: both $\mathbf{s}(x\vec{Q})$ and $\mathbf{s}(U)$ are unsolvable of order 0, hence $\mathbf{s}(x\vec{Q}) \uparrow_{\mathbf{V}}$ and $\mathbf{s}(U) \uparrow_{\mathbf{V}}$. In the second case, by induction $\mathbf{s}(Q) \uparrow_{\mathbf{V}}$; so, $\mathbf{s}((\lambda x. P)Q\vec{R}) \uparrow_{\mathbf{V}}$. In all cases, $\mathbf{s}(M) \uparrow_{\mathbf{V}}$.

On the other hand, $\mathbf{s}(\lambda^* z. Mz) = \lambda^* z. \mathbf{s}(M)z$, so $\mathbf{s}(\lambda^* z. Mz) \downarrow_{\mathbf{V}}$ by rule (lazy), against the hypothesis that $M \sim_{\mathbf{V}} \lambda^* z. Mz$.

(\Leftarrow) By definition, $\Gamma = \text{Var} \cup \{\lambda x. M \mid M \in \Lambda\}$. If $M \rightarrow_{\Gamma}^* \lambda x. P'$, for some P' , then $M =_{\Gamma} \lambda^* z. Mz$, so $M \approx_{\mathbf{V}} \lambda^* z. Mz$ and $M \sim_{\mathbf{V}} \lambda^* z. Mz$.

Let $M \rightarrow_{\Gamma}^* x$; for every substitution $\mathbf{s} : \text{Var} \rightarrow \Gamma^0$, it is easy to see that $\mathbf{s}(x) \in \Gamma^0$ and $\mathbf{s}(M) \rightarrow_{\Gamma}^* \mathbf{s}(x)$. By correctness, $\mathbf{s}(M) \downarrow_{\mathbf{V}}$ and $\mathbf{s}(\lambda^* z. Mz) = \lambda^* z. \mathbf{s}(M)z \in \Gamma^0$, so $\mathbf{s}(\lambda^* z. Mz) \downarrow_{\mathbf{V}}$. This implies, by definition, $M \sim_{\mathbf{V}} \lambda^* z. Mz$. \square

By the previous lemma, the $\mathbf{V}\eta$ -reduction ($\rightarrow_{\mathbf{V}\eta}$) can be restated, without any explicit reference to the evaluation relation \mathbf{V} , as follows:

$$\lambda^* x. Mx \rightarrow_{\mathbf{V}\eta} M \text{ if and only if } M \text{ is } \Gamma\text{-valuable.}$$

We prove that \mathbf{V} is an operational extensional Γ -theory by using some denotational tools, in Property 12.1.20.(i). An interesting overview on rewriting and extensionality can be found in [41].

8.1.1 Head-Discriminability

We introduced in Definition 5.0.10, the notion of context discriminating a pair of terms, for a given evaluation relation \mathbf{O} . We will refine such a notion defining \mathbf{O} head-discriminable, if the operational difference between two terms can be tested through an head context. Clearly, this notion is in some sense related to extensionality, since filling a head context $C[.]$ by a closed term M corresponds just to applying M to a suitable sequence of arguments.

Definition 8.1.10. $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ is head discriminable if and only if $M \not\approx_{\mathbf{O}} N$ implies there is a Δ -valuable head context $C[.]$ such that $C[M], C[N] \in \Lambda^0$ and $C[M] \Downarrow_{\mathbf{O}}$ while $C[N] \Uparrow_{\mathbf{O}}$ (or vice versa).

Now let us define a particular class of operational semantics.

Definition 8.1.11. An evaluation relation \mathbf{O} is uniform if and only if $\lambda y.M \sim_{\mathbf{O}} \lambda x.N$ implies $M \sim_{\mathbf{O}} N$.

Informally, the uniformity condition says that a reduction machine either computes under a λ -abstraction or not; in other words, it has either a lazy or a not lazy behaviour, but it cannot mix the two styles of computing. Note that all the semantics we defined in this chapter are uniform. Moreover, we would like to stress that uniformity is quite a natural property to expect for every reasonable operational semantics.

We will prove in the next theorem that, for all the uniform operational semantics, head discriminability implies operational extensionality.

Theorem 8.1.12. If $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ is uniform and head discriminable then it is operationally extensional.

Proof. Let \mathbf{O} be head discriminable, $M \sim_{\mathbf{O}} \lambda^*x.Mx$ and $N \sim_{\mathbf{O}} \lambda^*x.Nx$; we will prove that $M \not\approx_{\mathbf{O}} N$ implies $Mx \not\approx_{\mathbf{O}} Nx$, for all $x \notin \text{FV}(M) \cup \text{FV}(N)$. Since $\approx_{\mathbf{O}}$ is head discriminable, $M \not\approx_{\mathbf{O}} N$ implies that there is a Δ -valuable head context $(\lambda \vec{y}.[.])\vec{P}$ such that $(\lambda \vec{y}.M)\vec{P}, (\lambda \vec{y}.N)\vec{P} \in \Lambda^0$, and $(\lambda \vec{y}.M)\vec{P} \Downarrow_{\mathbf{O}}$ and $(\lambda \vec{y}.N)\vec{P} \Uparrow_{\mathbf{O}}$ (or vice versa).

- Let us consider the case $\|\vec{P}\| = \|\vec{y}\|$. Thus $M[\vec{P}/\vec{y}] \Downarrow_{\mathbf{O}}$ and $N[\vec{P}/\vec{y}] \Uparrow_{\mathbf{O}}$, by correctness. Since $M \sim_{\mathbf{O}} \lambda^*x.Mx$ and $N \sim_{\mathbf{O}} \lambda^*x.Nx$, $\lambda^*x.M[\vec{P}/\vec{y}]x \Downarrow_{\mathbf{O}}$ and $\lambda^*x.N[\vec{P}/\vec{y}]x \Uparrow_{\mathbf{O}}$. Thus the context $(\lambda \vec{y}x.[.])\vec{P}$ discriminates Mx and Nx , namely $Mx \not\approx_{\mathbf{O}} Nx$.

- Let $\|\vec{P}\| > \|\vec{y}\|$. Then $\vec{P} \equiv \vec{P}_1 \vec{P}_2$, where $\|\vec{P}_1\| = \|\vec{y}\|$, and so $M[\vec{P}_1/\vec{y}]\vec{P}_2 \Downarrow_{\mathbf{O}}$ and $N[\vec{P}_1/\vec{y}]\vec{P}_2 \Uparrow_{\mathbf{O}}$. Since $x \notin \text{FV}(M) \cup \text{FV}(N)$, $\approx_{\mathbf{O}}$ is closed under $=_{\Delta}$ and $\|\vec{P}_2\| > 0$, both $(\lambda x.M[\vec{P}_1/\vec{y}]x)\vec{P}_2 \Downarrow_{\mathbf{O}}$ and $(\lambda x.N[\vec{P}_1/\vec{y}]x)\vec{P}_2 \Uparrow_{\mathbf{O}}$, and consequently, by correctness, $(\lambda \vec{y}x.Mx)\vec{P}_1 \vec{P}_2 \Downarrow_{\mathbf{O}}$ and $(\lambda \vec{y}x.Nx)\vec{P}_1 \vec{P}_2 \Uparrow_{\mathbf{O}}$. So the context $(\lambda \vec{y}x.[.])\vec{P}_1 \vec{P}_2$ is a head context discriminating Mx and Nx , namely $Mx \not\approx_{\mathbf{O}} Nx$.
- Let $\|\vec{P}\| < \|\vec{y}\|$. Then $\vec{y} \equiv \vec{y}_1 \vec{y}_2$, where $\|\vec{P}\| = \|\vec{y}_1\|$, and, by Δ -reduction, $\lambda \vec{y}_2.M[\vec{P}/\vec{y}_1] \Downarrow_{\mathbf{O}}$ and $\lambda \vec{y}_2.N[\vec{P}/\vec{y}_1] \Uparrow_{\mathbf{O}}$. By uniformity, this implies there is a substitution s such that $s(M[\vec{P}/\vec{y}_1]) \Downarrow_{\mathbf{O}}$ and $s(N[\vec{P}/\vec{y}_1]) \Uparrow_{\mathbf{O}}$, and consequently there is a substitution s' such that $s'(M) \Downarrow_{\mathbf{O}}$ and $s'(N) \Uparrow_{\mathbf{O}}$. Since $M \sim_{\mathbf{O}} \lambda^*x.Mx$ and $N \sim_{\mathbf{O}} \lambda^*x.Nx$, $\lambda^*x.s'(M)x \Downarrow_{\mathbf{O}}$ and $\lambda^*x.s'(N)x \Uparrow_{\mathbf{O}}$. Let $\text{FV}(M) \cup \text{FV}(N) \subseteq \{z_1, \dots, z_k\}$ for some $k \in \mathbb{N}$, and $C'[] \equiv (\lambda z_1 \dots z_k.[.])s'(z_1) \dots s'(z_k)$. Then $\lambda x.C'[]$ is the context discriminating Mx and Nx , namely $Mx \not\approx_{\mathbf{O}} Nx$.

□

The previous theorem assures us that the notion of operational extensionality we defined is meaningful under the hypothesis of uniformity. In fact, head discriminability means that terms can be discriminated just observing their applicative behaviour, so by considering them as functions, may be of arity 0. All operational theories we considered are head-discriminable.

9. Further Reading

Operational semantics. An algebraic view of structural operational semantics (SOS) can be found in [94]. A formalization of the SOS. operational semantics based on natural deduction is given in [24]. In [77] the reader can find a presentation of the different approaches to structural operational semantics, concentrating on the advantages and disadvantages of each one for reasoning about operational equivalence of programs.

λ -Reduction machines. Some abstract machines for evaluating λ -terms according to different evaluation strategies have been performed. The CUCH machine, defined in [16] and whose implementation is described in [21], performs the **N**-evaluation. Krivine [60] designed an abstract machine performing a variant of the **H**-evaluation, inducing the same operational semantics. The Bologna Optimal Higher Order Machine (BOHM) machine implements an optimal evaluation relation, in the sense that it optimizes the number of λ -reduction performed in parallel [6]. The optimality is reached through a graph representation of terms based on *linear logic* [48]; a survey on the optimal implementation of $\lambda\Lambda$ -calculus can be found in [7].

Part III

Denotational Semantics

10. $\lambda\Delta$ -Models

To study the operational behaviour of λ -terms, we will use the *denotational (mathematical)* approach. A denotational semantics for a language is based on the choice of a space of *semantics values*, or *denotations*, where terms are to be interpreted. Choosing a space with nice mathematical properties can help in proving the semantic properties of terms, since to this aim standard mathematical techniques can be used.

In the next definition, we will give the properties that a structure must satisfy in order to be used as denotations space for the $\lambda\Delta$ -calculus, or, equivalently, to be a *model* for this calculus.

Definition 10.0.1 ($\lambda\Delta$ -Calculus model).

A $\lambda\Delta$ -model is a quadruple $\langle \mathbb{D}, \mathbb{I}, \circ, [\cdot] \rangle$, where:

\mathbb{D} is a set, \circ is a map from \mathbb{D}^2 in \mathbb{D} and $\mathbb{I} \subseteq \mathbb{D}$. Moreover, if \mathbb{E} is the collection of functions (environments) from Var to \mathbb{I} , ranged over by ρ, ρ', \dots then the interpretation function $[\cdot] : \Lambda \times \mathbb{E} \rightarrow \mathbb{D}$ satisfies the following conditions:

1. $[\![x]\!]_\rho = \rho(x)$,
2. $[\![MN]\!]_\rho = [\![M]\!]_\rho \circ [\![N]\!]_\rho$,
3. $[\![\lambda x.M]\!]_\rho \circ d = [\![M]\!]_{\rho[d/x]}$ if $d \in \mathbb{I}$,
4. if $[\![M]\!]_{\rho[d/x]} = [\![M']\!]_{\rho'[d/y]}$ for each $d \in \mathbb{I}$, then $[\![\lambda x.M]\!]_\rho = [\![\lambda y.M']\!]_{\rho'}$,
5. $M \in \Delta$ implies $\forall \rho. [\![M]\!]_\rho \in \mathbb{I}$,

where $\rho[d/x](y) = \text{if } y \equiv x \text{ then } d \text{ else } \rho(y)$.

This definition ensures that a $\lambda\Delta$ -model respects some elementary key properties, namely the interpretation of a term depends only on the behaviour of the environment on the free variables of the term itself, the α -rule is respected, the syntactical substitution is modeled by the environment and the interpretation is contextually closed [59, 50]. Moreover, \mathbb{I} is the semantical counterpart of the set of input values.

Property 10.0.2. Let $\langle \mathbb{D}, \mathbb{I}, \circ, [\cdot] \rangle$ be a $\lambda\Delta$ -model.

- (i) If $\rho(x) = \rho'(x)$, for all $x \in \text{FV}(M)$, then $[\![M]\!]_\rho = [\![M]\!]_{\rho'}$.
- (ii) If $y \notin \text{FV}(M)$ then $[\![M]\!]_{\rho[d/x]} = [\![M[y/x]]\!]_{\rho[d/y]}$, for all $d \in \mathbb{I}$.
- (iii) If $y \notin \text{FV}(M)$ then $[\![\lambda x.M]\!]_\rho = [\![\lambda y.M[y/x]]\!]_\rho$.
- (iv) If $N \in \Delta$ then $[\![M[N/x]]\!]_\rho = [\![M]\!]_{\rho[[N]\rho/x]}$.
- (v) If $[\![M]\!]_\rho = [\![N]\!]_\rho$ then, for every context $C[.]$, $[\![C[M]]\!]_\rho = [\![C[N]]\!]_\rho$.

- Proof.* (i) By induction on M . If $M \in \text{Var}$ then by Definition 10.0.1.1, the proof is immediate. If $M \equiv PQ$ then the proof follows by induction and Definition 10.0.1.2. If $M \equiv \lambda x.N$ then by induction, $\forall d \in \mathbb{I}, \llbracket N \rrbracket_{\rho[d/x]} = \llbracket N \rrbracket_{\rho'[d/x]}$; so $\llbracket \lambda x.N \rrbracket_\rho = \llbracket \lambda x.N \rrbracket_{\rho'}$ by Definition 10.0.1.4.
- (ii) By induction on M . If $M \in \text{Var}$ then by Definition 10.0.1.1, the proof is immediate. If $M \equiv PQ$ then the proof follows by induction and Definition 10.0.1.2. If $M \equiv \lambda z.N$ then by Definition 10.0.1.3, $\forall d' \in \mathbb{I}, \llbracket N \rrbracket_{\rho[d/x][d'/z]} = \llbracket N[y/x] \rrbracket_{\rho[d/y][d'/z]}$ (clearly $\rho[d_0/x_0][d_1/x_1] = \rho[d_1/x_1][d_0/x_0]$); hence, $\llbracket \lambda z.N \rrbracket_{\rho[d/x]} = \llbracket (\lambda z.N)[y/x] \rrbracket_{\rho'[d/y]}$ by Definition 10.0.1.4 and the proof is done.
- (iii) $\forall d \in \mathbb{I}, \llbracket M \rrbracket_{\rho[d/x]} = \llbracket M[y/x] \rrbracket_{\rho[d/y]}$ by the previous point of this property. The proof follows by Definition 10.0.1.4.
- (iv) By induction on M . If $M \in \text{Var}$ then the proof is immediate. If $M \equiv PQ$ then the proof follows by induction.
If $M \equiv \lambda z.P$ then $\forall d \in \mathbb{I}, \llbracket P[N/x] \rrbracket_{\rho[d/z]} = \llbracket P \rrbracket_{\rho[d/z][\llbracket N \rrbracket_{\rho}/x]}$, by Definition 10.0.1.3. Hence, $\llbracket M[N/x] \rrbracket_\rho = \llbracket M \rrbracket_{\rho[\llbracket N \rrbracket_{\rho}/x]}$, by Definition 10.0.1.4.
- (v) By induction on the context $C[.]$.
If $C[.]$ does not contain holes or $C[.] \equiv [.]$, then the proof is obvious.
If $C[.]$ is $C_1[.]C_2[.]$ then the proof follows immediately by induction.
Let $C[.]$ be $\lambda x.C'[.]$; thus, $\forall d \in \mathbb{I}, \llbracket C'[M] \rrbracket_{\rho[d/x]} = \llbracket C'[N] \rrbracket_{\rho[d/x]}$ by Definition 10.0.1.3. The proof follows by Definition 10.0.1.4. \square

The previous property implies that condition 3 of Definition 10.0.1 is the semantics counterpart of the Δ -reduction rule. It says that the interpretation of a term is closed under $=_\Delta$, as proved in the following.

Corollary 10.0.3. *Let $\langle \mathbb{D}, \mathbb{I}, \circ, \llbracket . \rrbracket \rangle$ be a $\lambda\Delta$ -model.*

If $M =_\Delta N$ then $\llbracket M \rrbracket_\rho = \llbracket N \rrbracket_\rho$, for all ρ .

Proof. It is sufficient to prove that if $M \rightarrow_\Delta N$ then $\llbracket M \rrbracket_\rho = \llbracket N \rrbracket_\rho$, for all ρ . Let $Q \in \Delta$; so $\llbracket (\lambda z.P)Q \rrbracket_\rho = \llbracket \lambda z.P \rrbracket_\rho \circ \llbracket Q \rrbracket_\rho = \llbracket P \rrbracket_{\rho[\llbracket Q \rrbracket_\rho/z]} = \llbracket P[Q/z] \rrbracket_\rho$ by the definition of the model and by Property 10.0.2.(iv). The proof follows by Property 10.0.2.(v). \square

Given a $\lambda\Delta$ -model \mathcal{M} , the interpretation function $\llbracket . \rrbracket^{\mathcal{M}}$ induces a denotational semantics on Λ . Namely, two terms M and N are denotationally equivalent in \mathcal{M} (and we write $M \sim_{\mathcal{M}} N$) if and only if:

$$\llbracket M \rrbracket_\rho^{\mathcal{M}} = \llbracket N \rrbracket_\rho^{\mathcal{M}}, \text{ for all environments } \rho.$$

Corollary 10.0.3 ensure us that $\sim_{\mathcal{M}}$ is a Δ -theory; moreover, it implies that if $M =_\Delta N \in \Delta$ then $\forall \rho. \llbracket M \rrbracket_\rho \in \mathbb{I}$.

The denotational semantics induced by a model \mathcal{M} is *correct* with respect to an operational equivalence \approx_O if:

$M \sim_{\mathcal{M}} N$ implies $M \approx_{\mathbf{O}} N$, for all M and N ;

while it is *complete* if:

$M \approx_{\mathbf{O}} N$ implies $M \sim_{\mathcal{M}} N$, for all M and N .

A model is called *fully abstract* [68] with respect to an operational equivalence if the induced denotational semantics is both correct and complete with respect to it.

As we will see in the rest of this section, if our aim is to study an operational equivalence then the correctness is the key point. The next lemma gives us a useful tool for testing the correctness of a model.

Lemma 10.0.4. *Let \mathcal{M} be a $\lambda\Delta$ -model such that:*

$M \Downarrow_{\mathbf{O}}$ and $N \Uparrow_{\mathbf{O}}$ imply $M \not\sim_{\mathcal{M}} N$, for all $M, N \in \Lambda$.

Then \mathcal{M} is correct with respect to the operational equivalence $\approx_{\mathbf{O}}$.

Proof. Let $M \sim_{\mathcal{M}} N$, so by Property 10.0.2.(v), for each context $C[.]$, $C[M] \sim_{\mathcal{M}} C[N]$; hence, by hypothesis, $C[M]$ and $C[N]$ either both $\Downarrow_{\mathbf{O}}$ or both $\Uparrow_{\mathbf{O}}$. Since this is true in particular when $C[M], C[N] \in \Lambda^0$, it follows that $M \approx_{\mathbf{O}} N$. \square

The simplest denotational model is the so-called *term model*. Let $| M |$ be the Δ -equivalence class of M , i.e. $| M | = \{N \mid N =_{\Delta} M\}$; let $|\Lambda|$ be the set all the equivalence classes of Λ with respect to $=_{\Delta}$, while $|\Delta| \subseteq |\Lambda|$ is the set of equivalence classes containing at least one input value. The term model $\mathcal{T}\mathcal{M}(\Delta)$ is the quadruple $<|\Lambda|, |\Delta|, \circ, [\![\cdot]\!]^{\mathcal{T}\mathcal{M}(\Delta)}>$, where \circ is defined as $|M| \circ |N| = |MN|$. The interpretation of a term M in $\mathcal{T}\mathcal{M}(\Delta)$, with $\text{FV}(M) = \{x_1, \dots, x_m\}$, is given by $[\![M]\!]_{\rho} = |M[N_1/x_1]..[N_m/x_m]|$, where $N_i \in \rho(x_i)$ ($1 \leq i \leq m$). It is easy to verify that $\mathcal{T}\mathcal{M}(\Delta)$ satisfies the conditions of Definition 10.0.1.

Theorem 10.0.5. *Let \mathbf{O} be an evaluation relation. If the $\lambda\Delta$ -calculus is correct with respect to $\approx_{\mathbf{O}}$ then $\mathcal{T}\mathcal{M}(\Delta)$ is correct for $\approx_{\mathbf{O}}$.*

Proof. Since the $\lambda\Delta$ -calculus is correct with respect to $\approx_{\mathbf{O}}$, $=_{\Delta}$ implies $\approx_{\mathbf{O}}$. Since $\sim_{\mathcal{T}\mathcal{M}(\Delta)}$ coincides with $=_{\Delta}$, the result follows. \square

It is easy to check that $\mathcal{T}\mathcal{M}(\Lambda)$ is not complete with respect to the operational semantics \mathbf{H} , \mathbf{N} and \mathbf{L} ; while $\mathcal{T}\mathcal{M}(\Gamma)$ is not complete with respect to the operational semantics \mathbf{V} . Just take two unsolvable terms of order 0, e.g. DD and $(\lambda x.xxx)(\lambda x.xxx)$. They are equated in all the operational semantics above, while they are different in both $\mathcal{T}\mathcal{M}(\Lambda)$ and $\mathcal{T}\mathcal{M}(\Gamma)$.

Remark 10.0.6. In case $\Delta = \Lambda$ and $\mathbb{D} = \mathbb{I}$, our definition of the $\lambda\Delta$ -calculus model becomes the well-known definition of a λ -calculus model. But it looks different from the original one, given by Hindley and Longo in [50]. In fact, they ask the interpretation function to satisfy the following six conditions:

1. $\llbracket x \rrbracket_\rho = \rho(x)$;
2. $\llbracket MN \rrbracket_\rho = \llbracket M \rrbracket_\rho \circ \llbracket N \rrbracket_\rho$;
3. $\llbracket \lambda x. M \rrbracket_\rho \circ d = \llbracket M \rrbracket_{\rho[d/x]}$;
4. $\rho(x) = \rho'(x)$ for all $x \in \text{FV}(M) \Rightarrow \llbracket M \rrbracket_\rho = \llbracket M \rrbracket_{\rho'}$;
5. $y \notin \text{FV}(M) \Rightarrow \llbracket \lambda x. M \rrbracket_\rho = \llbracket \lambda y. M[y/x] \rrbracket_\rho$;
6. if $\llbracket M \rrbracket_{\rho[d/x]} = \llbracket M' \rrbracket_{\rho[d/x]}$ for each $d \in \mathbb{I}$, then $\llbracket \lambda x. M \rrbracket_\rho = \llbracket \lambda x. M' \rrbracket_\rho$.

Conditions from 1 to 3 occur identical in our definition. Condition 4 is more restrictive than the corresponding one in Definition 10.0.1, while conditions 5 and 6 ask that the interpretation be closed by α -equivalence and λ -abstraction, respectively. Our definition is shorter, and the strengthening of condition 4 allows one to obtain, as side effect, both the α -equality and the contextual equality (see Property 10.0.2).

It is an useful exercise for the reader to prove that the two definitions are equivalent.

10.1 Filter $\lambda\Delta$ -Models

The idea of *filter $\lambda\Delta$ -model* is based on the notions of *type* and of *type assignment system*. Types represent properties of terms, and they are expressed through the language of the implicative and conjunctive fragment of intuitionistic logic, i.e. the predicate logic with just two connectives, the implication (\rightarrow) and the conjunction (\wedge), and the constant *true* (ω). For historical reasons, the conjunction will be called the *intersection*. Intersection types were first introduced in [27].

Very informally, a term M has the property $\sigma \rightarrow \tau$ if its application to every term N having the property σ has the property τ , and M has the property $\sigma \wedge \tau$ if and only if it has both property σ and property τ . The constant ω represents the property of being a term, so it holds for all terms. A formal description of types is sketched in Sect. 13.1 where types are interpreted as compact elements of suitable domains.

A type assignment system is a set of rules assigning types to terms, starting from a basis, i.e. a function assigning types to variables. A type assignment system can induce a $\lambda\Delta$ -model, where the interpretation of a term is given by the set of types that can be assigned to it.

This kind of model is particularly interesting, since the type assignment system is not only a support for defining the model itself, but it is a tool for reasoning, in a finitary way, on the interpretations of terms in it. So a

filter model, if it is correct with respect to an operational semantics \mathbf{O} , gives standard and powerful techniques for studying the \mathbf{O} -operational behaviour of terms.

Definition 10.1.1 (The intersection type assignment system).

- (i) Let C be a non empty countable set of type constants, containing at least the constant ω (the universal type).

The set $T(C)$ of types is inductively defined as follows:

$$\begin{array}{ll} \sigma \in C & \Rightarrow \sigma \in T(C), \\ \sigma, \tau \in T(C) & \Rightarrow (\sigma \rightarrow \tau) \in T(C), \\ \sigma, \tau \in T(C) & \Rightarrow (\sigma \wedge \tau) \in T(C). \end{array}$$

- (ii) An intersection relation \leq is a preorder relation on $T(C)$, closed under the following rules:

$$\begin{array}{cccc} \frac{}{\sigma \leq \omega} \text{ (a)} & \frac{}{\sigma \leq \sigma \wedge \sigma} \text{ (b)} & \frac{}{\sigma \wedge \tau \leq \sigma} \text{ (c)} & \frac{}{\sigma \wedge \tau \leq \tau} \text{ (c')} \\ \frac{(\sigma \rightarrow \tau) \wedge (\sigma \rightarrow \pi) \leq \sigma \rightarrow (\tau \wedge \pi)}{\sigma \rightarrow \tau \wedge \sigma \rightarrow \pi} \text{ (d)} & & \frac{\sigma \leq \sigma', \tau \leq \tau'}{\sigma \wedge \tau \leq \sigma' \wedge \tau'} \text{ (e)} & \\ \frac{\sigma' \leq \sigma, \tau \leq \tau'}{\sigma \rightarrow \tau \leq \sigma' \rightarrow \tau'} \text{ (f)} & \frac{}{\sigma \rightarrow \omega \leq \omega \rightarrow \omega} \text{ (g)} & \frac{}{\sigma \leq \sigma} \text{ (r)} & \frac{\sigma \leq \rho, \rho \leq \tau}{\sigma \leq \tau} \text{ (t)} \end{array}$$

- (iii) Let \leq be a intersection relation on $T(C)$.

\leq induce a type theory \simeq

$$\sigma \simeq \tau \text{ if and only if } \sigma \leq \tau \text{ and } \tau \leq \sigma.$$

- (iv) A type system ∇ is a triple $< C, \leq_\nabla, I(C) >$, where C is a set of type constants, \leq_∇ is an intersection relation on $T(C)$ and $I(C) \subseteq T(C)$ is a set of input types with respect to \leq_∇ ; namely, it is not empty and it is closed under the following conditions:

1. $\sigma \in I(C)$ and $\sigma \simeq_\nabla \tau$ imply $\tau \in I(C)$,
2. $\sigma \in I(C)$ and $\tau \notin I(C)$ imply $\sigma \leq_\nabla \tau$.

- (v) Given a type system ∇ , the corresponding type assignment system \vdash_∇ is a formal system proving statements of the shape:

$$B \vdash_\nabla M : \sigma$$

where M is a term, $\sigma \in T(C)$ and B is a basis, i.e a function from Var to $I(C)$.

$B[\sigma/x]$ denotes the basis such that $B[\sigma/x](y) = \text{if } y \equiv x \text{ then } \sigma \text{ else } B(y)$.

The type assignment system consists of the following rules:

$$\begin{array}{c}
 \frac{}{B[\sigma/x] \vdash_{\nabla} x : \sigma}^{(var)} \quad \frac{}{B \vdash_{\nabla} M : \omega}^{(\omega)} \\
 \frac{B[\sigma/x] \vdash_{\nabla} M : \tau \quad \sigma \in I(C) \quad B \vdash_{\nabla} M : \sigma \rightarrow \tau \quad B \vdash_{\nabla} N : \sigma}{B \vdash_{\nabla} \lambda x. M : \sigma \rightarrow \tau}^{(\rightarrow I)} \quad \frac{\sigma \in I(C) \quad B \vdash_{\nabla} M : \sigma \rightarrow \tau \quad B \vdash_{\nabla} N : \sigma}{B \vdash_{\nabla} MN : \tau}^{(\rightarrow E)} \\
 \frac{B \vdash_{\nabla} M : \sigma \quad B \vdash_{\nabla} M : \tau}{B \vdash_{\nabla} M : \sigma \wedge \tau}^{(\wedge I)} \quad \frac{B \vdash_{\nabla} M : \sigma \quad \sigma \leq_{\nabla} \tau}{B \vdash_{\nabla} M : \tau}^{(\leq_{\nabla})} \\
 \frac{B \vdash_{\nabla} M : \sigma \wedge \tau}{B \vdash_{\nabla} M : \sigma}^{(\wedge E_l)} \quad \frac{B \vdash_{\nabla} M : \sigma \wedge \tau}{B \vdash_{\nabla} M : \tau}^{(\wedge E_r)}
 \end{array}$$

Note that rules $(\wedge E_l)$ and $(\wedge E_r)$ are redundant, since the rule (\leq_{∇}) .

In the definition of intersection relation, note the controvariance of \leq_{∇} with respect to the left-hand argument of \rightarrow in rule (f) . Moreover, note that the rule $(\rightarrow E)$ imposes a condition on the type of the argument of application, namely that it belongs to the set of input types.

A *derivation* is a tree whose nodes are instances of rules, such that the premises of a rule are the consequences of its son nodes. If the root of a derivation d has as consequence the statement $B \vdash_{\nabla} M : \sigma$, we will say that d proves $B \vdash_{\nabla} M : \sigma$. The notion of *subderivation* then corresponds to the notion of subtree. We write $B \vdash_{\nabla} M : \sigma$ when *there is a derivation proving* $B \vdash_{\nabla} M : \sigma$; moreover, $B \vdash_{\nabla} M : \sigma$ is called a *typing*. If $B \vdash_{\nabla} M : \sigma$ and $B \vdash_{\nabla} N : \sigma$, then we will say that the terms M and N have a typing in common. $B \not\vdash_{\nabla} M : \sigma$ will denote that there are not derivations proving $B \vdash_{\nabla} M : \sigma$.

Example 10.1.2. Let $\mathfrak{U} = < \{\omega\}, \leq_{\mathfrak{U}}, I_{\mathfrak{U}} >$ where $I_{\mathfrak{U}} = T(\{\omega\})$ and $\leq_{\mathfrak{U}}$ is the least intersection relation such that $\omega \leq_{\mathfrak{U}} \sigma$, for all $\sigma \in T(\{\omega\})$.

It is easy to check that $\sigma \simeq_{\mathfrak{U}} \omega$, for all $\sigma \in T(\{\omega\})$; in particular, $\omega \simeq_{\mathfrak{U}} \omega \rightarrow \omega$. Let B_{ω} be the basis such that $B_{\omega}(x) = \omega$, for all $x \in \text{Var}$; hence, the following derivation proves $B_{\omega} \vdash_{\mathfrak{U}} (\lambda x. xx)(\lambda x. xx) : \omega \rightarrow \omega$:

$$\frac{\frac{}{B_{\omega} \vdash_{\mathfrak{U}} DD : \omega}^{(\omega)} \quad \omega \leq_{\mathfrak{U}} \omega \rightarrow \omega}{B_{\omega} \vdash_{\mathfrak{U}} DD : \omega \rightarrow \omega}^{(\leq_{\mathfrak{U}})}$$

It is easy to check that, for every set C of type-constants, it is correct to choose $I(C) = T(C)$.

Lemma 10.1.3. Let ∇ be the type system $< C, \leq_{\nabla}, I(C) >$.

- (i) If $\pi \in I(C)$ and $\sigma \leq_{\nabla} \pi$ then $\sigma \in I(C)$.
- (ii) If $\sigma \in I(C)$ then $\sigma \wedge \tau \in I(C)$, for all $\tau \in T(C)$.
- (iii) If $\pi \notin I(C)$ and $\pi \leq_{\nabla} \sigma$ then $\sigma \notin I(C)$.

- Proof.* (i) Assume $\pi \in I(C)$ and $\sigma \leq_{\nabla} \pi$ and $\sigma \notin I(C)$; so $\pi \leq_{\nabla} \sigma$ by condition 2 on the set of input types. Hence $\sigma \simeq_{\nabla} \pi$; by condition 1 on the set of input types, this is absurd.
- (ii) $\sigma \wedge \tau \leq_{\nabla} \sigma$, by rule (c) of intersection relations. The proof follows by the result proved in the previous point.
- (iii) Similar to the proof of point (i). \square

Note that if $I(C) \neq T(C)$ then $\omega \notin I(C)$, by the previous lemma.

In the next lemma some useful equivalences between types, true in all type theories, are proved.

Lemma 10.1.4. *Let ∇ be the type system $\langle C, \leq_{\nabla}, I(C) \rangle$.*

- (i) $\sigma \simeq_{\nabla} \sigma \wedge \sigma$;
- (ii) $\omega \wedge \sigma \simeq_{\nabla} \sigma$;
- (iii) $\sigma \rightarrow (\tau \wedge \pi) \simeq_{\nabla} (\sigma \rightarrow \tau) \wedge (\sigma \rightarrow \pi)$;
- (iv) $\sigma \rightarrow \omega \simeq_{\nabla} \omega \rightarrow \omega$;
- (v) $(\sigma \wedge \tau) \wedge \pi \simeq_{\nabla} \sigma \wedge (\tau \wedge \pi)$;
- (vi) $\sigma \wedge \tau \simeq_{\nabla} \tau \wedge \sigma$;
- (vii) $\pi \leq_{\nabla} \sigma$ and $\pi \leq_{\nabla} \tau$ if and only if $\pi \leq_{\nabla} \sigma \wedge \tau$.

- Proof.* (i) By rules (b) and (c) of the definition of intersection relations.
- (ii) By the rule (c) of definition of intersection relations, $\omega \wedge \sigma \leq_{\nabla} \sigma$. On the other side, $\sigma \leq_{\nabla} \sigma \wedge \sigma \leq_{\nabla} \omega \wedge \sigma$, by rules (a), (b) and (e) and by the reflexivity of \leq_{∇} . The proof follows by (t).
- (iii) Since rule (d), we need just to prove $\sigma \rightarrow (\tau \wedge \pi) \leq_{\nabla} (\sigma \rightarrow \tau) \wedge (\sigma \rightarrow \pi)$.
 $\sigma \rightarrow (\tau \wedge \pi) \leq_{\nabla} \sigma \rightarrow \tau$ and $\sigma \rightarrow (\tau \wedge \pi) \leq_{\nabla} \sigma \rightarrow \pi$ by rules (f), (c) and (c'); thus, $\sigma \rightarrow (\tau \wedge \pi) \leq_{\nabla} (\sigma \rightarrow (\tau \wedge \pi)) \wedge (\sigma \rightarrow (\tau \wedge \pi)) \leq_{\nabla} (\sigma \rightarrow \tau) \wedge (\sigma \rightarrow \pi)$ by rules (b) and (e). The proof follows by (t).
- (iv) Since rule (g), we need just to prove $\omega \rightarrow \omega \leq_{\nabla} \sigma \rightarrow \omega$, which follows by rules (a) and (f), and by the reflexivity of \leq_{∇} .
- (v) Let $\mu_0 \equiv (\sigma \wedge \tau) \wedge \pi$ and $\mu_1 \equiv \sigma \wedge (\tau \wedge \pi)$.
 $\mu_0, \mu_1 \leq_{\nabla} \sigma, \tau, \pi$ by rules (c), (c') and (t), hence $\mu_0 \wedge (\mu_0 \wedge \mu_0) \leq_{\nabla} \mu_1$ by rule (e). Thus $\mu_0 \leq_{\nabla} \mu_0 \wedge (\mu_0 \wedge \mu_0)$ by rule (b) and then $\mu_0 \leq_{\nabla} \mu_1$ by rule (t). The reverse relation can be proved in a symmetric way.
- (vi) Both $\sigma \wedge \tau \leq_{\nabla} \sigma, \tau$ and $\tau \wedge \sigma \leq_{\nabla} \sigma, \tau$ by rules (c) and (c'); hence $(\sigma \wedge \tau) \wedge (\sigma \wedge \tau) \leq_{\nabla} \tau \wedge \sigma$, and then $\sigma \wedge \tau \leq_{\nabla} \tau \wedge \sigma$ by rules (b) and (e). The reverse relation can be proved in a symmetric way.
- (vii) By rules (b), (c), (c') and (e). \square

In order to decrease the number of parenthesis in types, we will use the following precedence rules between connectives: \wedge binds stronger than \rightarrow , moreover \rightarrow associates to the right. For example, $\sigma \rightarrow \tau \wedge \rho$, $\sigma \rightarrow \tau \rightarrow \rho$ and $\sigma \wedge \tau \rightarrow \rho$ stand respectively for $\sigma \rightarrow (\tau \wedge \rho)$, $\sigma \rightarrow (\tau \rightarrow \rho)$ and $(\sigma \wedge \tau) \rightarrow \rho$.

Since the result of the previous lemma, at point (v), when no ambiguity can arise, we will use $\sigma \wedge \tau \wedge \rho$ for denoting both $\sigma \wedge (\tau \wedge \rho)$ and $\sigma \wedge (\tau \wedge \rho)$.

The notion of *legal type theory*, given in the next definition, is a key one, since we will prove that to be legal is a necessary condition for a type theory to induce a $\lambda\Delta$ -model.

Definition 10.1.5. Let ∇ be the type system $\langle C, \leq_\nabla, I(C) \rangle$.

∇ is legal if and only if for all $\sigma \in I(C)$ and $\tau \not\simeq_\nabla \omega$:

$$\begin{aligned} (\sigma_1 \rightarrow \tau_1) \wedge \dots \wedge (\sigma_n \rightarrow \tau_n) \leq_\nabla \sigma \rightarrow \tau \quad (1 \leq n) \text{ implies} \\ \exists \{i_1, \dots, i_k\} \subseteq \{1, \dots, n\} \text{ such that } (\sigma_{i_1} \wedge \dots \wedge \sigma_{i_k}) \geq_\nabla \sigma \\ \text{and } (\tau_{i_1} \wedge \dots \wedge \tau_{i_k}) \leq_\nabla \tau. \end{aligned}$$

Let ∇ be a type system $\langle C, \leq_\nabla, I(C) \rangle$ such that $I(C) = T(C)$ and \leq_∇ is the least inclusion relation: ∇ is legal.

In case of a legal type theory, rule (f) of the intersection relation defined in Definition 10.1.1 becomes a double implication. This will be useful in the following for proving properties of $\lambda\Delta$ -models.

Property 10.1.6. Let $\nabla = \langle C, \leq_\nabla, I(C) \rangle$ be legal.

If $\tau \not\simeq_\nabla \omega$ and $\sigma \in I(C)$ then

$$\sigma' \rightarrow \tau' \leq_\nabla \sigma \rightarrow \tau \text{ if and only if } \sigma \leq_\nabla \sigma' \text{ and } \tau' \leq_\nabla \tau.$$

Proof. By Definition 10.1.5 and by rule (f) of Definition 10.1.1.(ii). \square

In order to show that a legal type theory induces a $\lambda\Delta$ -model, first some syntactical properties of a type assignment system induced by a legal type theory must be proved.

Lemma 10.1.7. Let $\nabla = \langle C, \leq_\nabla, I(C) \rangle$ be legal.

- (i) $B \vdash_\nabla M : \sigma$ and $x \notin \text{FV}(M)$ imply $\forall \tau \in I(C), B[\tau/x] \vdash_\nabla M : \sigma$.
- (ii) $B \vdash_\nabla x : \sigma$ if and only if $B(x) \leq_\nabla \sigma$.
- (iii) $B[\tau/x] \vdash_\nabla M : \sigma$ and $\pi \leq_\nabla \tau$ imply $B[\pi/x] \vdash_\nabla M : \sigma$.
- (iv) If $B \vdash_\nabla \lambda x.M : \sigma$ then either $\sigma \simeq_\nabla \omega$ or $\sigma \geq_\nabla \pi_1 \wedge \dots \wedge \pi_n$ ($n \geq 1$), where $\pi_i \simeq_\nabla \mu_i \rightarrow \nu_i$, $\mu_i \in I(C)$ and $B[\mu_i/x] \vdash_\nabla M : \nu_i$ ($1 \leq i \leq n$).
- (v) If $B \vdash_\nabla MN : \sigma$ then either $\sigma \simeq_\nabla \omega$ or $\sigma \geq_\nabla \pi_1 \wedge \dots \wedge \pi_n$ ($n \geq 1$) such that $B \vdash_\nabla M : \tau_i \rightarrow \pi_i$ and $B \vdash_\nabla N : \tau_i$, for some $\tau_1, \dots, \tau_n \in I(C)$ ($1 \leq i \leq n$).
- (vi) Let $\tau \not\simeq_\nabla \omega$ and $\sigma \in I(C)$.
 $B \vdash_\nabla \lambda x.M : \sigma \rightarrow \tau$ if and only if $B[\sigma/x] \vdash_\nabla M : \tau$.
- (vii) Let $\sigma \not\simeq_\nabla \omega$. $B \vdash_\nabla MN : \sigma$ if and only if $B \vdash_\nabla M : \tau \rightarrow \sigma$ and $B \vdash_\nabla N : \tau$, for some $\tau \in I(C)$.

Proof. (i) Immediate, from the definition of the type assignment system.

(ii) (\Leftarrow) By rules (var) and (\leq_∇).

(\Rightarrow) By induction on the derivation. Note that ($\rightarrow I$) and ($\rightarrow E$) cannot be used.

(iii) By induction on the derivation. Note that $\pi \in I(C)$ by Lemma 10.1.3.(i).

- (iv) By induction on the derivation proving $B \vdash_{\nabla} \lambda x.M : \sigma$. If the last applied rule is either (ω) or $(\rightarrow I)$, then the proof is trivial. In case the last applied rule is (\leq_{∇}) , the proof follows from the inductive hypothesis and the transitivity of \leq_{∇} . In case the last applied rule is $(\wedge I)$, the proof follows from the inductive hypothesis. The case of rule $(\rightarrow E)$ is not possible.
- (v) By induction on the derivation proving $B \vdash_{\nabla} MN : \sigma$. If the last applied rule is (ω) or $(\rightarrow E)$, then the proof is trivial. In case the last applied rule is (\leq_{∇}) , the proof follows from the inductive hypothesis and the transitivity of \leq_{∇} . In case the last applied rule is $(\wedge I)$, the proof follows from the inductive hypothesis. $(\rightarrow I)$ cannot be the last applied rule.
- (vi) (\Leftarrow) By rule $(\rightarrow I)$.

(\Rightarrow) $B \vdash_{\nabla} \lambda x.M : \sigma \rightarrow \tau$ implies $(\mu_1 \rightarrow \nu_1) \wedge \dots \wedge (\mu_m \rightarrow \nu_m) \leq_{\nabla} \sigma \rightarrow \tau$ for some $m \geq 1$, where $\mu_i \in I(C)$ and $B[\mu_i/x] \vdash_{\nabla} M : \nu_i$ ($1 \leq i \leq m$), by point (iv). Since the type theory is legal, $\exists \{i_1, \dots, i_k\} \subseteq \{1, \dots, m\}$ such that $\mu_{i_1} \wedge \dots \wedge \mu_{i_k} \geq_{\nabla} \sigma$ and $\nu_{i_1} \wedge \dots \wedge \nu_{i_k} \leq_{\nabla} \tau$. By Lemma 10.1.3.(ii) $\mu_{i_1} \wedge \dots \wedge \mu_{i_k} \in I(C)$, so $B[\mu_{i_j}/x] \vdash_{\nabla} M : \nu_{i_j}$ ($1 \leq j \leq k$) imply $B[\mu_{i_1} \wedge \dots \wedge \mu_{i_k}/x] \vdash_{\nabla} M : \nu_{i_1} \wedge \dots \wedge \nu_{i_k}$, by rule $(\wedge I)$ and by point (iii) of this lemma. Again by point (iii) of this lemma, $B[\sigma/x] \vdash_{\nabla} M : \nu_{i_1} \wedge \dots \wedge \nu_{i_k}$ and by rule (\leq_{∇}) we can conclude $B[\sigma/x] \vdash_{\nabla} M : \tau$.
- (vii) (\Leftarrow) By rule $(\rightarrow E)$. Note that $\tau \in I(C)$ is a necessary hypothesis.

(\Rightarrow) By point (v), if $B \vdash_{\nabla} MN : \sigma$ and $\sigma \not\simeq_{\nabla} \omega$ then $\sigma \geq_{\nabla} \mu_1 \wedge \dots \wedge \mu_m$ for some $m \geq 1$; moreover, $B \vdash_{\nabla} M : \sigma_i \rightarrow \mu_i$ and $B \vdash_{\nabla} N : \sigma_i$, for some $\sigma_i \in I(C)$ ($1 \leq i \leq m$). Hence $\sigma_1 \wedge \dots \wedge \sigma_m \in I(C)$ by Lemma 10.1.3.(ii). By rule $(\wedge I)$, $B \vdash_{\nabla} M : (\sigma_1 \rightarrow \mu_1) \wedge \dots \wedge (\sigma_m \rightarrow \mu_m)$. Note that

$$\begin{aligned} (\sigma_1 \rightarrow \mu_1) \wedge \dots \wedge (\sigma_m \rightarrow \mu_m) &\leq_{\nabla} (\sigma_1 \wedge \dots \wedge \sigma_m \rightarrow \mu_1) \wedge \dots \wedge (\sigma_1 \wedge \dots \wedge \sigma_m \rightarrow \mu_m) \\ &\leq_{\nabla} \sigma_1 \wedge \dots \wedge \sigma_m \rightarrow \mu_1 \wedge \dots \wedge \mu_m \end{aligned}$$

since $\sigma_1 \wedge \dots \wedge \sigma_m \leq_{\nabla} \sigma_i$ ($1 \leq i \leq m$) and by Lemma 10.1.4.(iii).

So $B \vdash_{\nabla} M : \sigma_1 \wedge \dots \wedge \sigma_m \rightarrow \mu_1 \wedge \dots \wedge \mu_m$ by rule (\leq_{∇}) ; moreover, $B \vdash_{\nabla} N : \sigma_1 \wedge \dots \wedge \sigma_m$, by rule $(\wedge I)$.

Since $B \vdash_{\nabla} M : (\sigma_1 \wedge \dots \wedge \sigma_m) \rightarrow \sigma$ by rule (\leq_{∇}) , the proof is done. \square

Note that the notion of legality is essential in the proof of point (vi) of Lemma 10.1.7.

Remark 10.1.8. By Lemma 10.1.7.(i), it follows that the derivation of a type for a closed term is independent from the basis, i.e. if M is a closed term then $B \vdash_{\nabla} M : \sigma$ implies that $B' \vdash_{\nabla} M : \sigma$, for all B' .

Now we are ready to introduce the basic ingredients for defining a filter model.

Definition 10.1.9. Let ∇ be the type system $\langle C, \leq_{\nabla}, I(C) \rangle$.

- (i) A filter f on ∇ is any set containing ω and closed under \wedge and \leq_{∇} , namely:

- $\mu, \nu \in f$ implies $\mu \wedge \nu \in f$;
- $\mu \in f$ and $\mu \leq_{\nabla} \tau$ imply $\tau \in f$.

Let $\mathcal{F}(\nabla)$ be the set of all filters on ∇ , and let $\mathcal{I}(\nabla)$ be the set of filters containing at least one type belonging to $I(C)$.

- (ii) Let S be a set of types; $\uparrow S$ is the filter obtained from S by closing it under \wedge and \leq_{∇} , i.e. the least filter containing S .
- (iii) Let \circ_{∇} be the binary operation defined on $\mathcal{F}(\nabla)$ in the following way:

$$f_1 \circ_{\nabla} f_2 = \uparrow \{\omega\} \cup \{\tau \mid \sigma \rightarrow \tau \in f_1 \text{ and } \sigma \in f_2 \text{ and } \sigma \in I(C)\}.$$

Note that $f \in \mathcal{I}(\nabla)$ and $\sigma \notin I(C)$ imply $\sigma \in f$ by the conditions on the set of input types.

The following lemma shows that the definition of \circ_{∇} is correct.

Lemma 10.1.10. Let $\nabla = \langle C, \leq_{\nabla}, I(C) \rangle$.

If $f_1, f_2 \in \mathcal{F}(\nabla)$ then $f_1 \circ_{\nabla} f_2 \in \mathcal{F}(\nabla)$.

Proof. Clearly $\omega \in \mathcal{F}(\nabla)$.

Let us prove that $f_1 \circ_{\nabla} f_2$ is closed under intersection.

Let $\alpha, \beta \in f_1 \circ_{\nabla} f_2$; so, there are $\sigma, \tau \in f_2 \cap I(C)$ such that $\sigma \rightarrow \alpha, \tau \rightarrow \beta \in f_1$. Both f_1 and f_2 are filters, thus $\sigma \wedge \tau \in f_2 \cap I(C)$ and $\sigma \wedge \tau \rightarrow \alpha, \sigma \wedge \tau \rightarrow \beta \in f_1$ (since $\sigma \rightarrow \alpha \leq_{\nabla} \sigma \wedge \tau \rightarrow \alpha$ and $\tau \rightarrow \beta \leq_{\nabla} \sigma \wedge \tau \rightarrow \beta$). Hence $(\sigma \wedge \tau \rightarrow \alpha) \wedge (\sigma \wedge \tau \rightarrow \beta) \in f_1$ and $\sigma \wedge \tau \rightarrow \alpha \wedge \beta \in f_1$, by Lemma 10.1.4.(iii). Thus $\sigma \wedge \tau \in f_2 \cap I(C)$ implies $\alpha \wedge \beta \in f_1 \circ_{\nabla} f_2$, by definition of \circ_{∇} .

Now let us prove that $\alpha \in f_1 \circ_{\nabla} f_2$ implies $\tau \in f_1 \circ_{\nabla} f_2$, for every $\tau \geq_{\nabla} \alpha$. In fact, $\alpha \in f_1 \circ_{\nabla} f_2$ implies that there is $\sigma \in f_2 \cap I(C)$ such that $\sigma \rightarrow \alpha \in f_1$. $\sigma \rightarrow \alpha \leq_{\nabla} \sigma \rightarrow \tau$ and f_1 is a filter imply that $\sigma \rightarrow \tau \in f_1$, thus $\tau \in f_1 \circ_{\nabla} f_2$. \square

The interpretation function will associate to every term all the types that can be assigned to it.

Let $\nabla = \langle C, \leq_{\nabla}, I(C) \rangle$.

$[\cdot]^{F(\nabla)} : \Lambda \times (\text{Var} \rightarrow \mathcal{I}(\nabla)) \rightarrow \mathcal{F}(\nabla)$ is the *interpretation function*, defined as follows:

$$[M]_{\rho}^{F(\nabla)} = \{\sigma \in T(C) \mid \exists B \propto \rho \text{ such that } B \vdash_{\nabla} M : \sigma\},$$

where $B \propto \rho$ means that $\forall z \in \text{Var} \ B(z) \in \rho(z)$, and it can be read as “ B agree with ρ ”.

Remember that a basis B is a function from Var to $I(C)$.

Theorem 10.1.11 (Filter $\lambda\Delta$ -models).

Let $\nabla = \langle C, \leq_{\nabla}, I(C) \rangle$ be a legal type system, and let $M \in \Delta$ implies $[M]_{\rho} \in \mathcal{I}(\nabla)$, for all environments ρ . Then $\langle \mathcal{F}(\nabla), \mathcal{I}(\nabla), \circ_{\nabla}, [\cdot]^{F(\nabla)} \rangle$ is a $\lambda\Delta$ -model.

Proof. It is easy to see that $\llbracket M \rrbracket_{\rho}^{\mathcal{F}(\nabla)}$ is a filter, for all terms M . The proof is carried out by verifying the conditions of Definition 10.0.1.

1. We will prove $\rho(x) = \{\sigma \in T(C) \mid \exists B \propto \rho \text{ such that } B \vdash_{\nabla} x : \sigma\}$.

Let $\sigma \in I(C)$; so, $\sigma \in \rho(x)$ if and only if $\exists B \propto \rho$ such that $B(x) = \sigma$, which means $B \vdash_{\nabla} x : \sigma$ by rule (var). Note that $\rho(x) \in \mathcal{I}(\nabla)$ implies that there is $\mu \in I(C)$ such that $\mu \in \rho(x)$; if $\sigma \notin I(C)$ then $\sigma \in \rho(x)$ since $\mu \leq_{\nabla} \sigma$ by conditions on the input types, hence $B \vdash_{\nabla} x : \mu$ implies $B \vdash_{\nabla} x : \sigma$ by rule (\leq_{∇}).

2. Let $B_0 \cup B_1$ be the basis such that $(B_0 \cup B_1)(x) = B_0(x) \wedge B_1(x)$.

By using Lemmas 10.1.7.(iii) and 10.1.7.(vii):

$$\begin{aligned} \llbracket MN \rrbracket_{\rho}^{\mathcal{F}(\nabla)} &= \{\sigma \in T(C) \mid \exists B \propto \rho \text{ such that } B \vdash_{\nabla} MN : \sigma\} = \\ &\uparrow \{\omega\} \cup \left\{ \sigma \not\leq_{\nabla} \omega \mid \begin{array}{l} \exists B \propto \rho \\ \exists \tau \in I(C) \end{array} \text{ such that both } \begin{array}{l} B \vdash_{\nabla} N : \tau \text{ and} \\ B \vdash_{\nabla} M : \tau \rightarrow \sigma \end{array} \right\} = \\ &\uparrow \{\omega\} \cup \left\{ \sigma \not\leq_{\nabla} \omega \mid \begin{array}{l} \exists B_0, B_1 \propto \rho \\ \exists \tau \in I(C) \end{array} \text{ such that } \begin{array}{l} B_0 \cup B_1 \propto \rho \text{ and} \\ B_0 \vdash_{\nabla} N : \tau \text{ and} \\ B_1 \vdash_{\nabla} M : \tau \rightarrow \sigma \end{array} \right\} = \\ &\uparrow \{\omega\} \cup \left\{ \sigma \not\leq_{\nabla} \omega \mid \begin{array}{l} \exists B_0, B_1 \propto \rho \\ \exists \tau \in I(C) \end{array} \text{ such that } \begin{array}{l} B_0 \vdash_{\nabla} N : \tau \text{ and} \\ B_1 \vdash_{\nabla} M : \tau \rightarrow \sigma \end{array} \right\} = \\ &\uparrow \{\omega\} \cup \left\{ \sigma \not\leq_{\nabla} \omega \mid \begin{array}{l} \exists \tau \in I(C) \text{ such that} \\ \tau \in \llbracket N \rrbracket_{\rho}^{\mathcal{F}(\nabla)} \text{ and} \\ \tau \rightarrow \sigma \in \llbracket M \rrbracket_{\rho}^{\mathcal{F}(\nabla)} \end{array} \right\} = \\ &\llbracket M \rrbracket_{\rho}^{\mathcal{F}(\nabla)} \circ_{\nabla} \llbracket N \rrbracket_{\rho}^{\mathcal{F}(\nabla)} \end{aligned}$$

3. Let $f \in \mathcal{I}(\nabla)$; so by Lemma 10.1.7.(vi)

$$\begin{aligned} \llbracket \lambda x. M \rrbracket_{\rho}^{\mathcal{F}(\nabla)} \circ_{\nabla} f &= \\ &\uparrow \{\omega\} \cup \{\tau \not\leq_{\nabla} \omega \mid \sigma \in f \text{ and } \sigma \in I(C) \text{ and } \sigma \rightarrow \tau \in \llbracket \lambda x. M \rrbracket_{\rho}^{\mathcal{F}(\nabla)}\} = \\ &\uparrow \{\omega\} \cup \{\tau \not\leq_{\nabla} \omega \mid \sigma \in f \cap I(C) \text{ and } \exists B \propto \rho \text{ such that } B \vdash_{\nabla} \lambda x. M : \sigma \rightarrow \tau\} = \\ &\uparrow \{\omega\} \cup \{\tau \not\leq_{\nabla} \omega \mid \exists B \propto \rho \text{ such that } B[\sigma/x] \vdash_{\nabla} M : \tau \text{ and } \sigma \in I(C) \text{ and } \sigma \in f\} \end{aligned}$$

Thus the proof is done, since

$$\llbracket M \rrbracket_{\rho[f/x]}^{\mathcal{F}(\nabla)} = \{\tau \in T(C) \mid \exists B' \propto \rho[f/x] \text{ such that } B' \vdash_{\nabla} M : \tau\}.$$

4. We assume that $\llbracket M \rrbracket_{\rho[d/x]}^{\mathcal{F}(\nabla)} = \llbracket M' \rrbracket_{\rho'[d/y]}^{\mathcal{F}(\nabla)}$, for all $d \in \mathbb{I}$; namely $\forall d \in \mathbb{I}$

$\{\sigma \mid \exists B \propto \rho[d/x] \text{ s.t. } B \vdash_{\nabla} M : \sigma\} = \{\sigma \mid \exists B' \propto \rho'[d/y] \text{ s.t. } B' \vdash_{\nabla} M' : \sigma\}$, in particular, for all $\tau \in I(C)$

$$\{\sigma \mid \exists B \propto \rho[\uparrow \tau/x] \text{ s.t. } B \vdash_{\nabla} M : \sigma\} = \{\sigma \mid \exists B' \propto \rho'[\uparrow \tau/y] \text{ s.t. } B' \vdash_{\nabla} M' : \sigma\}.$$

Hence, it is easy to see that for all $\tau \in I(C)$:

$\exists B \propto \rho$ s.t. $B[\tau/x] \vdash_{\nabla} M : \sigma$ if and only if $\exists B' \propto \rho'$ s.t. $B'[\tau/y] \vdash_{\nabla} M' : \sigma$.

Let us assume $\pi \in \llbracket \lambda x.M \rrbracket_{\rho}^{\mathcal{F}(\nabla)}$, so $\exists B \propto \rho$ such that $B \vdash_{\nabla} \lambda x.M : \pi$ by the definition of the interpretation function. By Lemma 10.1.7.(iv), $\pi \geq_{\nabla} (\mu_1 \rightarrow \nu_1) \wedge \dots \wedge (\mu_m \rightarrow \nu_m)$ for some $m \geq 1$, where $\mu_i \in I(C)$ and $B[\mu_i/x] \vdash_{\nabla} M : \nu_i$ ($1 \leq i \leq m$). But $B[\mu_i/x] \vdash_{\nabla} M : \nu_i$ if and only if $\exists B' \propto \rho'$ such that $B'[\mu_i/y] \vdash_{\nabla} M' : \nu_i$. So by rules $(\rightarrow I)$, $(\wedge I)$ and (\leq_{∇}) $\exists B' \propto \rho'$ such that $B' \vdash_{\nabla} \lambda y.M' : \pi$. Hence, $\llbracket \lambda x.M \rrbracket_{\rho}^{\mathcal{F}(\nabla)} = \llbracket \lambda y.M' \rrbracket_{\rho'}^{\mathcal{F}(\nabla)}$ by the definition of the interpretation function.

5. By hypothesis. \square

Now we can define the notion of filter $\lambda\Delta$ -model (or briefly filter model, when Δ is either not instantiated or clear from the context).

Definition 10.1.12. (i) A filter $\lambda\Delta$ -model is any quadruple

$$\langle \mathcal{F}(\nabla), \mathcal{I}(\nabla), \circ_{\nabla}, \llbracket \cdot \rrbracket^{\mathcal{F}(\nabla)} \rangle$$

such that $\nabla = \langle C, \leq_{\nabla}, I(C) \rangle$ is a legal type system, and $M \in \Delta$ implies $\llbracket M \rrbracket_{\rho} \in \mathcal{I}(\nabla)$ for all environments ρ .

(ii) The partial order between terms induced by a filter $\lambda\Delta$ -model \mathcal{F} is defined as follows:

$$M \sqsubseteq_{\mathcal{F}} N \quad \text{if and only if} \quad \forall \rho, \llbracket M \rrbracket_{\rho}^{\mathcal{F}} \subseteq \llbracket N \rrbracket_{\rho}^{\mathcal{F}},$$

i.e.

$$\{\sigma | \exists B \propto \rho \text{ such that } B \vdash_{\nabla} M : \sigma\} \subseteq \{\sigma | \exists B \propto \rho \text{ such that } B \vdash_{\nabla} N : \sigma\}.$$

$M \sqsubset_{\mathcal{F}} N$ will denote the proper inclusion.

The two properties proved in the next lemma are quite useful for proving operational properties of terms.

Lemma 10.1.13. Let \mathcal{F} be a $\lambda\Delta$ -model.

- (i) If $M \sqsubseteq_{\mathcal{F}} N$ then $C[M] \sqsubseteq_{\mathcal{F}} C[N]$, for all context $C[.]$.
- (ii) If $M \sqsubseteq_{\mathcal{F}} N$ and $B \vdash_{\nabla} M : \sigma$ then $B \vdash_{\nabla} N : \sigma$, for all bases B .

Proof. (i) By induction on the context $C[.]$, similarly to the proof of Property 10.0.2.(v).

- (ii) $B \vdash_{\nabla} M : \sigma$ implies $\sigma \in \llbracket M \rrbracket_{\rho_B}^{\mathcal{F}}$, where $\rho_B(x) = \uparrow B(x)$, for each $x \in \text{Var}$. But $M \sqsubseteq_{\mathcal{F}} N$ implies $\sigma \in \llbracket N \rrbracket_{\rho_B}^{\mathcal{F}}$, so $\exists B' \propto \rho_B$ such that $B' \vdash_{\nabla} N : \sigma$. It is easy to see that $B(x) \leq_{\nabla} \tau \in \rho_B(x)$, so $B(x) \leq_{\nabla} B'(x)$, for each $x \in \text{Var}$ and, by Lemma 10.1.7.(iii), $B \vdash_{\nabla} N : \sigma$. \square

We can refine both the notion of correctness and completeness of a model with respect to a given operational semantics by taking into account the preorder relation instead of the equivalence one.

Definition 10.1.14. Let \mathcal{F} be a filter model. \mathcal{F} is correct with respect to the \mathbf{O} -operational semantics if and only if $M \sqsubseteq_{\mathcal{F}} N$ implies $M \preceq_{\mathbf{O}} N$, for all $M, N \in \Lambda$. \mathcal{F} is complete with respect to the \mathbf{O} -operational semantics if and only if the inverse implication holds.

Moreover, \mathcal{F} is fully abstract with respect to \mathbf{O} , in case it is both correct and complete with respect to \mathbf{O} .

The next property will be very useful in what follows in order to prove the correctness of some filter models.

Property 10.1.15. Let \mathcal{F} be a filter model such that

$$M \Downarrow_{\mathbf{O}} \text{ and } N \Uparrow_{\mathbf{O}} \text{ implies } N \sqsubset_{\mathcal{F}} M, \text{ for all } M, N \in \Lambda.$$

Then \mathcal{F} is correct with respect to \mathbf{O} .

Proof. We will prove that the given hypothesis implies the following implication: $Q \not\preceq_{\mathbf{O}} P$ implies $Q \not\sqsubseteq_{\mathcal{F}} P$. If $Q \not\preceq_{\mathbf{O}} P$ then there is a context $C[.]$ such that $C[Q] \Downarrow_{\mathbf{O}}$ while $C[P] \Uparrow_{\mathbf{O}}$, which implies, by hypothesis, $C[P] \sqsubset_{\mathcal{F}} C[Q]$, which in turn implies $C[Q] \not\sqsubseteq_{\mathcal{F}} C[P]$. So $Q \not\sqsubseteq_{\mathcal{F}} P$ by Property 10.1.13.(i). \square

The first filter model for the $\lambda\Lambda$ -calculus was built in [10]. A presentation of a class of filter models, which includes models of both $\lambda\Lambda$ -calculus and $\lambda\Gamma$ -calculus, can be found in [38]; it is less general than one given in this book, and it is based on the notion of partial intersection type assignment system.

11. Call-by-Name Denotational Semantics

To model a $\lambda\Lambda$ -theory, it is necessary to reflect in the model the fact that the set of input values coincides with the whole set Λ . So it is natural to ask that in every model the set \mathbb{D} and the set of semantic input values \mathbb{I} must coincide. In a filter model, it must be $\mathcal{F}(C, \nabla) = \mathcal{I}(C, \nabla)$, so $T(C) = I(C)$.

The characterization of a model of $\lambda\Lambda$ -calculus follows quite easily, remembering the notion of a legal type system.

Theorem 11.0.1. *Let $\nabla = < C, \leq_\nabla, T(C) >$.*

If ∇ is legal then $< \mathcal{F}(\nabla), \mathcal{F}(\nabla), \circ_\nabla, [\![\cdot]\!]^{\mathcal{F}(\nabla)} >$ is a $\lambda\Lambda$ -model.

Proof. The proof follows from Theorem 10.1.11, since $\Delta = \Lambda$. □

11.1 The Model \mathcal{H}

In this section, we will introduce a filter model that is fully abstract with respect to the \mathbf{H} -operational semantics. By keeping in mind Property 10.1.15 and by observing that if N is Λ -solvable and M Λ -unsolvable then $M \prec_{\mathbf{H}} N$, in order to define a model that is correct with respect to the \mathbf{H} -operational semantics, it is sufficient to ask for the property:

$$\forall M, N \text{ if } M \text{ is } \Lambda\text{-solvable and } N \text{ is } \Lambda\text{-unsolvable then } N \sqsubset_{\mathcal{H}} M.$$

So we must define a legal type system ∇ based on a set of constants C such that, for every Λ -solvable term M and Λ -unsolvable term N , there is a basis B and at least one type σ such that $B \vdash_\nabla M : \sigma$ while not $B \vdash_\nabla N : \sigma$. It is not possible to find a type σ having this property, because solvable terms are not closed under application (as we already observed at the end of Sect. 1.2), while a type must have a uniform functional behaviour.

In order to characterize the operational semantics \mathbf{H} , we will introduce the class of openly solvable terms, which is a proper subclass of Λ -solvable terms closed under application. We will prove that it is possible to characterize this class in the sense as before, and moreover the same model characterizes, in a weaker sense, the class of all Λ -solvable terms.

Definition 11.1.1. A term M is openly Λ -solvable if and only if $M \rightarrow_{\Lambda}^* \lambda x_1 \dots x_n. z M_1 \dots M_m$ where $z \in \text{FV}(M)$.

By abuse of notation, we will speak about the head variable of a term for denoting the head variable of its Λ -hnf.

Example 11.1.2. $I(Ix)$ and $I(\lambda y. xy)$ are openly Λ -solvable, while $I(\lambda x. xy)$ is Λ -solvable but non-openly Λ -solvable.

It is immediate to verify that the property of being openly Λ -solvable is closed under application; namely if M is openly solvable then $M \vec{N}$ is openly solvable too, for all \vec{N} . This behaviour suggests that a type σ characterizing the openly solvable terms must satisfy the equation $\sigma \simeq \omega \rightarrow \sigma$ (remember that the constant ω can be assigned to every term).

Definition 11.1.3. Let $C_{\infty} = \{\phi, \omega\}$ and $I(C_{\infty}) = T(C_{\infty})$.

∞ is the type system $\langle C_{\infty}, \leq_{\infty}, I(C_{\infty}) \rangle$, where \leq_{∞} is the least intersection relation induced by the rules in Fig. 11.1.

$\overline{\sigma \leq_{\infty} \omega} \quad (a)$	$\overline{\sigma \leq_{\infty} \sigma \wedge \sigma} \quad (b)$	$\overline{\sigma \wedge \tau \leq_{\infty} \sigma} \quad (c)$	$\overline{\sigma \wedge \tau \leq_{\infty} \tau} \quad (c')$	
		$\overline{(\sigma \rightarrow \tau) \wedge (\sigma \rightarrow \pi) \leq_{\infty} \sigma \rightarrow (\tau \wedge \pi)} \quad (d)$		
		$\overline{\sigma \rightarrow \omega \leq_{\infty} \omega \rightarrow \omega} \quad (g)$		
$\overline{\sigma \leq_{\infty} \sigma', \tau \leq_{\infty} \tau'} \quad (e)$	$\overline{\sigma' \leq_{\infty} \sigma, \tau \leq_{\infty} \tau'} \quad (f)$	$\overline{\sigma \leq_{\infty} \sigma} \quad (r)$	$\overline{\sigma \leq_{\infty} \rho \quad \rho \leq_{\infty} \tau} \quad (t)$	
		$\overline{\phi \leq_{\infty} \omega \rightarrow \phi} \quad (h1)$	$\overline{\omega \rightarrow \phi \leq_{\infty} \phi} \quad (h2)$	$\overline{\omega \leq_{\infty} \omega \rightarrow \omega} \quad (h3)$

Fig. 11.1. ∞ -intersection relation

Example 11.1.4. It is easy to check that $\phi \simeq_{\infty} \omega \rightarrow \phi$, $\omega \simeq_{\infty} \omega \rightarrow \omega$ and $\sigma \rightarrow \omega \simeq_{\infty} \omega$, for all σ . Moreover $B \vdash_{\infty} D : \phi \rightarrow \phi$, for every basis B . In fact, we can build the following derivation:

$$\frac{\begin{array}{c} \overline{B[\phi/x] \vdash_{\infty} x : \phi} \quad (var) \\ \hline \overline{B[\phi/x] \vdash_{\infty} x : \omega \rightarrow \phi} \quad (\leq_{\infty}) \end{array}}{B[\phi/x] \vdash_{\infty} xx : \phi \quad (\rightarrow E)}$$

$$\frac{B[\phi/x] \vdash_{\infty} x : \omega \quad (\omega)}{B \vdash_{\infty} D : \phi \rightarrow \phi \quad (\rightarrow I)}$$

In Sect. 11.1.1, the intersection relation \leq_{∞} is extensively studied.

Theorem 11.1.5. *The type system ∞ is legal.*

Proof. The proof is in Sect. 11.1.1. \square

Hence the following definition is well posed, by Theorem 11.0.1.

Definition 11.1.6. \mathcal{H} is the $\lambda\Lambda$ -model: $\langle \mathcal{F}(\infty), \mathcal{F}(\infty), \circ_\infty, \llbracket \cdot \rrbracket^{\mathcal{F}(\infty)} \rangle$.

We will prove that \mathcal{H} is fully abstract with respect to \mathbf{H} .

Property 11.1.7.

- (i) Let M be openly Λ -solvable and let z be its head variable. If B is the basis such that $B(z) = \phi$, then $B \vdash_\infty M : \phi$.
- (ii) Let $M \equiv \lambda x_1 \dots x_n. z M_1 \dots M_m$ ($m, n \in \mathbb{N}$). If $B \vdash_\infty M : \phi$ then M is openly solvable.
- (iii) If $M =_\Lambda \lambda x_1 \dots x_n. x_k M_1 \dots M_m$ ($1 \leq k \leq n$), then for all bases B :
 - $B \vdash_\infty M : \underbrace{\phi \rightarrow \dots \rightarrow \phi}_p$, for every p such that $k+1 \leq p \leq n+1$;
 - $B \not\vdash_\infty M : \underbrace{\phi \rightarrow \dots \rightarrow \phi}_p$, for any $p \leq k$.

Proof. (i) Let $M =_\Lambda \lambda x_1 \dots x_n. z M_1 \dots M_m$, where $z \in FV(M)$. Clearly

$$B[\omega/x_1, \dots, \omega/x_n] \vdash_\infty z : \phi \text{ and } B[\omega/x_1, \dots, \omega/x_n] \vdash_\infty z : \underbrace{\omega \rightarrow \dots \rightarrow \omega}_m \rightarrow \phi,$$

by rule (\leq_∞) . Since $B[\omega/x_1, \dots, \omega/x_n] \vdash_\infty M_i : \omega$ ($1 \leq i \leq m$) by rule (ω) , by m applications of rule $(\rightarrow E)$ we can build a derivation proving $B[\omega/x_1, \dots, \omega/x_n] \vdash_\infty z M_1 \dots M_m : \phi$. Hence, by n applications of rule $(\rightarrow I)$, we have $B \vdash_\infty \lambda x_1 \dots x_n. z M_1 \dots M_m : \underbrace{\omega \rightarrow \dots \rightarrow \omega}_n \rightarrow \phi$, which

implies $B \vdash_\infty \lambda x_1 \dots x_n. z M_1 \dots M_m : \phi$, by rule (\leq_∞) . Thus $B \vdash_\infty M : \phi$, by Corollary 10.0.3.

- (ii) By induction on n . If $n = 0$ then $M \equiv z M_1 \dots M_m$ and the proof is trivial; so let $n \geq 1$ and $B \vdash_\infty \lambda x_1 \dots x_n. z M_1 \dots M_m : \phi$. So $B \vdash_\infty M : \omega \rightarrow \phi$ by rule (\leq_∞) ; so $B[\omega/x_1] \vdash_\infty \lambda x_2 \dots x_n. z M_1 \dots M_m : \phi$, by Lemma 10.1.7.(vi). By inductive hypothesis $z \notin \{x_2, \dots, x_n\}$. Moreover, $B(z) \not\simeq_\infty \omega$; otherwise it would be $\phi \simeq_\infty \omega$ by Lemma 10.1.7.(ii), against the fact that $\phi \not\simeq_\infty \omega$, proved in Sect. 11.1.1. Then $z \neq x_1$ since $B(x_1) = \omega$, so M is openly solvable.
- (iii) Let $M =_\Lambda \lambda x_1 \dots x_n. x_k M_1 \dots M_m$, and let $p \geq k$. As for the point (i), we can build a derivation proving

$$B[\phi/x_1, \dots, \phi/x_p, \omega/x_{p+1}, \dots, \omega/x_n] \vdash_\infty x_k M_1 \dots M_m : \phi,$$

and so, by rules $(\rightarrow I)$ and (\leq_∞) ,

$$B[\phi/x_1, \dots, \phi/x_p] \vdash_{\infty} \lambda x_{p+1} \dots x_n. x_k M_1 \dots M_m : \phi.$$

Hence, by rule $(\rightarrow I)$, $B \vdash_{\infty} \lambda x_1 \dots x_n. x_k M_1 \dots M_m : \underbrace{\phi \rightarrow \dots \rightarrow \phi}_{p} \rightarrow \phi$.

A derivation assigning to M a type of the shape $\underbrace{\phi \rightarrow \dots \rightarrow \phi}_{p} \rightarrow \phi$, with

$p < k$ cannot exist, since the only possibility to derive it would be to assign to x_k a type equivalent to ω , and in this case only a type equivalent to ω can be assigned to M . \square

By using the type assignment system, we can easily prove that the Λ -theory induced by the model \mathcal{H} is a $\Lambda\eta$ -theory.

Property 11.1.8. $\sim_{\mathcal{H}}$ is a $\Lambda\eta$ -theory.

Proof. The theory induced by \mathcal{H} is a Λ -theory by Corollary 10.0.3, since \mathcal{H} is a $\lambda\Lambda$ -model by Theorem 11.0.1. In order to prove that it is also a $\Lambda\eta$ -theory, by Property 2.1.7, it is sufficient to prove that $I \sim_{\mathcal{H}} E$. We will prove both $I \sqsubseteq_{\mathcal{H}} E$ and $E \sqsubseteq_{\mathcal{H}} I$.

($I \sqsubseteq_{\mathcal{H}} E$) We will prove that $B \vdash_{\infty} I : \sigma$ implies $B \vdash_{\infty} E : \sigma$.

By Property 11.1.36.(i), either $\sigma \simeq_{\infty} \omega$ or $\sigma \simeq_{\infty} \sigma_0 \wedge \dots \wedge \sigma_n$ ($n \geq 0$) such that $\forall i \leq n$, $\sigma_i \simeq_{\infty} \tau_1^i \rightarrow \dots \rightarrow \tau_{m_i}^i \rightarrow \phi$ for some $m_i \in \mathbb{N}$. The case $\sigma \simeq_{\infty} \omega$ is trivial, by rule (ω) ; so the proof follows by induction on n .

Let $n = 0$, thus we can assume $\sigma \simeq_{\infty} \mu \rightarrow \nu \rightarrow \tau$, by Property 11.1.36.(ii). Note that $B \vdash_{\infty} I : \sigma$ implies $B \vdash_{\infty} I : \mu \rightarrow \nu \rightarrow \tau$, by rule (\leq_{∞}) ; thus $B[\mu/x] \vdash_{\infty} x : \nu \rightarrow \tau$ by Lemma 10.1.7.(vi), and $\mu \leq_{\infty} \nu \rightarrow \tau$ by Lemma 10.1.7.(ii). We can build the following derivation:

$$\frac{\overline{B[\mu/x][\nu/y] \vdash_{\infty} x : \mu} \quad \mu \leq_{\infty} \nu \rightarrow \tau}{B[\mu/x][\nu/y] \vdash_{\infty} x : \nu \rightarrow \tau} \quad (\text{var}) \quad (\leq_{\infty}) \quad \frac{\overline{B[\mu/x][\nu/y] \vdash_{\infty} y : \nu} \quad (\text{var})}{B[\mu/x][\nu/y] \vdash_{\infty} xy : \tau} \quad (\rightarrow E)$$

$$\frac{\overline{B[\mu/x][\nu/y] \vdash_{\infty} xy : \tau} \quad (\rightarrow I)}{B[\mu/x] \vdash_{\infty} \lambda y. xy : \nu \rightarrow \tau} \quad (\rightarrow I)$$

$$\frac{\overline{B[\mu/x] \vdash_{\infty} \lambda y. xy : \nu \rightarrow \tau} \quad (\rightarrow I) \quad \mu \rightarrow \nu \rightarrow \tau \leq_{\infty} \sigma}{B \vdash_{\infty} \lambda xy. xy : \sigma} \quad (\leq_{\infty})$$

If $n \geq 1$ then the proof follows by inductive hypothesis.

($E \sqsubseteq_{\mathcal{H}} I$) We will prove that $B \vdash_{\infty} E : \sigma$ implies $B \vdash_{\infty} I : \sigma$.

By Property 11.1.36.(i), either $\sigma \simeq_{\infty} \omega$ or $\sigma \simeq_{\infty} \sigma_0 \wedge \dots \wedge \sigma_n$ ($n \geq 0$) such that $\forall i \leq n$, $\sigma_i \simeq_{\infty} \tau_1^i \rightarrow \dots \rightarrow \tau_{m_i}^i \rightarrow \phi$ and $m_i \in \mathbb{N}$. The case $\sigma \simeq_{\infty} \omega$ is trivial, by rule (ω) ; so the proof follows by induction on n .

Let $n = 0$, thus we can assume $\sigma \simeq_{\infty} \mu \rightarrow \nu \rightarrow \tau$, by Property 11.1.36.(ii). Note that $B \vdash_{\infty} \lambda xy. xy : \sigma$ implies $B \vdash_{\infty} \lambda xy. xy :$

$\mu \rightarrow \nu \rightarrow \tau$, by rule (\leq_∞) ; thus, by Lemma 10.1.7.(vi), this implies $B[\mu/x][\nu/y] \vdash_\infty xy : \tau$, which in turn implies $B[\mu/x][\nu/y] \vdash_\infty x : \theta \rightarrow \tau$ and $B[\mu/x][\nu/y] \vdash_\infty y : \theta$, for some θ , by Lemma 10.1.7.(vii). Hence, by Lemma 10.1.7.(ii), $\mu \leq_\infty \theta \rightarrow \tau$ and $\nu \leq_\infty \theta$ (so $\mu \leq_\infty \nu \rightarrow \tau$). Then we can build the following derivation:

$$\frac{\frac{\frac{\overline{B[\mu/x] \vdash_\infty x : \mu}^{(var)} \quad \mu \leq_\infty \nu \rightarrow \tau}{B[\mu/x] \vdash_\infty x : \nu \rightarrow \tau}^{(\leq_\infty)}}{B \vdash_\infty \lambda x.x : \mu \rightarrow \nu \rightarrow \tau}^{(\rightarrow I)}}{B \vdash_\infty \lambda x.x : \sigma}^{(\leq_\infty)}$$

If $n \geq 1$ then the proof follows by inductive hypothesis. \square

In order to prove that \mathcal{H} is correct with respect to the **H**-operational semantics, it is necessary to prove that an unsolvable term cannot be assigned a type of the shape $\underbrace{\phi \rightarrow \dots \rightarrow \phi}_n$, for every n . To this aim we will introduce a

general tool for reasoning about interpretations of terms in a model, namely an approximation theorem. More precisely, we will prove that the meaning of a term (i.e. the set of types that can be derived for it) is the collection of the meanings of a set of normal forms of an extended language. So it will be possible to reason on the denotational semantics of terms simply by induction on normal forms.

First, we extend the language by adding a constant Ω to the formation rules of terms, then we define two new reduction rules on the so-obtained language.

Definition 11.1.9. (i) $\Lambda\Omega$ is the language obtained by adding to Λ the constant Ω , namely the language inductively defined as follows:

- $\Omega \in \Lambda\Omega$,
- $x \in \text{Var}$ implies $x \in \Lambda\Omega$,
- $M \in \Lambda\Omega$ and $x \in \text{Var}$ implies $(\lambda x.M) \in \Lambda\Omega$,
- $M \in \Lambda\Omega$ and $N \in \Lambda\Omega$ implies $(MN) \in \Lambda\Omega$.

(ii) \rightarrow_Ω is the reduction defined as the contextual closure of the following rules:

$$\Omega M \rightarrow \Omega, \quad \lambda x.\Omega \rightarrow \Omega.$$

(iii) The $\Lambda\Omega$ -reduction ($\rightarrow_{\Lambda\Omega}$) is the contextual closure of the following rules:

$$(\lambda x.M)N \rightarrow M[N/x], \quad \Omega M \rightarrow \Omega, \quad \lambda x.\Omega \rightarrow \Omega.$$

$\rightarrow_{\Lambda\Omega}^*$ is the reflexive and transitive closure of $\rightarrow_{\Lambda\Omega}$.

The η -reduction (\rightarrow_η) can be directly applied to the language $\Lambda\Omega$ (see Definition 1.3.7).

$M \in \Lambda\Omega$ is in $\Lambda\Omega$ -normal form ($\Lambda\Omega$ -nf) if and only if it does not contain $\Lambda\Omega$ -redexes.

As usual, terms of $\Lambda\Omega$ will be considered modulo α -equality. The type assignment system of Definition 10.1.1 can be applied to $\Lambda\Omega$ without modifications. It is easy to see that to the term Ω only the type ω can be assigned, by rule (ω) .

Property 11.1.10. The type assignment system \vdash_∞ is closed under $=_{\Lambda\Omega}$.

Proof. Easy. □

The intuitive interpretation of the constant Ω is that it represents a term with an unknown behaviour. The interpretation function is therefore naturally extended to $\Lambda\Omega$, i.e. the interpretation of a term of $\Lambda\Omega$ is the set of all types that can be assigned to it.

Definition 11.1.11. The set of approximants of a term M is defined as follows:

$$\mathcal{A}(M) = \left\{ A \mid \begin{array}{l} \text{there is } M' \text{ such that } M =_\Lambda M' \text{ and } A \text{ is a } \Lambda\Omega\text{-normal form} \\ \text{obtained from } M' \text{ by replacing some subterms with } \Omega. \end{array} \right\}$$

Example 11.1.12. Some sets of approximants are shown.

$$\begin{aligned} \mathcal{A}(I) &= \{\Omega, \lambda x.x\}, & \mathcal{A}(D) &= \{\Omega, \lambda x.xx, \lambda x.x\Omega\}, \\ \mathcal{A}(DD) &= \{\Omega\}, & \mathcal{A}(K(\lambda x.x(II))(DD)) &= \{\Omega, \lambda x.x\Omega, \lambda x.xI\}. \end{aligned}$$

Approximants can be defined inductively as follows.

Definition 11.1.13. The set \mathcal{A} of $\Lambda\Omega$ -normal forms can be inductively defined as follows:

- $\Omega \in \mathcal{A}$,
- $A_j \in \mathcal{A}$ imply $\lambda x_1\dots x_n.xA_1\dots A_m \in \mathcal{A}$ ($1 \leq j \leq m$).

$\Lambda\Omega$ -normal forms will be ranged over by A, A' , possibly indexed.

Clearly every Λ -normal form is an approximant of some term. So, we will simply call approximant a $\Lambda\Omega$ -normal form.

Property 11.1.14. (i) For every A , there is a term M such that $A \in \mathcal{A}(M)$.
(ii) If $M =_\Lambda N$ then $\mathcal{A}(M) = \mathcal{A}(N)$.

Proof. (i) Easy.

(ii) By definition of approximant. □

An approximant A is openly solvable if and only if $A \equiv \lambda x_1 \dots x_n.zA_1 \dots A_m$ and $z \in \text{FV}(A)$. It is easy to check that Property 11.1.7 holds for approximants too.

A key property of the approximants is that the $\sqsubseteq_{\mathcal{H}}$ order relation between them can be syntactically axiomatized.

Definition 11.1.15. Let A and A' be two approximants.

$A \ll A'$ if and only if one of the two following cases arises:

- (i) $A \equiv \Omega$;
- (ii) $A =_{\eta} \lambda x_1 \dots x_n.zA_1 \dots A_m$, and $A' =_{\eta} \lambda x_1 \dots x_n.zA'_1 \dots A'_m$,
and $A_i \ll A'_i$ for each i ($1 \leq i \leq m$; $n, m \in \mathbb{N}$).

Although the η -reduction (\rightarrow_{η}) can be directly applied to the language $\Lambda\Omega$, often we are only interested in its use on approximants. Note that $\Omega =_{\eta} \lambda z.\Omega z$ and $\lambda x.x =_{\eta} \lambda y.(\lambda x.x)y$, but both $\lambda z.\Omega z \notin \mathcal{A}$ and $\lambda y.(\lambda x.x)y \notin \mathcal{A}$, since they are not $\Lambda\Omega$ -normal forms. However, $\lambda z.\Omega z \rightarrow_{\Lambda\Omega} \Omega \in \mathcal{A}$, and $\lambda y.(\lambda x.x)y \rightarrow_{\Lambda\Omega} \lambda x.x \in \mathcal{A}$.

Example 11.1.16. $\mathcal{A}(\lambda x.xII) = \{\Omega, \lambda x.x\Omega\Omega, \lambda x.x\Omega I, \lambda x.xI\Omega, \lambda x.xII\}$. It is easy to see that $\lambda x.x\Omega I \not\ll \lambda x.xI\Omega$, while $\lambda x.x\Omega\Omega \ll \lambda x.xII$.

Clearly \ll is a preorder relation on \mathcal{A} and a partial order on the set of approximants considered up to η -equivalence.

The next theorem proves that two approximants A and A' such that $A \not\ll A'$ can be semiseparated, in the sense that there is a context that $\Lambda\Omega$ -reduces either to I or to Ω where it is filled respectively by A and A' . The fact that the model \mathcal{H} is fully abstract with respect to the **H**-operational semantics is based essentially on this property.

Theorem 11.1.17 (Semiseparability).

Let $A \not\ll A'$. Then there is a context $C[.]$ such that $C[A] \rightarrow_{\Lambda\Omega}^* I$ while $C[A'] \rightarrow_{\Lambda\Omega}^* \Omega$.

Proof. The proof is in Sect. 11.1.3. □

Hence, \ll is a syntactical axiomatization of the $\sqsubseteq_{\mathcal{H}}$ order relation between approximants.

Lemma 11.1.18. $A \sqsubseteq_{\mathcal{H}} A'$ if and only if $A \ll A'$.

Proof. (\Leftarrow) We will prove that, if $A \ll A'$, then $B \vdash_{\infty} A : \sigma$ implies $B \vdash_{\infty} A' : \sigma$, by induction on the definition of \ll . If $A \equiv \Omega$, then $\sigma \equiv \omega$ and the proof is trivial, by rule (ω) . Otherwise, let $A =_{\eta} \lambda x_1 \dots x_n.zA_1 \dots A_m$ and $A' =_{\eta} \lambda x_1 \dots x_n.zA'_1 \dots A'_m$ and $A_i \ll A'_i$ ($1 \leq i \leq m$). By Property 11.1.36.(i), either $\sigma \simeq_{\infty} \omega$ or $\sigma \simeq_{\infty} \sigma_0 \wedge \dots \wedge \sigma_k$ ($k \geq 1$), where $\sigma_i \simeq_{\infty} \tau_1^i \rightarrow \dots \rightarrow \tau_{h_i}^i \rightarrow \phi$ ($h_i \in \mathbb{N}, i \leq k$). The case $\sigma \simeq_{\infty} \omega$ is trivial; so let $\sigma \simeq_{\infty} \sigma_0 \wedge \dots \wedge \sigma_k$ ($k \geq 1$).

$B \vdash_{\infty} A : \sigma$ implies $B \vdash_{\infty} A : \sigma_i$ ($i \leq k$), by rule (\leq_{∞}) . As shown in Property 11.1.8, the type assignment system is closed under η -equality, thus $B \vdash_{\infty} \lambda x_1 \dots x_n. zA_1 \dots A_m : \tau_1^i \rightarrow \dots \rightarrow \tau_{h_i}^i \rightarrow \phi$.

Without loss of generality, we can assume $h_i \geq n$ ($\forall i \leq k$) by Property 11.1.36.(ii), thus $B[\tau_1^i/x_1, \dots, \tau_n^i/x_n] \vdash_{\infty} zA_1 \dots A_m : \tau_{n+1}^i \rightarrow \dots \rightarrow \tau_{h_i}^i \rightarrow \phi$, by Lemma 10.1.7.(vi). Hence, by Lemma 10.1.7.(vii) there are $\mu_1^i, \dots, \mu_{m_i}^i \in T(C_{\infty})$ such that

$$B[\tau_1^i/x_1, \dots, \tau_n^i/x_n] \vdash_{\infty} z : \mu_1^i \rightarrow \dots \rightarrow \mu_m^i \rightarrow \tau_{n+1}^i \rightarrow \dots \rightarrow \tau_{h_i}^i \rightarrow \phi,$$

and $B[\tau_1^i/x_1, \dots, \tau_n^i/x_n] \vdash_{\infty} A_j : \mu_j^i$ ($j \leq m, i \leq k$).

By induction $B[\tau_1^i/x_1, \dots, \tau_n^i/x_n] \vdash_{\infty} A'_j : \mu_j^i$ ($j \leq m, i \leq k$); so by m applications of rule $(\rightarrow E)$ and n applications of rule $(\rightarrow I)$, it follows that $B \vdash_{\infty} A' : \sigma_i$, for all $i \leq k$; the proof follows by rule $(\wedge I)$.

(\Rightarrow) By the semiseparability theorem, $A \ll A'$ implies there is a context $C[.]$ such that $C[A] \rightarrow_{\Lambda\Omega}^* I$ while $C[A'] \rightarrow_{\Lambda\Omega}^* \Omega$. By Property 11.1.10, this implies $C[A] \not\subseteq_{\mathcal{H}} C[A']$, which in turn implies $A \not\subseteq_{\mathcal{H}} A'$. \square

The following theorem states that the interpretation of a term is the collection of the interpretations of its approximants.

Theorem 11.1.19 (\mathcal{H} -Approximation).

$B \vdash_{\infty} M : \sigma$ if and only if $B \vdash_{\infty} A : \sigma$, for some $A \in \mathcal{A}(M)$.

Proof. The proof is in Sect. 11.1.2. \square

The syntactic shape of the approximants of a term having Λ -hnf is related to the syntactic shape of its Λ -hnf's, as showed in the following property.

Property 11.1.20. (i) M is Λ -unsolvable if and only if $\mathcal{A}(M) = \{\Omega\}$.

(ii) Let $A \in \mathcal{A}(M)$ and $A \not\equiv \Omega$.

$M \rightarrow_{\Lambda}^* \lambda x_1 \dots x_n. zM_1 \dots M_m$ if and only if $A \equiv \lambda x_1 \dots x_n. zA_1 \dots A_m$, for some $A_i \in \mathcal{A}(M_i)$.

(iii) Let $A \in \mathcal{A}(M)$ and $A \not\equiv \Omega$. $M =_{\Lambda\eta} \lambda x_1 \dots x_n. zM_1 \dots M_m$ if and only if $A =_{\eta} \lambda x_1 \dots x_n. zA_1 \dots A_m$, for some $A_i \in \mathcal{A}(M_i)$ ($1 \leq i \leq m$).

Proof. (i) If M is a Λ -unsolvable and $M =_{\Lambda} N \equiv \lambda x_1 \dots x_n. RN_1 \dots N_m$, then R is a Λ -redex. So the only $\Lambda\Omega$ -normal form that can be obtained from N by replacing some of its subterms by Ω is Ω itself.

(ii) By definition of approximants and Property 2.1.2.

(iii) By definition of approximants and by definition of $\Lambda\eta$. \square

Corollary 11.1.21. *The theory induced by \mathcal{H} is sensible.*

Proof. By Property 11.1.20.(i), remembering Definition 1.3.4. \square

So the correctness of \mathcal{H} with respect to the \mathbf{H} -operational semantics follows easily.

Theorem 11.1.22 (\mathcal{H} -Correctness).

The model \mathcal{H} is correct with respect to the \mathbf{H} -operational semantics.

Proof. By Property 10.1.15 it is sufficient to prove that, for all M, N , if M is Λ -solvable and N is Λ -unsolvable then $N \sqsubset_{\mathcal{H}} M$.

By Property 11.1.7 a solvable term can be assigned at least one type $\not\simeq_{\infty} \omega$, while only type $\simeq_{\infty} \omega$ can be assigned to an unsolvable term, since it has Ω as the only approximant. \square

The correctness of the model \mathcal{H} with respect to the operational semantics \mathbf{H} allows us to transfer some results from the denotational world to the operational one. So some interesting properties of \mathbf{H} can be proved. The reader can see that the approximation theorem is a quite useful tool.

Corollary 11.1.23. *The \mathbf{H} -operational semantics is fully extensional.*

Proof. By correctness and Property 11.1.8. \square

In Sect. 2.1 the notion of call-by-name fixed-point operator was introduced, and a term with this behaviour was shown in the proof of Theorem 2.1.8. An example of another fixed-point operator will be show in the last chapter. It is an easy exercise to prove that it is possible to build an infinite number of such operators (see [83]). The next theorem proves that all are equated in the \mathbf{H} -operational semantics.

Theorem 11.1.24. *All call-by-name fixed-point operators are \mathbf{H} -equivalent.*

Proof. Let P be any call-by-name fixed-point operator, so $PM =_{\Lambda} M(PM)$, for all $M \in \Lambda$. Then $P =_{\eta} \lambda z.Pz =_{\Lambda} \lambda z.z(Pz)$, where $z \notin FV(P)$; by iterating this reasoning, by the confluence theorem and by Properties 11.1.8 and 11.1.20, every approximant of P is either Ω or of the shape $\lambda z.z(\underbrace{z \dots (z \Omega)}_n \dots)(n \geq 1)$. Hence the result follows, by the approximation theorem. \square

The next theorem proves a key result, which implies directly the full-abstraction of the model \mathcal{H} with respect to the \mathbf{H} -operational semantics. It proves that, if $M \not\sqsubseteq_{\mathcal{H}} N$, then there is a context \mathbf{H} -discriminating them. Its proof uses the semiseparability property between approximants, namely if $M \not\sqsubseteq_{\mathcal{H}} N$ then it is possible to find two approximants $A \in \mathcal{A}(M)$ and $A' \in \mathcal{A}(N)$, such that $A \not\ll A'$ and the context discriminating M and N is a minor modification of the context semiseparating A and A' .

Theorem 11.1.25 (\mathcal{H} -Discriminability).

If $M \not\sqsubseteq_{\mathcal{H}} N$ then there is a context $C[.]$ such that $C[M], C[N] \in \Lambda^0$ and $C[M] \downarrow_{\mathbf{H}}$ while $C[N] \uparrow_{\mathbf{H}}$.

Proof. The proof is in Sect. 11.1.3. \square

Theorem 11.1.26 (\mathcal{H} -Completeness).

The model \mathcal{H} is complete with respect to the \mathbf{H} -operational semantics.

Proof. We must prove that $M \preceq_{\mathbf{H}} N$ implies $M \sqsubseteq_{\mathcal{H}} N$. Let $M \not\sqsubseteq_{\mathcal{H}} N$. By the \mathcal{H} -discriminability Theorem, there is a context $C[\cdot]$ such that both $C[M]$ and $C[N]$ are closed and $C[M] \Downarrow_{\mathbf{H}}$ while $C[N] \Uparrow_{\mathbf{H}}$, hence $M \not\preceq_{\mathbf{H}} N$. \square

Corollary 11.1.27 (\mathcal{H} -Full abstraction).

The model \mathcal{H} is fully abstract with respect to the \mathbf{H} -operational semantics.

Proof. By Theorems 11.1.22 and 11.1.26. \square

The \mathcal{H} -discriminability Theorem allows for a finite axiomatization of the preorder $\sqsubseteq_{\mathcal{H}}$ between terms.

Definition 11.1.28. (i) *The relation $\sqsubseteq_k \subseteq \Lambda \times \Lambda$ ($k \in \mathbb{N}$) is defined, by induction on k , as follows:*

- $M \sqsubseteq_k N$, for all k , if M is Λ -unsolvable;
 - $M \sqsubseteq_0 N$ if and only if $M =_{\Lambda\eta} \lambda x_1 \dots x_n . x M_1 \dots M_m$ and $N =_{\Lambda\eta} \lambda x_1 \dots x_n . x N_1 \dots N_m$;
 - $M \sqsubseteq_{k+1} N$ if and only if $M =_{\Lambda\eta} \lambda x_1 \dots x_n . x M_1 \dots M_m$ and $N =_{\Lambda\eta} \lambda x_1 \dots x_n . x N_1 \dots N_m$, where $M_i \sqsubseteq_k N_i$, for all i ($1 \leq i \leq m$).
- (ii) $M \sim_k N$ if both $M \sqsubseteq_k N$ and $N \sqsubseteq_k M$.

Example 11.1.29. $\lambda x.x(DD)I \sqsubseteq_0 \lambda x.xIK$ while $\lambda x.x(DD)I \not\sqsubseteq_1 \lambda x.xIK$.

Roughly speaking, two terms M and N are in the relation \sqsubseteq_k if there are two terms $M' =_{\Lambda\eta} M$ and $N' =_{\Lambda\eta} N$, having the same structure “up to level k ”. So, if $\forall k \geq 0, M \sim_k N$, any structural difference between them is pushed to the infinite.

Theorem 11.1.30 (\mathcal{H} -Characterization).

$M \sqsubseteq_{\mathcal{H}} N$ if and only if $\forall k \geq 0, M \sqsubseteq_k N$.

Proof. The proof is in Sect. 11.1.3. \square

The next theorem proves an unexpected result. It has been proved that the \mathbf{H} -operational semantics is fully-extensional, i.e. it is closed under η -equality. We will prove now that it also equates terms being different for an infinite number of η -reductions.

Let $E_\infty \equiv Y(\lambda xyz.y(xz))$ where Y is a call-by-name fixed point operator. Observe that, if $M \in \Lambda$ then $E_\infty M =_{\Lambda} \lambda z.M(E_\infty z) =_{\Lambda} \lambda z.M(\lambda z_1.z(E_\infty z_1)) =_{\Lambda} \lambda z.M(\lambda z_1.z(\lambda z_2.z_1(E_\infty z_2)))$, and so on. So z can be viewed as obtained from $E_\infty z$ by means of an infinite number of applications of the η -reduction rule. We will prove that $E_\infty M \approx_{\mathbf{H}} M$.

Theorem 11.1.31. $E_\infty M \approx_{\mathbf{H}} M$.

Proof. We prove $I \sim_{\mathcal{H}} E_{\infty}$; the result follows from Λ -reduction and correctness. Clearly $\mathcal{A}(I) = \{\Omega, \lambda x.x\}$. Moreover, it is straightforward to check that $\mathcal{A}(E_{\infty}) = \{\Omega, \lambda z_0 z_1.z_0 \Omega, \lambda z_0 z_1.z_0(\lambda z_2.z_1(\dots(\lambda z_{n+1}.z_n \Omega)\dots)) \mid n \geq 0\}$.

($E_{\infty} \sqsubseteq_{\mathcal{H}} I$) It is immediate to check that every $A \in \mathcal{A}(E_{\infty})$ is such that $A \ll I$. Thus the proof follows from Lemma 11.1.18 and from the \mathcal{H} -approximation theorem.

($I \sqsubseteq_{\mathcal{H}} E_{\infty}$) It is easy to see that $I \sqsubseteq_k E_{\infty}$, for all k . Then the proof follows from the \mathcal{H} -characterization theorem. \square

$E_{\infty} M$ has no normal form, also when M has a normal form. So a term with normal form is always \mathbf{H} -operationally equivalent to a term without normal form. A syntactical proof of the \mathbf{H} -equivalence between the two terms I and E_{∞} was done in [36].

11.1.1 The \leq_{∞} -Intersection Relation

\leq_{∞} is a preorder relation on $T(C_{\infty})$; so it induces a partial order on $T(C_{\infty})$ considered up to \simeq_{∞} .

Note that $\sigma \in T(C_{\infty})$ implies that $\sigma \equiv \sigma_0 \wedge \dots \wedge \sigma_n$ ($n \in \mathbb{N}$) where, $\sigma_i \in T(C_{\infty})$, and either $\sigma_i \equiv \omega$ or $\sigma_i \equiv \phi$ or $\sigma_i \equiv \pi_i \rightarrow \tau_i$ ($i \leq n$).

Some key properties of ∞ -type theory will be shown. The next theorem characterizes the set of types that are ∞ equivalent to ω .

Theorem 11.1.32. $\sigma \simeq_{\infty} \omega$ if and only if $\sigma \in \Omega^{\mathcal{H}}$, where

$$\Omega^{\mathcal{H}} = \left\{ \sigma \in T(C_{\infty}) \mid \begin{array}{l} \sigma \equiv \sigma_0 \wedge \dots \wedge \sigma_n \text{ } (n \geq 0) \text{ such that} \\ \forall i \leq n \text{ either } \sigma_i \equiv \omega, \text{ or } \sigma_i \equiv \pi_i \rightarrow \tau_i \text{ and } \tau_i \simeq_{\infty} \omega \end{array} \right\}.$$

Proof. Note that $\omega \simeq_{\infty} \sigma$ if and only if $\omega \leq_{\infty} \sigma$, by the rule (a).

\Leftarrow We will prove that $\sigma \in \Omega^{\mathcal{H}}$ implies $\omega \leq_{\infty} \sigma$, by induction on n . Let $n = 0$. If $\sigma_0 \equiv \omega$ then the proof is trivial; otherwise, $\sigma_0 \equiv \pi \rightarrow \tau$ and $\tau \simeq_{\infty} \omega$. Hence $\omega \leq_{\infty} \omega \rightarrow \omega \leq_{\infty} \pi \rightarrow \tau$, respectively, by rules (h3), (f) and (a); by rule (t), the proof follows.

Let $n \geq 1$ and $\sigma \equiv \sigma_0 \wedge \dots \wedge \sigma_n$; so $\forall i \leq n \omega \leq_{\infty} \sigma_i$ by induction. So

$$\omega \leq_{\infty} \underbrace{\omega \wedge \dots \wedge \omega}_n \leq_{\infty} \sigma_0 \wedge \dots \wedge \sigma_n$$

by rules (b), (e) and (t).

\Rightarrow Let us first prove that, if $\sigma \in \Omega^{\mathcal{H}}$ and $\sigma \leq_{\infty} \tau$ then $\tau \in \Omega^{\mathcal{H}}$, by induction on the rules of \leq_{∞} .

(a) Trivial, since $\omega \in \Omega^{\mathcal{H}}$.

(b), (c), (c') By definition of $\Omega^{\mathcal{H}}$.

- (d) $(\mu \rightarrow \rho) \wedge (\mu \rightarrow \pi) \in \Omega^{\mathcal{H}}$ implies $\rho, \pi \simeq_{\infty} \omega$ by definition of $\Omega^{\mathcal{H}}$, so $\omega \leq_{\infty} \omega \wedge \omega \leq_{\infty} \rho \wedge \pi$, by rules (b) and (e); hence $\rho \wedge \pi \simeq_{\infty} \omega$ and $\mu \rightarrow (\rho \wedge \pi) \in \Omega^{\mathcal{H}}$.
- (e) Let $\mu \leq_{\infty} \mu'$ and $\pi \leq_{\infty} \pi'$; $\mu \wedge \pi \in \Omega^{\mathcal{H}}$ implies $\mu, \pi \in \Omega^{\mathcal{H}}$, so $\mu', \pi' \in \Omega^{\mathcal{H}}$ by induction; hence $\mu' \wedge \pi' \in \Omega^{\mathcal{H}}$.
- (f) Let $\mu' \leq_{\infty} \mu$ and $\pi \leq_{\infty} \pi'$; $\mu \rightarrow \pi \in \Omega^{\mathcal{H}}$ implies $\pi \simeq_{\infty} \omega$, hence $\omega \leq_{\infty} \pi \leq_{\infty} \pi'$ implies $\pi' \simeq_{\infty} \omega$; thus, $\mu' \rightarrow \pi' \in \Omega^{\mathcal{H}}$ by definition of $\Omega^{\mathcal{H}}$.
- (g) trivial, since $\omega \rightarrow \omega \in \Omega^{\mathcal{H}}$.
- (r), (t), (h3) Easy.
- (h1), (h2) Not possible.

Then the proof is done, since $\sigma \simeq_{\infty} \omega$ implies $\omega \leq_{\infty} \sigma$, and therefore $\omega \in \Omega^{\mathcal{H}}$ implies $\sigma \in \Omega^{\mathcal{H}}$. \square

As a corollary, we can prove that \leq_{∞} is well posed, in the sense that it does not equate different type constants in C_{∞} .

Corollary 11.1.33. (i) $\phi \not\simeq_{\infty} \omega$.
(ii) $\sigma_1 \rightarrow \dots \rightarrow \sigma_n \rightarrow \omega \simeq_{\infty} \omega$, for all $\sigma_1, \dots, \sigma_n$ ($n \in \mathbb{N}$).

Proof. (i) Since $\phi \notin \Omega^{\mathcal{H}}$.

(ii) Easy, by Theorem 11.1.32. \square

ϕ is the “minimum” type with respect to \leq_{∞} .

Lemma 11.1.34. If $\sigma \in T(C_{\infty})$ then $\phi \leq_{\infty} \sigma$.

Proof. By induction on σ . If $\sigma \equiv \omega$ then the proof is trivial, by rule (a). If $\sigma \equiv \phi$ then the proof is trivial, by rule (r). If $\sigma \equiv \sigma_0 \wedge \sigma_1$ then $\phi \leq_{\infty} \sigma_i$ ($i \leq 1$) by induction, thus $\phi \leq_{\infty} \phi \wedge \phi \leq_{\infty} \sigma_0 \wedge \sigma_1$, by rules (b) and (e). If $\sigma \equiv \sigma_0 \rightarrow \sigma_1$ then $\phi \leq_{\infty} \sigma_1$ by induction, and $\sigma_0 \leq_{\infty} \omega$ by rule (a); thus $\phi \leq_{\infty} \omega \rightarrow \phi \leq_{\infty} \sigma_0 \rightarrow \sigma_1$ by rules (h1) and (f). \square

The class of types equivalent to ϕ is characterized in the following theorem.

Theorem 11.1.35. Let $\Phi^{\mathcal{H}} = \bigcup_{m \geq 0} \Phi_m^{\mathcal{H}}$, where $\Phi_0^{\mathcal{H}} = \{\phi\}$ and, for $m \in \mathbb{N}$,

$$\Phi_{m+1}^{\mathcal{H}} = \left\{ \sigma \in T(C_{\infty}) \mid \begin{array}{l} \sigma \equiv \sigma_0 \wedge \dots \wedge \sigma_n \text{ } (n \in \mathbb{N}) \text{ and } \exists k \leq n \text{ such that,} \\ \text{either } \sigma_k \in \Phi_m^{\mathcal{H}} \\ \text{or } \sigma_k \equiv \pi_k \rightarrow \tau_k \text{ with } \pi_k \simeq_{\infty} \omega \text{ and } \tau_k \in \Phi_m^{\mathcal{H}} \end{array} \right\}.$$

$\sigma \simeq_{\infty} \phi$ if and only if $\sigma \in \Phi^{\mathcal{H}}$, for all $\sigma \in T(C_{\infty})$.

Proof. Note that $\sigma \simeq_{\infty} \phi$ if and only if $\sigma \leq_{\infty} \phi$, by Lemma 11.1.34. Moreover, it is easy to check that

$$\Phi^{\mathcal{H}} = \left\{ \sigma \in T(C_{\infty}) \mid \begin{array}{l} \sigma \equiv \sigma_0 \wedge \dots \wedge \sigma_n \text{ } (n \in \mathbb{N}) \text{ and } \exists k \leq n, \text{ either } \sigma_k \equiv \phi \\ \text{or } \sigma_k \equiv \pi_k \rightarrow \tau_k \text{ with } \pi_k \simeq_{\infty} \omega \text{ and } \tau_k \simeq_{\infty} \phi \end{array} \right\}.$$

\Leftarrow By induction on m . If $m = 0$ then the proof is trivial; so let $m \geq 1$, $\sigma \equiv \sigma_0 \wedge \dots \wedge \sigma_n$ ($n \in \mathbb{N}$) and either $\sigma_k \in \Phi_{m-1}^{\mathcal{H}}$ or $\sigma_k \equiv \pi_k \rightarrow \tau_k$, $\pi_k \simeq_{\infty} \omega$ and $\tau_k \in \Phi_{m-1}^{\mathcal{H}}$. If $\sigma_k \in \Phi_{m-1}^{\mathcal{H}}$ then $\sigma_k \simeq_{\infty} \phi$ by induction; otherwise, $\sigma_k \leq_{\infty} \omega \rightarrow \phi \leq_{\infty} \phi$ by rules (h1) and (f) and by induction. Hence, by rules (e), (c), (r) and (t):

$$\sigma_0 \wedge \dots \wedge \sigma_k \dots \wedge \sigma_n \leq_{\infty} \sigma_0 \wedge \dots \wedge \phi \wedge \dots \wedge \sigma_n \leq_{\infty} \phi.$$

\Rightarrow Let us first prove that, if $\sigma \in \Phi^{\mathcal{H}}$ and $\tau \leq_{\infty} \sigma$ then $\tau \in \Phi^{\mathcal{H}}$, by induction on the rules of \leq_{∞} .

(a) Not possible, since $\omega \notin \Phi^{\mathcal{H}}$.

(b), (c), (c') By definition of $\Phi^{\mathcal{H}}$.

(d) $\mu \rightarrow (\rho \wedge \pi) \in \Phi^{\mathcal{H}}$ implies $\mu \simeq_{\infty} \omega$ and $\rho \wedge \pi \in \Phi^{\mathcal{H}}$, so it is easy to see that either $\rho \in \Phi^{\mathcal{H}}$ or $\pi \in \Phi^{\mathcal{H}}$; thus either $\mu \rightarrow \rho \in \Phi^{\mathcal{H}}$ or $\mu \rightarrow \pi \in \Phi^{\mathcal{H}}$; hence $(\mu \rightarrow \rho) \wedge (\mu \rightarrow \pi) \in \Phi^{\mathcal{H}}$.

(e) Let $\mu \leq_{\infty} \mu'$ and $\pi \leq_{\infty} \pi'$; so $\mu' \wedge \pi' \in \Phi^{\mathcal{H}}$ implies either $\mu' \in \Phi^{\mathcal{H}}$ or $\pi' \in \Phi^{\mathcal{H}}$, thus $\mu \in \Phi^{\mathcal{H}}$ or $\pi \in \Phi^{\mathcal{H}}$ by induction; hence $\mu \wedge \pi \in \Phi^{\mathcal{H}}$.

(f) Let $\mu' \leq_{\infty} \mu$ and $\pi \leq_{\infty} \pi'$; $\mu' \rightarrow \pi' \in \Phi^{\mathcal{H}}$ implies $\mu' \simeq_{\infty} \omega$ and $\pi' \in \Phi^{\mathcal{H}}$; hence, $\pi \in \Phi^{\mathcal{H}}$ by induction and $\omega \leq_{\infty} \mu$ by rule (t), so $\mu \rightarrow \pi \in \Phi^{\mathcal{H}}$.

(g), (h3) Not possible, since $\omega \rightarrow \omega \notin \Phi^{\mathcal{H}}$.

(r), (t), (h1), (h2) Easy.

Then the proof is done, since $\sigma \simeq_{\infty} \phi$ implies $\sigma \leq_{\infty} \phi$, and therefore $\phi \in \Phi^{\mathcal{H}}$ implies $\sigma \in \Phi^{\mathcal{H}}$. \square

The next property shows some type characterization.

Property 11.1.36. (i) If $\sigma \not\simeq_{\infty} \omega$ then $\sigma \simeq_{\infty} \sigma_0 \wedge \dots \wedge \sigma_n$ ($n \geq 0$) such that

$\forall i \leq n, \sigma_i \simeq_{\infty} \tau_1^i \rightarrow \dots \rightarrow \tau_{m_i}^i \rightarrow \phi$ for some $m_i \in \mathbb{N}$.

(ii) If $\sigma \simeq_{\infty} \tau_1 \rightarrow \dots \rightarrow \tau_n \rightarrow \phi$ ($n \in \mathbb{N}$) then, for all $p \in \mathbb{N}$,

$$\sigma \simeq_{\infty} \tau_1 \rightarrow \dots \rightarrow \tau_n \rightarrow \underbrace{\omega \rightarrow \dots \rightarrow \omega}_p \rightarrow \phi.$$

Proof. (i) By induction on σ . If either $\sigma \equiv \omega$ or $\sigma \equiv \phi$ then the proof is trivial. Let $\sigma \equiv \mu \rightarrow \pi$; so, by induction $\pi \simeq_{\infty} \pi_0 \wedge \dots \wedge \pi_h$ ($h \in \mathbb{N}$) such that $\forall i \leq h, \pi_i \simeq_{\infty} \nu_1^i \rightarrow \dots \rightarrow \nu_{m_i}^i \rightarrow \phi$. By Lemma 10.1.4.(iii), $\sigma \simeq_{\infty} \sigma_0 \wedge \dots \wedge \sigma_h$, where $\sigma_i \simeq_{\infty} \mu \rightarrow \nu_1^i \rightarrow \dots \rightarrow \nu_{m_i}^i \rightarrow \phi$.

The case $\sigma \equiv \mu \wedge \pi$ follows by induction.

(ii) By rules (h1), (h2) and (f) it is easy to see that:

$$\tau_1 \rightarrow \dots \rightarrow \tau_n \rightarrow \phi \simeq_{\infty} \tau_1 \rightarrow \dots \rightarrow \tau_n \rightarrow \omega \rightarrow \phi$$

so the proof follows by induction on p . \square

The next property implies the legality of ∞ . Let us notice that every type in $T(C_\infty)$ has the following syntactical shape:

$$(\sigma_1 \rightarrow \tau_1) \wedge \dots \wedge (\sigma_n \rightarrow \tau_n) \wedge c_1 \wedge \dots \wedge c_m \text{ where } m, n \geq 0, m + n \geq 1,$$

where $c_i \in \{\omega, \phi\}$ ($1 \leq i \leq m$). Moreover remember that every constant in C_∞ is equivalent to an arrow type; indeed, $\omega \simeq_\infty \omega \rightarrow \omega$ and $\phi \simeq_\infty \omega \rightarrow \phi$.

Property 11.1.37. Let $m, n, p, q \geq 0$, $m + n \geq 1$, $p + q \geq 1$, and

$$(\sigma_1 \rightarrow \tau_1) \wedge \dots \wedge (\sigma_n \rightarrow \tau_n) \wedge c_1 \wedge \dots \wedge c_m \leq_\infty (\sigma'_1 \rightarrow \tau'_1) \wedge \dots \wedge (\sigma'_p \rightarrow \tau'_p) \wedge d_1 \wedge \dots \wedge d_q$$

where $c_i, d_j \in \{\omega, \phi\}$, $c_i \simeq_\infty \sigma_{n+i} \rightarrow \tau_{n+i}$ and $d_j \simeq_\infty \sigma'_{p+j} \rightarrow \tau'_{p+j}$ ($\tau_{n+i}, \tau'_{p+j} \in \{\omega, \phi\}$, $1 \leq i \leq m, 1 \leq j \leq q$). If $\tau'_h \not\simeq \omega$ ($1 \leq h \leq p + q$), then there is $\{i_1, \dots, i_k\} \subseteq \{1, \dots, n + m\}$ such that $\sigma_{i_1} \wedge \dots \wedge \sigma_{i_k} \geq_\infty \sigma'_h$, $\tau_{i_1} \wedge \dots \wedge \tau_{i_k} \leq_\infty \tau'_h$ and $\tau_j \not\simeq \omega$, for each $j \in \{i_1, \dots, i_k\}$.

Proof. We reason by induction on the last rule of the derivation proving

$$(\sigma_1 \rightarrow \tau_1) \wedge \dots \wedge (\sigma_n \rightarrow \tau_n) \wedge c_1 \wedge \dots \wedge c_m \leq_\infty (\sigma'_1 \rightarrow \tau'_1) \wedge \dots \wedge (\sigma'_p \rightarrow \tau'_p) \wedge d_1 \wedge \dots \wedge d_q.$$

(a) Trivial, since $\omega \simeq_\infty \omega \rightarrow \omega$ makes the implication empty.

(g), (h3) Trivial, since $\omega \rightarrow \omega$ makes the implication empty.

(b), (c), (c'), (d), (r), (h1), (h2) Immediate.

(e), (f) By induction.

(t) Let $\rho \in T(C_\infty)$ be such that $(\sigma_1 \rightarrow \tau_1) \wedge \dots \wedge (\sigma_n \rightarrow \tau_n) \wedge c_1 \wedge \dots \wedge c_m \leq_\infty \rho$ and $\rho \leq_\infty (\sigma'_1 \rightarrow \tau'_1) \wedge \dots \wedge (\sigma'_p \rightarrow \tau'_p) \wedge d_1 \wedge \dots \wedge d_q$. ρ must be of the shape $(\mu_1 \rightarrow \nu_1) \wedge \dots \wedge (\mu_t \rightarrow \nu_t) \wedge e_1 \wedge \dots \wedge e_s$, where $e_i \in \{\omega, \phi\}$ and $e_i \simeq_\infty \mu_{t+i} \rightarrow \nu_{t+i}$ ($\nu_{t+i} \in \{\omega, \phi\}$, $1 \leq i \leq s$). By induction, for every h such that $\tau'_h \not\simeq \omega$, there is $\{i_1^h, \dots, i_{k_h}^h\} \subseteq \{1, \dots, t + s\}$ such that $\mu_{i_1^h} \wedge \dots \wedge \mu_{i_{k_h}^h} \geq_\infty \sigma'_h$, $\nu_{i_1^h} \wedge \dots \wedge \nu_{i_{k_h}^h} \leq_\infty \tau'_h$ and $\nu_{i_j^h} \not\simeq \omega$, for all $1 \leq j \leq k_h$. By induction, for every $\nu_{i_j^h}$, there is $\{r_1^{h,j}, \dots, r_{w_{h,j}}^{h,j}\} \subseteq \{1, 2, \dots, n + m\}$ such that $\sigma_{r_1^{h,j}} \wedge \dots \wedge \sigma_{r_{w_{h,j}}^{h,j}} \geq_\infty \mu_{i_j^h}$ and $\tau_{r_1^{h,j}} \wedge \dots \wedge \tau_{r_{w_{h,j}}^{h,j}} \leq_\infty \nu_{i_j^h}$ and $\tau_{r_u^{h,j}} \not\simeq \omega$, for all $1 \leq u \leq w_{h,j}$. So the proof follows by rule (e). \square

▼ Proof of Theorem 11.1.5 (pag. 121).

The legality of the type system ∞ is a particular case of Property 11.1.37. \blacksquare

11.1.2 Proof of the \mathcal{H} -Approximation Theorem

The most difficult part of the proof is to prove that $B \vdash_\infty M : \sigma$ implies $B \vdash_\infty A : \sigma$, for some $A \in \mathcal{A}(M)$.

The technique used here is a variant of Tait's notion of computability, first defined for intersection types in [30].

Let us sketch the general idea of the proof.

Definition 11.1.38. *App $_{\mathcal{H}}$ is the predicate defined as follows:*
App $_{\mathcal{H}}(B, \sigma, M)$ if and only if there is $A \in \mathcal{A}(M)$ such that $B \vdash_{\infty} A : \sigma$.

Our aim is to prove the following implication:

$$B \vdash_{\infty} M : \sigma \text{ implies } App_{\mathcal{H}}(B, \sigma, M).$$

We will build the proof in two steps. First, it will be proved that:

$$B \vdash_{\infty} M : \sigma \text{ implies } Comp_{\mathcal{H}}(B, \sigma, M), \quad (11.1)$$

and then

$$Comp_{\mathcal{H}}(B, \sigma, M) \text{ implies } App_{\mathcal{H}}(B, \sigma, M), \quad (11.2)$$

where $Comp_{\mathcal{H}}(B, \sigma, M)$ (read: the term M is *computable* of type σ with respect to a basis B) is a property of the triple $\langle B, \sigma, M \rangle$. We will prove Eq. (11.1) by induction on terms and Eq. (11.2) by induction on types.

A basis B is *finite* if and only if $B(y) \simeq_{\infty} \omega$ except in a finite number of variables. We will use $[\sigma_1/x_1, \dots, \sigma_n/x_n]$ to denote a finite basis. By Lemma 10.1.7.(i), in this section we limit ourselves to consider only such a kind of basis.

Let B and B' be two basis. $B \cup B'$ denotes the basis such that, for every x , $B \cup B'(x) = B(x) \wedge B'(x)$ (remember that, for every type σ , $\sigma \wedge \omega \simeq \sigma$).

Definition 11.1.39. *The predicate $Comp_{\mathcal{H}}$ is defined by induction on types as follows:*

- $Comp_{\mathcal{H}}(B, \omega, M)$ is true;
- $Comp_{\mathcal{H}}(B, \phi, M)$ if and only if $\forall \vec{N}, App_{\mathcal{H}}(B, \phi, M \vec{N})$;
- $Comp_{\mathcal{H}}(B, \sigma \rightarrow \tau, M)$ if and only if $\forall N \in \Lambda, Comp_{\mathcal{H}}(B', \sigma, N) \text{ implies } Comp_{\mathcal{H}}(B \cup B', \tau, MN)$;
- $Comp_{\mathcal{H}}(B, \sigma \wedge \tau, M)$ if and only if both $Comp_{\mathcal{H}}(B, \sigma, M)$ and $Comp_{\mathcal{H}}(B, \tau, M)$.

Note that $Comp_{\mathcal{H}}(B, \phi, M)$ implies $App_{\mathcal{H}}(B, \phi, M)$, as a particular case.

Lemma 11.1.40. *$Comp_{\mathcal{H}}(B, \sigma, M)$ and $M =_{\Lambda} M'$ imply $Comp_{\mathcal{H}}(B, \sigma, M')$, i.e. $Comp_{\mathcal{H}}$ is defined modulo $=_{\Lambda}$ on terms.*

Proof. The proof is given by induction on σ .

The case $\sigma \equiv \omega$ is obvious. Let $\sigma \equiv \phi$ and $M =_{\Lambda} M'$; so $Comp_{\mathcal{H}}(B, \phi, M)$ if and only if $\forall \vec{N}, App_{\mathcal{H}}(B, \phi, M \vec{N})$, thus $\forall \vec{N}, \exists A \in \mathcal{A}(M \vec{N}), B \vdash_{\infty} A : \phi$, which imply $\forall \vec{N}, \exists A' \in \mathcal{A}(M' \vec{N}), B \vdash_{\infty} A' : \phi$ (since approximants are defined up to $=_{\Lambda}$) if and only if $\forall \vec{N}, App_{\mathcal{H}}(B, \phi, M' \vec{N})$ (by definition of $App_{\mathcal{H}}$) if and only if $Comp_{\mathcal{H}}(B, \phi, M')$ (by definition of $Comp_{\mathcal{H}}$). The cases $\sigma \equiv \mu \rightarrow \nu$ and $\sigma \equiv \mu \wedge \nu$ follow immediately from the inductive hypothesis. \square

Point (ii) of the following lemma proves the implication of Eq. (11.2).

Lemma 11.1.41. (i) $\text{App}_{\mathcal{H}}(B, \sigma, x\vec{M})$ implies $\text{Comp}_{\mathcal{H}}(B, \sigma, x\vec{M})$.
(ii) $\text{Comp}_{\mathcal{H}}(B, \sigma, M)$ implies $\text{App}_{\mathcal{H}}(B, \sigma, M)$.

Proof. The proof is done by mutual induction on σ . The only not obvious case is when $\sigma \equiv \tau \rightarrow \rho$ and $\rho \not\simeq \omega$.

- (i) By induction on (ii), $\text{Comp}_{\mathcal{H}}(B', \tau, N)$ implies $\text{App}_{\mathcal{H}}(B', \tau, N)$. Thus, if $\text{App}_{\mathcal{H}}(B, \tau \rightarrow \rho, x\vec{M})$ then $\text{App}_{\mathcal{H}}(B \cup B', \rho, x\vec{M}N)$ by rule $(\rightarrow E)$, since if $x\vec{A} \in \mathcal{A}(\vec{M})$ and $A' \in \mathcal{A}(N)$ then $x\vec{A}A' \in \mathcal{A}(\vec{M}N)$. Hence, by induction, $\text{Comp}_{\mathcal{H}}(B \cup B', \rho, x\vec{M}N)$, so $\text{Comp}_{\mathcal{H}}(B, \tau \rightarrow \rho, x\vec{M})$ by definition of $\text{Comp}_{\mathcal{H}}$.
- (ii) Let $z \notin FV(M)$ be such that $B(z) \simeq_{\infty} \omega$. Note that there exists a such z , from the hypothesis that B is a finite basis. Clearly $z \in \mathcal{A}$ and $[\tau/z] \vdash_{\infty} z : \tau$, so by induction on (i), $\text{App}_{\mathcal{H}}([\tau/z], \tau, z)$ implies $\text{Comp}_{\mathcal{H}}([\tau/z], \tau, z)$. Thus $\text{Comp}_{\mathcal{H}}(B, \tau \rightarrow \rho, M)$ and $\text{Comp}_{\mathcal{H}}([\tau/z], \tau, z)$ imply $\text{Comp}_{\mathcal{H}}(B[\tau/z], \rho, Mz)$, and this implies $\text{App}_{\mathcal{H}}(B[\tau/z], \rho, Mz)$, by induction; which means there is $A \in \mathcal{A}(Mz)$ such that $B[\tau/z] \vdash_{\infty} A : \rho$. Note that $A \equiv \Omega$ is not possible, since by hypothesis $\rho \not\simeq \omega$. By rule $(\rightarrow I)$, $B \vdash_{\infty} \lambda z. A : \tau \rightarrow \rho$. By definition of approximant, $A \in \mathcal{A}(Mz)$ implies $\lambda z. A \in \mathcal{A}(\lambda z. Mz)$. Now there are two cases.
 - 1. $M \rightarrow_{\Lambda}^{*} xM_1...M_m$, thus A is of the shape $xA_1...A_mz$.
So $B \vdash_{\infty} xA_1...A_m : \tau \rightarrow \rho$, by Property 11.1.8. The proof is given, since $xA_1...A_m \in \mathcal{A}(M)$.
 - 2. Otherwise $M =_{\Lambda} \lambda y. M'$, so $\lambda z. Mz =_{\Lambda} \lambda z. M'[z/y] =_{\alpha} \lambda y. M'$, which implies $\lambda z. A \in \mathcal{A}(M)$, and the proof is given. \square

$\text{Comp}_{\mathcal{H}}$ is closed under \leq_{∞} .

Lemma 11.1.42. (i) $\text{Comp}_{\mathcal{H}}(B, \phi, M)$ if and only if $\text{Comp}_{\mathcal{H}}(B, \omega \rightarrow \phi, M)$.
(ii) If $\text{Comp}_{\mathcal{H}}(B, \sigma, M)$ and $\sigma \leq_{\infty} \tau$ then $\text{Comp}_{\mathcal{H}}(B, \tau, M)$.

Proof. (i) (\Rightarrow) $\text{Comp}_{\mathcal{H}}(B, \phi, M)$ implies $\forall N, \forall \vec{P}, \text{App}_{\mathcal{H}}(B, \phi, MN\vec{P})$ (by definition of $\text{Comp}_{\mathcal{H}}$), which imply $\forall N, \text{Comp}_{\mathcal{H}}(B, \phi, MN)$ (by definition of $\text{Comp}_{\mathcal{H}}$). Hence, $\text{Comp}_{\mathcal{H}}(B, \omega \rightarrow \phi, M)$ again by definition of $\text{Comp}_{\mathcal{H}}$.

(\Leftarrow) We prove that $\forall \vec{N}, \text{App}_{\mathcal{H}}(B, \phi, M\vec{N})$, by induction on $\|\vec{N}\|$. Then the proof follows, by definition of $\text{Comp}_{\mathcal{H}}$. Clearly, $\text{Comp}_{\mathcal{H}}(B, \omega \rightarrow \phi, M)$ implies $\text{App}_{\mathcal{H}}(B, \omega \rightarrow \phi, M)$ by Lemma 11.1.41.(ii), and this implies $\text{App}_{\mathcal{H}}(B, \phi, M)$ by rule (\leq_{∞}) .

In case $\|\vec{N}\| \geq 1$, let $\vec{N} \equiv \vec{N}'P$. Therefore $\text{App}_{\mathcal{H}}(B, \phi, M\vec{N}')$ by induction. Clearly $\text{App}_{\mathcal{H}}(B, \omega, P)$, hence the proof follows by rules (\leq_{∞}) and $(\rightarrow E)$.

- (ii) By induction on the rules of \leq_{∞} .
 - (a), (b), (c), (c'), (e), (r), (t) Trivial.

- (d) $\text{Comp}_{\mathcal{H}}(B, (\sigma_0 \rightarrow \tau_0) \wedge (\sigma_0 \rightarrow \pi_0), M)$ implies, by definition of $\text{Comp}_{\mathcal{H}}$, $\text{Comp}_{\mathcal{H}}(B, \sigma_0 \rightarrow \tau_0, M)$ and $\text{Comp}_{\mathcal{H}}(B, \sigma_0 \rightarrow \pi_0, M)$.
If $\text{Comp}_{\mathcal{H}}(B', \sigma_0, N)$ then $\text{Comp}_{\mathcal{H}}(B \cup B', \tau_0, MN)$ and $\text{Comp}_{\mathcal{H}}(B \cup B', \pi_0, MN)$; therefore, both $\text{Comp}_{\mathcal{H}}(B \cup B', \tau_0 \wedge \pi_0, MN)$ and $\text{Comp}_{\mathcal{H}}(B, \sigma_0 \rightarrow (\tau_0 \wedge \pi_0), M)$, by definition of $\text{Comp}_{\mathcal{H}}$.
- (f) Let $\sigma'_0 \leq_{\infty} \sigma_0$, $\tau_0 \leq_{\infty} \tau'_0$ and $\text{Comp}_{\mathcal{H}}(B, \sigma_0 \rightarrow \tau_0, M)$.
If $\text{Comp}_{\mathcal{H}}(B', \sigma'_0, N)$ then $\text{Comp}_{\mathcal{H}}(B', \sigma_0, N)$ by induction; hence $\text{Comp}_{\mathcal{H}}(B \cup B', \tau_0, MN)$ by definition of $\text{Comp}_{\mathcal{H}}$. Again, by induction, $\text{Comp}_{\mathcal{H}}(B \cup B', \tau'_0, MN)$, so the proof is done.
- (g), (h3) Easy, since $\text{Comp}_{\mathcal{H}}(B, \omega, MN)$ is always true.
- (h1), (h2) By point (i) of this lemma. \square

The following lemma allows us to prove the implication of Eq. (11.1).

Lemma 11.1.43. *Let $\text{FV}(M) \subseteq \{x_1, \dots, x_n\}$ and $B = [\sigma_1/x_1, \dots, \sigma_n/x_n]$. If $\text{Comp}_{\mathcal{H}}(B_i, \sigma_i, N_i)$ ($1 \leq i \leq n$) and $B \vdash_{\infty} M : \tau$ then*

$$\text{Comp}_{\mathcal{H}}(B_1 \cup \dots \cup B_n, \tau, M[N_1/x_1, \dots, N_n/x_n]).$$

Proof. By induction on the derivation of $B \vdash_{\infty} M : \tau$. The most interesting case is when the last applied rule is $(\rightarrow I)$; so $M \equiv \lambda x.M'$, $\tau \equiv \mu \rightarrow \nu$ and the derivation is:

$$\frac{\vdots}{\frac{B[\mu/x] \vdash_{\infty} M' : \nu}{B \vdash_{\infty} \lambda x.M' : \mu \rightarrow \nu}} \dots \quad (\rightarrow I)$$

If $\text{Comp}_{\mathcal{H}}(B', \mu, N)$ then $\text{Comp}_{\mathcal{H}}(B' \cup_{1 \leq j \leq n} B_j, \nu, M'[N_1/x_1, \dots, N_n/x_n, N/x])$, by induction; hence $\text{Comp}_{\mathcal{H}}(B' \cup_{1 \leq j \leq n} B_j, \nu, (\lambda x.M'[N_1/x_1, \dots, N_n/x_n])N)$ by Lemma 11.1.40, thus $\text{Comp}_{\mathcal{H}}(B_1 \cup \dots \cup B_n, \tau, M[N_1/x_1, \dots, N_n/x_n])$ by definition of $\text{Comp}_{\mathcal{H}}$.

All other cases follow directly from the inductive hypothesis. \square

▼ Proof of the \mathcal{H} -Approximation Theorem (Theorem 11.1.19 pag. 126).

- (\Rightarrow) Let us prove that $B \vdash_{\infty} M : \sigma$ implies $B \vdash_{\infty} A : \sigma$, for some $A \in \mathcal{A}(M)$. Clearly $\text{Comp}_{\mathcal{H}}([\tau/x], \tau, x)$, by Lemma 11.1.41.(i).
Let $\text{FV}(M) \subseteq \{x_1, \dots, x_n\}$, so we can assume $B = [\sigma_1/x_1, \dots, \sigma_n/x_n]$ without loss of generality, by Lemma 10.1.7.(i). Hence $B \vdash_{\infty} M : \sigma$ and $\text{Comp}_{\mathcal{H}}([\sigma_i/x_i], \sigma_i, x_i)$ ($1 \leq i \leq n$) imply $\text{Comp}_{\mathcal{H}}(B, \sigma, M)$ by Lemma 11.1.43. Thus by Lemma 11.1.41.(ii) and the definition of approximant, the proof is done.
- (\Leftarrow) We must prove that $B \vdash_{\infty} A : \sigma$ for some $A \in \mathcal{A}(M)$ implies $B \vdash_{\infty} M : \sigma$. By definition, there is M' such that $M =_A M'$ and A matches M' except at occurrences of Ω . A derivation of $B \vdash_{\infty} A : \sigma$ can be transformed into a derivation of $B \vdash_{\infty} M' : \sigma$, simply by replacing every subderivation

$$\frac{}{B \vdash_{\infty} \Omega : \omega}^{(\omega)} \quad \text{by} \quad \frac{}{B \vdash_{\infty} N : \omega}^{(\omega)},$$

where N is the subterm replaced by Ω in M' . $B \vdash_{\infty} M' : \sigma$ implies $B \vdash_{\infty} M : \sigma$, since the type assignment system is closed under $=_{\Lambda}$ on terms, as a consequence of the fact that it induces a $\lambda\Lambda$ -model, and the proof is given. ■

11.1.3 Proof of Semiseparability, \mathcal{H} -Discriminability and \mathcal{H} -Characterization Theorems

In order to prove the three theorems, we need a deeper investigation on the preorder relation induced by the model \mathcal{H} on \mathcal{A} . First of all, let us formalize a stratified version of the relation \ll and of its negation $\not\ll$.

Definition 11.1.44. Let a path c be a finite sequence of natural numbers greater than 0 (ϵ is the empty sequence).

- (i) $A \ll_c A'$ if and only if one of the following cases arises:
 - (a) $A \equiv \Omega$;
 - (b) $c \equiv \epsilon$, $A =_{\eta} \lambda x_1 \dots x_n . x A_1 \dots A_m$ and $A' =_{\eta} \lambda x_1 \dots x_n . x A'_1 \dots A'_m$ for some $n, m \in \mathbb{N}$;
 - (c) $c \equiv j, c'$, $A =_{\eta} \lambda x_1 \dots x_n . x A_1 \dots A_m$, $A' =_{\eta} \lambda x_1 \dots x_n . x A'_1 \dots A'_m$, and $A_j \ll_{c'} A'_j$ ($1 \leq j \leq m$).

- (ii) $A \not\ll_c A'$ if and only if one of the following cases arises:
 - (a) $A \not\equiv \Omega$ and $A' \equiv \Omega$;
 - (b) $A =_{\eta} \lambda x_1 \dots x_n . x A_1 \dots A_m$, $A' =_{\eta} \lambda x_1 \dots x_p . y A'_1 \dots A'_q$ and, either $x \not\equiv y$ or $|n - m| \neq |p - q|$;
 - (c) $c \equiv j, c'$, $A =_{\eta} \lambda x_1 \dots x_n . x A_1 \dots A_m$, $A' =_{\eta} \lambda x_1 \dots x_n . x A'_1 \dots A'_m$ and $A_j \not\ll_{c'} A'_j$ ($1 \leq j \leq m$).

Two approximants A and A' are structurally different in a path c if $A \not\ll_c A'$ is proved without using rule (a), they are structurally similar in a path c if and only if $A \ll_c A'$ is proved using rule (a).

Example 11.1.45. $x \ll_{3,1} x$; in fact, $x =_{\eta} \lambda x_1 x_2 x_3 . x x_1 x_2 x_3$ and $x_3 \ll_1 x_3$, since $x_3 =_{\eta} \lambda y_1 . x_3 y_1$ and $y_1 \ll_{\epsilon} y_1$.

Moreover, $\lambda x . x O \Omega \ll_{2,3} \lambda x . x K I$ since $\Omega \ll_3 I$, but $\lambda x . x O \Omega \not\ll_1 \lambda x . x K I$ since $O \not\ll_{\epsilon} K$; in particular, $\lambda x . x O \Omega$ and $\lambda x . x K I$ are structurally similar along path 2, but they are structurally different along the path 1.

Property 11.1.46. Let c be a path.

- (i) $A \ll A'$ if and only if $A \ll_c A'$, for all path c .
- (ii) $A \not\ll A'$ if and only if $A \not\ll_c A'$, for some path c .

Proof. (i) (\Rightarrow) Easy, by Definition 11.1.15 (pag. 125).

(\Leftarrow) By induction on A .

- (ii) We show that either $A \ll_c A'$ or $A \not\ll_c A'$ in an exclusive sense, for all paths c . In case $A \equiv \Omega$ then $A \ll_c A$ by Definition 11.1.44.(i).(a); while A, A' are not in relation $\not\ll_c$, for all paths c .

Let $A \not\equiv \Omega$ and $A' \equiv \Omega$ then $A \not\ll_c A'$, for all paths c ; while A, A' are not in relation \ll_c , for all path c .

Otherwise, $A \equiv \lambda x_1 \dots x_n.x A_1 \dots A_m$, $A' \equiv \lambda x_1 \dots x_p.y A'_1 \dots A'_q$, and the proof follows by taking into account all possible cases.

Thus by the previous point, the proof is done. \square

The proof of the semiseparability theorem will be done by showing a semiseparability algorithm, which is in some sense an extension to approximants of the Λ -separability algorithm. The main difference between the two algorithms is that the semiseparability one is defined depending on a particular path. Namely, given two approximants A and A' and a path c such that $A \not\ll_c A'$, it gives as output a context $C[.]$ such that $C[A] \rightarrow_{\Lambda\Omega}^* I$ while $C[A'] \rightarrow_{\Lambda\Omega}^* \Omega$. The path c is explicitly used by the algorithm.

The terms B^n , O^n and U_m^i are defined as in Sect. 2.1.2. Since approximants are $\Lambda\Omega$ -normal forms, the notion of args (see Definition 2.1.12 pag. 29) is naturally extended to them, by $\text{args}(\Omega) = 0$. Moreover, if $P \in \Lambda\Omega$ has a $\Lambda\Omega$ -normal form, then $\text{nf}_{\Lambda\Omega}(P)$ will denote the $\Lambda\Omega$ -normal form of P .

The algorithm is defined as a formal system, proving statements of the shape:

$$A, A' \Rightarrow_D^c C[.],$$

where $A \not\ll_c A'$ and $C[.]$ is a context.

The design of the algorithm follows the same pattern as the separability algorithm, but for some rules dealing with Ω ; it is presented in Fig. 11.2 (pag. 138). For the sake of simplicity, we assume that all bound and free variables have different names.

Lemma 11.1.47. *Let A, A' be two approximants, $r \geq \max\{\text{args}(A), \text{args}(A')\}$ and $C_x^r[.] \equiv (\lambda x.[.])B^r$.*

- (i) *There is $\bar{A} \in \mathcal{A}$ such that $C_x^r[A] \rightarrow_{\Lambda\Omega}^* \bar{A}$ and $r \geq \text{args}(\bar{A})$.*
- (ii) *If $A \not\ll_c A'$ then $\text{nf}_{\Lambda\Omega}(C_x^r[A]) \not\ll_c \text{nf}_{\Lambda\Omega}(C_x^r[A'])$.*

Proof. The proof is quite similar to the proof of Lemma 2.1.15, by replacing Λ -NF by \mathcal{A} and $\not\ll_c$ by $\not\ll_c$.

- (i) By induction on A .
- (ii) By induction on c . \square

Now we can prove that the algorithm is correct and complete.

Let A, A' be approximants such that $A \not\ll_c A'$ and $r \geq \max\{\text{args}(A), \text{args}(A')\}$.

The rules of the system proving statements $A, A' \Rightarrow_D^c C[.]$, are the following:

$$\frac{}{xA_1 \dots A_m, \Omega \Rightarrow_D^c (\lambda x.[.])O^m} (D1)$$

$$\frac{xA_1 \dots A_m, \Omega \Rightarrow_D^c C[.]}{\lambda x_1 \dots x_n. xA_1 \dots A_m, \Omega \Rightarrow_D^c C[[.]x_1 \dots x_n]} (D2)$$

$$\frac{p \leq q \quad xA_1 \dots A_m x_{p+1} \dots x_q, yA'_1 \dots A'_n \Rightarrow_D^c C[.]}{\lambda x_1 \dots x_p. xA_1 \dots A_m, \lambda x_1 \dots x_q. yA'_1 \dots A'_n \Rightarrow_D^c C[[.]x_1 \dots x_q]} (D3)$$

$$\frac{q < p \quad xA_1 \dots A_m, yA'_1 \dots A'_n x_{q+1} \dots x_p \Rightarrow_D^c C[.]}{\lambda x_1 \dots x_p. xA_1 \dots A_m, \lambda x_1 \dots x_q. yA'_1 \dots A'_n \Rightarrow_D^c C[[.]x_1 \dots x_p]} (D4)$$

$$\frac{n < m}{xA_1 \dots A_m, xA'_1 \dots A_n \Rightarrow_D^c (\lambda x.[.])O^m \underbrace{I \dots I}_{m-n-2} KI\Omega} (D5)$$

$$\frac{m < n}{xA_1 \dots A_m, xA'_1 \dots A'_n \Rightarrow_D^c (\lambda x.[.])O^n \underbrace{I \dots I}_{n-m-2} K\Omega I} (D6)$$

$$\frac{x \not\equiv y}{xA_1 \dots A_m, yA'_1 \dots A'_n \Rightarrow_D^c (\lambda xy.[.])(\lambda x_1 \dots x_m. I)(\lambda x_1 \dots x_n. \Omega)} (D7)$$

$$\frac{x \notin \text{FV}(A_k) \cup \text{FV}(A'_k) \quad A_k \not\ll A'_k \quad A_k, A'_k \Rightarrow_D^c C[.]}{xA_1 \dots A_m, xA'_1 \dots A'_m \Rightarrow_D^{k,c} C[(\lambda x.[.])U_m^k]} (D8)$$

$$\frac{x \in \text{FV}(A_k) \cup \text{FV}(A'_k) \quad A_k \not\ll A'_k \quad C_x^r[.] \equiv (\lambda x.[.])B^r \quad \text{nf}_{\Lambda\Omega}(C_x^r[A_k]), \text{nf}_{\Lambda\Omega}(C_x^r[A'_k]) \Rightarrow_D^c C[.]}{xA_1 \dots A_m, xA'_1 \dots A'_m \Rightarrow_D^{k,c} C[C_x^r[.] \underbrace{I \dots I}_{r-m} U_r^k]} (D9)$$

Fig. 11.2. Semiseparability algorithm

Lemma 11.1.48 (Termination).

If $A \ll_c A'$ then $A, A' \Rightarrow_D^c C[.]$.

Proof. The proof can be done by induction on c . The proof follows essentially the same pattern as the proof of termination of the Λ -separability algorithm (see Lemma 2.1.17). The only different cases are rules (D1) and (D2), and in both cases the proof is immediate. \square

Lemma 11.1.49 (Correctness).

Let $A \ll_c A'$, for some path c .

If $A, A' \Rightarrow_D^c C[.]$ then $C[A] \rightarrow_{\Lambda\Omega}^* I$, while $C[A'] \rightarrow_{\Lambda\Omega}^* \Omega$.

Proof. By induction on the derivation of $A, A' \Rightarrow_A^c C[.]$.

(D1),(D2) Obvious.

(D3),(D4),(D5),(D6),(D7),(D8) Respectively similar to case (A1), (A2), (A3), (A4), (A5) and (A6) in the proof of correctness of the Λ -separability algorithm (see Lemma 2.1.18).

(D9) The proof is similar to the proof of rule (A7) of the proof of correctness of the Λ -Separability algorithm, using Lemma 11.1.47 instead of Lemma 2.1.15. \square

▼ Proof of the Semiseparability Theorem (Theorem 11.1.17 pag. 125).

By Lemmas 11.1.48 and 11.1.49. \blacksquare

The proof of the \mathcal{H} -discriminability theorem is based on the semiseparability property. In fact, we will prove that, given two terms M and N such that $M \not\sqsubseteq_{\mathcal{H}} N$, it is always possible to find two approximants $A \in \mathcal{A}(M)$ and $A' \in \mathcal{A}(N)$ and a path c such that $A \ll_c A'$, and the context $C[.]$ such that $A, A' \Rightarrow_D^c C[.]$ can be easily transformed in a context discriminating M and N . In order to choose the correct approximants, we need a lemma, based on the following formalization of a “strict” version of \ll_c .

Definition 11.1.50. $A \ll_c^\circ A'$ if and only if one of the following cases arises:

- (a) $A \equiv \Omega$ and $A' \not\equiv \Omega$;
- (b) $c \equiv j, c'$, $A =_\eta \lambda x_1 \dots x_n . x A_1 \dots A_m$, $A' =_\eta \lambda x_1 \dots x_n . x A'_1 \dots A'_m$ where $j \leq m$, and $A_j \ll_{c'}^\circ A'_j$.

Moreover, $A \in \mathcal{A}(M)$ is maximal along c in $\mathcal{A}(M)$ if and only if there is no $A' \in \mathcal{A}(M)$ such that $A \ll_c^\circ A'$.

It is easy to check that $A \ll_c^\circ A'$ implies $A \ll_c A'$, while the opposite does not hold; in particular, $A \not\ll_c^\circ A$, for all $A \in \mathcal{A}$.

Example 11.1.51. Let E_∞ be defined as in Sect. 11.1, before Theorem 11.1.31. $\lambda x. x I \Omega$ is maximal along 1 in $\mathcal{A}(\lambda x. x I E_\infty)$, although $\lambda x. x I \Omega$ is not maximal along 2 in $\mathcal{A}(\lambda x. x E_\infty I)$. In particular, there is no maximal approximant along paths starting with 2 in $\mathcal{A}(\lambda x. x I E_\infty)$.

Property 11.1.52. Let $A \in \mathcal{A}$ be such that $B \vdash_{\infty} A : \sigma$. Let N be such that, for each $A' \in \mathcal{A}(N)$:

$$A \llcorner c A' \text{ implies both } \begin{cases} (a) \text{ } A, A' \text{ are structurally similar in } c, \\ (b) \text{ } A' \text{ is not maximal along } c \text{ in } \mathcal{A}(N). \end{cases}$$

Then $B \vdash_{\infty} N : \sigma$.

Proof. Let σ be a type generated from the following grammar:

$$\sigma ::= \omega \mid \sigma_1 \wedge \dots \wedge \sigma_n \mid \sigma_1 \rightarrow \dots \rightarrow \sigma_n \rightarrow \phi.$$

If $B \vdash_{\infty} A : \tau$ where $\tau \in T(C_{\infty})$ then by Property 11.1.36.(i), there is a σ produced from the previous grammar such that $\tau \simeq_{\infty} \sigma$; hence there is no loss of generality by considering only this kind of types.

The proof is given by induction on the pair (A, σ) endowed by the lexicographic order, where σ is a type of the considered grammar. If $\sigma \equiv \omega$ then the proof is trivial. If $\sigma \equiv \sigma_1 \wedge \dots \wedge \sigma_k$ for some $k \geq 1$ then the proof follows by induction. Thus, let $A \not\equiv \Omega$ and $\sigma \equiv \sigma_1 \rightarrow \dots \rightarrow \sigma_k \rightarrow \phi$ for some $k \geq 1$.

- Let $A \equiv xA_1\dots A_m$ for some $m \in \mathbb{N}$.

Clearly $\Omega \in \mathcal{A}(N)$ and $A \llcorner c \Omega$, for all paths c ; but Ω is not maximal along c in $\mathcal{A}(N)$, by hypotheses. Namely, there is $A' \in \mathcal{A}(N)$ such that $\Omega \ll_c^{\circ} A'$, i.e. $A' \not\equiv \Omega$, by definition of \ll_c° . So, there is $n \geq \max\{k, m\}$ such that $A =_{\eta} \lambda y_1\dots y_n. xA_1\dots A_my_1\dots y_n$ and $A' =_{\eta} \lambda y_1\dots y_n. xA'_1\dots A'_{m+n} \in \mathcal{A}(N)$ by hypothesis (a). Actually, all approximants of N but Ω have the shape of A' ; furthermore, $N =_{\Lambda\eta} \lambda y_1\dots y_n. xN_1\dots N_{m+n}$ by Property 11.1.20.(iii), where $A'_i \in \mathcal{A}(N_i)$ ($1 \leq i \leq m+n$). There are two cases.

- Let $k > m$. So $B \vdash_{\infty} \lambda y_1\dots y_n. xA_1\dots A_my_1\dots y_n : \sigma_0 \rightarrow \dots \rightarrow \sigma_k \rightarrow \phi$, by Property 11.1.8. By Property 11.1.36.(ii),

$$B \vdash_{\infty} \lambda y_1\dots y_n. xA_1\dots A_my_1\dots y_n : \sigma_0 \rightarrow \dots \rightarrow \sigma_k \rightarrow \underbrace{\omega \rightarrow \dots \rightarrow \omega}_{n-k} \rightarrow \phi.$$

By Lemma 10.1.7.(vi)

$$B[\sigma_1/y_1, \dots, \sigma_k/y_k, \omega/y_{k+1}, \dots, \omega/y_n] \vdash_{\infty} xA_1\dots A_my_1\dots y_n : \phi.$$

So $B[\sigma_1/y_1, \dots, \sigma_k/y_k, \omega/y_{k+1}, \dots, \omega/y_n] \vdash_{\infty} x : \sigma'_1 \rightarrow \dots \rightarrow \sigma'_{n+m} \rightarrow \phi$, $B[\sigma_1/y_1, \dots, \sigma_k/y_k, \omega/y_{k+1}, \dots, \omega/y_n] \vdash_{\infty} A_i : \sigma'_i$ ($1 \leq i \leq m$) and $B[\sigma_1/y_1, \dots, \sigma_k/y_k, \omega/y_{k+1}, \dots, \omega/y_n] \vdash_{\infty} y_i : \sigma'_{i+m}$ ($1 \leq i \leq n$) by Lemma 10.1.7.(vii). Note that, both $\sigma_i \leq_{\infty} \sigma'_{i+m}$ ($1 \leq i \leq k$) and $\omega \leq_{\infty} \sigma'_{i+m}$ ($k+1 \leq i \leq n$) by Lemma 10.1.7.(ii). By Property 10.1.6

$$\sigma'_1 \rightarrow \dots \rightarrow \sigma'_{n+m} \rightarrow \phi \leq_{\infty} \sigma'_1 \rightarrow \dots \rightarrow \sigma'_m \rightarrow \sigma_1 \rightarrow \dots \rightarrow \sigma_k \rightarrow \underbrace{\omega \rightarrow \dots \rightarrow \omega}_{n-k} \rightarrow \phi$$

and by rule (\leq_{∞})

$$\begin{aligned} B[\sigma_1/y_1, \dots, \sigma_k/y_k, \omega/y_{k+1}, \dots, \omega/y_n] \vdash_{\infty} x : \\ \sigma'_1 \rightarrow \dots \rightarrow \sigma'_m \rightarrow \sigma_1 \rightarrow \dots \rightarrow \sigma_k \rightarrow \underbrace{\omega \rightarrow \dots \rightarrow \omega}_{n-k} \rightarrow \phi. \end{aligned}$$

If $A'_i \in \mathcal{A}(N)$ ($1 \leq i \leq m$) then it would be clear that

$$A_i \not\ll_c A'_i \text{ implies both } \begin{cases} (a) A_i, A'_i \text{ are structurally similar in } c, \\ (b) A'_i \text{ is not maximal along } c \text{ in } \mathcal{A}(N_i). \end{cases}$$

Thus $B[\sigma_1/y_1, \dots, \sigma_k/y_k, \omega/y_{k+1}, \dots, \omega/y_n] \vdash_\infty N_i : \sigma'_i$ ($1 \leq i \leq m$) by induction on A .

$B[\sigma_1/y_1, \dots, \sigma_k/y_k, \omega/y_{k+1}, \dots, \omega/y_n] \vdash_\infty y_i : \sigma_i$ ($1 \leq i \leq k$), by rule (var) ; moreover, if $A'_i \in \mathcal{A}(N)$ ($m+1 \leq i \leq m+k$) then it would be clear that

$$y_i \not\ll_c A'_i \text{ implies both } \begin{cases} (a) y_i, A'_i \text{ are structurally similar in } c, \\ (b) A'_i \text{ is not maximal along } c \text{ in } \mathcal{A}(N_i). \end{cases} \text{ So}$$

$B[\sigma_1/y_1, \dots, \sigma_k/y_k, \omega/y_{k+1}, \dots, \omega/y_n] \vdash_\infty N_i : \sigma_i$ ($m+1 \leq i \leq m+k$) by induction on σ in case $m=0$, by induction on A otherwise. Moreover, $B[\sigma_1/y_1, \dots, \sigma_k/y_k, \omega/y_{k+1}, \dots, \omega/y_n] \vdash_\infty N_i : \omega$ ($m+k+1 \leq i \leq m+n$) by rule (ω) .

Hence $B[\sigma_1/y_1, \dots, \sigma_k/y_k, \omega/y_{k+1}, \dots, \omega/y_n] \vdash_\infty xN_1\dots N_{m+n} : \phi$ by $m+n$ applications of rule $(\rightarrow E)$, that implies $B \vdash_\infty \lambda y_1\dots y_n. xN_1\dots N_{m+n} : \sigma$ by n applications of rule $(\rightarrow I)$ and by an application of (\leq_∞) . Since typings are preserved by $=_{\Lambda\eta}$, the proof follows.

– In case $k \leq m$ the proof is simpler.

- Let $A \equiv \lambda x.\bar{A}$ and $\sigma \equiv \mu \rightarrow \nu$ (i.e. $k \geq 1$). Clearly $\Omega \in \mathcal{A}(N)$ and $A \not\ll_c \Omega$, so by hypothesis (b) there is $A' \in \mathcal{A}(N)$ such that $\Omega \ll_c^o A'$ and $A' =_{\Lambda\eta} \lambda x.A^*$. So by Property 11.1.20.(ii), $N =_{\Lambda\eta} \lambda x.N'$ and $A^* \in \mathcal{A}(N')$. Moreover, $B[\mu/x] \vdash_\infty \bar{A} : \nu$ by Lemma 10.1.7.(vi).

Note that \bar{A} and N' satisfy the hypothesis of this property (indeed, $\not\ll_c$ is defined modulo $=_{\Lambda\eta}$), thus $B[\mu/x] \vdash_\infty N' : \nu$ by induction. This implies $B \vdash_\infty \lambda x.N' : \mu \rightarrow \nu$, by rule $(\rightarrow I)$.

- Let $A \equiv \lambda x.\bar{A}$ and $k=0$. Since $\phi \simeq_\infty \omega \rightarrow \phi$, the proof can be done as in the previous point. \square

Example 11.1.53.

$$\mathcal{A}(E_\infty) = \{\Omega, \lambda z_0 z_1. z_0 \Omega, \lambda z_0 z_1. z_0 (\lambda z_2. z_1 (\dots (\lambda z_{n+1}. z_n \Omega) \dots)) \mid n \geq 0\}.$$

It is easy to see that $I \not\ll A'$, for all $A' \in \mathcal{A}(E_\infty)$.

Let $A_n \equiv \lambda z_0 z_1. z_0 (\lambda z_2. z_1 (\dots (\lambda z_{n+1}. z_n \Omega) \dots))$, therefore $A_n \ll_c^o A_{n+1}$ where $c \equiv \underbrace{1\dots 1}_n$. It is easy to see that $B \vdash_\infty I : \sigma$ implies that there is $A \in \mathcal{A}(E_\infty)$

such that $B \vdash_\infty A : \sigma$, by the previous property.

This agrees with the Theorem 11.1.31.

The next lemma shows that, if $M \not\sqsubseteq_{\mathcal{H}} N$, there are always two approximants of them which are the “witness” of the difference between M and N , and such that the context semiseparating the two approximants can be transformed in a context discriminating the two terms.

Lemma 11.1.54. $M \not\sqsubseteq_{\mathcal{H}} N$ implies that there are two approximants $A \in \mathcal{A}(M)$ and $A' \in \mathcal{A}(N)$, and a path c such that $A \not\ll_c A'$, and either A and A' are structurally different in c , or A' is maximal along c in $\mathcal{A}(N)$.

Proof. $M \not\sqsubseteq_{\mathcal{H}} N$ means that there are a basis B and a type σ such that $B \vdash_{\infty} M : \sigma$ while $B \not\vdash_{\infty} N : \sigma$. By the approximation theorem, this means that there is $A \in \mathcal{A}(M)$ such that $B \vdash A : \sigma$, while there is not $A' \in \mathcal{A}(N)$, such that $B \vdash A' : \sigma$. This implies: $A \not\ll A'$, for all $A' \in \mathcal{A}(N)$.

The proof follows by Property 11.1.52. \square

Now we are ready to prove the \mathcal{H} -discriminability theorem.

▼ Proof of the \mathcal{H} -Discriminability Theorem (Theorem 11.1.25 pag. 127).

Let $M \not\sqsubseteq_{\mathcal{H}} N$. Choose $A \in \mathcal{A}(M)$, $A' \in \mathcal{A}(N)$ and a path c satisfying Lemma 11.1.54. Hence $A, A' \Rightarrow_D^c C[.]$ such that $C[A] \rightarrow_{\Lambda\Omega}^* I$ and $C[A'] \rightarrow_{\Lambda\Omega}^* \Omega$. Let $C'[.]$ be the context obtained by $C[.]$ by replacing every occurrence of Ω by a Λ -unsolvable term (say DD). We will prove that $C'[.]$ is a \mathbf{H} -discriminating context between M and N , by induction on the derivation d of $A, A' \Rightarrow^c C[.]$. If the only used rule is an axiom, then if the axiom is either (D5) or (D6) or (D7), the proof follows from the fact that M and N have both Λ -hnf, and by Property 11.1.20.(iii), their Λ -hnf's have the same shape respectively of A and A' . If axiom (D1) was used, then $A' \equiv \Omega$ and A' is maximal along the empty path in $\mathcal{A}(N)$. Hence, N is Λ -unsolvable and $C[.]$ is the discriminating context.

If the last rule applied is (D2), (D3), (D4), (D8) or (D9), then the proof follows by induction, always taking into account the fact that M and N both have Λ -hnf, and by Property 11.1.20.(iii), their Λ -hnf's have the same shape respectively of A and A' . \blacksquare

Example 11.1.55. Let $M \equiv \lambda x.xII$ and $N \equiv \lambda x.xE_{\infty}(DD)$.

Let B_{ω} be the basis such that $B_{\omega}(x) = \omega$, for all $x \in \text{Var}$; it is easy to see that:

$$\begin{aligned} B_{\omega} \vdash_{\infty} M : (\omega \rightarrow (\phi \rightarrow \phi) \rightarrow \phi \rightarrow \phi) \rightarrow \phi \rightarrow \phi, \\ B_{\omega} \not\vdash_{\infty} N : (\omega \rightarrow (\phi \rightarrow \phi) \rightarrow \phi \rightarrow \phi) \rightarrow \phi \rightarrow \phi. \end{aligned}$$

Hence $M \not\sqsubseteq_{\mathcal{H}} N$.

Two approximants respectively of $\lambda x.xII$ and $\lambda x.xE_{\infty}(DD)$ and a path satisfying the conditions of Lemma 11.1.54 are $\lambda x.x\Omega I$, $\lambda x.x\Omega\Omega$ and path 2. In fact, all approximants of $\lambda x.xE_{\infty}(DD)$ are of the shape $\lambda x.xA\Omega$, where $A \in \mathcal{A}(E_{\infty})$, and so $\lambda x.x\Omega\Omega$ is maximal along path 2 in $\mathcal{A}(N)$.

$\lambda x.x\Omega I, \lambda x.x\Omega\Omega \Rightarrow_D^2 C[.]$ where $C[.] \equiv \lambda y.(\lambda x.([.]x))U_2^2y$; in fact,

$$\begin{array}{c} \overline{y, \Omega \Rightarrow_D^{\epsilon} \lambda y.[.]}^{(D1)} \\ \overline{\lambda y.y, \Omega \Rightarrow_D^{\epsilon} \lambda y.[.]y}^{(D2)} \\ \overline{x\Omega I, x\Omega\Omega \Rightarrow_D^2 \lambda y.(\lambda x.([.]x))U_2^2y}^{(D8)} \\ \overline{\lambda x.x\Omega I, \lambda x.x\Omega\Omega \Rightarrow_D^2 \lambda y.(\lambda x.([.]x))U_2^2y}^{(D3)} \end{array}$$

It is easy to see that $C[\lambda x.x\Omega I] \rightarrow_{\Lambda\Omega} I$ and $C[\lambda x.xII] \Downarrow_{\mathbf{H}} I$, while $C[\lambda x.x\Omega\Omega] \rightarrow_{\Lambda\Omega} \Omega$ and $C[\lambda x.xE_{\infty}(DD)] \uparrow_{\mathbf{H}}$.

Finally, the proof of the \mathcal{H} -characterization theorem can be done.

▼ **Proof of the \mathcal{H} -Characterization Theorem** (Theorem 11.1.30 pag. 128).

($M \sqsubseteq_{\mathcal{H}} N$ implies $\forall k \geq 0, M \sqsubseteq_k N$) We will prove that if there is k such that $M \not\sqsubseteq_k N$ then there are $A \in \mathcal{A}(M)$ and $A' \in \mathcal{A}(N)$ and a path c satisfying the conditions of Lemma 11.1.54. Then, by the proof of the \mathcal{H} -discriminability theorem, there is a context $C[.]$ discriminating between M and N , i.e. $C[M] \rightarrow_{\Lambda}^* I$ and $C[N]$ is a Λ -unsolvable. By the completeness of the model with respect to the **H**-operational semantics, this implies $M \not\sqsubseteq_{\mathcal{H}} N$. The existence of $A \in \mathcal{A}(M)$ and $A' \in \mathcal{A}(N)$ and a path c satisfying the conditions of Lemma 11.1.54 can be proved by induction on k . If $k = 0$, then there are two cases.

- (i) N is Λ -unsolvable and M is Λ -solvable; take an $A \not\equiv \Omega$ and the only approximant of N , namely $A' \equiv \Omega$: so the empty path satisfies the given conditions.
- (ii) Every Λ -hnf of M is of the shape $\lambda x_1 \dots x_n . x M_1 \dots M_m$, every Λ -hnf of N is of the shape $\lambda x_1 \dots x_p . y N_1 \dots N_q$, and either $x \not\equiv y$ or $|m - n| \neq |p - q|$. So every $A \in \mathcal{A}(M)$ and $A' \in \mathcal{A}(N)$ such that $A, A' \not\equiv \Omega$ and the empty path satisfy the given conditions.

If $k \geq 1$, then $M =_{\Lambda\eta} \lambda x_1 \dots x_n . x M_1 \dots M_m$, $N =_{\Lambda\eta} \lambda x_1 \dots x_n . x N_1 \dots N_m$ and $M_i \not\sqsubseteq_{k-1} N_i$, for some $1 \leq i \leq m$. Then by induction there are $A^* \in \mathcal{A}(M_i)$, $A^{**} \in \mathcal{A}(N_i)$ and a path c satisfying Lemma 11.1.54.

Hence, take approximants $A \in \mathcal{A}(M)$ and $A' \in \mathcal{A}(N)$ such that both

$$A =_{\eta} \lambda x_1 \dots x_n . x \underbrace{\Omega \dots \Omega}_{i-1} A^* \underbrace{\Omega \dots \Omega}_{m-i} \text{ and } A' =_{\eta} \lambda x_1 \dots x_n . x \underbrace{\Omega \dots \Omega}_{i-1} A^{**} \underbrace{\Omega \dots \Omega}_{m-i}$$

so the path i, c satisfies the conditions.

($\forall k \geq 0, M \sqsubseteq_k N$ implies $M \sqsubseteq_{\mathcal{H}} N$) Assume $M \not\sqsubseteq_{\mathcal{H}} N$; by Lemma 11.1.54 there are $A \in \mathcal{A}(M)$, $A' \in \mathcal{A}(N)$ and there is a path c such that $A \not\ll_c A'$ and either A, A' are structurally different in c , or A' is maximal along c in $\mathcal{A}(N)$. Note that $A \not\equiv \Omega$, since $\Omega \ll A'$.

By induction on c , we prove that there is k such that $M \not\sqsubseteq_k N$.

- Let $c \equiv \epsilon$ and $A' \equiv \Omega$; so A' is maximal implies that N is Λ -unsolvable and so $M \not\sqsubseteq_0 N$.
- Let $A \equiv \lambda x_1 \dots x_n . x A_1 \dots A_m$ and $A' \equiv \lambda x_1 \dots x_p . y A_1 \dots A_q$, where either $|n - m| \neq |p - q|$ or $x \not\equiv y$; so $M \not\sqsubseteq_0 N$.
- Let $c \equiv i, c'$ and $A =_{\eta} \lambda x_1 \dots x_n . x A_1 \dots A_m$, $A' =_{\eta} \lambda x_1 \dots x_n . x A_1 \dots A_m$. Then $A_i \not\ll_{c'} A'_i$ and the proof follows by induction and by Property 11.1.20.(iii). ■

Remark 11.1.56. It is possible to transform the semiseparability algorithm in such a way that it always produces a head context. So, from the \mathcal{H} -characterization theorem and from the completeness of \mathcal{H} with respect to the **H**-operational semantics it follows that **H** is head-discriminable.

11.2 The Model \mathcal{N}

In this section, we will introduce the filter model \mathcal{N} , which is fully abstract with respect to the \mathbf{N} -operational semantics.

By keeping in mind Property 10.1.15 and by observing that, if N has Λ -nf and M has not Λ -nf then $M \prec_{\mathbf{N}} N$, in order to define a model correct with respect to the \mathbf{N} -operational semantics, it is sufficient to ask for the property:

$$\forall M, N \text{ if } M \text{ has } \Lambda\text{-nf and } N \text{ has not } \Lambda\text{-nf then } N \sqsubset_{\mathcal{N}} M.$$

So we must define a legal type system ∇ based on a set of constants C such that there is a type σ characterizing the terms having Λ -nf. While the class of Λ -nf's is not closed under application, as the class of Λ -solvable terms, such a characterization is possible, thanks to the existence of a particular class of terms, which is a proper subclass of Λ -NF, and such that the two classes are mutually closed under application.

Definition 11.2.1. A term M has a Λ -persistent normal form (Λ -pnf) if and only if it has Λ -normal form and moreover, for each sequence \vec{N} of terms having Λ -nf, $M\vec{N}$ has a Λ -persistent normal form too.

The class of Λ -persistent normal forms is not empty, since at least the variables belong to it.

Example 11.2.2. If $M \rightarrow_{\Lambda}^{*} \lambda x_1 \dots x_n . z M_1 \dots M_m \in \Lambda\text{-NF}$ and $z \notin \{x_1, \dots, x_n\}$ then M has a Λ -pnf.

Now an alternative definition of Λ -normal forms can be given, thanks to the following property.

Property 11.2.3. A term M has a Λ -nf if and only if, for each sequence \vec{N} of terms having Λ -pnf, $M\vec{N}$ has a Λ -nf too.

Proof. Let $\vec{N} \equiv N_1, \dots, N_h$ ($h \geq 1$).

(\Rightarrow) Let $M \rightarrow_{\Lambda}^{*} \bar{M} \in \Lambda\text{-NF}$; we reason by induction on \bar{M} .

If $\bar{M} \equiv z M_1 \dots M_m$ ($m \in \mathbb{N}$) then the proof is immediate; otherwise $\bar{M} \equiv \lambda x_1 \dots x_n . z M_1 \dots M_m$ ($n \geq 1$). If $h \leq n$ then $M\vec{N} \rightarrow_{\Lambda}^{*} \lambda x_{h+1} \dots x_n . \zeta M'_1 \dots M'_m$ where $\zeta \equiv z[N_1/x_1, \dots, N_h/x_h]$ and $M'_j \equiv M_j[N_1/x_1, \dots, N_h/x_h] =_{\Lambda} (\lambda x_1 \dots x_n . M_j)\vec{N}$ ($1 \leq j \leq m$). Since N_i ($1 \leq i \leq h$) has a Λ -pnf, by induction M'_j has a Λ -nf ($1 \leq j \leq m$).

If $\zeta \equiv z$ then the proof is given. Otherwise $\zeta \equiv N_j$, for some j , and the proof follows from the definition of Λ -persistent normal form.

Let $h > n$, $\vec{P} \equiv N_1 \dots N_n$, $\vec{Q} \equiv N_{n+1} \dots N_h$; so $M\vec{P}\vec{Q} \rightarrow_{\Lambda}^{*} \zeta M'_1 \dots M'_m \vec{Q}$, where $M'_j \equiv M_j[N_1/x_1, \dots, N_n/x_n] =_{\Lambda} (\lambda x_1 \dots x_n . M_j)\vec{P}$ and $\zeta \equiv z[N_1/x_1, \dots, N_n/x_n]$ ($1 \leq j \leq m$). M'_j has a Λ -nf ($1 \leq j \leq m$) by induction, while each term in \vec{Q} has a Λ -nf too by hypothesis. If $\zeta \equiv z$ then the proof is trivial; otherwise it follows from the fact that ζ has Λ -pnf.

(\Leftarrow) It is easy to check that Mx has Λ -nf if and only if M has Λ -nf, so the proof is done since x is a Λ -pnf. \square

So the class of Λ -normal forms and the class of Λ -persistent normal forms are mutually closed under application.

We will take the set of constants $C_{\bowtie} = \{\phi, \psi, \omega\}$ and we will use the type constants ϕ and ψ , for characterizing respectively the class of Λ -normal forms and the class of Λ -persistent normal forms. Since a Λ -persistent normal form is a Λ -normal form too, it must be that $\psi \leq \phi$. Moreover, by definition of Λ -persistent normal form, it must be that $\psi \simeq \underbrace{\phi \rightarrow \dots \rightarrow \phi}_n \rightarrow \psi$, and

$\phi \simeq \underbrace{\psi \rightarrow \dots \rightarrow \psi}_n \rightarrow \phi$, for all $n \geq 1$, by Property 11.2.3 ($n \in \mathbb{N}$). We want also $\omega \simeq \omega \rightarrow \omega$, since all terms must have a functional behavior.

Definition 11.2.4. Let $C_{\bowtie} = \{\phi, \psi, \omega\}$ and $I(C_{\bowtie}) = T(C_{\bowtie})$.

\bowtie is the type system $< C_{\bowtie}, \leq_{\bowtie}, I(C_{\bowtie}) >$, where \leq_{\bowtie} is the least intersection relation induced from the rules in Fig. 11.3.

$\sigma \leq_{\bowtie} \omega$ <small>(a)</small>	$\sigma \leq_{\bowtie} \sigma \wedge \sigma$ <small>(b)</small>	$\sigma \wedge \tau \leq_{\bowtie} \sigma$ <small>(c)</small>	$\sigma \wedge \tau \leq_{\bowtie} \tau$ <small>(c')</small>
$(\sigma \rightarrow \tau) \wedge (\sigma \rightarrow \pi) \leq_{\bowtie} \sigma \rightarrow (\tau \wedge \pi)$ <small>(d)</small>		$\sigma \rightarrow \omega \leq_{\bowtie} \omega \rightarrow \omega$ <small>(g)</small>	
$\sigma \leq_{\bowtie} \sigma', \tau \leq_{\bowtie} \tau'$ <small>(e)</small>	$\sigma' \leq_{\bowtie} \sigma, \tau \leq_{\bowtie} \tau'$ <small>(f)</small>	$\sigma \leq_{\bowtie} \sigma$ <small>(r)</small>	$\sigma \leq \rho \quad \rho \leq \tau$ <small>(t)</small>
$\psi \leq_{\bowtie} \phi$ <small>(n0)</small>	$\psi \leq_{\bowtie} \phi \rightarrow \psi$ <small>(n1)</small>	$\phi \rightarrow \psi \leq_{\bowtie} \psi$ <small>(n2)</small>	
$\phi \leq_{\bowtie} \psi \rightarrow \phi$ <small>(n3)</small>	$\psi \rightarrow \phi \leq_{\bowtie} \phi$ <small>(n4)</small>	$\omega \leq_{\bowtie} \omega \rightarrow \omega$ <small>(n5)</small>	

Fig. 11.3. \bowtie -intersection relation

Note that $\phi \simeq_{\bowtie} \psi \rightarrow \phi$, $\psi \simeq_{\bowtie} \phi \rightarrow \psi$ and $\omega \simeq_{\bowtie} \omega \rightarrow \omega$. Some key properties of the \bowtie -intersection relation are shown in the Sect. 11.2.1.

Theorem 11.2.5. The type system \bowtie is legal.

Proof. The proof is in Sect. 11.2.1. \square

So the model \mathcal{N} can be defined.

Definition 11.2.6. \mathcal{N} is the $\lambda\Lambda$ -model $< \mathcal{F}(\bowtie), \mathcal{F}(\bowtie), \circ_{\bowtie}, [\cdot]^{F(\bowtie)} >$.

We will prove that the type assignment system \vdash_{\bowtie} characterizes the terms having normal form. Let B_ψ be the basis assigning ψ to each variable; the typing $B_\psi \vdash_{\bowtie} M : \phi$ can be proved if and only if M has Λ -normal form. So the correctness of \mathcal{N} with respect to the \mathbf{N} -operational semantics follows by Theorem 10.0.4.

Lemma 11.2.7. *If M has a Λ -normal form then there is a derivation proving $B_\psi \vdash_{\bowtie} M : \phi$, where $B_\psi(x) = \psi$ for every x .*

Proof. Since \mathcal{N} is a $\lambda\Lambda$ -model, by Lemma 10.1.13.(ii), the \bowtie -type assignment system is closed under Λ -reduction; so, let $M \equiv \lambda x_1 \dots x_n. x M_1 \dots M_m$ where $M_i \in \Lambda\text{-NF}$ ($1 \leq i \leq m$), without loss of generality. The proof is given by induction on M . Thus $B_\psi \vdash_{\bowtie} M_i : \phi$ ($1 \leq i \leq m$) by induction. But $B_\psi \vdash_{\bowtie} x : \psi$ by rule *(var)*, hence $B_\psi \vdash_{\bowtie} x : \underbrace{\phi \rightarrow \dots \rightarrow \phi}_m \rightarrow \psi$ by rule (\leq_{\bowtie}) . We obtain $B_\psi \vdash_{\bowtie} x M_1 \dots M_m : \psi$ by m application of the rule $(\rightarrow E)$, therefore $B_\psi \vdash_{\bowtie} \lambda x_1 \dots x_n. x M_1 \dots M_m : \underbrace{\psi \rightarrow \dots \rightarrow \psi}_n \rightarrow \phi$, by rule (\leq_{\bowtie}) and n applications of rule $(\rightarrow I)$. Yet, by applying the rule (\leq_{\bowtie}) the proof is done. \square

To prove the correctness of the model with respect to the \mathbf{N} -operational semantics, we need an approximation theorem. We will prove that such a theorem holds for the same definition of approximants given for the model \mathcal{H} . First, we need to prove that the theory induced by the model \mathcal{N} is a $\Lambda\eta$ -theory.

Property 11.2.8. $\sim_{\mathcal{N}}$ is a $\Lambda\eta$ -theory.

Proof. The theory induced by \mathcal{N} is a Λ -theory by Corollary 10.0.3, since \mathcal{N} is a $\lambda\Lambda$ -model by Theorem 11.0.1. In order to prove that it is also a $\Lambda\eta$ -theory, by Property 2.1.7, it is sufficient to prove that $I \sim_{\mathcal{N}} E$. We will prove both $I \sqsubseteq_{\mathcal{N}} E$ and $E \sqsubseteq_{\mathcal{N}} I$.

($I \sqsubseteq_{\mathcal{N}} E$) We prove that $B \vdash_{\bowtie} I : \sigma$ implies $B \vdash_{\bowtie} E : \sigma$.

By Property 11.2.27.(i), either $\sigma \not\simeq_{\bowtie} \omega$ or $\sigma \simeq_{\bowtie} \sigma_0 \wedge \dots \wedge \sigma_n$ ($n \geq 0$) such that $\forall i \leq n$, $\sigma_i \simeq_{\bowtie} \tau_1^i \rightarrow \dots \rightarrow \tau_{m_i}^i \rightarrow \rho$, $m_i \in \mathbb{N}$ and $\rho \in \{\phi, \psi\}$. The case $\sigma \simeq_{\bowtie} \omega$ is trivial, by rule *(ω)*; otherwise the proof can be done by induction on n .

Let $n = 0$, thus we can assume $\sigma \simeq_{\bowtie} \mu \rightarrow \nu \rightarrow \tau$, by Property 11.2.27.(ii). Note that $B \vdash_{\infty} I : \sigma$ implies $B \vdash_{\infty} I : \mu \rightarrow \nu \rightarrow \tau$, by rule (\leq_{∞}) ; thus $B[\mu/x] \vdash_{\bowtie} x : \nu \rightarrow \tau$ by Lemma 10.1.7.(vi), and $\mu \leq_{\bowtie} \nu \rightarrow \tau$ by Lemma 10.1.7.(iii). We can build the following derivation:

$$\begin{array}{c}
 \frac{}{B[\mu/x][\nu/y] \vdash_{\bowtie} x : \mu} \text{(var)} \quad \mu \leq_{\bowtie} \nu \rightarrow \tau \quad \frac{}{B[\mu/x][\nu/y] \vdash_{\bowtie} x : \nu \rightarrow \tau} \text{(\leq_{\bowtie})} \quad \frac{}{B[\mu/x][\nu/y] \vdash_{\bowtie} y : \nu} \text{(var)} \\
 \hline
 \frac{}{B[\mu/x][\nu/y] \vdash_{\bowtie} xy : \tau} \text{(\rightarrow I)} \quad \frac{}{B \vdash_{\bowtie} \lambda xy.xy : \nu \rightarrow \tau} \text{(\rightarrow I)} \\
 \frac{B[\mu/x] \vdash_{\bowtie} \lambda y.xy : \nu \rightarrow \tau \quad \mu \rightarrow \nu \rightarrow \tau \leq_{\bowtie} \sigma}{B \vdash_{\bowtie} \lambda xy.xy : \sigma} \text{(\leq_{\bowtie})}
 \end{array}$$

If $n \geq 1$ then the proof follows by inductive hypothesis.

($E \sqsubseteq_{\mathcal{N}} I$) We prove that $B \vdash_{\bowtie} E : \sigma$ implies $B \vdash_{\bowtie} I : \sigma$.

By Property 11.2.27.(i), either $\sigma \not\leq_{\bowtie} \omega$ or $\sigma \simeq_{\bowtie} \sigma_0 \wedge \dots \wedge \sigma_n$ ($n \geq 0$) such that $\forall i \leq n, \sigma_i \simeq_{\bowtie} \tau_1^i \rightarrow \dots \rightarrow \tau_{m_i}^i \rightarrow \rho$, $m_i \in \mathbb{N}$ and $\rho \in \{\phi, \psi\}$. The case $\sigma \simeq_{\bowtie} \omega$ is trivial, by rule (ω) ; otherwise the proof can be done by induction on n .

Let $n = 0$, thus we can assume $\sigma \simeq_{\bowtie} \mu \rightarrow \nu \rightarrow \tau$, by Property 11.2.27.(ii). Note that $B \vdash_{\bowtie} \lambda xy.xy : \sigma$ implies $B \vdash_{\bowtie} \lambda xy.xy : \mu \rightarrow \nu \rightarrow \tau$, by rule (\leq_{\bowtie}) ; thus, by Lemma 10.1.7.vi), this implies $B[\mu/x][\nu/y] \vdash_{\bowtie} xy : \tau$, which in turn implies $B[\mu/x][\nu/y] \vdash_{\bowtie} x : \theta \rightarrow \tau$ and $B[\mu/x][\nu/y] \vdash_{\bowtie} y : \theta$, for some θ , by Lemma 10.1.7.(vii). Hence, by Lemma 10.1.7.(iii), $\mu \leq_{\bowtie} \theta \rightarrow \tau$ and $\nu \leq_{\bowtie} \theta$ (so $\mu \leq_{\bowtie} \nu \rightarrow \tau$). Then we can build the following derivation:

$$\begin{array}{c}
 \frac{}{B[\mu/x] \vdash_{\bowtie} x : \mu} \text{(var)} \quad \mu \leq_{\bowtie} \nu \rightarrow \tau \quad \frac{}{B[\mu/x] \vdash_{\bowtie} x : \nu \rightarrow \tau} \text{(\leq_{\bowtie})} \\
 \hline
 \frac{}{B \vdash_{\bowtie} \lambda x.x : \mu \rightarrow \nu \rightarrow \tau} \text{(\rightarrow I)} \quad \mu \rightarrow \nu \rightarrow \tau \leq_{\bowtie} \sigma \quad \frac{}{B \vdash_{\bowtie} \lambda x.x : \sigma} \text{(\leq_{\bowtie})}
 \end{array}$$

If $n \geq 1$ then the proof follows by inductive hypothesis. \square

Let $\Lambda\Omega$ and the set $\mathcal{A}(M)$, for a term M , be defined as in Definitions 11.1.9 and 11.1.11. As before, we will use A, A' for ranging over the class of approximants. We can extend to $\Lambda\Omega$ the type assignment system \vdash_{\bowtie} , as we did for the system \vdash_{∞} . It is immediate to verify that \vdash_{\bowtie} is closed under $=_{\Lambda\Omega}$.

Theorem 11.2.9 (\mathcal{N} -Approximation).

$B \vdash_{\bowtie} M : \sigma$ if and only if $B \vdash_{\bowtie} A : \sigma$, for some $A \in \mathcal{A}(M)$.

Proof. The proof is in Sect. 11.2.3. \square

As already observed, Λ -normal forms are approximants.

Property 11.2.10. M has Λ -normal form if and only if there is $A \in \mathcal{A}(M)$ having no occurrences of the constant Ω .

Proof. Trivial. \square

Let B_ψ be the basis such that $B(x) = \psi$, for all $x \in \text{Var}$, as in Lemma 11.2.7; moreover, let B_ω be the basis such that $B_\omega(x) = \omega$, for all $x \in \text{Var}$.

Lemma 11.2.11. (i) $B_\psi \vdash_{\bowtie} M : \phi$ if and only if M has Λ -normal form.
(ii) $B \vdash_{\bowtie} M : \sigma$ and $\sigma \not\leq_{\bowtie} \omega$ if and only if M has a Λ -head normal form.

Proof. (i) (\Rightarrow) By the approximation theorem, $B_\psi \vdash_{\bowtie} M : \phi$ implies there is $A \in \mathcal{A}(M)$ such that $B_\psi \vdash_{\bowtie} A : \phi$. We will prove that A does not contain occurrences of Ω , by induction on A . Hence M has Λ -normal form, by Property 11.2.10.

$A \equiv \Omega$ is not possible, since types derivable for Ω are equivalent to ω .

If $A \equiv \lambda x_1 \dots x_n. z A_1 \dots A_m$ and $B_\psi \vdash_{\bowtie} A : \phi$ then, by rule (\leq_{\bowtie}) , $B_\psi \vdash_{\bowtie} z A_1 \dots A_m : \phi$ by Lemma 10.1.7.(vi).

By Lemma 10.1.7.(vii) this implies both (*) $B_\psi \vdash_{\bowtie} z : \sigma_1 \rightarrow \dots \rightarrow \sigma_m \rightarrow \phi$, and (***) $B_\psi \vdash_{\bowtie} A_i : \sigma_i$, for some $\sigma_1, \dots, \sigma_m$ ($1 \leq i \leq m$). (*) implies $\psi \leq_{\bowtie} \sigma_1 \rightarrow \dots \rightarrow \sigma_m \rightarrow \phi$ by Lemma 10.1.7.(iii) and so $\sigma_i \leq_{\bowtie} \phi$, since \bowtie is legal and $\psi \simeq_{\bowtie} \underbrace{\phi \rightarrow \dots \rightarrow \phi}_{m} \rightarrow \psi$.

Hence $B_\psi \vdash_{\bowtie} A_i : \phi$ ($1 \leq i \leq m$), by rule (\leq_{\bowtie}) . By inductive hypothesis A_i ($1 \leq i \leq m$) has no occurrences of Ω and the proof is done.

(\Leftarrow) By Lemma 11.2.7.

(ii) (\Rightarrow) If M is Λ -unsolvable then the only approximant of M is Ω , so the proof follows from the \mathcal{N} -approximation theorem.

(\Leftarrow) Let $M \rightarrow_A^* \lambda x_1 \dots x_n. z M_1 \dots M_m$, for some $n, m \in \mathbb{N}$.

Let $\mu \equiv \underbrace{\omega \rightarrow \dots \rightarrow \omega}_{m} \rightarrow \pi$, where $\pi \not\simeq \omega$. Clearly $B_\omega[\mu/z] \vdash_{\bowtie} M_i : \omega$ ($1 \leq i \leq m$) by rule (ω) ; so $B_\omega[\mu/z] \vdash_{\bowtie} z M_1 \dots M_m : \pi$ by m applications of rule $(\rightarrow E)$. If $z \notin \text{FV}(M)$ then, by applying n times the rule $(\rightarrow I)$,

$$B_\omega[\mu/z] \vdash_{\bowtie} \lambda x_1 \dots x_n. z M_1 \dots M_m : \underbrace{\omega \rightarrow \dots \rightarrow \omega}_n \rightarrow \pi.$$

So $B_\omega[\mu/z] \vdash_{\bowtie} M : \underbrace{\omega \rightarrow \dots \rightarrow \omega}_n \rightarrow \pi$, since the type assignment system is closed under $=_\Lambda$. If $z \equiv x_k$ for some $k \leq n$ then, it is easy to see that

$$B_\omega \vdash_{\bowtie} M : \underbrace{\omega \rightarrow \dots \rightarrow \omega}_{k-1} \rightarrow (\underbrace{\omega \rightarrow \dots \rightarrow \omega}_m \rightarrow \pi) \rightarrow \underbrace{\omega \rightarrow \dots \rightarrow \omega}_{n-k} \rightarrow \pi,$$

hence the proof follows by Theorem 11.2.23. \square

The order relation $\sqsubseteq_{\mathcal{N}}$ between approximants can be axiomatized through the same syntactical relation \ll defined in Definition 11.1.15 for the model \mathcal{H} . Namely $A \ll A'$ if and only if one of the two following cases arises:

- (i) $A \equiv \Omega$;
- (ii) $A =_{\eta} \lambda x_1 \dots x_n. z A_1 \dots A_m$ and $A' =_{\eta} \lambda x_1 \dots x_n. z A'_1 \dots A'_m$
and $A_i \ll A'_i$, for all i ($1 \leq i \leq m$), for some $n, m \in \mathbb{N}$.

Lemma 11.2.12. $A \sqsubseteq_{\mathcal{N}} A'$ if and only if $A \ll A'$.

Proof. (\Leftarrow) We prove that, if $A \ll A'$, then $B \vdash_{\bowtie} A : \sigma$ implies $B \vdash_{\bowtie} A' : \sigma$. The proof is by induction on the definition of \ll , and it is similar to the proof of the (\Rightarrow) part of the proof of Lemma 11.1.18, where, instead of Properties 11.1.36.(ii) and 11.1.8, Properties 11.2.27.(ii) and 11.2.8 must be respectively used.

(\Rightarrow) We prove, by contraposition, that $A \not\ll A'$ implies $A \not\sqsubseteq_{\mathcal{N}} A'$.

By the semiseparability theorem (Theorem 11.1.17), $A \not\ll A'$ implies there is a context $C[.]$ such that $C[A] \rightarrow_{\Lambda\Omega}^* I$ while $C[A'] \rightarrow_{\Lambda\Omega}^* \Omega$. Since the type assignment system is closed under $=_{\Lambda\Omega}$, this implies $C[A] \not\sqsubseteq_{\mathcal{N}} C[A']$, which in turn implies $A \not\sqsubseteq_{\mathcal{N}} A'$. \square

Now we are able to prove the correctness of the model, with respect to the **N**-operational semantics.

Theorem 11.2.13 (\mathcal{N} -Correctness).

*The model \mathcal{N} is correct with respect to the **N**-operational semantics.*

Proof. By Property 10.1.15 it is sufficient to prove that, for all M, N , if M has Λ -nf and N has not Λ -nf then $M \not\sqsubseteq_{\mathcal{N}} N$.

Hence, either N is a Λ -unsolvable term or it has a Λ -head normal form. Then the proof follows respectively from Lemmas 11.2.11.(i) and 11.2.11.(ii). \square

The model \mathcal{N} also characterizes the class of Λ -persistent normal forms. Here we will prove just a partial characterization, namely that if a term can be assigned type ψ , from the basis B_{ψ} , then it has Λ -persistent normal form. The inverse implication has a rather technical proof, which is out of the scope of this book. The interested reader can read it in [30].

Let us notice that the fact that a term is a Λ -persistent normal form if and only if $B_{\psi} \vdash_{\bowtie} M : \psi$ implies that Λ -persistent normal forms are closed under substitution. An interesting consequence of this fact is that the set of Λ -persistent normal forms is a set of input values.

Theorem 11.2.14. *If $B_{\psi} \vdash_{\bowtie} M : \psi$ then M has a Λ -persistent normal form.*

Proof. $B_{\psi} \vdash_{\bowtie} M : \psi$ implies $B_{\psi} \vdash_{\bowtie} M : \underbrace{\phi \rightarrow \dots \rightarrow \phi}_{n} \rightarrow \psi$ ($n \in \mathbb{N}$) by rule (\leq_{\bowtie}) . Let \vec{N} be a sequence of normal forms, so by Lemma 11.2.11 $B_{\psi} \vdash_{\bowtie} N : \phi$, for every $N \in \vec{N}$. Hence $B_{\psi} \vdash_{\bowtie} MN : \psi$ implies, by rule (\leq_{\bowtie}) , that $B_{\psi} \vdash_{\bowtie} MN : \phi$, and so $M\vec{N}$ has a normal form. \square

Using the same techniques as for model \mathcal{H} , we can use the correctness of the model \mathcal{N} with respect to the \mathbf{N} -operational semantics for proving some of its properties. The proofs of the following properties can be easily carried out in a similar way to the analogous properties for model \mathcal{H} .

- Property 11.2.15.* (i) $\approx_{\mathbf{N}}$ is a sensible theory.
(ii) $\approx_{\mathbf{N}}$ is fully extensional.
(iii) All fixed-point operators are equated by $\approx_{\mathbf{N}}$.

For proving that \mathcal{N} is also complete with respect to the \mathbf{N} -operational semantics, we can state a discriminability property between terms which are not in the $\sqsubseteq_{\mathcal{N}}$ relation, which is based on a minor variant of the semiseparability algorithm between approximants showed in Sect. 11.1.3.

Theorem 11.2.16 (\mathcal{N} -Discriminability).

If $M \not\sqsubseteq_{\mathcal{N}} N$ then there is a context $C[.]$ such that both $C[M]$ and $C[N]$ are closed and $C[M] \Downarrow_{\mathbf{N}}$ while $C[N] \Uparrow_{\mathbf{N}}$.

Proof. The proof is in Sect. 11.2.3. □

Theorem 11.2.17 (\mathcal{N} -Completeness).

The model \mathcal{N} is complete with respect to the \mathbf{N} -operational semantics.

Proof. Let $M \not\sqsubseteq_{\mathcal{N}} N$. By the \mathcal{N} -discriminability theorem, there is a context $C[.]$ such that both $C[M]$ and $C[N]$ are closed and $C[M] \Downarrow_{\mathbf{N}}$ while $C[N] \Uparrow_{\mathbf{N}}$, which implies, by definition, that $M \not\leq_{\mathbf{N}} N$. □

Finally, we can state the full abstraction result.

Corollary 11.2.18 (\mathcal{N} -Full abstraction).

The model \mathcal{N} is fully abstract with respect to the \mathbf{N} -operational semantics.

Proof. By the \mathcal{N} -correctness and \mathcal{N} -completeness theorems. □

Thanks to the \mathcal{N} -discriminability theorem, a finite axiomatization of the preorder $\sqsubseteq_{\mathcal{H}}$ between terms can be given. Note that this characterization is simpler than the characterization of $\sqsubseteq_{\mathcal{H}}$.

Definition 11.2.19. *The relation $\sqsubseteq \subseteq \Lambda \times \Lambda$ is defined as follows:*

$M \sqsubseteq N$ if and only if one of the following conditions holds:

- M is Λ -unsolvable;
- $M =_{\Lambda\eta} \lambda x_1 \dots x_n. x M_1 \dots M_m$, $N =_{\Lambda\eta} \lambda x_1 \dots x_n. x N_1 \dots N_m$, where $M_i \sqsubseteq N_i$, and M has Λ -nf implies N has Λ -nf ($1 \leq i \leq m$).

Theorem 11.2.20 (\mathcal{N} -Characterization).

$M \sqsubseteq_{\mathcal{N}} N$ if and only if $M \sqsubseteq N$.

Proof. The proof is in Sect. 11.2.3. □

Then the following corollary can be easily proved.

Corollary 11.2.21. $M \sqsubseteq_{\mathcal{N}} N$ if and only if $\forall A \in \mathcal{A}(M), \exists A' \in \mathcal{A}(N)$ such that $A \sqsubseteq_{\mathcal{N}} A'$.

Note, for model \mathcal{H} , only the if implication of a similar corollary holds.

It seems that the **H** and **N** operational theories coincide. The next property shows that this is not true.

Property 11.2.22. $E_{\infty} \prec_{\mathbf{N}} I$.

Proof. Let $E_{\infty} \equiv Y(\lambda xyz.y(xz))$, where Y is a call-by-name fixed-point operator, so $\mathcal{A}(E_{\infty}) = \{\Omega, \lambda z_0 z_1.z_0\Omega, \lambda z_0 z_1.z_0(\lambda z_2.z_1(\dots(\lambda z_{n+1}.z_n)\Omega)\dots)) \mid n \geq 0\}$ (see Example 11.1.53). For all $A' \in \mathcal{A}(E_{\infty})$ it is easy to see that $I \not\ll A'$ and $A' \ll I$, hence $E_{\infty} \sqsubseteq_{\mathcal{N}} I$ but $I \not\sqsubseteq_{\mathcal{N}} E_{\infty}$. \square

So the **N**-operational semantics is fully extensional, but it is not able to grasp the $=_{\eta}$ “up to infinite”, as the model \mathcal{H} does.

11.2.1 The \leq_{\bowtie} -Intersection Relation

First, let us prove that \leq_{\bowtie} is well posed, i.e. different type constants in C_{\bowtie} are not \simeq_{\bowtie} . The following theorem characterizes the syntactic shape of types that are $\simeq_{\bowtie} \omega$.

Theorem 11.2.23. $\sigma \simeq_{\bowtie} \omega$ if and only if $\sigma \in \Omega^{\mathcal{N}}$, where

$$\Omega^{\mathcal{N}} = \left\{ \sigma \in T(C_{\bowtie}) \mid \begin{array}{l} \sigma \equiv \sigma_0 \wedge \dots \wedge \sigma_n \ (n \in \mathbb{N}), \text{ where } \forall i \leq n \text{ either} \\ \sigma_i \equiv \omega, \text{ or } \sigma_i \equiv \pi_i \rightarrow \tau_i \text{ and } \tau_i \simeq_{\bowtie} \omega, 1 \leq i \leq n \end{array} \right\}.$$

Proof. Note that $\omega \simeq_{\bowtie} \sigma$ if and only if $\omega \leq_{\bowtie} \sigma$, by rule (a).

\Leftarrow The proof can be obtained from that of Theorem 11.1.32, taking into account that now rule (h3) is named (n5).

\Rightarrow First we will prove that, if $\sigma \in \Omega^{\mathcal{N}}$ and $\sigma \leq_{\bowtie} \tau$ then, $\tau \in \Omega^{\mathcal{N}}$, by induction on the rules of \leq_{\bowtie} .

(a),(b),(c),(c'),(d),(e),(f),(g),(r),(t) See Theorem 11.1.32.

(n0),(n1),(n2),(n3),(n4) Not possible.

(n5) Corresponds to case (h3) in Theorem 11.1.32.

The proof is done, since $\sigma \simeq_{\bowtie} \omega$ implies $\omega \leq_{\bowtie} \sigma$, but $\omega \in \Omega^{\mathcal{N}}$ implies $\sigma \in \Omega^{\mathcal{N}}$. \square

Note that $\sigma_1 \rightarrow \dots \rightarrow \sigma_n \rightarrow \omega \simeq_{\bowtie} \omega$, for all $\sigma_1, \dots, \sigma_n$ ($n \in \mathbb{N}$).

Corollary 11.2.24. (i) $\phi \not\simeq_{\bowtie} \omega$,
(ii) $\psi \not\simeq_{\bowtie} \omega$.

Proof. For proving both points, it is sufficient to observe that both ϕ and ψ do not belong to the set Ω^N . \square

In order to prove that $\phi \not\leq_{\bowtie} \psi$, we need the following lemma.

Lemma 11.2.25. $\sigma \in \Psi^N$ and $\tau \leq_{\bowtie} \sigma$ imply $\tau \in \Psi^N$, where

$$\Psi^N = \left\{ \sigma \in T(C_{\bowtie}) \mid \begin{array}{l} \sigma \equiv \sigma_0 \wedge \dots \wedge \sigma_n \ (n \in \mathbb{N}), \text{ where} \\ \exists i \leq n \text{ such that } \sigma_i \equiv \pi_1 \rightarrow \dots \rightarrow \pi_{h_i} \rightarrow \psi \ (h_i \in \mathbb{N}) \end{array} \right\}.$$

Proof. By induction on the definition of \leq_{\bowtie} . If the last applied rule is either (b), (c), (c'), (r), (n1) or (n2) the proof is obvious.

If the last applied rule is (f), then $\tau \equiv \mu \rightarrow \nu$ and $\sigma \equiv \rho \rightarrow \pi$, where $\rho \leq_{\bowtie} \mu$ and $\nu \leq_{\bowtie} \pi$; so, by induction, $\nu \in \Psi^N$, which implies $\tau \in \Psi^N$.

Cases (t) and (e) follow easily by induction. All other cases are not possible. \square

Note that $\sigma \in \Psi^N$ does not imply $\sigma \simeq_{\bowtie} \psi$.

Corollary 11.2.26. $\phi \not\leq_{\bowtie} \psi$.

Proof. Since $\psi \leq_{\bowtie} \phi$, $\phi \simeq_{\bowtie} \psi$ if and only if $\phi \leq_{\bowtie} \psi$. But $\psi \in \Psi^N$, so this would imply $\phi \in \Psi^N$, by Lemma 11.2.25. But this is not possible, by the definition of Ψ^N . \square

The following property characterizes the shape of types in $T(C_{\bowtie})$, modulo \simeq_{\bowtie} . It will be extensively used in order to prove some properties about the type assignment system \vdash_{\bowtie} .

Property 11.2.27. (i) If $\sigma \not\leq_{\bowtie} \omega$ then $\sigma \simeq_{\bowtie} \sigma_0 \wedge \dots \wedge \sigma_n \ (n \geq 0)$ such that $\forall i \leq n, \sigma_i \simeq_{\bowtie} \tau_1^i \rightarrow \dots \rightarrow \tau_{m_i}^i \rightarrow \rho_i$ for some $m_i \in \mathbb{N}$, and $\rho_i \in \{\phi, \psi\}$.
(ii) If $\sigma \simeq_{\bowtie} \tau_1 \rightarrow \dots \rightarrow \tau_n \rightarrow \rho \ (n \geq 0)$, where $\rho \in \{\phi, \psi\}$, then for all $p \in \mathbb{N}$,

$$\sigma \simeq_{\bowtie} \tau_1 \rightarrow \dots \rightarrow \tau_n \rightarrow \underbrace{\pi \rightarrow \dots \rightarrow \pi}_p \rightarrow \rho,$$

where $\pi \in \{\phi, \psi\}$ but $\pi \not\equiv \rho$.

Proof. (i) By induction on σ . If $\sigma \equiv \omega, \phi, \psi$ then the proof is trivial.

Let $\sigma \equiv \mu \rightarrow \pi$; so, by induction $\pi \simeq_{\bowtie} \pi_0 \wedge \dots \wedge \pi_h \ (h \in \mathbb{N})$ such that $\forall i \leq h, \pi_i \simeq_{\bowtie} \nu_1^i \rightarrow \dots \rightarrow \nu_{m_i}^i \rightarrow \rho$ and $\rho \in \{\phi, \psi\}$. By Lemma 10.1.4.(iii), $\sigma \simeq_{\bowtie} \sigma_0 \wedge \dots \wedge \sigma_h$, where $\sigma_i \simeq_{\bowtie} \mu \rightarrow \nu_1^i \rightarrow \dots \rightarrow \nu_{m_i}^i \rightarrow \rho$.

The case $\sigma \equiv \mu \wedge \pi$ follows by induction.

(ii) By rules (n1), (n2), (n3), (n4) and (f) it is easy to see that

$$\tau_1 \rightarrow \dots \rightarrow \tau_n \rightarrow \rho \simeq_{\bowtie} \tau_1 \rightarrow \dots \rightarrow \tau_n \rightarrow \pi \rightarrow \rho,$$

where $\pi \in \{\phi, \psi\}$ but $\pi \not\equiv \rho$, so the proof follows by induction on p . \square

The following lemma implies the legality of \bowtie . Note that a type in $T(C_\bowtie)$ has the following syntactical shape:

$$(\sigma_1 \rightarrow \tau_1) \wedge \dots \wedge (\sigma_n \rightarrow \tau_n) \wedge c_1 \wedge \dots \wedge c_m \text{ where } m, n \geq 0, m + n \geq 1,$$

where $c_j \in \{\phi, \psi\}$ ($1 \leq j \leq m$). Moreover, let us recall that every constant in C_\bowtie is \simeq_\bowtie to an arrow type, indeed $\omega \simeq \omega \rightarrow \omega$, $\phi \simeq_\bowtie \psi \rightarrow \phi$ and $\psi \simeq_\bowtie \phi \rightarrow \psi$.

Property 11.2.28. Let $n, m, p, q \geq 0$, $n + m \geq 1$, $p + q \geq 1$, and

$$(\sigma_1 \rightarrow \tau_1) \wedge \dots \wedge (\sigma_n \rightarrow \tau_n) \wedge c_1 \wedge \dots \wedge c_m \leq_\bowtie (\sigma'_1 \rightarrow \tau'_1) \wedge \dots \wedge (\sigma'_p \rightarrow \tau'_p) \wedge d_1 \wedge \dots \wedge d_q$$

where $c_i, d_j \in \{\omega, \phi, \psi\}$, $c_i \simeq_\bowtie \sigma_{n+i} \rightarrow \tau_{n+i}$ and $d_j \simeq_\bowtie \mu_{p+j} \rightarrow \nu_{p+j}$ ($\tau_{n+i}, \tau'_{p+j} \in \{\omega, \phi, \psi\}$, $1 \leq i \leq m$, $1 \leq j \leq q$). If $\tau'_h \not\simeq_\bowtie \omega$ ($1 \leq h \leq p + q$), then there is $\{i_1, \dots, i_k\} \subseteq \{1, \dots, n + m\}$ such that $\sigma_{i_1} \wedge \dots \wedge \sigma_{i_k} \geq_\bowtie \sigma'_h$, $\tau_{i_1} \wedge \dots \wedge \tau_{i_k} \leq_\bowtie \tau'_h$ and $\tau_j \not\simeq_\bowtie \omega$, for each $j \in \{i_1, \dots, i_k\}$.

Proof. We reason by induction on the last rule of the derivation proving

$$(\sigma_1 \rightarrow \tau_1) \wedge \dots \wedge (\sigma_n \rightarrow \tau_n) \wedge c_1 \wedge \dots \wedge c_m \leq_\bowtie (\sigma'_1 \rightarrow \tau'_1) \wedge \dots \wedge (\sigma'_p \rightarrow \tau'_p) \wedge d_1 \wedge \dots \wedge d_q.$$

(b), (r), (a) Obvious.

(c), (c'), (d) Easy.

(e) By induction.

(f) $n = 1$, $m = 0$, $p = 1$ and $q = 0$. Then $\sigma_1 \rightarrow \tau_1 \leq_\bowtie \sigma'_1 \rightarrow \tau'_1$ if and only if $\sigma'_1 \leq_\bowtie \sigma_1$ and $\tau_1 \leq_\bowtie \tau'_1$.

(n0), (n1), (n2), (n3), (n4) Immediate, since $\psi \simeq_\bowtie \phi \rightarrow \psi$ and $\phi \leq_\bowtie \psi \rightarrow \phi$.

(n5), (g) Trivial.

(t) Let $\rho \in T(C_\bowtie)$ be such that $(\sigma_1 \rightarrow \tau_1) \wedge \dots \wedge (\sigma_n \rightarrow \tau_n) \wedge c_1 \wedge \dots \wedge c_m \leq_\bowtie \rho$ and $\rho \leq_\bowtie (\sigma'_1 \rightarrow \tau'_1) \wedge \dots \wedge (\sigma'_p \rightarrow \tau'_p) \wedge d_1 \wedge \dots \wedge d_q$. ρ must be of the shape $(\mu_1 \rightarrow \nu_1) \wedge \dots \wedge (\mu_t \rightarrow \nu_t) \wedge e_1 \wedge \dots \wedge e_s$, where $e_i \in \{\omega, \phi, \psi\}$ and $e_i \simeq_\bowtie \mu_{t+i} \rightarrow \nu_{t+i}$ ($1 \leq i \leq s$). By induction, for all h such that $\tau'_h \not\simeq_\bowtie \omega$, there is $\{i_1^h, \dots, i_{k_h}^h\} \subseteq \{1, \dots, t + s\}$ such that $\mu_{i_1^h} \wedge \dots \wedge \mu_{i_{k_h}^h} \geq_\bowtie \sigma'_h$, $\nu_{i_1^h} \wedge \dots \wedge \nu_{i_{k_h}^h} \leq_\bowtie \tau'_h$ and $\nu_{i_j^h} \not\simeq_\bowtie \omega$, for all $1 \leq j \leq k_h$. By induction, for every $\nu_{i_j^h}$, there is $\{r_1^{h,j}, \dots, r_{w_{h,j}}^{h,j}\} \subseteq \{1, 2, \dots, n + m\}$ such that $\sigma_{r_1^{h,j}} \wedge \dots \wedge \sigma_{r_{w_{h,j}}^{h,j}} \geq_\bowtie \mu_{i_j^h}$ and $\tau_{r_1^{h,j}} \wedge \dots \wedge \tau_{r_{w_{h,j}}^{h,j}} \leq_\bowtie \nu_{i_j^h}$.

So the proof follows from rule (e). \square

▼ Proof of Theorem 11.2.5 (pag. 145).

The legality of the type system \bowtie is a particular case of Property 11.2.28. ■

11.2.2 Proof of \mathcal{N} -Approximation Theorem

Also in this case, the proof is very similar to the proof of the approximation theorem for the model \mathcal{H} . We will show here just the differences with respect that proof.

Definition 11.2.29. Let the predicate $App_{\mathcal{N}}$ be so defined:

$App_{\mathcal{N}}(B, \sigma, M)$ if and only if there is $A \in \mathcal{A}(M)$ such that $B \vdash_{\mathcal{N}} A : \sigma$.

In order to prove the (\Leftarrow) part of the theorem, we need to prove the following implication:

$$B \vdash_{\mathcal{N}} M : \sigma \Rightarrow App_{\mathcal{N}}(B, \sigma, M).$$

We will build the proof in two steps. First, it will be proved that

$$B \vdash_{\mathcal{N}} M : \sigma \Rightarrow Comp_{\mathcal{N}}(B, \sigma, M), \quad (11.3)$$

and then

$$Comp_{\mathcal{N}}(B, \sigma, M) \Rightarrow App_{\mathcal{N}}(B, \sigma, M), \quad (11.4)$$

where $Comp_{\mathcal{N}}(B, \sigma, M)$ (read: the term M is *computable* of type σ with respect to a basis B) is a property of the triple $\langle B, \sigma, M \rangle$. We will prove Eq. (11.3) by induction on terms and Eq. (11.4) by induction on types.

A basis B is *finite* if and only if $B(y) \simeq_{\mathcal{N}} \omega$ except in a finite number of variables. We will use $[\sigma_1/x_1, \dots, \sigma_n/x_n]$ to denote a finite basis. By Lemma 10.1.7.(i), in this section we limit ourselves to consider only such a kind of basis.

Let B and B' be two basis. Recall that $B \cup B'$ denotes the basis such that, for every x , $B \cup B'(x) = B(x) \wedge B'(x)$ (remember that, for every type σ , $\sigma \wedge \omega \simeq \sigma$). The key point is the difference in the definition of the computability predicate.

Definition 11.2.30. The predicate $Comp_{\mathcal{N}}$ is defined by induction on types as follows:

- $Comp_{\mathcal{N}}(B, \omega, M)$ is true;
- $Comp_{\mathcal{N}}(B, \phi, M)$ if and only if $App_{\mathcal{N}}(B, \phi, M)$;
- $Comp_{\mathcal{N}}(B, \psi, M)$ if and only if $App_{\mathcal{N}}(B, \psi, M)$
and $\forall N \in \vec{N}, App_{\mathcal{N}}(B', \phi, N)$ implies $App_{\mathcal{N}}(B \cup B', \psi, MN)$;
- $Comp_{\mathcal{N}}(B, \sigma \rightarrow \tau, M)$ if and only if
 $\forall N, Comp_{\mathcal{N}}(B', \sigma, N)$ implies $Comp_{\mathcal{N}}(B \cup B', \tau, MN)$;
- $Comp_{\mathcal{N}}(B, \sigma \wedge \tau, M)$ if and only if $Comp_{\mathcal{N}}(B, \sigma, M)$ and $Comp_{\mathcal{N}}(B, \tau, M)$.

It can be proved that $Comp_{\mathcal{N}}$ is defined modulo $=_A$, as for the model \mathcal{H} .

Lemma 11.2.31. (i) $App_{\mathcal{N}}(B, \sigma, x\vec{M})$ implies $Comp_{\mathcal{N}}(B, \sigma, x\vec{M})$.
(ii) $Comp_{\mathcal{N}}(B, \sigma, M)$ implies $App_{\mathcal{N}}(B, \sigma, M)$.

Proof. As for model \mathcal{H} , the proof can be done by mutual induction on σ . \square

$Comp_{\mathcal{N}}$ is closed under \leq_{\bowtie} .

Lemma 11.2.32. (i) If $Comp_{\mathcal{N}}(B, \psi, M)$ then $Comp_{\mathcal{N}}(B, \phi, M)$.
(ii) $Comp_{\mathcal{N}}(B, \psi, M)$ if and only if $Comp_{\mathcal{N}}(B, \phi \rightarrow \psi, M)$.
(iii) $App_{\mathcal{N}}(B[\psi/z], \phi, M)$ and $Comp_{\mathcal{N}}(B', \psi, N)$ imply $App_{\mathcal{N}}(B \cup B', \phi, M[N/z])$.
(iv) $Comp_{\mathcal{N}}(B, \phi, M)$ if and only if $Comp_{\mathcal{N}}(B, \psi \rightarrow \phi, M)$.
(v) If $Comp_{\mathcal{N}}(B, \sigma, M)$ and $\sigma \leq_{\bowtie} \tau$ then $Comp_{\mathcal{N}}(B, \tau, M)$.

Proof. (i) Trivial.

(ii) (\Rightarrow) We prove that $Comp_{\mathcal{N}}(B', \phi, N)$ implies $Comp_{\mathcal{N}}(B \cup B', \psi, MN)$ under the hypothesis that $Comp_{\mathcal{N}}(B, \psi, M)$.

$Comp_{\mathcal{N}}(B', \phi, N)$ implies $App_{\mathcal{N}}(B', \phi, N)$, by Lemma 11.2.31.(ii).

$Comp_{\mathcal{N}}(B, \psi, M)$ implies, by definition, that if $\forall Q \in \vec{Q}, App_{\mathcal{N}}(B', \phi, Q)$ then $App_{\mathcal{N}}(B \cup B', \psi, M\vec{Q})$. Hence $App_{\mathcal{N}}(B \cup B', \psi, MN)$ and $\forall P \in \vec{P}, App_{\mathcal{N}}(B', \phi, P)$ imply $App_{\mathcal{N}}(B \cup B', \psi, MN\vec{P})$.

Thus $Comp_{\mathcal{N}}(B \cup B', \psi, MN)$; so, by definition $Comp_{\mathcal{N}}(B, \phi \rightarrow \psi, M)$.

(\Leftarrow) We prove that $App_{\mathcal{N}}(B, \psi, M)$ and if $\forall N \in \vec{N} App_{\mathcal{N}}(B', \phi, N)$ then $App_{\mathcal{N}}(B \cup B', \psi, MN\vec{N})$, under the hypothesis that $Comp_{\mathcal{N}}(B, \phi \rightarrow \psi, M)$. $App_{\mathcal{N}}(B, \phi \rightarrow \psi, M)$, by Lemma 11.2.31.(ii); thus, by rule (\leq_{\bowtie}) , we obtain both $App_{\mathcal{N}}(B, \psi, M)$ and $App_{\mathcal{N}}(B, \underbrace{\phi \rightarrow \dots \rightarrow \phi}_{h} \rightarrow \psi, M)$, for all

$h \in \mathbb{N}$. If $\vec{N} \equiv N_1 \dots N_m$ is such that $App_{\mathcal{N}}(B', \phi, N_i)$ ($1 \leq i \leq m$) then $Comp_{\mathcal{N}}(B', \psi, N)$ implies $App_{\mathcal{N}}(B \cup B', \psi, MN\vec{N})$, by m applications of the rule $(\rightarrow E)$.

Hence, by definition $Comp_{\mathcal{N}}(B \cup B', \psi, MN\vec{N})$.

(iii) $App_{\mathcal{N}}(B[\psi/z], \phi, M)$ means that there is $A \in \mathcal{A}(M)$ such that $B[\psi/z] \vdash_{\bowtie} A : \phi$. The proof is given by induction on A .

Clearly, $A \not\equiv \Omega$ since $B \not\vdash_{\bowtie} \Omega : \phi$. Let $A \equiv z$; the proof follows since $Comp_{\mathcal{N}}(B', \psi, N)$ implies $App_{\mathcal{N}}(B', \psi, N)$, therefore $App_{\mathcal{N}}(B', \phi, N)$ by rule (\leq_{\bowtie}) , and $App_{\mathcal{N}}(B \cup B', \phi, M[N/z])$ by Lemma 10.1.7.(iii).

The case $A \in \text{Var}$, but $A \not\equiv z$ is trivial.

Let $A \equiv zA_1 \dots A_m$ ($m \geq 1$), thus $M =_A zM_1 \dots M_m$ and $A_i \in \mathcal{A}(M_i)$, for all $i \leq m$. Hence $B[\psi/z] \vdash_{\bowtie} z : \tau_1 \rightarrow \dots \rightarrow \tau_m \rightarrow \phi$ and $B[\psi/z] \vdash_{\bowtie} A_i : \tau_i$ for some τ_i , by Lemma 10.1.7.(vi). Since $\underbrace{\phi \rightarrow \dots \rightarrow \phi}_{m} \rightarrow \psi \leq_{\bowtie} \psi$ by rule

(\leq_{\bowtie}) and $\psi \leq_{\bowtie} \tau_1 \rightarrow \dots \rightarrow \tau_m \rightarrow \phi$ by Lemma 10.1.7.(ii), it follows that $\tau_i \leq_{\bowtie} \phi$ by Property 10.1.6. Thus $B[\psi/z] \vdash_{\bowtie} A_i : \phi$ and this implies $App_{\mathcal{N}}(B[\psi/z], \phi, M_i)$. Therefore $App_{\mathcal{N}}(B \cup B', \phi, M_i[N/z])$ by induction. So $Comp_{\mathcal{N}}(B', \psi, N)$ implies $App_{\mathcal{N}}(B \cup B', \phi, NM_1[N/z] \dots M_m[N/z])$ by definition, i.e. $App_{\mathcal{N}}(B \cup B', \phi, M[N/z])$.

The case $A \equiv yA_1\dots A_m$ ($m \geq 1$) where $y \not\equiv z$ is simpler.

Let $A \equiv \lambda y.A'$, for some $A' \in \mathcal{A}$, thus $M =_{\Lambda\eta} \lambda y.M'$ and $A' \in \mathcal{A}(M')$. Hence, $B[\psi/z] \vdash_{\bowtie} \lambda y.A' : \psi \rightarrow \phi$ by rule (\leq_{\bowtie}) , and $B[\psi/z, \psi/y] \vdash_{\bowtie} A' : \phi$ by Lemma 10.1.7.(vi). Without loss of generality, let $B'(y) = \psi$; so the proof follows by induction on $\text{App}_{\mathcal{N}}(B[\psi/z, \psi/y], \phi, M')$.

- (iv) (\Rightarrow) We will prove that $\text{Comp}_{\mathcal{N}}(B', \psi, N)$ implies $\text{Comp}_{\mathcal{N}}(B \cup B', \phi, MN)$, under the hypothesis that $\text{Comp}_{\mathcal{N}}(B, \phi, M)$.

By definition $\text{App}_{\mathcal{N}}(B, \phi, M)$, so $\text{App}_{\mathcal{N}}(B, \psi \rightarrow \phi, M)$ by rule (\leq_{\bowtie}) , which means there is $A \in \mathcal{A}(M)$ such that $B \vdash_{\bowtie} A : \psi \rightarrow \phi$.

If A is of the shape $\lambda x.A'$ then $M =_{\Lambda} \lambda x.M'$ and $A' \in \mathcal{A}(M')$; thus $B[\psi/x] \vdash_{\bowtie} A' : \phi$, by Lemma 10.1.7.(vi) and so $\text{App}_{\mathcal{N}}(B[\psi/x], \phi, M')$. Hence, $\text{App}_{\mathcal{N}}(B \cup B', \phi, M'[N/x])$ by the point (iii) of this Lemma. But $MN =_{\Lambda} M'[N/x]$, so $\text{App}_{\mathcal{N}}(B \cup B', \phi, MN)$ and by definition of $\text{Comp}_{\mathcal{N}}$ the proof follows. The case $A \equiv z\tilde{A}$ is simpler.

(\Leftarrow) If $\text{Comp}_{\mathcal{N}}(B, \psi \rightarrow \phi, M)$ then $\text{App}_{\mathcal{N}}(B, \psi \rightarrow \phi, M)$, by Lemma 11.2.31.(ii), hence $\text{App}_{\mathcal{N}}(B, \phi, M)$, by rule (\leq_{\bowtie}) .

- (v) By induction on the rules of \leq_{\bowtie} .

(a), (b), (c), (c'), (e), (r), (t) Trivial.

(d) $\text{Comp}_{\mathcal{H}}(B, (\sigma_0 \rightarrow \tau_0) \wedge (\sigma_0 \rightarrow \pi_0), M)$ implies, by definition of $\text{Comp}_{\mathcal{N}}$, $\text{Comp}_{\mathcal{N}}(B, \sigma_0 \rightarrow \tau_0, M)$ and $\text{Comp}_{\mathcal{N}}(B, \sigma_0 \rightarrow \pi_0, M)$.

If $\text{Comp}_{\mathcal{N}}(B', \sigma_0, N)$ then $\text{Comp}_{\mathcal{N}}(B \cup B', \tau_0, MN)$ and $\text{Comp}_{\mathcal{N}}(B \cup B', \pi_0, MN)$; hence, $\text{Comp}_{\mathcal{N}}(B \cup B', \tau_0 \wedge \pi_0, MN)$ and $\text{Comp}_{\mathcal{N}}(B, \sigma_0 \rightarrow (\tau_0 \wedge \pi_0), M)$, by definition of $\text{Comp}_{\mathcal{N}}$.

(f) Let $\sigma'_0 \leq_{\bowtie} \sigma_0$, $\tau_0 \leq_{\bowtie} \tau'_0$ and $\text{Comp}_{\mathcal{N}}(B, \sigma_0 \rightarrow \tau_0, M)$.

If $\text{Comp}_{\mathcal{N}}(B', \sigma'_0, N)$ then $\text{Comp}_{\mathcal{N}}(B', \sigma_0, N)$ by induction, hence $\text{Comp}_{\mathcal{N}}(B \cup B', \tau_0, MN)$ by definition of $\text{Comp}_{\mathcal{N}}$. Again by induction $\text{Comp}_{\mathcal{N}}(B \cup B', \tau'_0, MN)$, so the proof is done.

(g), (n5) Easy, since $\text{Comp}_{\mathcal{N}}(B, \omega, MN)$ is always true.

(n0), (n1), (n2), (n3), (n4) By using points (i), (ii) and (iii) of this lemma. □

Now we will prove the implication of Eq. (11.3).

Lemma 11.2.33. *Let $\text{FV}(M) \subseteq \{x_1, \dots, x_n\}$ and $B = [\sigma_1/x_1, \dots, \sigma_n/x_n]$. If $\text{Comp}_{\mathcal{N}}(B_i, \sigma_i, N_i)$ ($1 \leq i \leq n$) and $B \vdash_{\bowtie} M : \tau$ then*

$$\text{Comp}_{\mathcal{N}}(B_1 \cup \dots \cup B_n, \tau, M[N_1/x_1, \dots, N_n/x_n]).$$

Proof. Similar to the proof of Lemma 11.1.43, for model \mathcal{H} . □

▼ Proof of \mathcal{N} -Approximation Theorem (Theorem 11.2.9 pag. 147).

(\Rightarrow) We prove that $B \vdash_{\bowtie} M : \sigma$ implies $B \vdash_{\bowtie} A : \sigma$, for some $A \in \mathcal{A}(M)$.

Clearly $\text{Comp}_{\mathcal{H}}([\tau/x], \tau, x)$ from Lemma 11.2.31.(i).

Let $\text{FV}(M) \subseteq \{x_1, \dots, x_n\}$, so we can assume $B = [\sigma_1/x_1, \dots, \sigma_n/x_n]$ without loss of generality, by Lemma 10.1.7.(i). Therefore $B \vdash_{\bowtie} M : \sigma$ and

$Comp_{\mathcal{H}}([\sigma_i/x_i], \sigma_i, x_i)$ ($1 \leq i \leq n$) imply $Comp_{\mathcal{H}}(B, \sigma, M)$ by Lemma 11.2.33. So, by Lemma 11.2.31.(ii) and by definition of $App_{\mathcal{N}}$, the proof is given.

(\Leftarrow) By definition, there is M' such that $M =_A M'$, and A matches M' except at occurrences of Ω . A derivation of $B \vdash_{\bowtie} A : \sigma$ can be transformed into a derivation of $B \vdash_{\bowtie} M' : \sigma$ simply by replacing every subderivation

$$\overline{B \vdash_{\bowtie} \Omega : \omega}^{(\omega)} \quad \text{by} \quad \overline{B \vdash_{\bowtie} N : \omega}^{(\omega)},$$

where N are the subterms replaced by Ω in M' .

$B \vdash_{\bowtie} M' : \sigma$ implies $B \vdash_{\bowtie} M : \sigma$, since the type assignment system is closed under $=_A$ on terms, as a consequence of the fact that it induces a $\lambda\Lambda$ -model, and the proof is given. ■

11.2.3 Proof of \mathcal{N} -Discriminability and \mathcal{N} -Characterization Theorems

In this section we use the notion of path c and of the relation \ll_c between approximants, as defined in Sect. 11.1.3. Moreover, a minor modification of the semiseparability algorithm presented in Fig. 11.2 (pag.138) is introduced.

Let M and N be two terms such that $M \not\sqsubseteq_N N$; it is always possible to find two approximants $A \in \mathcal{A}(M)$ and $A' \in \mathcal{A}(N)$ and a path c such that $A \ll_c A'$, and the context $C[.]$ such that $A, A' \Rightarrow_D^c C[.]$ can be easily transformed in a context \mathbf{N} -discriminating M and N .

In order to choose the correct approximants, we use the next definition.

Definition 11.2.34. A path c is deep on A under the following conditions:

- if $c \equiv \epsilon$ then $A \in \Lambda\text{-NF}$, namely in A there are no occurrences of Ω ;
- if $c \equiv i, c'$ then $A =_{\eta} \lambda x_1..x_n.x A_1 ... A_m$ ($i \leq m$) and c' is deep on A_i .

Example 11.2.35. Let $A \in \mathcal{A}(\lambda x.x E_{\infty} I)$; it is easy to see that ϵ is not deep on A , paths having 1 as prefix are not deep on A , while if $A \equiv \lambda x.x A_1 A_2$ and $A_2 \not\equiv \Omega$, then every path having 2 as prefix is deep on A .

Note that if c is deep on A then c, c' is deep on A , for all paths c' .

Property 11.2.36. Let $B \vdash_{\bowtie} A : \sigma$ and let N be such that:

1. for each $A' \in \mathcal{A}(N)$, $A \ll_c A'$ implies A, A' are structurally similar along c and A' is not maximal along c in $\mathcal{A}(N)$;
2. c is deep on A implies that there is $A' \in \mathcal{A}(N)$ such that c is deep on A' .

Then $B \vdash_{\bowtie} N : \sigma$.

Proof. The proof is given by induction on the derivation proving $B \vdash_{\bowtie} A : \sigma$.

(var) Then $A \equiv x$, and the derivation is

$$\overline{B[\sigma/x] \vdash_{\bowtie} x : \sigma}^{(var)}$$

Note that the path ϵ is deep on x , so there is $A^* \in \mathcal{A}(N)$ such that ϵ is deep on A^* . Hence $A^* \not\equiv \Omega$, i.e. A^* is of the shape $\lambda x_1 \dots x_p . z A'_1 \dots A'_q$. From the condition that $A \lessdot_c A^*$ implies A, A^* are structurally similar along c , it follows that it must be $x \equiv z$; otherwise it would be $A \lessdot_\epsilon A^*$, and A, A^* not structurally similar along ϵ , against the hypothesis. By the same reasoning, it must be $p = q$.

Clearly $A^* \in A\text{-NF}$ implies that $x =_n A^*$ and $A'_i =_n x_i$; moreover, $N =_{A\eta} A^*$. Since typings are preserved by $=_{A\eta}$, the proof follows.

($\rightarrow I$) Then $A \equiv \lambda x. \bar{A}$, $\sigma \equiv \mu \rightarrow \nu$, and the derivation is

$$\frac{B[\mu/x] \vdash_{\bowtie} \bar{A} : \nu}{B \vdash_{\bowtie} \lambda x. \bar{A} : \mu \rightarrow \nu}^{(\rightarrow I)}$$

Clearly $\Omega \in \mathcal{A}(N)$ and $A \lessdot_c \Omega$, so by hypothesis 1 there is $A' \in \mathcal{A}(N)$ such that $\Omega \lessdot_c^\circ A'$ and $A' =_n \lambda x. A^*$. Hence, by Property 11.1.20.(ii), $N =_{A\eta} \lambda x. N'$ and $A^* \in \mathcal{A}(N')$.

Note that $B[\mu/x] \vdash_{\bowtie} \bar{A} : \nu$ and N' satisfy the hypothesis of this property (indeed, \lessdot_c is defined modulo $=_{A\eta}$), thus $B[\mu/x] \vdash_{\bowtie} N' : \nu$ by induction. This implies $B \vdash_{\bowtie} \lambda x. N' : \mu \rightarrow \nu$, by rule ($\rightarrow I$).

($\rightarrow E$) Then $A \equiv x A_1 \dots A_m A_{m+1}$, since $A \in \mathcal{A}$, and the derivation is

$$\frac{B \vdash_{\bowtie} x A_1 \dots A_m : \tau \rightarrow \sigma \quad B \vdash_{\bowtie} A_{m+1} : \tau}{B \vdash_{\bowtie} x A_1 \dots A_m A_{m+1} : \sigma}^{(\rightarrow I)}$$

for some τ . Clearly $\Omega \in \mathcal{A}(N)$ and $A \lessdot_c \Omega$, so by hypothesis 1 there is $A' \in \mathcal{A}(N)$ such that $\Omega \lessdot_c^\circ A'$. Assume that for every $A' \in \mathcal{A}(N)$, if $A' \not\equiv \Omega$ then A' is of the shape $\lambda x_1 \dots x_p . z A'_1 \dots A'_q$. The given conditions assure us that $z \equiv x$, $q = m + 1 + p$ and $A'_{m+i+1} =_n x_i$ ($1 \leq i \leq p$), i.e.

$$A' \equiv \lambda x_1 \dots x_p . x A'_1 \dots A'_{m+1} x_1 \dots x_p =_n x A'_1 \dots A'_{m+1}.$$

So $N =_{A\eta} x N_1 \dots N_{m+1}$ and $A'_i \in \mathcal{A}(N_i)$ ($1 \leq i \leq m + 1$). Note that, for A_i and every $A'_i \in \mathcal{A}(N_i)$, the conditions of the theorem are satisfied. So, by induction, $B \vdash_{\bowtie} N_{m+1} : \tau$ and $B \vdash_{\bowtie} x N_1 \dots N_m : \tau \rightarrow \sigma$. So, by rule ($\rightarrow E$), $B \vdash_{\bowtie} x N_1 \dots N_m N_{m+1} : \sigma$, and the proof follows by the fact that typings are preserved by $=_{A\eta}$.

($\wedge I$), ($\wedge E$), (\leq_{\bowtie}) The proof follows directly from the inductive hypothesis.

(ω) Trivial. □

We need a further definition.

Definition 11.2.37. Let c a path and A an approximant:

- if $c \equiv \epsilon$ then c is defined on A ;
- if $c \equiv k, c'$ then c is defined on A , whenever $A =_{\eta} \lambda x_1 \dots x_p . z A_1 \dots A_q$ ($q \geq k$) and c' is defined on A .

Example 11.2.38. 1 is defined on $\lambda z. x \Omega I$, 1 is defined on $\lambda z. x I \Omega$, but 1 is not defined on Ω .

Lemma 11.2.39. Let $M \not\subseteq_{\mathcal{N}} N$. Then there are $A \in \mathcal{A}(M)$, $A' \in \mathcal{A}(N)$ and a path c such that $A \llcorner_c A'$, c is defined on A, A' and one of the following conditions holds:

1. A and A' are structurally different in c ;
2. A' is maximal along c in $\mathcal{A}(N)$;
3. c is deep on A but c is not deep on A'' , for all $A'' \in \mathcal{A}(N)$.

Proof. $M \not\subseteq_{\mathcal{N}} N$ implies there are B, σ such that $B \vdash_{\mathbb{M}} M : \sigma$ while $B \not\vdash_{\mathbb{M}} N : \sigma$. By the \mathcal{N} -approximation theorem, this means there is $A \in \mathcal{A}(M)$ such that $B \vdash_{\mathbb{M}} A : \sigma$. Then, by Property 11.2.36, there are three cases.

1. There is $A' \in \mathcal{A}(N)$ such that $A \llcorner_c A'$, and A and A' are structurally different along c . Then A and A' are the desired approximants.
2. There is $A' \in \mathcal{A}(N)$ such that $A \llcorner_c A'$, and A and A' are structurally similar along c , and A' is maximal along c in $\mathcal{A}(N)$. Then A and A' are the desired approximants.
3. Otherwise, c is deep in A and not deep in A'' , for all $A'' \in \mathcal{A}(N)$. So each A'' such that $A \llcorner_c A''$ and c is defined on A'' can be chosen. \square

The \mathcal{N} -semiseparability algorithm is presented in Fig. 11.4. The algorithm is defined as a formal system, proving statements of the shape: $A, A' \Rightarrow_N^c C[.]$, where $A \llcorner_c A'$ and $C[.]$ is a context. The \mathcal{N} -semiseparability algorithm is a minor modification of that we given in Fig. 11.2 (pag.138). Rule (D1) has been divided in two rules, namely (N0) and (N1), according to the fact that the given path is deep or not in A .

For the sake of simplicity, in the \mathcal{N} -semiseparability algorithm we assume that all bound and free variables have different names.

Theorem 11.2.40. (i) If $A \llcorner_c A'$ then $A, A' \Rightarrow_N^c C[.]$.
(ii) If $A, A' \Rightarrow_N^c C[.]$ then $C[A]$ has A -normal form, while $C[A'] \rightarrow_{A\Omega}^* \Omega$.

Proof. (i) Similar to the proof of Lemma 11.1.48.

- (ii) The proof is carried out by induction on the derivation proving $A, A' \Rightarrow_N^c C[.]$. In case the last applied rule is (N0), then the proof follows from definition of deep. In case the last applied rule is (N1), then obviously $C[A] \rightarrow_{A\Omega}^* I$, while $C[A'] \rightarrow_{A\Omega}^* \Omega$. In the remaining cases, the proof is similar to that of the Lemma 11.1.49. \square

Let A, A' be approximants such that $A \not\ll_c A'$ and $r \geq \max\{\text{args}(A), \text{args}(A')\}$.

The rules of the system proving statements $A, A' \Rightarrow_N^c C[.]$, are the following:

$$\frac{\epsilon \text{ is deep on } xA_1\dots A_m}{xA_1\dots A_m, \Omega \Rightarrow_N^c [.]} \quad (N0) \quad \frac{\epsilon \text{ is not deep on } xA_1\dots A_m}{xA_1\dots A_m, \Omega \Rightarrow_N^c (\lambda x.[.])O^m} \quad (N1)$$

$$\frac{xA_1\dots A_m, \Omega \Rightarrow_N^c C[.]}{\lambda x_1\dots x_n.xA_1\dots A_m, \Omega \Rightarrow_N^c C[[.]x_1\dots x_n]} \quad (D2)$$

$$\frac{p \leq q \quad xA_1\dots A_mx_{p+1}\dots x_q, yA'_1\dots A'_n \Rightarrow_N^c C[.]}{\lambda x_1\dots x_p.xA_1\dots A_m, \lambda x_1\dots x_q.yA'_1\dots A'_n \Rightarrow_N^c C[[.]x_1\dots x_q]} \quad (D3)$$

$$\frac{q < p \quad xA_1\dots A_m, yA'_1\dots A'_nx_{q+1}\dots x_p \Rightarrow_N^c C[.]}{\lambda x_1\dots x_p.xA_1\dots A_m, \lambda x_1\dots x_q.yA'_1\dots A'_n \Rightarrow_N^c C[[.]x_1\dots x_p]} \quad (D4)$$

$$\frac{n < m}{xA_1\dots A_m, xA'_1\dots A_n \Rightarrow_N^c (\lambda x.[.])O^m \underbrace{I\dots I}_{m-n-2} KI\Omega} \quad (D5)$$

$$\frac{m < n}{xA_1\dots A_m, xA'_1\dots A_n \Rightarrow_N^c (\lambda x.[.])O^n \underbrace{I\dots I}_{n-m-2} K\Omega I} \quad (D6)$$

$$\frac{x \not\equiv y}{xA_1\dots A_m, yA'_1\dots A'_n \Rightarrow_N^c (\lambda xy.[.])(\lambda x_1\dots x_m.I)(\lambda x_1\dots x_n.\Omega)} \quad (D7)$$

$$\frac{x \notin \text{FV}(A_k) \cup \text{FV}(A'_k) \quad A_k \not\ll A'_k \quad A_k, A'_k \Rightarrow_N^c C[.]}{xA_1\dots A_m, xA'_1\dots A'_m \Rightarrow_N^{k,c} C[(\lambda x.[.])U_m^k]} \quad (D8)$$

$$\frac{x \in \text{FV}(A_k) \cup \text{FV}(A'_k) \quad A_k \not\ll A'_k \quad C_x^r[.] \equiv (\lambda x.[.])B^r \quad \text{nf}_{\Lambda\Omega}(C_x^r[A_k]), \text{nf}_{\Lambda\Omega}(C_x^r[A'_k]) \Rightarrow_N^c C[.]}{xA_1\dots A_m, xA'_1\dots A'_m \Rightarrow_N^{k,c} C[C_x^r[.] \underbrace{I\dots I}_{r-m} U_r^k]} \quad (D9)$$

Fig. 11.4. \mathcal{N} -Semiseparability algorithm

▼ **Proof of the \mathcal{N} -Discriminability Theorem** (Theorem 11.2.16 pag. 150).

Let $M \not\subseteq_{\mathcal{N}} N$. Choose $A \in \mathcal{A}(M)$, $A' \in \mathcal{A}(N)$ and a path c satisfying the conditions of Lemma 11.2.39. Then $A, A' \Rightarrow_N^c C[.]$. Let $C'[.]$ be the context obtained from $C[.]$ by replacing every occurrence of Ω by a Λ -unsolvable term, say DD . We will prove that $C'[.]$ is a context \mathbf{N} -discriminating M and N . The proof will be done by induction on the derivation of $A, A' \Rightarrow_N^c C[.]$. If the derivation coincides with an application of the axiom $(N0)$, then c is deep in A but not deep in A' . Thus M has Λ -nf while N does not have Λ -nf; so $[.]$ is a discriminating context.

If the derivation coincides with an application of the axiom $(N1)$, then $A' \equiv \Omega$ and A' is maximal along the empty path in $\mathcal{A}(N)$. Thus N is Λ -unsolvable and $(\lambda x.[.])O^m$ is the discriminating context.

If the used axiom is either $(D5)$, $(D6)$ or $(D7)$, the proof follows from the fact that M and N have both Λ -hnf, and by Property 11.1.20, their Λ -hnf's have the same shape respectively as A and A' . If the last rule applied is $(D2)$, $(D3)$, $(D4)$, $(D8)$ or $(D9)$, the proof follows by induction, always taking into account that M and N have both Λ -hnf, and that by Property 11.1.20, their Λ -hnf's have the same shape respectively as A and A' . ■

Example 11.2.41. (i) Let $M \equiv I$ and $N \equiv E_\infty$, where $E_\infty \equiv Y(\lambda xyz.y(xz))$.

Clearly $I \in \mathcal{A}(M)$, and, for every $A' \in \mathcal{A}(N)$, for every path c , I and A' are structurally similar along c , and c is not maximal in A' . Moreover, every path c is deep on I , while c is not deep in A' , for all $A' \in \mathcal{A}(N)$. So, a choice satisfying the conditions of Lemma 11.2.39 is $\lambda x.x \in \mathcal{A}(M)$, $\Omega \in \mathcal{A}(N)$ and $c \equiv \epsilon$. In fact,

$$\frac{}{\lambda x.x, \Omega \Rightarrow_N^\epsilon \lambda x.[.]} \quad (N0)$$

$$\frac{x, \Omega \Rightarrow_N^\epsilon \lambda x.[.]}{\lambda x.x, \Omega \Rightarrow_N^\epsilon \lambda x.[.]x} \quad (D2)$$

Therefore $(\lambda x.Ix) \rightarrow_\Lambda (\lambda x.x)$ and $(\lambda x.Ix) \Downarrow_{\mathbf{N}}$, while $(\lambda x.E_\infty x) \uparrow_{\mathbf{N}}$, since it has no Λ -nf.

(ii) Let $M \equiv \lambda xz.xzI$ and $N \equiv \lambda xz.xzE_\infty$. Then two approximants of M and N and a path c satisfying the conditions of Lemma 11.2.39 are respectively $A \equiv \lambda xz.xzI$, $A' \equiv \lambda xz.x\Omega\Omega$ and $c \equiv 2$.

Note that the A , A' and path $c' \equiv 1$ does not satisfy the requirements of Lemma 11.2.39. In fact, c' is not deep on all $A'' \in \mathcal{A}(N)$. Note that also $A \equiv \lambda xz.x\Omega I$, $A' \equiv \lambda xz.x\Omega\Omega$ and $c \equiv 2$ can be safely chosen.

▼ **Proof of the \mathcal{N} -Characterization Theorem** (Theorem 11.2.20 pag. 150).

(\Leftarrow) We will prove that if $M \not\subseteq N$ then there are $A \in \mathcal{A}(M)$ and $A' \in \mathcal{A}(N)$ and a path c satisfying the conditions of Lemma 11.2.39. By the proof of

the \mathcal{N} -discriminability theorem, there is a context $C[.]$ \mathbf{N} -discriminating M and N , i.e. $C[M] \downarrow_{\mathbf{N}}$ while $C[N] \uparrow_{\mathbf{N}}$. By the correctness of the model with respect to the \mathbf{N} -operational semantics, this implies $M \not\subseteq_{\mathcal{N}} N$. The existence of $A \in \mathcal{A}(M)$ and $A' \in \mathcal{A}(N)$ and a path c satisfying the conditions of Lemma 11.2.39 can be proved by induction on the definition of \sqsubseteq . There are four cases:

1. N is Λ -unsolvable and M is Λ -solvable; let $A \in \mathcal{A}(M)$ and $A \not\equiv \Omega$. Thus A , Ω and the empty path satisfy the given conditions.
2. $M =_{\Lambda\eta} \lambda x_1 \dots x_n . x M_1 \dots M_m$, $N =_{\Lambda\eta} \lambda x_1 \dots x_p . y M_1 \dots M_q$, and M has Λ -nf while N has not Λ -nf; so $M \in \mathcal{A}(M)$, every $A' \in \mathcal{A}(N)$ such that $A' \not\equiv \Omega$ and the empty path satisfy the given conditions.
3. Each Λ -hnf of M is of the shape $\lambda x_1 \dots x_n . x M_1 \dots M_m$, each Λ -hnf of N is of the shape $\lambda x_1 \dots x_p . y N_1 \dots N_q$, and either $x \not\equiv y$ or $|m - n| \neq |p - q|$. So every $A \in \mathcal{A}(M)$ and $A' \in \mathcal{A}(N)$ such that $A, A' \not\equiv \Omega$ and the empty path satisfy the given conditions.
4. Let $M =_{\Lambda\eta} \lambda x_1 \dots x_n . x M_1 \dots M_m$, $N =_{\Lambda\eta} \lambda x_1 \dots x_n . x N_1 \dots N_m$ and $M_k \not\subseteq N_k$, for some k ($1 \leq k \leq m$). Then by induction there are $A^* \in \mathcal{A}(M_k)$, $A^{**} \in \mathcal{A}(N_k)$ and a path c satisfying Lemma 11.2.39. Hence $A \in \mathcal{A}(M)$ and $A' \in \mathcal{A}(M')$ having the following shapes,

$$\begin{aligned} A &=_{\eta} \lambda x_1 \dots x_n . x \underbrace{\Omega \dots \Omega}_{k-1} A^* \underbrace{\Omega \dots \Omega}_{m-k}, \\ A' &=_{\eta} \lambda x_1 \dots x_n . x \underbrace{\Omega \dots \Omega}_{k-1} A^{**} \underbrace{\Omega \dots \Omega}_{m-k}, \end{aligned}$$

and the path k, c satisfy the conditions.

(\Rightarrow) Assume $M \not\subseteq_{\mathcal{N}} N$; by Lemma 11.2.39 there are $A \in \mathcal{A}(M)$, $A' \in \mathcal{A}(N)$, and there is a path c such that $A \ll_c A'$ and either A, A' are structurally different in c , or A' is maximal along c in $\mathcal{A}(N)$ or there is not A'' maximal along c, c' in $\mathcal{A}(N)$, for all c' , and c is deep in A and not deep in A' . Note that $A \not\equiv \Omega$, since $\Omega \ll A'$.

By induction on c , we will prove that $M \not\subseteq N$.

Let $c \equiv \epsilon$. Let $A' \equiv \Omega$. If A' is maximal then N is Λ -unsolvable, and so $M \not\subseteq N$. If there is not A'' maximal along c, c' in $\mathcal{A}(N)$, for all c' , and c is deep in A and not deep in A' , then M has Λ -nf while N has not Λ -nf, and so $M \not\subseteq N$. Let $A \equiv \lambda x_1 \dots x_n . x A_1 \dots A_m$ and $A' \equiv \lambda x_1 \dots x_p . y A_1 \dots A_q$, where either $|n - m| \neq |p - q|$ or $x \not\equiv y$. Then $M \not\subseteq N$ by definition. Let $c \equiv i, c'$. Then $A =_{\eta} \lambda x_1 \dots x_n . x A_1 \dots A_m$ and $A' =_{\eta} \lambda x_1 \dots x_n . x A'_1 \dots A'_m$ where $A_i \ll_c A'_i$, and the proof follows by induction. ■

11.3 The Model \mathcal{L}

In this section we will introduce a filter model correct with respect to the \mathbf{L} -operational semantics. Property 10.1.15 says that a filter model such that

for all M, N , if $M \Downarrow_{\mathbf{L}}$ (i.e. it has Λ -lazy head normal form) and $N \uparrow_{\mathbf{L}}$ (i.e. it has not Λ -lazy head normal form) then $N \sqsubset_{\mathcal{L}} M$.

is correct with respect to the \mathbf{L} -operational semantics.

So we define a legal type theory, say \angle , based on a set of constants C_{\angle} such that, for every M with Λ -lhnf and N without Λ -lhnf, there is a basis B and at least one type σ such that $B \vdash_{\angle} M : \sigma$ while not $B \vdash_{\angle} N : \sigma$. Since terms without Λ -lhnf are all and only the Λ -unsolvable terms of order 0, following the same approach as for the previous two models, it seems natural to characterize them by assigning them only type ω . Let us recall that \mathbf{L} induces a not sensible Λ -theory; in fact, $DD \uparrow_{\mathbf{L}}$ while $\lambda x.DD \Downarrow_{\mathbf{L}}$. Since, by rules (ω) and $(\rightarrow I)$, $\lambda x.DD$ can always be assigned type $\omega \rightarrow \omega$, a natural choice is to characterize the convergent terms by this type. This allows us to have as set of type constants just the singleton $\{\omega\}$. Clearly the inequality $\omega \leq \omega \rightarrow \omega$, which holds in the two previous models, is no longer correct in this setting.

Definition 11.3.1. Let $C_{\angle} = \{\omega\}$ and $I(C_{\angle}) = T(C_{\angle})$.

\angle is the type system $\langle C_{\angle}, \leq_{\angle}, I(C_{\angle}) \rangle$, where \leq_{\angle} is the least intersection relation induced by the rules of Fig. 11.5.

$\frac{}{\sigma \leq_{\angle} \omega}$ (a)	$\frac{}{\sigma \leq_{\angle} \sigma \wedge \sigma}$ (b)	$\frac{}{\sigma \wedge \tau \leq_{\angle} \sigma}$ (c)	$\frac{}{\sigma \wedge \tau \leq_{\angle} \tau}$ (c')
$\frac{(\sigma \rightarrow \tau) \wedge (\sigma \rightarrow \pi) \leq_{\angle} \sigma \rightarrow (\tau \wedge \pi)}{\sigma' \leq_{\angle} \sigma, \tau \leq_{\angle} \tau'}$ (d)	$\frac{\sigma \leq_{\angle} \sigma', \tau \leq_{\angle} \tau'}{\sigma \wedge \tau \leq_{\angle} \sigma' \wedge \tau'}$ (e)		
$\frac{\sigma' \leq_{\angle} \sigma, \tau \leq_{\angle} \tau'}{\sigma \rightarrow \tau \leq_{\angle} \sigma' \rightarrow \tau'}$ (f)	$\frac{\sigma \rightarrow \omega \leq_{\angle} \omega \rightarrow \omega}{\sigma \leq_{\angle} \sigma}$ (g)	$\frac{\sigma \leq \rho \quad \rho \leq \tau}{\sigma \leq_{\angle} \tau}$ (t)	

Fig. 11.5. \angle -intersection relation

Note that \leq_{\angle} is the minimum intersection relation, and we already noticed that \angle is a legal type system.

Definition 11.3.2. \mathcal{L} is the $\lambda\Lambda$ -model $\langle \mathcal{F}(\angle), \mathcal{F}(\angle), \circ_{\angle}, [\cdot]^{\mathcal{F}(\angle)} \rangle$.

Now we will state some properties of the \angle -intersection relation that will be useful in this chapter. Namely, it will be proved that ω is $\not\leq_{\angle}$ to any arrow type, and the general shape of a type, modulo \simeq_{\angle} will be shown. To do this, we need to characterize the set of types that are $\simeq \omega$.

Theorem 11.3.3. $\sigma \simeq_{\angle} \omega$ if and only if $\sigma \equiv \underbrace{\omega \wedge \dots \wedge \omega}_n$ ($n \geq 1$).

Proof. Note that $\omega \simeq_{\angle} \sigma$ if and only if $\omega \leq_{\angle} \sigma$, by rule (a).

(\Leftarrow) The proof follows by rules (b), (e) and (t).

(\Rightarrow) It is easy to prove, by induction on the rules of \leq_{\angle} , that if $\sigma \equiv \underbrace{\omega \wedge \dots \wedge \omega}_k$, for some $k \geq 1$, and $\sigma \leq_{\angle} \tau$ then $\tau \equiv \underbrace{\omega \wedge \dots \wedge \omega}_h$, for some $h \geq 1$. Then the proof is done, since $\sigma \simeq_{\angle} \omega$ implies $\omega \leq_{\angle} \sigma$, so $\sigma \equiv \underbrace{\omega \wedge \dots \wedge \omega}_n$, for some $n \geq 1$. \square

Corollary 11.3.4. $\sigma \equiv \mu \rightarrow \nu$ implies $\sigma \not\simeq_{\angle} \omega$.

The following property states a characterization of the shape of types in the theory \angle .

Property 11.3.5. (i) If $\sigma \not\simeq_{\angle} \omega$ then $\sigma \simeq_{\angle} \sigma_0 \wedge \dots \wedge \sigma_n$ such that $\forall i \leq n$, $\sigma_i \simeq_{\angle} \tau_1^i \rightarrow \dots \rightarrow \tau_{m_i}^i \rightarrow \omega \rightarrow \omega$, for some $n, m_i \in \mathbb{N}$.

(ii) There is $p \in \mathbb{N}$ such that $\underbrace{\omega \rightarrow \dots \rightarrow \omega}_k \rightarrow \omega \rightarrow \omega \leq_{\angle} \sigma$, for all $k \geq p$.

(iii) If $\sigma \not\simeq_{\angle} \omega$ then $\sigma \leq_{\angle} \omega \rightarrow \omega$.

Proof. (i) By induction on σ . The case $\sigma \equiv \omega$ is against the hypothesis. If $\sigma \equiv \pi \wedge \tau$ then the proof follows by induction. Let $\sigma \equiv \tau \rightarrow \pi$. If $\pi \simeq_{\angle} \omega$ then by rules (g), (f) and (a) it is easy to check that $\omega \rightarrow \omega \simeq_{\angle} \tau \rightarrow \pi$. Otherwise, by induction $\pi \simeq_{\angle} \pi_0 \wedge \dots \wedge \pi_k$ for some $k \in \mathbb{N}$ and $\pi_i \simeq_{\angle} \pi_1^i \rightarrow \dots \rightarrow \pi_{k_i}^i \rightarrow \omega \rightarrow \omega$ for some $k_i \in \mathbb{N}$. Hence, by Lemma 10.1.4.(iii), $\sigma \simeq_{\angle} (\tau \rightarrow \pi_0) \wedge \dots \wedge (\tau \rightarrow \pi_k)$, and the proof is done.

(ii) If $\sigma \simeq_{\angle} \omega$ then $p = 0$, and the proof follows from rule (a). Otherwise, by the previous point, $\sigma \simeq_{\angle} \sigma_0 \wedge \dots \wedge \sigma_n$ ($n \in \mathbb{N}$) such that $\forall i \leq n$, $\sigma_i \simeq_{\angle} \tau_1^i \rightarrow \dots \rightarrow \tau_{m_i}^i \rightarrow \omega \rightarrow \omega$ for some $m_i \in \mathbb{N}$. Notice that

$$\begin{aligned} & \underbrace{\omega \rightarrow \dots \rightarrow \omega}_{m_i} \rightarrow \omega \rightarrow \omega \leq_{\angle} \tau_1^i \rightarrow \dots \rightarrow \tau_{m_i}^i \rightarrow \omega \rightarrow \omega \\ & \underbrace{\omega \rightarrow \dots \rightarrow \omega}_{m_i} \rightarrow \omega \rightarrow \omega \rightarrow \omega \rightarrow \omega \leq_{\angle} \underbrace{\omega \rightarrow \dots \rightarrow \omega}_{m_i} \rightarrow \omega \rightarrow \omega \end{aligned}$$

so, posing $p = \max\{m_1, \dots, m_n\}$, the proof is easy.

(iii) By point (i), $\sigma \simeq_{\angle} \sigma_0 \wedge \dots \wedge \sigma_n$ ($n \in \mathbb{N}$) such that $\forall i \leq n$, $\sigma_i \simeq_{\angle} \tau_1^i \rightarrow \dots \rightarrow \tau_{m_i}^i \rightarrow \omega \rightarrow \omega$ for some $m_i \in \mathbb{N}$. It is easy to see that $\sigma_i \leq_{\angle} \tau_1^i \rightarrow \omega \leq_{\angle} \omega \rightarrow \omega$ ($1 \leq i \leq n$) by rules (a), (f) and (g), so the proof follows by rule (b). \square

In the related type assignment system, every term having Λ -lhnf can be assigned at least the type $\omega \rightarrow \omega$.

Lemma 11.3.6. If M has Λ -lhnf then $B \vdash_{\angle} M : \omega \rightarrow \omega$, for some basis B .

Proof. Let $M \equiv \lambda x.P$ and let B_ω be a basis such that $B_\omega(x) = \omega$, for all $x \in \text{Var}$. So $B \vdash_{\angle} P : \omega$ by rule (ω) , and $B \vdash_{\angle} \lambda x.P : \omega \rightarrow \omega$ by rule $(\rightarrow I)$. Let $M \equiv xM_1\dots M_m$. Let B be such that $B(x) = \underbrace{\omega \rightarrow \dots \rightarrow \omega}_m \rightarrow \omega \rightarrow \omega$.

Clearly $B \vdash_{\angle} M_i : \omega$ ($1 \leq i \leq m$) by rule (ω) , therefore by rule $(\rightarrow E)$, $B \vdash_{\angle} xM_1\dots M_m : \omega \rightarrow \omega$.

The proof is done, since \mathcal{L} is a λA -model, hence it is closed under $=_A$. \square

In order to show that this model is correct with respect to the **L**-operational semantics, we must show that if M has no A -lhnf, i.e. it is a A -unsolvable term of order 0, and $B \vdash_{\angle} M : \sigma$ then $\sigma \simeq_{\angle} \omega$. To prove this, we need an approximation theorem.

The notion of approximant needed for studying the lazy evaluation is different from that used in the previous sections.

Definition 11.3.7. Let $\Lambda\Omega$ be defined as in Definition 11.1.9 pag. 123.

- (i) The $\angle\Omega$ -reduction $(\rightarrow_{\angle\Omega})$ is defined as the contextual closure of the following rule:

$$\Omega M \rightarrow \Omega.$$

- (ii) The $\Lambda\angle\Omega$ -reduction $(\rightarrow_{\Lambda\angle\Omega})$ is the contextual closure of the following rules:

$$(\lambda x.M)N \rightarrow M[N/x], \text{ for all } N \in \Lambda\Omega, \quad \Omega M \rightarrow \Omega.$$

$\rightarrow_{\angle\Omega}^*$ denotes the symmetric and transitive closure of $\rightarrow_{\angle\Omega}$.

The η -reduction (\rightarrow_η) can be directly applied to the language $\Lambda\Omega$ (see Definition 1.3.7). $M \in \Lambda\Omega$ is in $\Lambda\angle\Omega$ -normal form ($\Lambda\angle\Omega$ -nf) if and only if it does not contain $\Lambda\angle\Omega$ -redexes.

Note that the reduction rule $\lambda x.\Omega \rightarrow_\Omega \Omega$, which has been used in both models \mathcal{H} and \mathcal{N} , is no longer correct for the lazy semantics.

Definition 11.3.8. The set of $\angle\Omega$ -approximants of a term M is defined as follows:

$$\mathcal{A}^{\angle}(M) = \left\{ A \mid \begin{array}{l} \exists M' \text{ such that } M =_A M' \text{ and } A \text{ is a } \Lambda\angle\Omega\text{-normal form} \\ \text{obtained from } M' \text{ by replacing some subterms with } \Omega. \end{array} \right\}$$

In this chapter, we will simply call approximant a $\Lambda\angle\Omega$ -normal form.

Example 11.3.9. Some sets of approximants are shown.

$$\mathcal{A}^{\angle}(I) = \{\Omega, \lambda x.\Omega, \lambda x.x\};$$

$$\mathcal{A}^{\angle}(D) = \{\Omega, \lambda x.\Omega, \lambda x.xx, \lambda x.x\Omega\};$$

$$\mathcal{A}^{\angle}(DD) = \{\Omega\};$$

$$\mathcal{A}^{\angle}(K(\lambda x.x(II))(DD)) = \{\Omega, \lambda x.\Omega, \lambda x.x\Omega, \lambda x.x(\lambda x.\Omega), \lambda x.xI\}.$$

Theorem 11.3.10 (\mathcal{L} -Approximation).

$B \vdash_{\angle} M : \sigma$ if and only if $B \vdash_{\angle} A : \sigma$, for some $A \in \mathcal{A}^{\angle}(M)$.

Proof. The proof is in Sect. 11.3.1. \square

The following property shows that the theory of the model \mathcal{L} is not sensible.

Property 11.3.11.

- (i) If M is Λ -unsolvable of order n then $\mathcal{A}^{\angle}(M) = \{\lambda x_1 \dots x_p . \Omega \mid 0 \leq p \leq n\}$.
- (ii) If M is Λ -unsolvable of infinite order then $\mathcal{A}^{\angle}(M) = \{\lambda x_1 \dots x_p . \Omega \mid p \in \mathbb{N}\}$.
- (iii) Let M and N be Λ -unsolvable respectively of order p and q ($p, q \in \mathbb{N}$).
If $p < q$ then $M \sqsubset_{\mathcal{L}} N$.

Proof. Easy. \square

Corollary 11.3.12. $M \Downarrow_{\mathbf{L}}$ if and only if $B \vdash_{\angle} M : \omega \rightarrow \omega$, for some B .

Proof. By Lemma 11.3.6, by Property 11.3.11 and by the approximation theorem. \square

Thus we can prove the correctness of the model.

Theorem 11.3.13 (\mathcal{L} -Correctness).

The model \mathcal{L} is correct with respect to the \mathbf{L} -operational semantics.

Proof. From Property 10.1.15 and Corollary 11.3.12. \square

The correctness implies some properties of the \mathbf{L} -operational semantics.

Property 11.3.14. (i) $M \rightarrow_{\eta} N$ implies $N \preceq_{\mathbf{L}} M$.

- (ii) All call-by-name fixed-point operators are equated in \mathbf{L} .
- (iii) Let Z be a call-by-name fixed-point operator. $M \preceq_{\mathbf{L}} ZK$, for all $M \in \Lambda$.
- (iv) All Λ -unsolvable of the same order are equated in \mathbf{L} .

Proof. (i) It is sufficient to prove that $I \preceq_{\mathbf{L}} E$. Let $B \vdash_{\angle} I : \sigma$; by Lemma 11.3.5.(i), either $\sigma \simeq_{\angle} \omega$ or $\sigma \simeq_{\angle} (\mu_1 \rightarrow \nu_1) \wedge \dots \wedge (\mu_k \rightarrow \nu_k)$, for some $k \in \mathbb{N}$. The case $\sigma \simeq_{\angle} \omega$ is trivial, so let $\sigma \simeq_{\angle} (\mu_1 \rightarrow \nu_1) \wedge \dots \wedge (\mu_k \rightarrow \nu_k)$. But $B \vdash_{\angle} I : \mu_i \rightarrow \nu_i$ by rule (\leq_{\angle}) , therefore by Lemma 10.1.7.(vi) $B[\mu_i/x] \vdash_{\angle} x : \nu_i$, and by Lemma 10.1.7.(ii) $\mu_i \leq_{\angle} \nu_i$ ($i \leq k$). We prove that, if $\mu_i \leq_{\angle} \nu_i$ then $B \vdash_{\angle} E : \mu_i \rightarrow \nu_i$; thus the proof follows by rule $(\wedge I)$. The proof is given by induction on μ_i .

If $\mu_i \simeq_{\angle} \omega$ then $\nu_i \simeq_{\angle} \omega$, so it is easy to see that $B \vdash_{\angle} E : \omega \rightarrow \omega$, by rules (ω) and $(\rightarrow I)$. If $\mu_i \equiv \tau_i \rightarrow \rho_i$ then $B[\mu_i/x, \tau_i/y] \vdash_{\angle} xy : \rho_i$, so $B \vdash_{\angle} \lambda xy. xy : \mu_i \rightarrow \mu_i$, by applying rule $(\rightarrow I)$ twice. Finally, by rule (\leq_{\angle}) , $B \vdash_{\angle} \lambda xy. xy : \mu_i \rightarrow \nu_i$. If μ_i is an intersection type then the proof follows by induction.

Note that this inclusion is strict. In fact, $B \vdash_{\angle} \lambda xy. xy : \omega \rightarrow \omega \rightarrow \omega$, for all B , while this typing is not derivable for I .

- (ii) Similar to the proof of Theorem 11.1.24.
- (iii) By definition of call-by-name fixed-point, $ZM =_{\Lambda} M(ZM)$, for all terms M , so $ZK =_{\Lambda} K(ZK) =_{\Lambda} \lambda x.ZK$. So $\mathcal{A}^{\mathcal{L}}(ZK) = \{\lambda x_1 \dots x_p.\Omega \mid 0 \leq p\}$, and $\underbrace{\omega \rightarrow \dots \rightarrow \omega}_p \rightarrow \omega \rightarrow \omega \in \llbracket ZK \rrbracket_{\rho}^{\mathcal{F}(\mathcal{L})}$ for all environments ρ and natural numbers p .
The proof follows by Property 11.3.5.(ii) and by Correctness.
- (iv) By Property 11.3.11.(iii) and by correctness. \square

It is interesting to compare the proof of point (i) of the previous lemma with the proof of the Lemma 6.3.10, proving the same statement. The denotational proof is much easier, this fact is a witness of the powerful of the type assignment system for proving properties of filter models.

The model \mathcal{L} is not fully abstract with respect to the **L**-operational semantics. In fact, an incompleteness result holds.

Theorem 11.3.15 (\mathcal{L} -Incompleteness).

*The model \mathcal{L} is incomplete with respect to the **L**-operational semantics.*

Proof. The proof is in Sect. 11.3.2. \square

There is no a filter $\lambda\Lambda$ -model fully abstract with respect to the **L**-operational semantics, as will be proved in the next theorem.

Theorem 11.3.16. *There is no a filter $\lambda\Lambda$ -model that is fully abstract with respect to the **L**-operational semantics.*

Proof. The proof is in Sect. 11.3.2. \square

Until now, a syntactical axiomatization of the relation $\preceq_{\mathbf{L}}$ has not been found. Note that the $\sim_{\mathcal{L}}$ relation has an unusual behaviour under application, as can be seen from the following example, first proved in [2].

Example 11.3.17. $x \not\sim_{\mathcal{L}} \lambda y.xy$ but $xx \sim_{\mathcal{L}} x(\lambda y.xy)$.

$x \not\sim_{\mathcal{L}} \lambda y.xy$ follows from the fact that, if B is a basis such that $B(x) = \omega$, then $B \not\vdash_{\mathcal{L}} x : \omega \rightarrow \omega$, while $B \vdash_{\mathcal{L}} \lambda y.xy : \omega \rightarrow \omega$.

To prove $xx \sim_{\mathcal{L}} x(\lambda y.xy)$, by Property 11.3.14.(i), it is sufficient to prove $x(\lambda y.xy) \sqsubseteq_{\mathcal{L}} xx$, namely that $B \vdash_{\mathcal{L}} x(\lambda y.xy) : \sigma$ implies $B \vdash_{\mathcal{L}} xx : \sigma$.

Without loss of generality let $\sigma \not\simeq_{\nabla} \omega$; thus $B \vdash_{\mathcal{L}} x(\lambda y.xy) : \sigma$ implies, by Lemma 10.1.7.(vii), that $B \vdash_{\mathcal{L}} x : \rho \rightarrow \sigma$ and $B \vdash_{\mathcal{L}} \lambda y.xy : \rho$, for some ρ . Moreover, $B(x) \leq_{\mathcal{L}} \rho \rightarrow \sigma$ by Lemma 10.1.7.(ii).

If $\rho \simeq_{\mathcal{L}} \omega$ then $\rho \rightarrow \sigma \leq_{\mathcal{L}} \omega \rightarrow \sigma$ thus

$$\frac{\overline{B \vdash_{\mathcal{L}} x : B(x)}^{(var)} \quad \overline{B \vdash_{\mathcal{L}} x : \omega \rightarrow \sigma}^{(\leq_{\mathcal{L}})} \quad \overline{B \vdash_{\mathcal{L}} x : \omega}^{(\omega)}}{B \vdash_{\mathcal{L}} xx : \sigma}^{(\rightarrow E)}$$

Otherwise, $\rho \simeq_{\angle} \sigma_0 \wedge \dots \wedge \sigma_n$ ($n \geq 1$) such that $\forall i \leq n, \sigma_i \simeq_{\angle} \mu_i \rightarrow \nu_i$ by Property 11.3.5.(i). Hence $B[\mu_i/y] \vdash_{\angle} xy : \nu_i$ by Lemma 10.1.7.(vi), and by Lemma 10.1.7.(vii), $B[\mu_i/y] \vdash_{\angle} y : \tau_i$ and $B[\mu_i/y] \vdash_{\angle} x : \tau_i \rightarrow \nu_i$, for some τ_i . So by Lemma 10.1.7.(ii), $\mu_i \leq_{\angle} \tau_i$ and $B(x) \leq_{\angle} \tau_i \rightarrow \nu_i$, hence by rule (f) of the \angle -intersection relation, $B(x) \leq_{\angle} \mu_i \rightarrow \nu_i$ ($i \leq n$); it is easy to see that $B(x) \leq_{\angle} \sigma_0 \wedge \dots \wedge \sigma_n$. So it is possible to build the following derivation:

$$\frac{\begin{array}{c} \overline{B \vdash_{\angle} x : B(x)}^{(var)} \\ \overline{B \vdash_{\angle} x : \sigma_0 \wedge \dots \wedge \sigma_n \rightarrow \sigma}^{(\leq_{\angle})} \end{array}}{B \vdash_{\angle} xx : \sigma} \quad \frac{\begin{array}{c} \overline{B \vdash_{\angle} x : B(x)}^{(var)} \\ \overline{B \vdash_{\angle} x : \sigma_0 \wedge \dots \wedge \sigma_n}^{(\leq_{\angle})} \end{array}}{B \vdash_{\angle} x : \sigma_0 \wedge \dots \wedge \sigma_n}^{(\rightarrow E)}$$

11.3.1 Proof of \mathcal{L} -Approximation Theorem

We define a computability predicate as for the previous models. The notion of approximants is different from the previous one given in Definition 11.1.39, since it takes into account the different behaviour of the \vdash_{\angle} type assignment system with respect to \vdash_{∞} .

A basis B is *finite* if and only if $B(y) \simeq_{\angle} \omega$ except in a finite number of variables. We will use $[\sigma_1/x_1, \dots, \sigma_n/x_n]$ to denote a finite basis. By Lemma 10.1.7.(i), in this section we limit ourselves to consider only such a kind of basis.

Let B and B' be two basis. $B \cup B'$ denotes the basis such that, for every x , $B \cup B'(x) = B(x) \wedge B'(x)$ (remember that, for every type σ , $\sigma \wedge \omega \simeq \sigma$).

Definition 11.3.18. (i) $App_{\mathcal{L}}(B, \sigma, M)$ if and only if there is $A \in \mathcal{A}^{\angle}(M)$ such that $B \vdash_{\angle} A : \sigma$.

(ii) The predicate $Comp_{\mathcal{L}}$ is defined by induction on types as follows:

- $Comp_{\mathcal{L}}(B, \omega, M)$ is true;
- $Comp_{\mathcal{L}}(B, \sigma \rightarrow \tau, M)$ where $\tau \simeq_{\angle} \omega$, if and only if $App_{\mathcal{L}}(B, \omega \rightarrow \omega, M)$;
- $Comp_{\mathcal{L}}(B, \sigma \rightarrow \tau, M)$ where $\tau \not\simeq_{\angle} \omega$, if and only if $\forall N, Comp_{\mathcal{L}}(B', \sigma, N)$ implies $Comp_{\mathcal{L}}(B \cup B', \tau, MN)$;
- $Comp_{\mathcal{L}}(B, \sigma \wedge \tau, M)$ if and only if $Comp_{\mathcal{L}}(B, \sigma, M)$ and $Comp_{\mathcal{L}}(B, \tau, M)$.

To prove the (\Rightarrow) part of the \mathcal{L} -approximation theorem, we will prove, in the usual way, that $B \vdash_{\angle} M : \sigma$ implies $Comp_{\mathcal{L}}(B, M, \sigma)$, which in turn implies $App(B, \sigma, M)$.

It is easy to check, by induction on σ , that $Comp_{\mathcal{L}}(B, \sigma, M)$ and $M =_A M'$ imply $Comp_{\mathcal{L}}(B, \sigma, M')$; by induction on σ , the proof is easier than that of Lemma 11.1.40 pag. 133.

Note that typings are not preserved by the η -reduction, as we observed in the proof of Property 11.3.14.(i). This property was used for proving the approximation theorem in both models \mathcal{H} and \mathcal{N} . Here a weak version of this property holds, just for approximants of a particular shape, but it is sufficient for the rest of the proof.

Property 11.3.19. Let $A \equiv \lambda z.xA_1\dots A_mz$, where $z \notin \text{FV}(xA_1\dots A_m)$. If $B \vdash_{\angle} A : \sigma \rightarrow \tau$, with $\tau \not\simeq_{\angle} \omega$ then $B \vdash_{\angle} xA_1\dots A_m : \sigma \rightarrow \tau$.

Proof. $B \vdash_{\angle} A : \sigma \rightarrow \tau$ and $\tau \not\simeq_{\angle} \omega$ imply $B[\sigma/z] \vdash_{\angle} xA_1\dots A_mz : \tau$, by Lemma 10.1.7.(vi), so $B[\sigma/z] \vdash_{\angle} xA_1\dots A_m : \epsilon \rightarrow \tau$ and $B[\sigma/z] \vdash_{\angle} z : \epsilon$ for some ϵ , by Lemma 10.1.7.(vii). Thus $\sigma \leq_{\angle} \epsilon$, so $\epsilon \rightarrow \tau \leq_{\angle} \sigma \rightarrow \tau$; hence by rule (\leq_{\angle}) $B[\sigma/z] \vdash_{\angle} xA_1\dots A_m : \sigma \rightarrow \tau$.

$B \vdash_{\angle} xA_1\dots A_m : \sigma \rightarrow \tau$, by Lemma 10.1.7.(i) since $z \notin \text{FV}(xA_1\dots A_m)$. \square

Lemma 11.3.20. (i) $\text{App}_{\mathcal{L}}(B, \sigma, x\vec{M})$ implies $\text{Comp}_{\mathcal{L}}(B, \sigma, x\vec{M})$.

(ii) $\text{Comp}_{\mathcal{L}}(B, \sigma, M)$ implies $\text{App}_{\mathcal{L}}(B, \sigma, M)$.

Proof. The proof is done by mutual induction on σ . The only non obvious case is when $\sigma \equiv \tau \rightarrow \rho$ and $\rho \not\simeq_{\angle} \omega$.

- (i) We will prove that $\text{Comp}_{\mathcal{L}}(B', \tau, N)$ implies $\text{Comp}_{\mathcal{L}}(B \cup B', \rho, x\vec{M}N)$, thus $\text{Comp}_{\mathcal{L}}(B, \tau \rightarrow \rho, x\vec{M})$ follows by definition.
 $\text{Comp}_{\mathcal{L}}(B', \tau, N)$ implies $\text{App}_{\mathcal{L}}(B', \tau, N)$, by induction on (ii). By hypothesis $\text{App}_{\mathcal{L}}(B, \tau \rightarrow \rho, x\vec{M})$, so $\text{App}_{\mathcal{L}}(B \cup B', \rho, x\vec{M}N)$ by rule $(\rightarrow E)$, since $\vec{A} \in \mathcal{A}^{\angle}(\vec{M})$ and $A' \in \mathcal{A}^{\angle}(N)$ imply $x\vec{A}A' \in \mathcal{A}^{\angle}(x\vec{M}N)$; hence, $\text{Comp}_{\mathcal{L}}(B \cup B', \rho, x\vec{M}N)$ by induction.
- (ii) Let $z \notin \text{FV}(M)$ and $B(z) \simeq_{\angle} \omega$. Note that both $z \in \mathcal{A}^{\angle}$ and $[\tau/z] \vdash_{\angle} z : \tau$, hence $\text{App}_{\mathcal{L}}([\tau/z], \tau, z)$. Thus by induction on (i), $\text{Comp}_{\mathcal{L}}([\tau/z], \tau, z)$. $\text{Comp}_{\mathcal{L}}(B, \tau \rightarrow \rho, M)$ and $\text{Comp}_{\mathcal{L}}([\tau/z], \tau, z)$ imply $\text{Comp}_{\mathcal{L}}(B[\tau/z], \rho, Mz)$ and this implies $\text{App}_{\mathcal{L}}(B[\tau/z], \rho, Mz)$, by induction; which means there is $A \in \mathcal{A}^{\angle}(Mz)$ such that $B[\tau/z] \vdash_{\angle} A : \rho$. The case $A \equiv \Omega$ is not possible, since by the hypothesis $\rho \not\simeq_{\angle} \omega$. By rule $(\rightarrow I)$, $B \vdash_{\angle} \lambda z.A : \tau \rightarrow \rho$. By definition of the \angle -approximants of a term, $A \in \mathcal{A}^{\angle}(Mz)$ implies $\lambda z.A \in \mathcal{A}^{\angle}(\lambda z.Mz)$. Now there are two cases.
 1. If M is of order 0 then A is of the shape $xA_1\dots A_mz$, where $z \notin \text{FV}(xA_1\dots A_m)$ and $xA_1\dots A_m \in \mathcal{A}^{\angle}(M)$. So $B \vdash_{\angle} xA_1\dots A_m : \tau \rightarrow \rho$, by Property 11.3.19 and the proof is given.
 2. Otherwise $M =_A \lambda y.M'$, so $\lambda z.Mz =_A \lambda z.M'[z/y] =_{\alpha} \lambda y.M'$, which implies $\lambda z.A \in \mathcal{A}^{\vee}(M)$, and the proof is given. \square

Lemma 11.3.21. If $\text{Comp}_{\mathcal{L}}(B, \sigma, M)$ and $\sigma \leq_{\angle} \tau$ then $\text{Comp}_{\mathcal{L}}(B, \tau, M)$.

Proof. By induction on the rules of \leq_{\angle} . The more complex case is the rule (f) , so let $\pi' \leq_{\angle} \pi$, $\tau \leq_{\angle} \tau'$ and $\text{Comp}_{\mathcal{L}}(B, \pi \rightarrow \tau, M)$.

If $\tau \simeq_{\angle} \omega$ then $\tau' \simeq_{\angle} \omega$, so the proof is immediate. If $\tau, \pi \not\simeq_{\angle} \omega$ then the proof follows by induction. If $\tau \not\simeq_{\angle} \omega$ but $\tau' \simeq_{\angle} \omega$ then, by Lemma 11.3.20.(ii), $\text{App}_{\mathcal{L}}(B, \pi \rightarrow \tau, M)$. By definition of $\text{App}_{\mathcal{L}}$, there is an $A \in \mathcal{A}^{\angle}(M)$ such that $B \vdash_{\angle} A : \pi \rightarrow \tau$, thus $B \vdash_{\angle} A : \omega \rightarrow \omega$, and the proof follows by the definition of $\text{Comp}_{\mathcal{L}}$. \square

The following lemma will be used in the proof of the \mathcal{L} -approximation theorem.

Lemma 11.3.22. Let $\text{FV}(M) \subseteq \{x_1, \dots, x_n\}$ and $B = [\sigma_1/x_1, \dots, \sigma_n/x_n]$. If $\text{Comp}_{\mathcal{L}}(B_i, \sigma_i, N_i)$ ($1 \leq i \leq n$) and $B \vdash_{\angle} M : \tau$ then

$$\text{Comp}_{\mathcal{L}}(B_1 \cup \dots \cup B_n, \tau, M[N_1/x_1, \dots, N_n/x_n]).$$

Proof. By induction on the derivation $B \vdash_{\angle} M : \tau$. \square

▼ Proof of \mathcal{L} -Approximation Theorem (Theorem 11.3.10 pag. 166).

(\Rightarrow) Clearly $\text{Comp}_{\mathcal{L}}([\tau/x], \tau, x)$, by Lemma 11.3.20.(i).

Let $\text{FV}(M) \subseteq \{x_1, \dots, x_n\}$, so we can assume $B = [\sigma_1/x_1, \dots, \sigma_n/x_n]$ without loss of generality, by Lemma 10.1.7.(i). Therefore $B \vdash_{\angle} M : \sigma$ and $\text{Comp}_{\mathcal{L}}([\sigma_i/x_i], \sigma_i, x_i)$ ($1 \leq i \leq n$) imply $\text{Comp}(B, \sigma, M)$ by Lemma 11.3.22, which in turn implies $\text{App}_{\mathcal{L}}(B, \sigma, M)$, by Lemma 11.3.20.(ii).

(\Leftarrow) By definition, there is M' such that $M =_A M'$ and A matches M' except at occurrences of Ω . A derivation of $B \vdash_{\angle} A : \sigma$ can be transformed into a derivation of $B \vdash_{\angle} M' : \sigma$, simply by replacing every subderivation

$$\overline{B \vdash_{\angle} \Omega : \omega}^{(\omega)} \quad \text{by} \quad \overline{B \vdash_{\angle} N : \omega}^{(\omega)},$$

where N is the subterm replaced by Ω in M' . $B \vdash_{\angle} M' : \sigma$ implies $B \vdash_{\angle} M : \sigma$, since the type assignment system is closed under $=_A$ on terms as a consequence of the fact that it induces a $\lambda\Lambda$ -model, so the proof is given. \blacksquare

11.3.2 Proof of Theorems 11.3.15 and 11.3.16

Let $L_0 \equiv \lambda x.x(x(\lambda x.DD)(DD))(\lambda x.DD)$,

$L_1 \equiv \lambda x.x(\lambda y.x(\lambda x.DD)(DD)y)(\lambda x.DD)$.

We already showed, in Sect. 6.3, that $L_0 \approx_{\mathbf{L}} L_1$. Now we will prove $L_0 \not\sim_{\mathcal{L}} L_1$, so \mathcal{L} is not complete with respect to the \mathbf{L} -operational semantics.

▼ Proof of \mathcal{L} -Incompleteness Theorem (Theorem 11.3.15 pag. 167).

Let us prove that $L_1 \not\sim_{\mathcal{L}} L_0$; namely we will show that, for every basis B , $B \vdash_{\angle} L_1 : ((\omega \rightarrow \omega) \rightarrow (\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega$, while L_0 has no such a typing. Let $\sigma \equiv (\omega \rightarrow \omega) \rightarrow (\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega$.

$$\begin{array}{c} \frac{}{B[\sigma/x] \vdash_{\angle} x : \sigma}^{(\text{var})} \quad \frac{\overline{B[\sigma/x][\omega/y] \vdash_{\angle} x(\lambda x.DD)(DD)y : \omega}^{(\omega)}}{B[\sigma/x] \vdash_{\angle} \lambda y.x(\lambda x.DD)(DD)y : \omega \rightarrow \omega}^{(\rightarrow I)} \quad \frac{\overline{(B[\sigma/x])[\omega/x] \vdash_{\angle} DD : \omega}^{(\omega)}}{B[\sigma/x] \vdash_{\angle} \lambda x.DD : \omega \rightarrow \omega}^{(\rightarrow I)} \\ \frac{}{B[\sigma/x] \vdash_{\angle} x(\lambda y.x(\lambda x.DD)(DD)y) : (\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega}^{(\rightarrow E)} \quad \frac{}{B[\sigma/x] \vdash_{\angle} x(\lambda y.x(\lambda x.DD)(DD)y)(\lambda x.DD) : \omega \rightarrow \omega}^{(\rightarrow E)} \\ \frac{}{B \vdash_{\angle} L_1 : ((\omega \rightarrow \omega) \rightarrow (\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega}^{(\rightarrow I)} \end{array}$$

On the other hand, assume $B \vdash_{\angle} L_0 : \sigma \rightarrow \omega \rightarrow \omega$; so, by Lemma 10.1.7.(vi) $B[\sigma/x] \vdash_{\angle} x(x(\lambda x.DD)(DD))(\lambda x.DD) : \omega \rightarrow \omega$.

By Lemma 10.1.7.(vii), there is μ such that $B[\sigma/x] \vdash_{\angle} (\lambda x.DD) : \mu$ and $B[\sigma/x] \vdash_{\angle} x(x(\lambda x.DD)(DD)) : \mu \rightarrow \omega \rightarrow \omega$.

By Lemma 10.1.7.(vii), there is ν such that $B[\sigma/x] \vdash_{\angle} x(\lambda x.DD)(DD) : \nu$ and $B[\sigma/x] \vdash_{\angle} x : \nu \rightarrow \mu \rightarrow \omega \rightarrow \omega$.

By Lemma 10.1.7.(ii), $\sigma \leq_{\angle} \nu \rightarrow \mu \rightarrow \omega \rightarrow \omega$, hence $\nu, \mu \leq_{\angle} \omega \rightarrow \omega$ by Property 10.1.6; but $B[\sigma/x] \vdash_{\angle} \lambda x.DD : \mu$ implies, by the \mathcal{L} -approximation theorem and Property 11.3.11, that $\mu \geq_{\angle} \omega \rightarrow \omega$, so $\mu \simeq_{\angle} \omega \rightarrow \omega$ and moreover $\nu \not\leq_{\angle} \omega$.

By Lemma 10.1.7.(vii), there is π such that $B[\sigma/x] \vdash_{\angle} x(\lambda x.DD) : \pi \rightarrow \nu$ and

$$B[\sigma/x] \vdash_{\angle} DD : \pi. \quad (11.5)$$

Again by Lemma 10.1.7.(vii), there is τ such that $B[\sigma/x] \vdash_{\angle} x : \tau \rightarrow \pi \rightarrow \nu$ and $B[\sigma/x] \vdash_{\angle} \lambda x.DD : \tau$. By Lemma 10.1.7.(ii), $\sigma \leq_{\angle} \tau \rightarrow \pi \rightarrow \nu$, so by Property 10.1.6 it follows that $\pi \leq_{\angle} \omega \rightarrow \omega$, which together with Eq. (11.5) and Corollary 11.3.12 is absurd. ■

In order to prove the nonexistence of a fully abstract filter model for the \mathbf{L} -operational semantics, we will prove that in every such model, if it did exist, the two terms L_0 and L_1 would be denotationally different, so reaching an absurdum. Let us notice that, while we said that a natural choice for a call-by-name filter $\lambda\Lambda$ -model is $T(C) = I(C)$, a priori we cannot exclude that a different choice could be made. So, in order to prove the nonexistence of a fully abstract filter model, we must prove it in the case of every (correct) choice of the set of input types.

First, we need some properties.

Property 11.3.23. Let U be any closed Λ -unsolvable term of order 0, and let the type system $\nabla \equiv < C, \leq_{\nabla}, I(C) >$ induce a $\lambda\Lambda$ -model \mathcal{M} that is fully abstract with respect to the \mathbf{L} -operational semantics.

- (i) $U \sqsubset_{\mathcal{M}} \lambda x.U \sqsubset_{\mathcal{M}} \lambda y.x(\lambda x.DD)(DD)y$.
- (ii) There is $\theta \in I(C)$, such that $B \vdash_{\nabla} \lambda x.U : \theta$ but $B \not\vdash_{\nabla} U : \theta$, for all bases B ; furthermore, $B \vdash_{\nabla} \lambda y.x(\lambda x.DD)(DD)y : \theta$, for all bases B .
- (iii) If θ is the type considered in point (ii) then $\theta \rightarrow \theta, \theta \rightarrow \theta \rightarrow \theta \in I(C)$.

Proof. (i) Clearly $U \prec_{\mathbf{L}} \lambda x.U \prec_{\mathbf{L}} \lambda y.x(\lambda x.DD)(DD)y$. So the proof follows from the definition of a fully abstract model.

- (ii) By point (i), by the fact that all Λ -unsolvables of the same order are equated in \mathbf{L} (by Property 11.3.14.(iv)) and by definition of filter model, there is $\theta \in T(C)$ such that $B \vdash_{\nabla} \lambda x.U : \theta$ but $B \not\vdash_{\nabla} U : \theta$, for every basis B (since U is closed). So, always by point (i) and by Property 10.1.13.(ii), for every B , $B \vdash_{\nabla} \lambda y.x(\lambda x.DD)(DD)y : \theta$. Note that U is

an input value for the λA -calculus, so by definition of filter λA -model, there is a type $\alpha \in I(C)$ such that $\alpha \in \llbracket U \rrbracket_{\rho}^{\mathcal{M}}$, for all environments ρ . If we assume $\theta \notin I(C)$ then $\alpha \leq_{\nabla} \theta$, by conditions on input types, thus $B \vdash_{\nabla} U : \theta$, against hypotheses. Hence it must be that $\theta \in I(C)$.

- (iii) Since U is an input value for the λA -calculus, there exists a type $\alpha \in I(C)$ such that $\alpha \in \llbracket U \rrbracket_{\rho}^{\mathcal{M}}$, for all environments ρ . If $\theta \rightarrow \theta \notin I(C)$ then $\alpha \leq_{\nabla} \theta \rightarrow \theta$, thus $B \vdash_{\nabla} U : \theta \rightarrow \theta$. But $B \vdash_{\nabla} \lambda x.U : \theta$ and $\theta \in I(C)$ imply that $B \vdash_{\nabla} U(\lambda x.U) : \theta$, by rule $(\rightarrow E)$; this is an absurdum, since $U(\lambda x.U)$ is itself a closed A -unsolvable term of degree 0, by Property 1.2.18.(ii). In a similar way it is easy to check that $\theta \rightarrow \theta \rightarrow \theta \in I(C)$. \square

▼ **Proof of the Theorem 11.3.16** (pag.167).

Let $\nabla = < C, \leq_{\nabla}, I(C) >$ be a legal type system, inducing a filter λA -model that is fully abstract with respect to the **L**-operational semantics. Let θ be the input type considered in Property 11.3.23 and note that $\theta \not\simeq_{\nabla} \omega$.

Let σ denote $\theta \rightarrow \theta \rightarrow \theta$; since $\sigma \in I(C)$, $B[\sigma/x] \vdash_{\nabla} \lambda y.x(\lambda x.DD)(DD)y : \theta$ and $B \vdash_{\nabla} DD : \theta$, by Property 11.3.23.(ii); so $B \vdash_{\nabla} L_1 : \sigma \rightarrow \theta$, by rules $(\rightarrow E)$ and $(\rightarrow I)$. Now, let us assume $B \vdash_{\nabla} L_0 : \sigma \rightarrow \theta$.

$B[\sigma/x] \vdash_{\nabla} x(x(\lambda x.DD)(DD))(\lambda x.DD) : \theta$, by Lemma 10.1.7.(vi); thus there is an input type μ such that $B[\sigma/x] \vdash_{\nabla} x(x(\lambda x.DD)(DD)) : \mu \rightarrow \theta$ and $B[\sigma/x] \vdash_{\nabla} \lambda x.DD : \mu$, by Lemma 10.1.7.(vii).

If $\mu \rightarrow \theta \simeq_{\nabla} \omega$ then $\mu \rightarrow \omega \leq_{\nabla} \mu \rightarrow \theta$, since $\mu \rightarrow \omega \leq_{\nabla} \omega$ by rule (a) of intersection relations. By Property 10.1.6, $\omega \leq_{\nabla} \theta$ and thus $\theta \simeq_{\nabla} \omega$, which is not possible, since $B \not\vdash_{\nabla} U : \theta$; hence $\mu \rightarrow \theta \not\simeq_{\nabla} \omega$.

So $B[\sigma/x] \vdash_{\nabla} x(\lambda x.DD)(DD) : \nu$ and $B[\sigma/x] \vdash_{\nabla} x : \nu \rightarrow \mu \rightarrow \theta$ for some $\nu \in I(C)$, by Lemma 10.1.7.(vii). By Lemma 10.1.7.(ii), $\sigma \leq_{\nabla} \nu \rightarrow \mu \rightarrow \theta$, hence $\nu, \mu \leq_{\nabla} \theta$ by Property 10.1.6. Again by Lemma 10.1.7.(vii), there is $\pi \in I(C)$ such that $B[\sigma/x] \vdash_{\nabla} x(\lambda x.DD) : \pi \rightarrow \nu$ and

$$B[\sigma/x] \vdash_{\nabla} DD : \pi. \tag{11.6}$$

If $\pi \rightarrow \nu \simeq_{\nabla} \omega$ then $\pi \rightarrow \omega \leq_{\nabla} \omega \leq_{\nabla} \pi \rightarrow \nu$ thus, by Property 10.1.6, $\omega \leq_{\nabla} \nu$, which would be in contradiction with $\theta \not\simeq_{\nabla} \omega$, hence we assume $\pi \rightarrow \nu \not\simeq_{\nabla} \omega$. Yet by Lemma 10.1.7.(vii), there is $\tau \in I(C)$ such that $B[\sigma/x] \vdash_{\nabla} x : \tau \rightarrow \pi \rightarrow \nu$ and $B[\sigma/x] \vdash_{\nabla} (\lambda x.DD) : \tau$. Hence by Lemma 10.1.7.(ii), $\sigma \leq_{\nabla} \tau \rightarrow \pi \rightarrow \nu$, so by Lemma 10.1.6, it follows that $\theta \leq_{\nabla} \nu$ and $\pi, \tau \leq_{\nabla} \theta$. But $\pi \leq_{\nabla} \theta$ and Eq. (11.6) imply an absurd, since from rule (\leq_{∇}) , it follows $B[\sigma/x] \vdash_{\nabla} DD : \theta$, against Property 11.3.23.(ii). ■

11.4 A Fully Abstract Model for the L-Operational Semantics

It was proved, in Theorem 11.3.16, that there is not a filter model that is fully abstract with respect to the **L**-operational semantics. But we will show

now that it is possible to build, starting from the model \mathcal{L} , the desired fully abstract model. In a very general way, the idea is to start from the model \mathcal{L} , to build a space of filters that is a subspace of $\mathcal{F}(\angle)$, consisting just of filters that are interpretations of closed terms, and then to identify those filters that are interpretations of terms L-operationally equivalent. The so-obtained model is no longer a filter model, and the interpretation of a term is no longer the collection of the types that can be assigned to it. The fully abstract model amounts just to the closed term model of the Λ -theory \mathbf{L} , equipped with a partial order relation, which is obtained from a preorder on terms, defined in a stratified way by using the type assignment \vdash_{\angle} .

Definition 11.4.1. (i) \trianglelefteq_{σ} is a relation on Λ^0 defined as follows:

- $M \trianglelefteq_{\omega} N$ is true;
- $M \trianglelefteq_{\sigma \rightarrow \tau} N$, where $\tau \simeq_{\angle} \omega$, if and only if
 $B \vdash_{\angle} M : \omega \rightarrow \omega$ implies $B \vdash_{\angle} N : \omega \rightarrow \omega$, for all bases B ;
- $M \trianglelefteq_{\sigma \rightarrow \tau} N$, where $\tau \not\simeq_{\angle} \omega$, if and only if
 $\forall P \in \Lambda^0, B \vdash_{\angle} P : \sigma$ implies $MP \trianglelefteq_{\tau} NP$;
- $M \trianglelefteq_{\sigma \wedge \tau} N$ if and only if both $M \trianglelefteq_{\sigma} N$ and $M \trianglelefteq_{\tau} N$.

(ii) $M \trianglelefteq N$ if and only if $M \trianglelefteq_{\sigma} N$, for all σ .

The next property will be useful in order to better understand the previous definition.

Property 11.4.2. There is $P \in \Lambda^0$ such that $B \vdash_{\angle} P : \sigma$, for all B and σ .

Proof. We will prove, by induction on σ , that there is P of the shape: $\lambda x_1 \dots x_n. DD$, for $n \geq 0$, to which σ can be assigned.

If $\sigma \simeq_{\angle} \omega$, then $B \vdash_{\angle} DD : \sigma$, by rules (ω) and (\leq_{\angle}) .

If $\sigma \simeq_{\angle} \omega \rightarrow \omega$, then, by rule (ω) , $B[\tau/x] \vdash_{\angle} DD : \omega$, and, by rule $(\rightarrow I)$, $B \vdash_{\angle} \lambda x. DD : \tau \rightarrow \omega$. The proof follows by rule (\leq_{\angle}) .

If $\sigma \equiv \mu \rightarrow \nu$, where $\nu \not\simeq_{\angle} \omega$, then by induction there is $P \in \Lambda^0$ such that $B \vdash_{\angle} P : \nu$. Hence $B[\mu/x] \vdash_{\angle} P : \nu$, and the proof follows by rule $(\rightarrow I)$.

Let $\sigma \equiv \mu \wedge \nu$. By induction, there are $\lambda x_1 \dots x_p. DD$ and $\lambda x_1 \dots x_q. DD$ such that $B \vdash_{\angle} \lambda x_1 \dots x_p. DD : \mu$ and $B \vdash_{\angle} \lambda x_1 \dots x_q. DD : \nu$. Let $n = \max\{p, q\}$: $\lambda x_1 \dots x_n. DD$ is the desired term by Lemma 11.3.5.(ii). \square

If a type can be assigned to a closed term, then it is said to be *inhabited*. Note that although in the model \mathcal{L} all types are inhabited, this does not imply that all filters are inhabited, i.e. it does not imply that every filter is the interpretation of a closed term. Indeed, the filter

$$\uparrow \{(\omega \rightarrow \omega) \rightarrow (\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega\}$$

is not the interpretation of any term, since the reader can check that every term having type $(\omega \rightarrow \omega) \rightarrow (\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega$ also has the type $\omega \rightarrow \omega \rightarrow \omega \rightarrow \omega$, which is not in the filter. If this filter were inhabited, then it would be $L_0 \not\approx_{\mathbf{L}} L_1$ (see Sect. 11.3.2).

Property 11.4.3. Let $M, N \in \Lambda^0$.

- (i) If $M \sqsubseteq_{\mathcal{L}} N$ then $M \sqsubseteq N$.
- (ii) \sqsubseteq is reflexive.
- (iii) \sqsubseteq is transitive.

Proof. (i) We will prove that $M \not\sqsubseteq N$ implies $M \not\sqsubseteq_{\mathcal{L}} N$. By definition, $M \not\sqsubseteq N$ means there is σ such that $M \not\sqsubseteq_{\sigma} N$. The proof is given by induction on σ .

Clearly $\sigma \not\simeq_{\angle} \omega$, since by definition $M \sqsubseteq_{\omega} N$ is true. If $\sigma \equiv \mu \rightarrow \nu$ where $\nu \simeq_{\angle} \omega$, then $B \vdash_{\angle} M : \omega \rightarrow \omega$ and $B \not\vdash_{\angle} N : \omega \rightarrow \omega$ by definition 11.4.1, so the proof is immediate.

If $\sigma \equiv \mu \rightarrow \nu$ where $\nu \not\simeq_{\angle} \omega$, then there is $P \in \Lambda^0$ such that $MP \not\sqsubseteq_{\nu} NP$, by definition of \sqsubseteq . Hence, $MP \not\sqsubseteq_{\mathcal{L}} NP$ by induction, so $M \not\sqsubseteq_{\mathcal{L}} N$ by Lemma 10.1.13.(i).

If $\sigma \equiv \mu \wedge \nu$ then the proof follows by induction.

- (ii) We will prove that $M \sqsubseteq_{\sigma} M$, for all σ , by induction on σ . The case ω is obvious. Let $\sigma \equiv \mu \rightarrow \nu$; the case $\nu \simeq_{\angle} \omega$ is obvious. Let $\nu \not\simeq_{\angle} \omega$, thus $P \in \Lambda^0$ and $B \vdash_{\angle} P : \mu$ imply $MP \sqsubseteq_{\nu} MP$, by induction; the proof follows by definition of \sqsubseteq . The case $\sigma \equiv \mu \wedge \nu$ follows by induction.

- (iii) We will prove that \sqsubseteq_{σ} is transitive, for all σ , by induction on σ .

The cases ω and $\mu \rightarrow \nu$ where $\nu \simeq_{\angle} \omega$ are obvious.

Let $\sigma \equiv \mu \rightarrow \nu$, $\nu \not\simeq_{\angle} \omega$ and $M \sqsubseteq_{\mu \rightarrow \nu} N \sqsubseteq_{\mu \rightarrow \nu} P$. If $Q \in \Lambda^0$ and $B \vdash_{\angle} Q : \mu$ then $MQ \sqsubseteq_{\nu} NQ \sqsubseteq_{\nu} PQ$, and by induction $MQ \sqsubseteq_{\nu} PQ$. Thus, the proof follows by the definition of $\sqsubseteq_{\mu \rightarrow \nu}$.

The case $\sigma \equiv \mu \wedge \nu$ follows by induction. \square

In next two lemmas it is proved that \sqsubseteq corresponds exactly to the operational inclusion $\preceq_{\mathbf{L}}$.

Lemma 11.4.4. Let $M, N \in \Lambda^0$.

$M \sqsubseteq N$ if and only if $M\vec{P} \sqsubseteq_{\omega \rightarrow \omega} N\vec{P}$ for each sequence of closed terms \vec{P} .

Proof. (\Leftarrow) We will prove that $M \not\sqsubseteq N$ implies that there is a closed sequence of terms \vec{P} such that $M\vec{P} \not\sqsubseteq_{\omega \rightarrow \omega} N\vec{P}$. By hypothesis there is a type σ such that $M \not\sqsubseteq_{\sigma} N$, so the proof is done by induction on σ .

If $\sigma \simeq_{\angle} \omega$ then $\sigma \equiv \underbrace{\omega \wedge \dots \wedge \omega}_n$ ($n \geq 1$), by Theorem 11.3.3; but, since $M \sqsubseteq_{\omega} N$ by definition, this is not possible.

Let $\sigma \not\simeq_{\angle} \omega$. If $\sigma \equiv \mu \rightarrow \nu$ where $\nu \simeq_{\angle} \omega$, then the proof is vacuous. If $\sigma \equiv \mu \rightarrow \nu$ where $\nu \not\simeq_{\angle} \omega$, then there is $P \in \Lambda^0$ such that $MP \not\sqsubseteq_{\nu} NP$, so the proof follows by induction. If $\sigma \equiv \mu \wedge \nu$ then the proof follows by induction.

(\Rightarrow) We will prove that if there is a sequence of closed terms \vec{P} and a type $\tau \not\simeq_{\angle} \omega$ such that $M\vec{P} \not\sqsubseteq_{\tau} N\vec{P}$ then $M \not\sqsubseteq N$, by induction on $\|\vec{P}\|$.

If $\|\vec{P}\| = 0$ then the proof is trivial, so let $\|\vec{P}\| \geq 1$ and $\vec{P} \equiv \vec{Q}\vec{Q}'$. But

$B \vdash_{\angle} Q' : \omega$ by rule (ω) implies $M\vec{Q} \not\trianglelefteq_{\omega \rightarrow \tau} N\vec{Q}$ by definition of \trianglelefteq ; so the proof follows by induction. \square

Note that, although $\lambda x.DD \not\trianglelefteq_{\omega \rightarrow \omega} DD$, $\lambda x.DD \trianglelefteq_{\omega \rightarrow \omega \rightarrow \omega} DD$; in fact for each $P \in \Lambda^0$, $B \vdash_{\angle} P : \omega$, and so, by definition of \trianglelefteq , $(\lambda x.DD)P \trianglelefteq_{\omega \rightarrow \omega} (DD)P$ is true. Hence $M \trianglelefteq_{\sigma} N$ and $\sigma \leq_{\angle} \tau$ does not imply $M \trianglelefteq_{\tau} N$.

Lemma 11.4.5. *Let $M, N \in \Lambda^0$.*

$M \preceq_{\mathbf{L}} N$ if and only if $M\vec{P} \trianglelefteq_{\omega \rightarrow \omega} N\vec{P}$, for each sequence of closed terms \vec{P} .

Proof. Remember that, for every term Q , $Q \Downarrow_{\mathbf{L}}$ if and only if $B \vdash_{\angle} Q : \omega \rightarrow \omega$, by Corollary 11.3.12.

(\Rightarrow) Let \vec{P} be a sequence of closed terms and let B be a basis.

If $M \preceq_{\mathbf{L}} N$ then $M\vec{P} \Downarrow_{\mathbf{L}}$ implies $N\vec{P} \Downarrow_{\mathbf{L}}$; therefore, by Corollary 11.3.12, $B \vdash_{\angle} M\vec{P} : \omega \rightarrow \omega$ implies $B \vdash_{\angle} N\vec{P} : \omega \rightarrow \omega$. Hence, by definition of $\trianglelefteq_{\omega \rightarrow \omega}$ the proof is done.

(\Leftarrow) Let $M\vec{P} \trianglelefteq_{\omega \rightarrow \omega} N\vec{P}$, for each sequence of closed terms \vec{P} . We will prove that, if $C[M], C[N] \in \Lambda^0$ and $C[M] \Downarrow_{\mathbf{L}}$, then $C[N] \Downarrow_{\mathbf{L}}$, for all contexts $C[.]$. The proof is done by induction on the derivation proving $C[M] \Downarrow_{\mathbf{L}}$. If the last applied rule is either *(var)* or *(lazy)*, then the proof is immediate. If the last applied rule is *(head)*, then there are two cases, according to the possible shape of $C[.]$.

- $C[.] \equiv [.]C_1[.] \dots C_m[.]$ ($m \in \mathbb{N}$).

If $m = 0$ then $M \Downarrow_{\mathbf{L}}$ implies $B \vdash_{\angle} M : \omega \rightarrow \omega$, so $B \vdash_{\angle} N : \omega \rightarrow \omega$ by definition of $\trianglelefteq_{\omega \rightarrow \omega}$, and the proof follows by Corollary 11.3.12.

Now, let $m \geq 1$ and $D[.] \equiv MC_1[.] \dots C_m[.]$. Clearly $D[M] \equiv C[M]$ and $D[.] \equiv (\lambda z.M_0)\vec{MC}_1[.] \dots C_m[.]$ ($m \in \mathbb{N}$), since $M \in \Lambda^0$.

If $\|\vec{M}\| = 0$ then let $D^*[.] \equiv M_0[C_1[.]/z]C_2[.] \dots C_m[.]$, otherwise let $D^*[.] \equiv M_0[M_1/z]\vec{R}C_1[.] \dots C_m[.]$ where $\vec{M} \equiv M_1\vec{R}$. In all cases $D^*[M] \Downarrow_{\mathbf{L}}$ and by induction $D^*[N] \Downarrow_{\mathbf{L}}$, so $D[N] \Downarrow_{\mathbf{L}}$ by rule *(head)*. But $MC_1[N] \dots C_m[N] \Downarrow_{\mathbf{L}}$ implies $B \vdash_{\angle} MC_1[N] \dots C_m[N] : \omega \rightarrow \omega$, so by hypothesis $B \vdash_{\angle} NC_1[N] \dots C_m[N] : \omega \rightarrow \omega$. Hence, $NC_1[N] \dots C_m[N] \Downarrow_{\mathbf{L}}$ by Corollary 11.3.12.

- $C[.] \equiv (\lambda y.C_0[.])C_1[.] \dots C_m[.]$ ($m \in \mathbb{N}$).

The case $m = 0$ is not possible, otherwise the proof follows by induction on the derivation proving $C_0[M][C_1[M]/y]C_2[M] \dots C_m[M] \Downarrow_{\mathbf{L}}$. \square

Theorem 11.4.6. *$M \trianglelefteq N$ if and only if $M \preceq_{\mathbf{L}} N$, for all $M, N \in \Lambda^0$.*

Proof. By Lemmas 11.4.4 and 11.4.5. \square

The next definition overloads the meaning of \trianglelefteq on a subset of filters.

Definition 11.4.7. *Let $f, g \in \mathcal{F}^0(\angle)$ and let ρ be an environment.*

$f \trianglelefteq g$ if and only if $M, N \in \Lambda^0$, $\llbracket M \rrbracket_{\rho}^{\mathcal{F}(\angle)} = f$ and $\llbracket N \rrbracket_{\rho}^{\mathcal{F}(\angle)} = g$ imply $M \trianglelefteq N$. Moreover, $f \triangleq g$ if and only if $f \trianglelefteq g$ and $g \trianglelefteq f$.

Note that if M is closed then $\llbracket M \rrbracket_{\rho}^{\mathcal{L}} = \llbracket M \rrbracket_{\rho'}^{\mathcal{L}}$, for all ρ, ρ' ; moreover, if M, N are closed then $\llbracket M \rrbracket_{\rho}^{\mathcal{L}} = \llbracket N \rrbracket_{\rho'}^{\mathcal{L}}$ implies $M \trianglelefteq N$ and $N \trianglelefteq M$, by Property 11.4.3.(i). Now we can define the new $\lambda\Lambda$ -model.

Definition 11.4.8. Let $f, g \in \mathcal{F}^0(\angle)$.

- (i) $[f]$ is the equivalence class of f with respect to the equivalence relation \trianglelefteq , while $\mathcal{F}_{\trianglelefteq}^0$ is the set of equivalence classes induced from \trianglelefteq on $\mathcal{F}^0(\angle)$.
- (ii) $\circ_{\trianglelefteq} : \mathcal{F}_{\trianglelefteq}^0 \times \mathcal{F}_{\trianglelefteq}^0 \rightarrow \mathcal{F}_{\trianglelefteq}^0$ is defined as $[f] \circ_{\trianglelefteq} [g] = [f \circ_{\angle} g]$, for all $[f], [g] \in \mathcal{F}_{\trianglelefteq}^0$.
- (iii) The interpretation function $\llbracket \cdot \rrbracket^{\mathcal{LL}} : A \times (\text{Var} \rightarrow \mathcal{F}_{\trianglelefteq}^0) \rightarrow \mathcal{F}_{\trianglelefteq}^0$ is defined as: $\llbracket M \rrbracket_{\zeta}^{\mathcal{LL}} = \llbracket \llbracket M \rrbracket_{\rho}^{\mathcal{F}(\angle)} \rrbracket$, where ρ is such that $\rho(x) \in \zeta(x)$ for all $x \in \text{Var}$.
- (iv) Let \mathcal{LL} be the quadruple: $\langle \mathcal{F}_{\trianglelefteq}^0, \mathcal{F}_{\trianglelefteq}^0, \circ_{\trianglelefteq}, \llbracket \cdot \rrbracket^{\mathcal{LL}} \rangle$.

Note that the interpretation is defined for open terms too.

Property 11.4.9. Let $M, N, P, Q \in A^0$. If $M \trianglelefteq N$ and $P \trianglelefteq Q$ then $MP \trianglelefteq NQ$.

Proof. Clearly $M \preceq_L N$ and $P \preceq_L Q$ imply $MP \preceq_L NQ$, therefore the proof follows by Theorem 11.4.6. \square

It is easy to see that \circ_{\trianglelefteq} is well defined, by using the previous property.

Lemma 11.4.10. \mathcal{LL} is a $\lambda\Lambda$ -model.

Proof. We check that \mathcal{LL} satisfies the conditions of Definition 10.0.1.

If $\zeta \in (\text{Var} \rightarrow \mathcal{F}_{\trianglelefteq}^0)$ then let ρ be such that $\rho(x) \in \zeta(x)$ for all $x \in \text{Var}$.

1. $\llbracket x \rrbracket_{\zeta}^{\mathcal{LL}} = \llbracket \llbracket x \rrbracket_{\rho}^{\mathcal{F}(\angle)} \rrbracket = [\rho(x)] = \zeta(x)$.
2. $\llbracket MN \rrbracket_{\zeta}^{\mathcal{LL}} = \llbracket \llbracket MN \rrbracket_{\rho}^{\mathcal{F}(\angle)} \rrbracket = \llbracket \llbracket M \rrbracket_{\rho}^{\mathcal{F}(\angle)} \circ_{\angle} \llbracket N \rrbracket_{\rho}^{\mathcal{F}(\angle)} \rrbracket = \llbracket \llbracket M \rrbracket_{\rho}^{\mathcal{F}(\angle)} \rrbracket \circ_{\trianglelefteq} \llbracket \llbracket N \rrbracket_{\rho}^{\mathcal{F}(\angle)} \rrbracket = \llbracket M \rrbracket_{\zeta}^{\mathcal{LL}} \circ_{\trianglelefteq} \llbracket N \rrbracket_{\zeta}^{\mathcal{LL}}$.
3. $\llbracket \lambda x. M \rrbracket_{\zeta}^{\mathcal{LL}} \circ_{\trianglelefteq} d = \llbracket \llbracket \lambda x. M \rrbracket_{\rho}^{\mathcal{F}(\angle)} \rrbracket \circ_{\trianglelefteq} d = \llbracket \llbracket \lambda x. M \rrbracket_{\zeta}^{\mathcal{F}(\angle)} \circ_{\angle} f \rrbracket = \llbracket \llbracket M \rrbracket_{\zeta[f/x]}^{\mathcal{F}(\angle)} \rrbracket = \llbracket M \rrbracket_{\zeta[d/x]}^{\mathcal{LL}}$, for all $d \in \mathcal{F}_{\trianglelefteq}^0$ and $f \in d$.
4. Let $\llbracket M \rrbracket_{\zeta[d/x]}^{\mathcal{LL}} = \llbracket N \rrbracket_{\zeta'[d'/x']}^{\mathcal{LL}}$, where $d, d' \in \mathcal{F}_{\trianglelefteq}^0$, $f \in d$ and $f' \in d'$. $\llbracket M \rrbracket_{\rho[f/x]}^{\mathcal{F}(\angle)} = \llbracket N \rrbracket_{\rho[f'/x']}^{\mathcal{F}(\angle)}$ since \mathcal{L} is a model, so $\llbracket \llbracket M \rrbracket_{\rho[f/x]}^{\mathcal{F}(\angle)} \rrbracket = \llbracket \llbracket N \rrbracket_{\rho[f'/x']}^{\mathcal{F}(\angle)} \rrbracket$, therefore $\llbracket \llbracket \lambda x. M \rrbracket_{\rho} \rrbracket = \llbracket \llbracket \lambda x'. N \rrbracket_{\rho'} \rrbracket$ thus $\llbracket \llbracket \lambda x. M \rrbracket_{\zeta}^{\mathcal{LL}} \rrbracket = \llbracket \llbracket \lambda x'. N \rrbracket_{\zeta'}^{\mathcal{LL}} \rrbracket$.
5. Trivial. \square

Since \trianglelefteq is a preorder on $\mathcal{F}^0(\angle)$ it induces a partial order on $\mathcal{F}_{\trianglelefteq}^0$.

Definition 11.4.11. Let $M \sqsubseteq_{\mathcal{LL}} N$ denote $\llbracket M \rrbracket_{\zeta}^{\mathcal{LL}} \trianglelefteq \llbracket N \rrbracket_{\zeta}^{\mathcal{LL}}$, for each $\zeta \in (\text{Var} \rightarrow \mathcal{F}_{\trianglelefteq}^0)$. Moreover, let $M \sim_{\mathcal{LL}} N$ denote $M \sqsubseteq_{\mathcal{LL}} N$ and $N \sqsubseteq_{\mathcal{LL}} M$.

Consequently, the model \mathcal{LL} induces a partial order on the interpretation of terms (not only closed terms).

Property 11.4.12. Let $M, N \in \Lambda^0$. $M \sqsubseteq_{\mathcal{LL}} N$ if and only if $M \trianglelefteq N$.

Proof. Let $\zeta \in (\text{Var} \rightarrow \mathcal{F}_{\triangleleft}^0)$ and let ρ be such that $\rho(x) \in \zeta(x)$ for all $x \in \text{Var}$. $M \sqsubseteq_{\mathcal{LL}} N$ if and only if $\llbracket M \rrbracket_{\zeta}^{\mathcal{LL}} \trianglelefteq \llbracket N \rrbracket_{\zeta}^{\mathcal{LL}}$ if and only if $\llbracket \llbracket M \rrbracket_{\rho}^{\mathcal{F}(\zeta)} \rrbracket \trianglelefteq \llbracket \llbracket N \rrbracket_{\rho}^{\mathcal{F}(\zeta)} \rrbracket$ if and only if $\llbracket M \rrbracket_{\rho}^{\mathcal{F}(\zeta)} \trianglelefteq \llbracket N \rrbracket_{\rho}^{\mathcal{F}(\zeta)}$ if and only if $M \trianglelefteq N$. \square

The correctness is easy.

Theorem 11.4.13 (\mathcal{LL} -Correctness).

*The model \mathcal{LL} is correct with respect to the **L**-operational semantics.*

Proof. We will prove that $M \sqsubseteq_{\mathcal{LL}} N$ implies $M \trianglelefteq_{\mathbf{L}} N$, by definition of correctness. $M \sqsubseteq_{\mathcal{LL}} N$ implies $C[M] \sqsubseteq_{\mathcal{LL}} C[N]$, for each closing context $C[.]$, by Property 10.0.2.(v). Thus $C[M] \trianglelefteq C[N]$, by Lemma 11.4.12; hence $C[M] \trianglelefteq_{\omega \rightarrow \omega} C[N]$, thus $B \vdash_{\angle} C[M] : \omega \rightarrow \omega$ implies $B \vdash_{\angle} C[N] : \omega \rightarrow \omega$, for all bases B . So, by Corollary 11.3.12, $C[M] \Downarrow_{\mathbf{L}}$ implies $C[N] \Downarrow_{\mathbf{L}}$, and so $M \trianglelefteq_{\mathbf{L}} N$. \square

The following theorem implies the full abstraction of \mathcal{LL} with respect to the **L**-operational semantics.

Theorem 11.4.14 (\mathcal{LL} -Completeness).

*The model \mathcal{LL} is complete with respect to the **L**-operational semantics.*

Proof. We will prove $\not\sqsubseteq_{\mathcal{LL}}$ implies $\not\trianglelefteq_{\mathbf{L}}$.

$M \not\sqsubseteq_{\mathcal{LL}} N$ means $\llbracket M \rrbracket_{\zeta}^{\mathcal{LL}} \not\trianglelefteq \llbracket N \rrbracket_{\zeta}^{\mathcal{LL}}$, for some $\zeta \in (\text{Var} \rightarrow \mathcal{F}_{\triangleleft}^0)$.

Since the codomain of ζ is $\mathcal{F}_{\triangleleft}^0$, if $\text{FV}(M) \cup \text{FV}(N) = \{x_1, \dots, x_m\}$ then there are $P_i \in \Lambda^0$ such that $\zeta(x_i) = \llbracket \llbracket P_i \rrbracket_{\rho}^{\mathcal{F}(\zeta)} \rrbracket$. Thus, let \mathbf{s} be such that $\mathbf{s}(x_i) = P_i$ ($1 \leq i \leq m$), hence $\mathbf{s}(M), \mathbf{s}(N) \in \Lambda^0$.

$\llbracket \mathbf{s}(M) \rrbracket_{\zeta'}^{\mathcal{LL}} \not\trianglelefteq \llbracket \mathbf{s}(N) \rrbracket_{\zeta'}^{\mathcal{LL}}$, for all $\zeta' \in (\text{Var} \rightarrow \mathcal{F}_{\triangleleft}^0)$ by Property 10.0.2.(iv), since $\mathbf{s}(M)$ and $\mathbf{s}(N)$ are closed. Therefore $\mathbf{s}(M) \not\sqsubseteq_{\mathcal{LL}} \mathbf{s}(N)$ and, by Lemma 11.4.12, $\mathbf{s}(M) \not\trianglelefteq \mathbf{s}(N)$. So there is a sequence of closed terms \vec{Q} such that $\mathbf{s}(M)\vec{Q} \not\trianglelefteq_{\omega \rightarrow \omega} \mathbf{s}(N)\vec{Q}$, by Lemma 11.4.4.

Let $C[.] \equiv (\lambda x_1 \dots x_m . [.]) \mathbf{s}(x_1) \dots \mathbf{s}(x_m) \vec{Q}$; clearly $C[M], C[N] \in \Lambda^0$ and, moreover it is such that $C[M] \Downarrow_{\mathbf{L}}$ and $C[N] \Uparrow_{\mathbf{L}}$, so $M \not\sqsubseteq_{\mathcal{LL}} N$. \square

Hence, \mathcal{LL} is fully abstract with respect to the **L**-operational semantics.

Corollary 11.4.15. If $M \not\trianglelefteq_{\mathbf{L}} N$ then there is a head context separating M and N .

Proof. Immediate, by the proof of Theorem 11.4.14. \square

The technique used here for building the fully abstract model of the **L**-operational semantics is similar to that used in [71] and [44], for different calculi. The use of intersection types and filter models allows for the application of such techniques to a wider class of models. A further fully abstract model for the **L**-operational semantics, based on a variant of the game semantics, was built in [42].

11.5 Crossing Models

It can be interesting to ask for the behaviour of the three filter Λ -models we defined, with respect to all the considered call-by-name operational semantics.

11.5.1 The Model \mathcal{H}

In Sect. 11.1, we already proved that \mathcal{H} is fully abstract with respect to the **H**-operational semantics.

It can be readily noticed that \mathcal{H} is not correct with respect to the **N**-operational semantics. In fact, by Property 11.2.22 and by Theorem 11.1.31, $I \sqsubseteq_{\mathcal{H}} E_{\infty}$, but $I \not\sqsubseteq_{\mathbf{N}} E_{\infty}$.

On the other hand, \mathcal{H} is complete with respect to the **N**-operational semantics, namely $M \preceq_{\mathbf{N}} N$ implies $M \sqsubseteq_{\mathcal{H}} N$. Let us prove that $M \not\sqsubseteq_{\mathcal{H}} N$ implies $M \not\preceq_{\mathbf{N}} N$. Let $M \not\sqsubseteq_{\mathcal{H}} N$; so by the correctness of \mathcal{H} with respect to **H**-operational semantics, this implies that there is a closing context $C[.]$ such that $C[M]$ has Λ -hnf, while $C[N]$ has not Λ -hnf. So there are $P_1 \dots P_n$ such that $C[M]P_1 \dots P_n =_{\Lambda} I$, while $C[N]P_1 \dots P_n$ has no Λ -hnf. Since I is a Λ -nf and a term without Λ -hnf in particular does not have Λ -nf, $C'[.] \equiv C[.]P_1 \dots P_n$ is a context such that $C'[M] \Downarrow_{\mathbf{N}}$ while $C'[N] \Uparrow_{\mathbf{N}}$, so $M \not\preceq_{\mathbf{N}} N$.

\mathcal{H} is not correct with respect to the **L**-operational semantics. In fact, $\lambda y.xy \sqsubseteq_{\mathcal{H}} x$ while $\lambda y.xy \not\sqsubseteq_{\mathbf{L}} x$, as shown in Example 11.3.17. Another counterexample is the pair of terms DD and $\lambda x.DD$; indeed, by Corollary 11.1.21 and Property 11.3.11, $\lambda x.DD \sqsubseteq_{\mathcal{H}} DD$ but $\lambda x.DD \not\sqsubseteq_{\mathbf{L}} DD$.

\mathcal{H} is not complete with respect to the **L**-operational semantics. In fact, for every call-by-name fixed-point operator Z , ZK is a Λ -unsolvable term of infinite order, so $ZK \sqsubseteq_{\mathcal{H}} I$ while $ZK \not\sqsubseteq_{\mathbf{L}} I$ (see Property 11.3.14.(iii)).

But if we take into account the equivalence relation, we have that $\approx_{\mathbf{L}}$ implies $\sim_{\mathcal{H}}$. In fact, the following lemma holds, which, together with the completeness of \mathcal{H} with respect to the **H**-operational semantics, proves this implication.

Lemma 11.5.1. *If $M \approx_{\mathbf{L}} N$ then $M \approx_{\mathbf{H}} N$.*

Proof. We prove that $M \not\approx_{\mathbf{H}} N$ implies $M \not\approx_{\mathbf{L}} N$. By hypothesis there is a context $C[.]$ such that $C[M], C[N] \in \Lambda^0$, $C[M] \Downarrow_{\mathbf{H}}$ and $C[N] \Uparrow_{\mathbf{H}}$ (or vice versa). Thus there is a sequence of terms $P_1 \dots P_n$ such that $C[M]P_1 \dots P_n =_{\Lambda} I$ and $C[N]P_1 \dots P_n$ is Λ -unsolvable, since the unsolvability is closed under head contexts. If $C[N]P_1 \dots P_n$ is Λ -unsolvable of the finite order p then let $C^*[.] \equiv C[.]P_1 \dots P_n \underbrace{I \dots I}_p$. Clearly $C^*[M] \Downarrow_{\mathbf{L}}$ while $C^*[N] \Uparrow_{\mathbf{L}}$, so $M \not\approx_{\mathbf{L}} N$.

Otherwise, $C[N]P_1 \dots P_n$ must be a Λ -unsolvable of infinite order, therefore let $C^*[.] \equiv C[.]P_1 \dots P_n(DD)$. Clearly $C^*[N] \Downarrow_{\mathbf{L}}$ while $C^*[M] \Uparrow_{\mathbf{L}}$, so $M \not\approx_{\mathbf{L}} N$. \square

11.5.2 The Model \mathcal{N}

In Sect. 11.2, we already proved \mathcal{N} is fully abstract with respect to the **N**-operational semantics.

Lemma 10.0.4 and Property 11.2.11.(ii) imply that \mathcal{N} is correct with respect to the **H**-operational semantics. On the other hand, it is easy to see that it is not complete; in fact, by Property 11.2.22 and by Theorem 11.1.31, $I \preceq_{\mathbf{H}} E_\infty$ while $I \not\sqsubseteq_{\mathcal{N}} E_\infty$.

Finally, \mathcal{N} is not correct with respect to the **L**-operational semantics; in fact, by Property 11.2.15.(ii) and by Example 11.3.17, $\lambda y.xy \sqsubseteq_{\mathcal{N}} x$, but $\lambda y.xy \not\preceq_{\mathbf{L}} x$. On the other hand, \mathcal{N} is not complete with respect to the **L**-operational semantics; in fact, $I \preceq_{\mathbf{L}} ZK$ while $I \not\sqsubseteq_{\mathcal{N}} ZK$, for every call-by-name fixed-point operator Z (see Property 11.3.14.(iii)). Note that ZK is a Λ -unsolvable term of infinite order, hence it is easy to see $ZK \sqsubset_{\mathcal{N}} I$.

11.5.3 The Model \mathcal{L}

In Sect. 11.3, we already proved that \mathcal{L} is correct but not complete with respect to the **L**-operational semantics.

\mathcal{L} is not correct with respect to both **H** and **N**-operational semantics; in fact, $I \sqsubseteq_{\mathcal{L}} ZK$ (see Property 11.3.14.(iii)) while $I \not\preceq_{\mathbf{H}} ZK$ and $I \not\preceq_{\mathbf{N}} ZK$, for every call-by-name fixed point operator Z , by Corollary 11.1.21 and by Property 11.2.15.(i).

\mathcal{L} is not complete with respect to both **H** and **N**-operational semantics; in fact, $\lambda x.DD \preceq_{\mathbf{H}} DD$ and $\lambda x.DD \preceq_{\mathbf{N}} DD$ while $\lambda x.DD \not\sqsubseteq_{\mathcal{L}} DD$.

12. Call-by-Value Denotational Semantics

For modeling the $\lambda\Gamma$ -calculus, we must reflect in the model the fact that the set Γ of input values is a proper subset of the whole set Λ . In the setting of filter $\lambda\Gamma$ -models, this implies that every type system ∇ inducing a filter $\lambda\Gamma$ -model must be such that $\mathcal{I}(\nabla) \subset \mathcal{F}(\nabla)$.

Since ω is the universal type it cannot characterize any property of terms; note that from $B \vdash_{\nabla} \lambda x.M : \omega \rightarrow \sigma$ we cannot conclude $B \vdash_{\nabla} (\lambda x.M)N : \sigma$ for every N in a correct $\lambda\Gamma$ -model. Indeed, $(\lambda x.M)(DD) \uparrow_{\mathbf{V}}$ and so every type σ such that $B \vdash_{\nabla} (\lambda x.M)(DD) : \sigma$ must be such that $\sigma \notin I(C)$. Thus the type $\omega \rightarrow \sigma$ cannot have a meaningful applicative power. This is simply formalized by assuming $\omega \notin I(C)$, since the condition on the rule $(\rightarrow E)$ of the intersection type assignment system, namely

$$\frac{\sigma \in I(C) \quad B \vdash_{\nabla} M : \sigma \rightarrow \tau \quad B \vdash_{\nabla} N : \sigma}{B \vdash_{\nabla} MN : \tau} (\rightarrow E).$$

The simplest choice in order to build a filter $\lambda\Gamma$ -model is to choose a legal type system $\nabla = < C, \leq_{\nabla}, I(C) >$ such that $I(C) = \{\sigma \in T(C) \mid \sigma \not\simeq_{\nabla} \omega\}$.

12.1 The Model \mathcal{V}

In this section, we will define a filter $\lambda\Gamma$ -model that is correct with respect to the \mathbf{V} -operational semantics. In the $\lambda\Gamma$ -calculus, there are terms, as $(\lambda z.D)(xI)D$, that are Γ -unsolvable of order 0 but Γ -normal form. So $(\lambda z.D)(xI)D \downarrow_{\mathbf{V}}$ and $DD \uparrow_{\mathbf{V}}$, while they are $\approx_{\mathbf{V}}$. This implies that Property 10.1.15 cannot be a guideline for building a correct model. But we can use the fact, proved in Property 7.1.10, that if M is potentially Γ -valuable and N is not potentially Γ -valuable then $N \prec_{\mathbf{V}} M$.

Property 12.1.1. Let \mathcal{F} be a filter model such that

$$P \text{ potentially } \Gamma\text{-valuable and } Q \text{ not potentially } \Gamma\text{-valuable} \\ \text{imply } Q \sqsubset_{\mathcal{F}} P, \text{ for all } P, Q \in \Lambda.$$

Then \mathcal{F} is correct with respect to $\preceq_{\mathbf{V}}$.

Proof. If $Q \sqsubset_{\mathcal{F}} P$ then $P \not\sqsubseteq_{\mathcal{F}} Q$, so the hypothesis implies the following statement:

" $P \sqsubseteq_{\mathcal{F}} Q$ and P is potentially Γ -valuable imply Q is potentially Γ -valuable."

Let $M \sqsubseteq_{\mathcal{F}} N$, so by Property 10.1.13 $C[M] \sqsubseteq_{\mathcal{F}} C[N]$, for each closing context $C[.]$; hence, by hypothesis, if $C[M]$ is potentially Γ -valuable then $C[N]$ is potentially Γ -valuable too. Since $C[M], C[N] \in \Lambda^0$, the potential Γ -valuability coincides with the Γ -valuability, and $C[M] \Downarrow_{\mathbf{V}}$ implies $C[N] \Downarrow_{\mathbf{V}}$; hence $M \preceq_{\mathbf{V}} N$. \square

So, we want a legal type system $\sqrt{\cdot}$, based on a set of constants $C_{\sqrt{\cdot}}$, such that there is at least one type σ and a basis B such that, if M is potentially Γ -valuable then $B \vdash_{\sqrt{\cdot}} M : \sigma$, otherwise $B \nvdash_{\sqrt{\cdot}} M : \sigma$. A term is potentially Γ -valuable if and only if it has $\Xi\ell$ -normal form, so it is natural to characterize terms without $\Xi\ell$ -normal form by assigning them only type ω . Moreover, the \mathbf{V} -operational semantics performs a lazy evaluation (so it cannot be $\omega \simeq_{\sqrt{\cdot}} \omega \rightarrow \omega$), and this allows us to use the type $\omega \rightarrow \omega$ to characterize $\Xi\ell$ -normal form.

Definition 12.1.2. $\sqrt{\cdot}$ is the type system $\langle C_{\sqrt{\cdot}}, \leq_{\sqrt{\cdot}}, I(C_{\sqrt{\cdot}}) \rangle$ where $C_{\sqrt{\cdot}} = \{\omega\}$,

$$I(C_{\sqrt{\cdot}}) = \{\sigma_0 \wedge \dots \wedge \sigma_n \mid \exists k \leq n \quad \exists \sigma, \tau \in T(C) \quad \sigma_k \equiv \sigma \rightarrow \tau\},$$

and $\leq_{\sqrt{\cdot}}$ is the intersection relation induced by the rules in Fig. 12.1.

$\frac{}{\sigma \leq_{\sqrt{\cdot}} \omega} \text{(a)}$	$\frac{}{\sigma \leq_{\sqrt{\cdot}} \sigma \wedge \sigma} \text{(b)}$	$\frac{}{\sigma \wedge \tau \leq_{\sqrt{\cdot}} \sigma} \text{(c)}$	$\frac{}{\sigma \wedge \tau \leq_{\sqrt{\cdot}} \tau} \text{(c')}$
$\frac{(\sigma \rightarrow \tau) \wedge (\sigma \rightarrow \pi) \leq_{\sqrt{\cdot}} \sigma \rightarrow (\tau \wedge \pi)}{\sigma \leq_{\sqrt{\cdot}} \sigma \wedge \pi} \text{(d)}$	$\frac{\sigma \leq_{\sqrt{\cdot}} \sigma', \tau \leq_{\sqrt{\cdot}} \tau'}{\sigma \wedge \tau \leq_{\sqrt{\cdot}} \sigma' \wedge \tau'} \text{(e)}$		
$\frac{\sigma' \leq_{\sqrt{\cdot}} \sigma, \tau \leq_{\sqrt{\cdot}} \tau'}{\sigma \rightarrow \tau \leq_{\sqrt{\cdot}} \sigma' \rightarrow \tau'} \text{(f)}$	$\frac{\sigma \rightarrow \omega \leq_{\sqrt{\cdot}} \omega \rightarrow \omega}{\sigma \leq_{\sqrt{\cdot}} \sigma} \text{(g)}$	$\frac{}{\sigma \leq_{\sqrt{\cdot}} \sigma} \text{(r)}$	$\frac{\sigma \leq_{\sqrt{\cdot}} \rho, \rho \leq_{\sqrt{\cdot}} \tau}{\sigma \leq_{\sqrt{\cdot}} \tau} \text{(t)}$
$\frac{(\omega \rightarrow \omega) \rightarrow \tau \leq_{\sqrt{\cdot}} \omega \rightarrow \tau}{\sigma \leq_{\sqrt{\cdot}} \tau} \text{(v)}$			

Fig. 12.1. $\sqrt{\cdot}$ -Intersection relation

Some properties of the $\leq_{\sqrt{\cdot}}$ -intersection relation are proved in Sect. 12.1.1. In particular notice that $(\omega \rightarrow \omega) \rightarrow \tau \simeq_{\sqrt{\cdot}} \omega \rightarrow \tau$, for all $\tau \in T(C_{\sqrt{\cdot}})$.

By the definition of $I(C_{\sqrt{\cdot}})$ and the rule $(\rightarrow E)$, a type of the shape $\omega \rightarrow \sigma$ does not have applicative power. So a typing as $B[\omega \rightarrow \sigma/x] \vdash_{\sqrt{\cdot}} xM : \sigma$

cannot be proved, for all M , while it is a correct typing in every other type assignment system we have seen before.

Let $\mathcal{I}(\sqrt{\cdot}) = \{f \in \mathcal{F}(\sqrt{\cdot}) \mid f \neq \uparrow \{\omega\}\}$, so the type system $\sqrt{\cdot}$ induces a $\lambda\Gamma$ -model, by Theorems 10.1.11 and 12.1.3.

Theorem 12.1.3. (i) *The type system $\sqrt{\cdot}$ is legal.*

(ii) *If $M \in \Gamma$ then $\llbracket M \rrbracket_\rho \in \mathcal{I}(C_\sqrt{\cdot})$, for all environments ρ .*

Proof. (i) The proof is in Sect. 12.1.1.

(ii) $\omega \rightarrow \omega \in \mathcal{I}(\sqrt{\cdot})$; thus it is sufficient to check that $B \vdash_\sqrt{\cdot} M : \omega \rightarrow \omega$, for all $M \in \Gamma$ and all bases B . Let $M \equiv \lambda x.P$ and $\sigma \in I(C_\sqrt{\cdot})$, so

$$\frac{\overline{B[\sigma/x] \vdash_\sqrt{\cdot} P : \omega}^{(\omega)} \quad B \vdash_\sqrt{\cdot} \lambda x.P : \sigma \rightarrow \omega \quad \sigma \rightarrow \omega \leq_\sqrt{\cdot} \omega \rightarrow \omega}{B \vdash_\sqrt{\cdot} \lambda x.P : \omega \rightarrow \omega}^{(\leq_\sqrt{\cdot})}$$

If $\sigma \in I(C_\sqrt{\cdot})$ then $\sigma \not\leq_\sqrt{\cdot} \omega$, hence $\sigma \leq_\sqrt{\cdot} \omega \rightarrow \omega$ by Property 12.1.28.(ii). Let $M \equiv x$, so

$$\frac{B \vdash_\sqrt{\cdot} x : B(x) \quad B(x) \leq_\sqrt{\cdot} \omega \rightarrow \omega}{B \vdash_\sqrt{\cdot} x : \omega \rightarrow \omega}^{(var)}$$

□

The next definition is well given by Corollary 12.1.27.

Definition 12.1.4. \mathcal{V} is the $\lambda\Gamma$ -model $< \mathcal{F}(\sqrt{\cdot}), \mathcal{I}(\sqrt{\cdot}), \circ_\sqrt{\cdot}, \llbracket \cdot \rrbracket^{\mathcal{F}(\sqrt{\cdot})} >$.

Since \mathcal{V} is a $\lambda\Gamma$ -model the interpretation of terms is closed under $=_T$; hence, by Corollary 10.1.13, typings are closed under $=_T$. We will prove that they are closed under the $\Xi\ell$ -reduction too. First of all, let us prove that a $\Xi\ell$ -nf can always be assigned the type $\omega \rightarrow \omega$.

Lemma 12.1.5. *If M is a $\Xi\ell$ -nf then $B \vdash_\sqrt{\cdot} M : \omega \rightarrow \omega$, for some basis B .*

Proof. By induction on M . If $M \in \Gamma$ then the proof is similar to that of Theorem 12.1.3.(ii). Let $M \equiv xM_1\dots M_m$ ($m \geq 1$) where M_i is a $\Xi\ell$ -nf, so by induction $B_i \vdash_\sqrt{\cdot} M_i : \omega \rightarrow \omega$ ($1 \leq i \leq m$).

Let B be a basis such that $B(y) = B_1(y) \wedge \dots \wedge B_m(y)$ for $y \not\equiv x$, while

$$B(x) = B_1(x) \wedge \dots \wedge B_m(x) \wedge \underbrace{(\omega \rightarrow \omega) \rightarrow \dots \rightarrow (\omega \rightarrow \omega)}_m \rightarrow \omega \rightarrow \omega.$$

$B \vdash_\sqrt{\cdot} M_i : \omega \rightarrow \omega$, by Lemma 10.1.7.(iii); so $B \vdash_\sqrt{\cdot} M : \omega \rightarrow \omega$, by rules (var) , $(\leq_\sqrt{\cdot})$ and m applications of rule $(\rightarrow E)$. □

Theorem 12.1.6. (i) $B \vdash_{\vee} M : \sigma$ and $M \rightarrow_{\Xi\ell} N$ imply $B \vdash_{\vee} N : \sigma$;
(ii) $B \vdash_{\vee} M : \sigma$ and $N \rightarrow_{\Xi\ell} M$ imply there is B' such that, for all x ,
 $B'(x) \leq_{\vee} B(x)$, and $B' \vdash_{\vee} N : \sigma$.

Proof. The proof is in Sect. 12.1.2. \square

In order to understand the difference between the two points of the previous theorem, consider the two terms I and $(\lambda z t. t)(xy)$. They are $=_{\Xi\ell}$ but they are not $\sim_{\mathcal{V}}$. Let B_ω be the basis such that, for all x , $B_\omega(x) = \omega$; thus

$$B_\omega \vdash_{\vee} I : ((\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega) \rightarrow (\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega$$

nevertheless, in order to assign the same type to the term $(\lambda z t. t)(xy)$ it is necessary to change the basis, choosing, for example, a basis B such that $B(x) = (\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega$ and $B(y) = \omega \rightarrow \omega$. So the following corollary holds for closed terms only.

Corollary 12.1.7. If $M, N \in \Lambda^0$ and $M =_{\Xi\ell} N$ then $M \sim_{\mathcal{V}} N$.

Proof. From Theorem 12.1.6, since the typings of a closed term are independent of the basis. \square

So the following property holds.

Property 12.1.8. If M is potentially Γ -valuable then $B \vdash_{\vee} M : \omega \rightarrow \omega$, for some basis B .

Proof. M potentially valuable means M has $\Xi\ell$ -nf. The proof follows from Lemma 12.1.5 and Theorem 12.1.6.(ii). \square

In order to prove that a term that is not potentially Γ -valuable cannot be assigned the type $\omega \rightarrow \omega$ from any basis, we need an approximation theorem. As usual, we extend the language by adding a constant Ω to the formation rules of terms; hence we define some new reduction rules on the so-obtained language.

Definition 12.1.9. Let $\Lambda\Omega$ be defined as in Definition 11.1.9.

(i) $\rightarrow_{\Lambda\Omega}$ is defined as the contextual closure of the following rules:

$$\Omega M \rightarrow \Omega, \quad M\Omega \rightarrow \Omega.$$

(ii) The set of approximated input values is $\Gamma_\Omega = \text{Var} \cup \{\lambda x. M \mid M \in \Lambda\Omega\}$.
The $\Gamma\Omega$ -reduction ($\rightarrow_{\Gamma\Omega}$) is the contextual closure of the following rules:

$$\begin{aligned} \Omega M &\rightarrow \Omega, & M\Omega &\rightarrow \Omega, \\ (\lambda x. M)N &\rightarrow M[N/x] & \text{if } N \in \Gamma_\Omega. \end{aligned}$$

$\rightarrow_{\Gamma\Omega}^*$ is the symmetric and transitive closure of $\rightarrow_{\Gamma\Omega}$. The η -reduction can be directly applied to the language $\Lambda\Omega$ (see Definition 1.3.7 pag. 23). $M \in \Lambda\Omega$ is in $\Gamma\Omega$ -normal form ($\Gamma\Omega$ -nf) if and only if it does not contain $\Gamma\Omega$ -redexes.

- (iii) Let $\Xi_\Omega = \Gamma_\Omega \cup \{xM_1\dots M_m \mid M_i \in \Xi_\Omega \ (i \leq m)\}$. The $\Xi\ell\Omega$ -reduction ($\rightarrow_{\Xi\ell\Omega}$) is the applicative closure (see Definition 3.1.3 pag. 36) of the following rules:

$$\begin{aligned} \Omega M &\rightarrow \Omega, & M\Omega &\rightarrow \Omega, \\ (\lambda x.M)N &\rightarrow M[N/x] \quad \text{if } N \in \Xi_\Omega \end{aligned}$$

$\rightarrow_{\Xi\ell\Omega}^*$ is the symmetric and transitive closure of $\rightarrow_{\Xi\ell\Omega}$. $M \in \Lambda\Omega$ is in $\Xi\ell\Omega$ -normal form ($\Xi\ell\Omega$ -nf) if and only if it does not contain $\Xi\ell\Omega$ -redexes. Note that M is a $\Xi\ell\Omega$ -normal form if and only if $M \in \Xi_\Omega \cup \{\Omega\}$.

The first reduction rule defined in point (i) of Definition 12.1.9, namely $\Omega M \rightarrow \Omega$, reflects the lazy behaviour of the Γ -calculus, while $M\Omega \rightarrow \Omega$ reflects its call-by-value behaviour. Note that the $\Xi\ell\Omega$ -reduction is lazy, since it is closed under application, but not under abstraction. As usual, terms of $\Lambda\Omega$ will be considered modulo $=_\alpha$.

Example 12.1.10. The term $\lambda x.x(II)\Omega$ is a $\Xi\ell\Omega$ -normal form, but it is not a $\Gamma\Omega$ -normal form. Indeed, its subterm II is not a $\Xi\ell\Omega$ -redex (since it occurs under the scope of a λ -abstraction), but it is a $\Gamma\Omega$ -redex.

$\lambda x.x(II)\Omega \rightarrow_{\Gamma\Omega} \lambda x.\Omega$, which is a $\Gamma\Omega$ -normal form.

The type assignment system of Definition 10.1.1 can be applied to $\Lambda\Omega$ without modifications. It is easy to see that to the term Ω only the type ω can be assigned, by using rule (ω) . The intuitive interpretation of the constant Ω is that it represents a term with an unknown behaviour. The interpretation function is naturally extended to $\Lambda\Omega$, i.e. the interpretation of a term of $\Lambda\Omega$ is the set of types that can be assigned to it.

Definition 12.1.11. The set of $\sqrt{\Omega}$ -approximants of a term M is defined as follows:

$$\mathcal{A}^\sqrt{\Omega}(M) = \left\{ A \mid \begin{array}{l} \exists M' \text{ such that } M =_\Gamma M' \text{ and } A \text{ is a } \Gamma\Omega\text{-nf} \\ \text{obtained from } M' \text{ by replacing some subterms by } \Omega. \end{array} \right\}$$

The set of upper approximants of a term M is defined as follows:

$$\mathcal{U}^\sqrt{\Omega}(M) = \{U \mid \exists A \in \mathcal{A}^\sqrt{\Omega}(M) \text{ such that } A \rightarrow_{\Xi\ell\Omega}^* U \in \Xi\ell\Omega\text{-NF.}\}$$

Example 12.1.12. Some sets of approximants are shown.

- $\mathcal{A}^\sqrt{\Omega}((\lambda y.I)(DD)) = \mathcal{U}^\sqrt{\Omega}((\lambda y.I)(DD)) = \{\Omega\}$.
- $\mathcal{A}^\sqrt{\Omega}((\lambda zt.t)(xy)) = \{\Omega, (\lambda z.\Omega)(xy), (\lambda zt.\Omega)(xy), (\lambda zt.t)(xy)\}$,
 $\mathcal{U}^\sqrt{\Omega}((\lambda zt.t)(xy)) = \{\Omega, \lambda t.\Omega, \lambda t.t\}$ and note that $(\lambda zt.t)\Omega \notin \mathcal{U}^\sqrt{\Omega}((\lambda zt.t)(xy))$.
- $\mathcal{A}^\sqrt{\Omega}((\lambda zx.xD)(yI)D) \supseteq \{\Omega, (\lambda z.\Omega)(yI)(\lambda x.\Omega), (\lambda zx.xD)(yI)D\}$,
 $\mathcal{U}((\lambda zx.xD)(yI)D) = \{\Omega\}$.
- $\mathcal{A}^\sqrt{\Omega}(\lambda z.II) = \mathcal{U}^\sqrt{\Omega}(\lambda z.II) = \{\Omega, \lambda z.\Omega, \lambda zx.\Omega, \lambda zx.x\}$.

- $\mathcal{A}^\vee((\lambda xyz.yI)(uv)I) \supseteq \{(\lambda xyz.yI)(uv)I\}$.
- $\mathcal{U}((\lambda xyz.yI)(uv)I) \supseteq \{\lambda z.II\}$.

Approximants can be inductively defined.

Definition 12.1.13. *The set \mathcal{A}^\vee of approximants can be inductively defined as follows:*

- $\Omega \in \mathcal{A}^\vee$;
- If $A_j \in \mathcal{A}^\vee$ and $A_j \not\equiv \Omega$ then $\lambda x_1 \dots x_n.x A_1 \dots A_m \in \mathcal{A}^\vee$ ($1 \leq j \leq m$ and $n, m \in \mathbb{N}$).
- If $A, A_j \in \mathcal{A}^\vee$, $A_1 \notin \Gamma_\Omega$ and $A_j \not\equiv \Omega$ then $\lambda x_1 \dots x_n.(\lambda x.A) A_1 \dots A_m \in \mathcal{A}^\vee$ ($1 \leq j \leq m$, $m \geq 1$ and $n \in \mathbb{N}$).

Approximants will be ranged over by A, A' , possibly indexed. Upper approximants will be ranged over by U, U' , possibly indexed.

Property 12.1.14. $A \in \mathcal{A}^\vee$ if and only if there is M such that $A \in \mathcal{A}^\vee(M)$.

Proof. Easy. \square

An approximation theorem holds, relating the interpretation of a term to the interpretations both of its approximants and of its upper approximants.

Theorem 12.1.15 (V-Approximation).

- (i) $B \vdash_\vee M : \sigma$ if and only if $B \vdash_\vee A : \sigma$, for some $A \in \mathcal{A}^\vee(M)$.
- (ii) $B \vdash_\vee M : \sigma$ if and only if $B' \vdash_\vee U : \sigma$ where $U \in \mathcal{U}^\vee(M)$, for some basis B, B' such that $B(x) \leq_\vee B'(x)$, for all x .

Proof. Both proofs are in Sect. 12.1.3. \square

The next property implies $\mathcal{A}^\vee(M) = \{\Omega\}$ if and only if M is not Γ -valuable, for each closed term M . It also implies $\mathcal{U}^\vee(M) = \{\Omega\}$ if and only if M is not potentially Γ -valuable, for each term M (not necessarily closed).

Property 12.1.16. (i) M is Γ -valuable if and only if there is $A \in \mathcal{A}^\vee(M)$ such that $A \in \Gamma_\Omega$.

(ii) M is potentially Γ -valuable if and only if there is $U \in \mathcal{U}^\vee(M)$ such that $U \not\equiv \Omega$.

Proof. (i) (\Leftarrow) If $x \in \mathcal{A}^\vee(M)$ then $M =_\Gamma x$, by definition of $\mathcal{A}^\vee(M)$. If $\lambda x.A \in \mathcal{A}^\vee(M)$ then there is $M' \in \Lambda$ such that $M =_\Gamma \lambda x.M'$ and $A \in \mathcal{A}^\vee(M')$, by definition of $\mathcal{A}^\vee(M)$. (\Rightarrow) Trivial, by definition of Γ -valuable terms and by definition of $\mathcal{A}^\vee(M)$.

(ii) (\Leftarrow) If there is $U \in \mathcal{U}^{\checkmark}(M)$ such that $U \not\equiv \Omega$ then $\exists A \in \mathcal{A}^{\checkmark}(M)$ such that $A \xrightarrow{*_{\Xi\ell\Omega}} U \in \Xi\ell\Omega\text{-NF}$. As done in the proof of Lemma 3.1.35, it is possible to show that there is $r \in \mathbb{N}$ and $\bar{A}^r \in \Gamma_{\Omega}$ such that both $A[O^r/x_1, \dots, O^r/x_n] \xrightarrow{*_{\Gamma}} \bar{A}^r$ and $U[O^r/x_1, \dots, O^r/x_n] \xrightarrow{*_{\Gamma}} \bar{A}^r$, where $O^r \equiv \lambda x_1 \dots x_{r+1}.x_{r+1}$. Since $\bar{A}^r \in \mathcal{A}^{\checkmark}(M[O^r/x_1, \dots, O^r/x_n])$, by the point (i) of this Lemma the proof is done.

(\Rightarrow) By Property 12.1.8 and by point (ii) of the \mathcal{V} -approximation theorem. \square

Note that point (i) of the \mathcal{V} -approximation theorem is not sufficient in order to give a complete characterization of the not potentially Γ -valuable terms, through the syntactical shape of the approximants. In fact, $(\lambda zx.xD)(yI)D$ is not potentially Γ -valuable, but it is an approximant, as shown in Example 12.1.12.

Moreover, point (ii) of the \mathcal{V} -approximation theorem cannot be used to characterize the equivalence in the model, if we want to take into account also open terms (see Corollary 12.1.7).

Lemma 12.1.17. *Let M be not potentially Γ -valuable.*

If $B \vdash_{\checkmark} M : \sigma$ then $\sigma \simeq_{\checkmark} \omega$.

Proof. M not potentially Γ -valuable implies $\mathcal{U}^{\checkmark}(M) = \{\Omega\}$, by Property 12.1.16.(ii). Hence, the result follows from point (ii) of the \mathcal{V} -approximation theorem. \square

Now we are able to state the correctness of the model.

Theorem 12.1.18 (V-Correctness).

*The model \mathcal{V} is correct with respect to the **V**-operational semantics.*

Proof. By Property 12.1.1, it is sufficient to check that M is potentially Γ -valuable and N is not potentially Γ -valuable imply $N \not\leq_{\mathcal{V}} M$. The proof follows by Property 12.1.8 and Lemma 12.1.17. \square

The model gives also a (partial) characterization of the Γ -solvable terms.

Theorem 12.1.19. (i) *Let M be either a Γ -solvable term or a Γ -unsolvable term of infinite Γ -order. Then for all $p \geq 0$ there is a basis B and types $\sigma_1, \dots, \sigma_p$ such that $B \vdash_{\checkmark} M : \sigma_1 \rightarrow \sigma_2 \rightarrow \dots \rightarrow \sigma_p \rightarrow \omega$.*
(ii) *Let M be a Γ -unsolvable term of Γ -order p . If $B \vdash_{\checkmark} M : \sigma$ then for some $k \geq 1$, $\sigma \equiv \sigma_1 \wedge \dots \wedge \sigma_k$, where $\sigma_i \equiv \tau_1 \rightarrow \tau_2 \rightarrow \dots \rightarrow \tau_n \rightarrow \omega$, and $n \leq p$.*

Proof. (i) By Property 3.1.15 (pag. 39), M is Γ -solvable if and only if there are terms M_1, \dots, M_n ($n \in \mathbb{N}$) such that $M \xrightarrow{*_{\Xi\ell}} \lambda x_1.M_1$, $M_i \xrightarrow{*_{\Xi\ell}} \lambda x_{i+1}.M_{i+1}$ ($1 \leq i \leq n$) and $M_n \equiv xP_1 \dots P_m$ where $P_i \in \Xi$ for some $m \in \mathbb{N}$. Let M be Γ -solvable. The proof will be given by induction on n . If $n = 0$, then choose a basis B such that

$$B(x) = \underbrace{(\omega \rightarrow \omega) \rightarrow \dots \rightarrow (\omega \rightarrow \omega)}_m \rightarrow \sigma_1 \rightarrow \sigma_2 \rightarrow \dots \rightarrow \sigma_p \rightarrow \omega.$$

Then $B \vdash_{\vee} xP_1 \dots P_m : \sigma_1 \rightarrow \sigma_2 \rightarrow \dots \rightarrow \sigma_p \rightarrow \omega$, by rule $(\rightarrow E)$ and Property 12.1.8. The inductive step is easy, using the basic case, Theorem 12.1.6 and rule $(\rightarrow I)$.

Let M be Γ -unsolvable of Γ -order ∞ . For every $i \geq 1$, $M \rightarrow_{\Xi\ell}^* \lambda x_1.M_1, M_i \rightarrow_{\Xi\ell}^* \lambda x_{i+1}.M_{i+1}$. So choose $i = p$, and obviously there is a typing $B \vdash_{\vee} \lambda x_p.M_p : \omega \rightarrow \omega$, by Property 12.1.8. Then the proof is similar to the previous case.

- (ii) By induction on p . If $p = 0$, then M is not potentially Γ -valuable, and the result follows from Lemma 12.1.17. The case $p \geq 0$ follows easily from the definition of Γ -solvability and of Γ -order. \square

As the previous theorem shows, the \mathcal{V} -model does not permit us to discriminate between Γ -solvable terms and Γ -unsolvable terms of infinite order. An intersection type assignment system giving a complete characterization of the Γ -solvable terms is shown in [73].

Some properties of the \mathbf{V} -operational semantics can be proved by the approximation theorem.

- Property 12.1.20.* (i) The theory $\approx_{\mathbf{V}}$ is operationally extensional.
(ii) All call-by-value fixed-point operators are equated in \mathcal{V} .
(iii) All call-by-value recursion operators are equated in \mathcal{V} .
(iv) Let Z be a call-by-value recursion operator. Then every term M is such that $M \preceq_{\mathbf{V}} ZK$.

Proof. (i) It is sufficient to prove that $I \sim_{\mathbf{V}} E$, then the result follows by Property 3.1.18 and Lemma 8.1.9. The proof of $I \sim_{\mathbf{V}} E$ can be easily constructed in a way similar to the proof of Lemma 11.1.8, taking into account the differences between the two type assignment systems, namely Property 12.1.28 must be used in place of Property 11.1.36. The proof follows by correctness.

- (ii) Let Z be a fixed-point operator, i.e. $ZM =_{\Gamma} M(ZM)$, for all Γ -valuable terms M . Then $\mathcal{A}^{\vee}(Z) = \{\Omega, \lambda x.\Omega\}$.
- (iii) Let Z be a call-by-value recursion operator, i.e. $ZM =_{\Gamma} M(\lambda z.ZMz)$, for all Γ -valuable terms M .
Then $\mathcal{A}^{\vee}(Z) = \{\Omega, \lambda z.\Omega, \lambda x.x(\lambda z_0.(\dots(\lambda z_n.\Omega)\dots)z_0) \mid n \geq 1\}$.
- (iv) $ZK =_{\Gamma} K(\lambda z.ZKz) =_{\Gamma} \lambda yz.ZKz$. So $\mathcal{A}^{\vee}(ZK) = \{\lambda x_1 \dots x_n.\Omega \mid n \in \mathbb{N}\}$.
As a result, it is easy to check that $B \vdash_{\vee} ZK : \sigma$, for all $\sigma \in T(C_{\vee})$ and for all bases B . \square

As we anticipated at the end of Sect. 3.1, the notion of fixed-point in the call-by-value setting is in some sense meaningless. In fact, it follows from the proof of Property 12.1.20.(ii), that a fixed point operator Z is such that, for every Γ -valuable term M , ZM is Γ -unsolvable of order 0.

The model \mathcal{V} induces a not semisensible Γ -theory (this notion was defined in Definition 1.3.4). In order to prove this result, we need to show that there is an infinite class of approximants which behaves, with respect to typing, as free variables.

Lemma 12.1.21. *Let $A_0 \equiv \lambda z. \Omega$ and $A_{n+1} \equiv \lambda z. (\lambda x. A_n)(xz)$.*

- (i) *For all $\sigma \not\leq_{\vee} \omega$ there is n such that $B[\sigma/x] \vdash_{\vee} A_n : \sigma$.*
- (ii) *$B[\sigma/x] \vdash_{\vee} A_n : \tau$ implies $\sigma \leq_{\vee} \tau$.*

Proof. (i) $\sigma \simeq_{\vee} \sigma_0 \wedge \dots \wedge \sigma_n$ ($n \in \mathbb{N}$), where $\sigma_i \simeq_{\vee} \tau_1^i \rightarrow \dots \rightarrow \tau_{m_i}^i \rightarrow \omega \rightarrow \omega$ ($m_i \in \mathbb{N}, i \leq n$) by Property 12.1.28.(iii). Moreover, without loss of generality, we can assume that $\tau_r^i \not\leq_{\vee} \omega$ for all r , by Property 12.1.28.(i).

We will show that if $p_i \geq m_i$ then $B[\sigma_i/x] \vdash_{\vee} A_{p_i} : \sigma_i$ ($i \leq n$), by induction on m_i . Since $r \geq \max\{p_1, \dots, p_n\}$ implies $B[\sigma_i/x] \vdash_{\vee} A_r : \sigma_i$ ($i \leq n$), the proof follows by rule $(\wedge I)$ and Lemma 10.1.7.(iii).

If $m_i = 0$ then the proof is trivial, by Lemma 12.1.5. Otherwise

$B[\tau_2 \rightarrow \dots \rightarrow \tau_{m_i} \rightarrow \omega \rightarrow \omega/x] \vdash_{\vee} A_p : \tau_2 \rightarrow \dots \rightarrow \tau_{m_i} \rightarrow \omega \rightarrow \omega$ where $p \geq m_i - 1$, by induction. By Lemma 10.1.7.(i), it follows that $B[\tau_2 \rightarrow \dots \rightarrow \tau_{m_i} \rightarrow \omega \rightarrow \omega/x, \tau_1/z] \vdash_{\vee} A_p : \tau_2 \rightarrow \dots \rightarrow \tau_{m_i} \rightarrow \omega \rightarrow \omega$. So $B[\tau_1/z] \vdash_{\vee} \lambda x. A_p : (\tau_2 \rightarrow \dots \rightarrow \tau_{m_i} \rightarrow \omega \rightarrow \omega) \rightarrow \tau_2 \rightarrow \dots \rightarrow \tau_{m_i} \rightarrow \omega \rightarrow \omega$ by rule $(\rightarrow I)$. Thus, always by Lemma 10.1.7.(i),

$$\begin{aligned} B[\tau_1/z, \tau_1 \rightarrow \dots \rightarrow \tau_{m_i} \rightarrow \omega \rightarrow \omega/x] &\vdash_{\vee} \lambda x. A_p : \\ &(\tau_2 \rightarrow \dots \rightarrow \tau_{m_i} \rightarrow \omega \rightarrow \omega) \rightarrow \tau_2 \rightarrow \dots \rightarrow \tau_{m_i} \rightarrow \omega \rightarrow \omega \end{aligned}$$

and by rule $(\rightarrow E)$

$$B[\tau_1/z, \tau_1 \rightarrow \dots \rightarrow \tau_{m_i} \rightarrow \omega \rightarrow \omega/x] \vdash_{\vee} (\lambda x. A_p)(xz) : \tau_2 \rightarrow \dots \rightarrow \tau_{m_i} \rightarrow \omega \rightarrow \omega.$$

Finally, $B[\tau_1 \rightarrow \dots \rightarrow \tau_{m_i} \rightarrow \omega \rightarrow \omega/x] \vdash_{\vee} \lambda z. (\lambda x. A_p)(xz) : \sigma$.

- (ii) By induction on n . If $n = 0$ then the proof is obvious, since $\sigma \in I(C_{\vee})$. Let $n \geq 1$. If $\tau \simeq_{\vee} \omega \rightarrow \omega$ $\tau \simeq_{\vee} \omega \rightarrow \omega$ then the proof is obvious. Otherwise, $B[\sigma/x] \vdash_{\vee} A_n : \tau$ implies, by Lemma 10.1.7.(iv), $\tau \geq_{\vee} \tau_1 \wedge \dots \wedge \tau_r$ ($r \geq 1$) where $\tau_i \equiv \mu_i \rightarrow \nu_i$, $B[\sigma/x, \mu_i/z] \vdash_{\vee} (\lambda x. A_{n-1})(xz) : \nu_i$ and $\mu_i \not\leq_{\vee} \omega$ ($1 \leq i \leq r$). Since $\tau \simeq_{\vee} \omega \rightarrow \omega$ has been already considered, we can assume $\nu_i \not\leq_{\vee} \omega$ without loss of generality. Therefore $B[\sigma/x, \mu_i/z] \vdash_{\vee} (\lambda x. A_{n-1}) : \pi_i \rightarrow \nu_i$ and $B[\sigma/x, \mu_i/z] \vdash_{\vee} xz : \pi_i$ for some $\pi_i \not\leq_{\vee} \omega$, by Lemma 10.1.7.(vii). So, by Lemma 10.1.7.(iv), $B[\sigma/x, \mu_i/z, \pi_i/y] \vdash_{\vee} A_{n-1}[y/x] : \nu_i$, and by induction $\pi_i \leq_{\vee} \nu_i$. Moreover, by Lemma 10.1.7.(vii) and (ii), $B[\sigma/x, \mu_i/z] \vdash_{\vee} xz : \pi_i$ implies $\sigma \leq_{\vee} \mu_i \rightarrow \pi_i$. Thus $\sigma \leq \tau_i$, and the proof follows. \square

Now we are ready to prove the next theorem.

Theorem 12.1.22. *Let Z be a call-by-value recursion operator.*

If $B \equiv \lambda xyz. x(yz)$ then $I \sim_{\mathcal{V}} ZB$.

Proof. It is easy, but boring, to check that $\mathcal{A}^{\vee}(ZB) = \{\Omega, \lambda x.\Omega, \lambda xy.\Omega\} \cup \{\lambda x.A_n \mid n \geq 1\}$, where $A_n \equiv \lambda z.(\lambda x.A_{n-1})(xz)$. Then the proof follows from Lemma 12.1.21, since $B \vdash I : \sigma$ if and only if either $\sigma \simeq_{\vee} \omega$ or $\sigma \simeq_{\vee} \omega \rightarrow \omega$ or $\sigma \simeq_{\vee} \sigma_1 \wedge \sigma_2$ or $\sigma \simeq_{\vee} \mu \rightarrow \nu$, where $\mu \leq_{\vee} \nu$. \square

Corollary 12.1.23. *The \mathbf{V} -operational semantics is not semisensible.*

Proof. Since ZB is a Γ -unsolvable term of infinite order, the Γ -theory \mathcal{V} is not semisensible, by Theorem 12.1.22. Then, by correctness, \mathbf{V} is also not semisensible. \square

The fact that \mathbf{V} is not semisensible was first proved in [73], using syntactical tools. The model \mathcal{V} is not fully abstract with respect to \mathbf{V} -operational semantics. In fact, the next theorem shows its incompleteness.

Theorem 12.1.24 (\mathcal{V} -Incompleteness).

The model \mathcal{V} is incomplete with respect to the \mathbf{V} -operational semantics.

Proof. The proof is in Sect. 12.1.4 \square

The next theorem proves that there is not a filter $\lambda\Gamma$ -model that is fully abstract with respect to the \mathbf{V} -operational semantics. The proof is similar to the corresponding one for the \mathbf{L} -operational semantics, given in Sect. 11.3.2. Namely, we will show that there are two terms that are $\approx_{\mathbf{V}}$, but they cannot be equated in every filter model correct with respect to the \mathbf{V} -operational semantics.

Theorem 12.1.25. *There is not a filter $\lambda\Gamma$ -model that is fully abstract with respect to the \mathbf{V} -operational semantics.*

Proof. The proof is in Sect. 12.1.4. \square

12.1.1 The \leq_{\vee} -Intersection Relation

In order to prove that the \leq_{\vee} -intersection relation is well posed, i.e. it does not identify all types, we need to characterize the types $\simeq_{\vee} \omega$.

Theorem 12.1.26. $\sigma \simeq_{\vee} \omega$ if and only if $\sigma \notin I(C_{\vee})$, for all $\sigma \in T(C_{\vee})$.

Proof. Note that $\omega \simeq_{\vee} \sigma$ if and only if $\omega \leq_{\vee} \sigma$, by the rule (a).

(\Leftarrow) By induction on σ . The case $\sigma \equiv \omega$ is obvious. The case $\sigma \equiv \tau \rightarrow \pi$ is not possible. The case of intersection follows by induction.

(\Rightarrow) Let us first prove that if $\sigma \notin I(C_{\vee})$ and $\sigma \leq_{\vee} \tau$ then $\tau \notin I(C_{\vee})$, by induction on the rule of \leq_{\vee} .

(a),(b),(c),(c'),(r) Trivial.

(d),(f),(g),(v) Not possible.

(e),(t) By induction.

Since $\omega \notin I(C_{\vee})$, if $\sigma \simeq_{\vee} \omega$ then $\omega \leq_{\vee} \sigma$, so the proof is done. \square

Corollary 12.1.27. Let $\Omega^{\mathcal{V}} = \{\sigma_0 \wedge \dots \wedge \sigma_n \ (n \in \mathbb{N}) \mid \forall i \leq n \ \sigma_i \equiv \omega\}$.
 $\Omega^{\mathcal{V}} = T(C_{\vee}) - I(C_{\vee})$.

The previous theorem implies that $I(C_{\vee})$ is a well-defined set of input types, since it satisfies the conditions of Definition 10.1.1.(iv). Moreover, not all types in $T(C_{\vee})$ are equated by \simeq_{\vee} ; in fact, $\omega \rightarrow \omega \not\simeq_{\vee} \omega$.

- Property 12.1.28.* (i) $\omega \rightarrow \sigma \simeq_{\vee} (\omega \rightarrow \omega) \rightarrow \sigma$.
(ii) $\sigma \rightarrow \tau \leq_{\vee} \omega \rightarrow \omega$, for all $\sigma, \tau \in T(C_{\vee})$.
(iii) If $\sigma \not\simeq_{\vee} \omega$ then $\sigma \simeq_{\vee} \sigma_0 \wedge \dots \wedge \sigma_n \ (n \in \mathbb{N})$ where

$$\sigma_i \simeq_{\vee} \tau_1^i \rightarrow \dots \rightarrow \tau_{m_i}^i \rightarrow \omega \rightarrow \omega \quad (m_i \in \mathbb{N}, i \leq n).$$

Proof. (i) By rule (a), (f) and (v).

(ii) Clearly $\sigma \rightarrow \tau \not\simeq_{\vee} \omega$ and $\sigma \rightarrow \tau \leq_{\vee} \sigma \rightarrow \omega \leq_{\vee} \omega \rightarrow \omega$ by rules (a), (f) and (g).

(iii) By induction on σ . The case $\sigma \equiv \omega$ is against the hypothesis. If $\sigma \equiv \pi \wedge \tau$ and $\pi, \tau \not\simeq_{\vee} \omega$ then the proof follows by induction. If $\sigma \equiv \pi \wedge \tau$ and $\pi \not\simeq_{\vee} \omega$ but $\tau \simeq_{\vee} \omega$ the proof follows by induction on π , since $\sigma \simeq_{\vee} \pi$. If $\sigma \equiv \pi \wedge \tau$ and $\tau \not\simeq_{\vee} \omega$ but $\pi \simeq_{\vee} \omega$ the proof follows by induction on τ , since $\sigma \simeq_{\vee} \tau$.

Let $\sigma \equiv \tau \rightarrow \pi$. If $\pi \simeq_{\vee} \omega$ then $\sigma \simeq_{\vee} \omega \rightarrow \omega$ by rules (g), (f) and (a); so, let $\pi \not\simeq_{\vee} \omega$. By induction $\pi \simeq_{\vee} \pi_0 \wedge \dots \wedge \pi_k \ (k \in \mathbb{N})$, where

$$\pi_i \simeq_{\vee} \pi_1^i \rightarrow \dots \rightarrow \pi_{k_i}^i \rightarrow \omega \rightarrow \omega \quad (k_i \in \mathbb{N}, i \leq k).$$

Hence, $\sigma \simeq_{\vee} (\tau \rightarrow \pi_0) \wedge \dots \wedge (\tau \rightarrow \pi_k)$ by Lemma 10.1.4.(iii). \square

It is easy to check that a type in $T(C_{\vee})$ has the following shape:

$$(\sigma_1 \rightarrow \tau_1) \wedge \dots \wedge (\sigma_n \rightarrow \tau_n) \wedge \underbrace{\omega \wedge \dots \wedge \omega}_m$$

where $n, m \geq 0, m + n \geq 1$. The following lemma implies the legality of \vee .

Property 12.1.29. Let $n, m, p, q \in \mathbb{N}$ be such that $n, p \geq 1$ and

$$(\sigma_1 \rightarrow \tau_1) \wedge \dots \wedge (\sigma_n \rightarrow \tau_n) \wedge \underbrace{\omega \wedge \dots \wedge \omega}_m \leq_{\vee} (\sigma'_1 \rightarrow \tau'_1) \wedge \dots \wedge (\sigma'_p \rightarrow \tau'_p) \wedge \underbrace{\omega \wedge \dots \wedge \omega}_q.$$

Let $h \leq p$; if $\tau'_h \not\simeq_{\vee} \omega$ then $\exists \{i_1, \dots, i_k\} \subseteq \{1, \dots, n\}$, for some $k \in \mathbb{N}$, such that $\sigma_{i_1} \wedge \dots \wedge \sigma_{i_k} \geq_{\vee} \sigma'_h \wedge (\omega \rightarrow \omega)$ and $\tau_{i_1} \wedge \dots \wedge \tau_{i_k} \leq_{\vee} \tau'_h$, where $\tau_{i_j} \not\simeq_{\vee} \omega$ ($i_j \in \{i_1, \dots, i_k\}$).

Proof. By induction on the definition of \leq_{\vee} .

(a), (g), (b), (c), (c'), (e), (f), (r) Easy.

- (d) Let $(\sigma \rightarrow \tau) \wedge (\sigma \rightarrow \pi) \leq_{\vee} \sigma \rightarrow (\tau \wedge \pi)$. The case $\tau, \pi \not\simeq_{\vee} \omega$ is easy. If $\tau \simeq_{\vee} \omega$ and $\pi \not\simeq_{\vee} \omega$ then it is easy to see that $\tau \wedge \pi \simeq_{\vee} \pi$, so the proof is immediate. The remaining case is similar to the previous one.
- (v) It is sufficient to check that $\omega \rightarrow \omega \geq_{\vee} \omega \wedge (\omega \rightarrow \omega)$; the proof follows by rule (c').
- (t) Let ρ be such that $(\sigma_1 \rightarrow \tau_1) \wedge \dots \wedge (\sigma_n \rightarrow \tau_n) \wedge \underbrace{\omega \wedge \dots \wedge \omega}_m \leq_{\vee} \rho$ and $\rho \leq_{\vee} (\sigma'_1 \rightarrow \tau'_1) \wedge \dots \wedge (\sigma'_p \rightarrow \tau'_p) \wedge \underbrace{\omega \wedge \dots \wedge \omega}_q$. If $\rho \simeq_{\vee} \omega$ then $\omega \leq_{\vee} (\sigma'_1 \rightarrow \tau'_1) \wedge \dots \wedge (\sigma'_p \rightarrow \tau'_p) \wedge \underbrace{\omega \wedge \dots \wedge \omega}_q \leq_{\vee} \sigma'_j \rightarrow \tau'_j$ implies $\omega \simeq_{\vee} \sigma'_j \rightarrow \tau'_j$ ($j \leq p$) and so $p = 0$, by Theorem 12.1.26. Let $\rho \equiv (\mu_1 \rightarrow \nu_1) \wedge \dots \wedge (\mu_r \rightarrow \nu_r) \wedge \underbrace{\omega \wedge \dots \wedge \omega}_s$, for some $r, s \in \mathbb{N}$ such that $r \geq 1$. If $\tau'_h \not\simeq \omega$ ($h \leq p$) then, by inductive hypothesis, $\exists \{i_1, \dots, i_h\} \subseteq \{1, \dots, r\}$ such that $\mu_{i_1} \wedge \dots \wedge \mu_{i_h} \geq_{\vee} \sigma'_h \wedge (\omega \rightarrow \omega)$ and $\nu_{i_1} \wedge \dots \wedge \nu_{i_h} \leq_{\vee} \tau'_h$. Since $\nu_{i_j} \not\simeq_{\vee} \omega$ ($i_j \in \{i_1, \dots, i_h\}$), the proof follows by applying the inductive hypothesis to each arrow $\mu_{i_j} \rightarrow \nu_{i_j}$. \square

▼ Proof of Theorem 12.1.3.(i) (pag. 183).

The legality of the type system \checkmark is a particular case of Property 12.1.29. ■

12.1.2 Proof of Theorem 12.1.6

In order to show that the type assignment system \vdash_{\vee} is closed under $\Xi\ell$ -reduction, first we prove that the type assignment system \vdash_{\vee} is closed under Λ -reduction.

Lemma 12.1.30 (Λ -Subject reduction).

If $M \rightarrow_{\Lambda} N$ and $B \vdash_{\vee} M : \sigma$ then $B \vdash_{\vee} N : \sigma$.

Proof. Let $M \equiv (\lambda x.P)Q$ and $N \equiv P[Q/x]$. If $\sigma \simeq_{\vee} \omega$ then the proof is trivial, so let $\sigma \not\simeq_{\vee} \omega$. $B \vdash_{\vee} (\lambda x.P)Q : \sigma$ implies, by Lemma 10.1.7.(vii), both $B \vdash_{\vee} (\lambda x.P) : \tau \rightarrow \sigma$ and $B \vdash_{\vee} Q : \tau$, for some $\tau \in I(C_{\vee})$. By Lemma 10.1.7.(vi), $B \vdash_{\vee} \lambda x.P : \tau \rightarrow \sigma$ if and only if $B[\tau/x] \vdash_{\vee} P : \sigma$.

Without loss of generality, we can assume that there is a derivation d proving $B[\tau/x] \vdash_{\vee} P : \sigma$ such that all typings occurring in it have the same basis $B[\tau/x]$. Indeed, the only rule that can change the basis is $(\rightarrow I)$, and we can assume that free and bound variables have different names in P . Derivation d can be transformed into a derivation d' proving $B \vdash_{\vee} P[Q/x] : \sigma$, by performing the following operations:

1. replace each subderivation of d of the shape

$$\frac{}{B[\tau/x] \vdash_{\vee} x : \tau}^{(var)}$$

- by a copy of a derivation proving $B \vdash_{\vee} Q : \tau$;
 2. replace each typing $B[\tau/x] \vdash_{\vee} P^* : \mu$ in d by $B \vdash_{\vee} P^*[Q/x] : \mu$.

By induction on the derivation d , it is easy to check that d' is well defined. Let $M \equiv C[(\lambda x.P)Q]$ and $N \equiv C[P[Q/x]]$. If an occurrence of $(\lambda x.P)Q$ in M is inside a subterm of M typed by the rule (ω) , then just replace Q to each free occurrence of x in term being subject of typings in d . Otherwise, replace each subderivation d proving a typing for $(\lambda x.P)Q$ by a subderivation d' built as described before. \square

Obviously, \vdash_{\vee} cannot be closed under Λ -expansion, i.e. $B \vdash_{\vee} P[Q/x]$ cannot imply $B \vdash_{\vee} (\lambda x.P)Q$, since in this case the model would be incorrect. But a restricted form of Λ -expansion can be proved. First, we will prove a property.

Property 12.1.31. Let d be a derivation proving $B \vdash_{\vee} M : \sigma$ where $\sigma \not\leq_{\vee} \omega$. If N is a subterm of M not occurring under the scope of a λ -abstraction, then in d there is a subderivation d' proving $B \vdash_{\vee} N : \tau$, where $\tau \not\leq_{\vee} \omega$.

Proof. We will prove that all subterms S of M not occurring under the scope of a λ -abstraction are typed in d by a subderivation d_S proving a typing $B_S \vdash_{\vee} S : \tau_s$ where $\tau_s \not\leq_{\vee} \omega$. The proof is given by induction on M .

The proof is obvious for $M \equiv x$; so let $M \equiv M_1 M_2$.

$B \vdash_{\vee} M_1 M_2 : \sigma$ and $\sigma \not\leq_{\vee} \omega$ imply that there is $\tau \in I(C_{\vee})$ such that $B \vdash_{\vee} M_1 : \tau \rightarrow \sigma$ and $B \vdash_{\vee} M_2 : \tau$, by Lemma 10.1.7.(vii). Moreover, by induction the property is true for all subterms S of M_i not occurring under the scope of a λ -abstraction ($1 \leq i \leq 2$).

The case $M \equiv \lambda x.N$ and S occur in N is against the hypothesis. \square

Lemma 12.1.32. Let d be a derivation proving $B \vdash_{\vee} C[M] : \sigma$, where $\sigma \not\leq_{\vee} \omega$. If M occurs in $C[M]$, and there is at least one subderivation of d assigning to M a type $\not\leq_{\vee} \omega$, then $B \vdash_{\vee} (\lambda x.C[x])M : \sigma$.

Proof. Without loss of generality, let each typing in d have the same basis B ; indeed, the only rule that can change the basis is $(\rightarrow I)$, and we can assume that free and bound variables have different names in $C[M]$.

Let M occur in $C[M]$ and let there exist $n \geq 1$ subderivations d_i in d proving $B \vdash_{\vee} M : \tau_i$, where $\tau_i \not\leq_{\vee} \omega$ ($1 \leq i \leq n$).

Let x be a fresh variable, so d can be transformed into a derivation d' proving $B[\tau_1 \wedge \dots \wedge \tau_n/x] \vdash_{\vee} (\lambda x.C[x])M : \sigma$ by performing the following operations. First,

- replace d_i by

$$\frac{}{B[\tau_1 \wedge \dots \wedge \tau_n/x] \vdash_{\vee} x : \tau_1 \wedge \dots \wedge \tau_n}^{(var)} \quad \frac{}{B[\tau_1 \wedge \dots \wedge \tau_n/x] \vdash_{\vee} x : \tau_i}^{(\leq_{\vee})}$$

- replace each typing $B \vdash_{\vee} P[M/x] : \mu$ occurring in the derivation d by the typing $B[\tau_1 \wedge \dots \wedge \tau_n/x] \vdash_{\vee} P : \mu$.

It is easy to check that d' is well defined, by induction on d .

So, by rule $(\rightarrow I)$, $B \vdash_{\vee} \lambda x.C[x] : (\tau_1 \wedge \dots \wedge \tau_n) \rightarrow \sigma$ and, by rule $(\wedge I)$, $B \vdash_{\vee} M : \tau_1 \wedge \dots \wedge \tau_n$, so the proof follows by rule $(\rightarrow E)$, since $\tau_1 \wedge \dots \wedge \tau_n \in I(C_{\vee})$. \square

Lemma 12.1.33 (Weak $\exists\ell$ -subject expansion).

$M \rightarrow_{\exists\ell} N$ and $B \vdash_{\vee} N : \sigma$ imply that there is B' such that $B' \vdash_{\vee} M : \sigma$ and $B'(x) \leq_{\vee} B(x)$, for each $x \in \text{Var}$.

Proof. Let $M \equiv C[(\lambda x.P)Q]$, let $N \equiv C[P[Q/x]]$ and, let d be the derivation proving $B \vdash_{\vee} C[P[Q/x]] : \sigma$. If $\sigma \simeq_{\vee} \omega$ then the proof is trivial, so let $\sigma \in I(C_{\vee})$. The proof is given by induction on $C[.]$.

Without loss of generality, let each typing in d have the same basis B . Indeed, the only rule that can change the basis is $(\rightarrow I)$, and we can assume that free and bound variables have different names in P . Let $C[.] \equiv [.]$. There are two cases.

- (i) Either Q does not occur in $P[Q/x]$, so $x \notin \text{FV}(P)$ and $N \equiv P$, or Q occurs in subterms of P that are subjects of an application of the rule (ω) . Since Q is a $\exists\ell$ -nf, then there is a basis B^* such that $B^* \vdash_{\vee} Q : \omega \rightarrow \omega$ by Lemma 12.1.5. Let $B'(y) = B(y) \wedge B^*(y)$, for each $y \in \text{Var}$; so, by Lemma 10.1.7.(ii) $B'[\omega \rightarrow \omega/x] \vdash_{\vee} N : \sigma$. Thus, by rule $(\rightarrow I)$, $B' \vdash_{\vee} \lambda x.P : (\omega \rightarrow \omega) \rightarrow \sigma$. Hence $B' \vdash_{\vee} (\lambda x.P)Q : \sigma$ by rule $(\rightarrow E)$.
- (ii) In the case where Q occurs in $P[Q/x]$ and there is at least one subderivation of d assigning to Q a type $\not\simeq_{\vee} \omega$, the proof follows by Lemma 12.1.32.

In the general case either $C[.] \equiv M'C'[.]$ or $C[.] \equiv C'[.]M'$, since the reduction is lazy. Let us consider the first case. By Lemma 10.1.7.(vi), there are subderivations d_0 and d_1 of d proving respectively $B \vdash_{\vee} C'[P[Q/x]] : \sigma_0$ and $B \vdash_{\vee} M' : \sigma_0 \rightarrow \sigma$, for $\sigma_0 \not\simeq_{\vee} \omega$. By induction there is a derivation d^* proving $B^* \vdash_{\vee} C'[(\lambda x.P)Q] : \sigma_0$ where $B^*(x) \leq_{\vee} B(x)$, for each $x \in \text{Var}$. By Lemma 10.1.7.(iii) $B^* \vdash_{\vee} M' : \sigma_0 \rightarrow \sigma$; hence it is easy to build a derivation proving $B^* \vdash_{\vee} C[(\lambda x.P)Q] : \sigma$. The second case is similar. \square

Now we are able to prove the theorem.

▼ Proof of Theorem 12.1.6 (pag.184).

- (i) By the Λ -subject reduction lemma, taking into account that $\exists\ell$ -reduction is a special case of Λ -reduction.
- (ii) By the weak $\exists\ell$ -subject expansion lemma. \blacksquare

12.1.3 Proof of the \mathcal{V} -Approximation Theorem

The proof follows the same lines as the corresponding proof in the previous models. In order to prove the (\Rightarrow) implication both parts of the theorem, we need to define a computability predicate.

A basis B is *finite* if and only if $B(y) \simeq_{\sqrt{\omega}} \omega \rightarrow \omega$ except in a finite number of variables. We will use $[\sigma_1/x_1, \dots, \sigma_n/x_n]$ to denote a finite basis. By Lemma 10.1.7.(i), in this section we limit ourselves to consider only such a kind of basis.

Let B and B' be two basis. $B \cup B'$ denotes the basis such that, for every x , $B \cup B'(x) = B(x) \wedge B'(x)$ (remember that $\sigma \wedge (\omega \rightarrow \omega) \simeq_{\sqrt{\omega}} \sigma$, for every type $\sigma \not\simeq_{\sqrt{\omega}} \omega$).

Definition 12.1.34. (i) $App_{\mathcal{V}}(B, \sigma, M)$ if and only if there is $A \in \mathcal{A}^{\sqrt{\omega}}(M)$ such that $B \vdash_{\sqrt{\omega}} A : \sigma$.

(ii) The predicate $Comp_{\mathcal{V}}$ is defined by induction on types as follows:

- $Comp_{\mathcal{V}}(B, \omega, M)$ is true;
- $Comp_{\mathcal{V}}(B, \sigma \rightarrow \tau, M)$ where $\tau \simeq_{\sqrt{\omega}} \omega$, if and only if $App_{\mathcal{V}}(B, \omega \rightarrow \omega, M)$;
- $Comp_{\mathcal{V}}(B, \sigma \rightarrow \tau, M)$ where $\tau \not\simeq_{\sqrt{\omega}} \omega$, if and only if $\forall N \in \Gamma, Comp_{\mathcal{V}}(B', \sigma, N)$ implies $Comp_{\mathcal{V}}(B \cup B', \tau, MN)$;
- $Comp_{\mathcal{L}}(B, \sigma \wedge \tau, M)$ if and only if $Comp_{\mathcal{V}}(B, \sigma, M)$ and $Comp_{\mathcal{V}}(B, \tau, M)$.

In the usual way, we prove that $B \vdash_{\sqrt{\omega}} M : \sigma$ implies $Comp_{\mathcal{V}}(B, \sigma, M)$, which in turn implies $App_{\mathcal{V}}(B, \sigma, M)$.

Lemma 12.1.35. $Comp_{\mathcal{V}}(B, \sigma, M)$ and $M =_{\Gamma} M'$ imply $Comp_{\mathcal{V}}(B, \sigma, M')$.

Proof. The proof is given by induction on σ . The case $\sigma \equiv \omega$ is obvious. If $\sigma \equiv \sigma \rightarrow \tau$ where $\tau \simeq_{\sqrt{\omega}} \omega$, then the proof follows from the definition of $App_{\mathcal{V}}$, since $App_{\mathcal{V}}$ is closed under $=_{\Gamma}$.

The other cases follow by the inductive hypothesis. \square

Hence, $Comp_{\mathcal{V}}$ is defined modulo $=_{\Gamma}$ on terms. The following property holds.

Property 12.1.36. Let B be a basis, M be a term and τ a type. $Comp_{\mathcal{V}}(B, \omega \rightarrow \tau, M)$ if and only if $Comp_{\mathcal{V}}(B, (\omega \rightarrow \omega) \rightarrow \tau, M)$.

Proof. The proof is easy by induction on the definition of $Comp_{\mathcal{V}}$, since $N \in \Gamma$ implies $B' \vdash N : \omega \rightarrow \omega$ for some B' , by Theorem 12.1.3.(ii). \square

In order to prove that $Comp_{\mathcal{V}}(B, \sigma, M)$ implies $App_{\mathcal{V}}(B, \sigma, M)$, we need the following property.

Property 12.1.37. Let A be an approximant such that $A \equiv \lambda z. \zeta A_1 \dots A_m z$, where ζ is either a variable or a head block, and $z \notin FV(\zeta A_1 \dots A_m)$.

If $B \vdash_{\sqrt{\omega}} A : \sigma \rightarrow \tau$ where $\sigma, \tau \not\simeq_{\sqrt{\omega}} \omega$ then $B \vdash_{\sqrt{\omega}} \zeta A_1 \dots A_m : \sigma \rightarrow \tau$.

Proof. $\sigma \in I(C_{\sqrt{}})$ and $B \vdash_{\sqrt{}} A : \sigma \rightarrow \tau$ imply $B[\sigma/z] \vdash_{\sqrt{}} \zeta A_1 \dots A_m z : \tau$, by Lemma 10.1.7.(vi). Since $\tau \not\simeq_{\sqrt{}} \omega$, by Lemma 10.1.7.(vii) there is $\epsilon \in I(C_{\sqrt{}})$ such that $B[\sigma/z] \vdash_{\sqrt{}} z : \epsilon$ and $B[\sigma/z] \vdash_{\sqrt{}} \zeta A_1 \dots A_m : \epsilon \rightarrow \tau$. By Lemma 10.1.7.(ii) $\sigma \leq_{\sqrt{}} \epsilon$; hence $\epsilon \rightarrow \tau \leq_{\sqrt{}} \sigma \rightarrow \tau$. So $B[\sigma/z] \vdash_{\sqrt{}} \zeta A_1 \dots A_m : \sigma \rightarrow \tau$ by rule $(\leq_{\sqrt{}})$. Clearly $B \vdash_{\sqrt{}} \zeta A_1 \dots A_m : \sigma \rightarrow \tau$, since $z \notin \text{FV}(\zeta A_1 \dots A_m)$. \square

Lemma 12.1.38. (i) $\text{App}_{\mathcal{V}}(B, \sigma, x\vec{M})$ implies $\text{Comp}_{\mathcal{V}}(B, \sigma, x\vec{M})$.
(ii) $\text{Comp}_{\mathcal{V}}(B, \sigma, M)$ implies $\text{App}_{\mathcal{V}}(B, \sigma, M)$.

Proof. The proof is done by mutual induction on σ .

The only nonobvious case is when $\sigma \equiv \tau \rightarrow \rho$, where $\rho \not\simeq_{\sqrt{}} \omega$.

- (i) We will prove that $N \in \Gamma$ and $\text{Comp}_{\mathcal{V}}(B', \tau, N)$ imply $\text{Comp}_{\mathcal{V}}(B \cup B', \rho, x\vec{M}N)$, thus $\text{Comp}_{\mathcal{V}}(B, \tau \rightarrow \rho, x\vec{M})$ follows by definition.
 $\text{Comp}_{\mathcal{V}}(B', \tau, N)$ implies $\text{App}_{\mathcal{V}}(B', \tau, N)$, by induction on (ii).
By hypothesis $\text{App}_{\mathcal{V}}(B, \tau \rightarrow \rho, x\vec{M})$; thus $B \cup B' \vdash_{\sqrt{}} A^* : \rho$, for some $A^* \in \mathcal{A}^{\sqrt{}}(x\vec{M}N)$ by rule $(\rightarrow E)$, since $x\vec{A} \in \mathcal{A}^{\sqrt{}}(x\vec{M})$ and $A' \in \mathcal{A}^{\sqrt{}}(N)$ imply $x\vec{A}A' \in \mathcal{A}^{\sqrt{}}(x\vec{M}N)$. Thus $\text{App}_{\mathcal{V}}(B \cup B', \rho, x\vec{M}N)$ and by induction, $\text{Comp}_{\mathcal{V}}(B \cup B', \rho, x\vec{M}N)$.
- (ii) Let $z \notin \text{FV}(M)$ and $B(z) \simeq_{\sqrt{}} \omega \rightarrow \omega$. Note that both $z \in \mathcal{A}^{\sqrt{}}$ and $[\tau/x] \vdash_{\sqrt{}} x : \tau$, thus $\text{App}_{\mathcal{V}}([\tau/z], \tau, z)$. Hence, $\text{Comp}_{\mathcal{V}}([\tau/z], \tau, z)$ by induction on (i).
 $\text{Comp}_{\mathcal{V}}(B, \tau \rightarrow \rho, M)$ and $\text{Comp}_{\mathcal{V}}([\tau/z], \tau, z)$ imply $\text{Comp}_{\mathcal{V}}(B[\tau/z], \rho, Mz)$ and this implies $\text{App}_{\mathcal{V}}(B[\tau/z], \rho, Mz)$, by induction; which means there is $A \in \mathcal{A}^{\sqrt{}}(Mz)$ such that $B[\tau/z] \vdash_{\sqrt{}} A : \rho$.
The case $A \equiv \Omega$ is not possible, since by the hypothesis $\rho \not\simeq_{\sqrt{}} \omega$. Hence $B \vdash_{\sqrt{}} \lambda z. A : \tau \rightarrow \rho$, by rule $(\rightarrow I)$. By definition of the $\sqrt{}$ -approximants of a term, $A \in \mathcal{A}(Mz)$ implies $\lambda z. A \in \mathcal{A}(\lambda z. Mz)$. Now there are two cases.
 - 1. M is of order 0, so A is of the shape $A'z$, where either $A' \equiv xA_1 \dots A_m$ or $A' \equiv (\lambda x. A')A''A_1 \dots A_m z$ and $z \notin \text{FV}(A')$.
In both cases $A' \in \mathcal{A}^{\sqrt{}}(M)$. By Property 12.1.37, $B \vdash A' : \tau \rightarrow \rho$, and so $\text{App}_{\mathcal{V}}(B, \tau \rightarrow \rho, M)$.
 - 2. Otherwise $M =_{\Gamma} \lambda y. M'$, so $\lambda z. Mz =_{\Gamma} \lambda z. M'[z/y] =_{\alpha} \lambda y. M'$, which implies $\lambda z. A \in \mathcal{A}^{\sqrt{}}(M)$ and the proof is given. \square

Lemma 12.1.39. $\text{Comp}_{\mathcal{V}}(B, \sigma, M)$ and $\sigma \leq_{\sqrt{}} \tau$ implies $\text{Comp}_{\mathcal{V}}(B, \tau, M)$.

Proof. By induction on the definition of $\leq_{\sqrt{}}$. The more complex case is that of rule (f) , so let $\pi' \leq_{\sqrt{}} \pi$, $\tau \leq_{\sqrt{}} \tau'$ and $\text{Comp}_{\mathcal{V}}(B, \pi \rightarrow \tau, M)$.

If $\tau \simeq_{\sqrt{}} \omega$ then $\tau' \simeq_{\sqrt{}} \omega$, so the proof is immediate. If $\tau, \pi \not\simeq_{\sqrt{}} \omega$ then the proof follows by induction. If $\tau \not\simeq_{\sqrt{}} \omega$ but $\tau' \simeq_{\sqrt{}} \omega$ then, by Lemma 12.1.38.(ii), $\text{App}_{\mathcal{V}}(B, \pi \rightarrow \tau, M)$. By definition of $\text{App}_{\mathcal{V}}$, there is an $A \in \mathcal{A}^{\leq{}}(M)$ such that $B \vdash_{\sqrt{}} A : \pi \rightarrow \tau$; thus $B \vdash_{\sqrt{}} A : \omega \rightarrow \omega$, and the proof follows by definition of $\text{Comp}_{\mathcal{V}}$. \square

Lemma 12.1.40. Let $\text{FV}(M) \subseteq \{x_1, \dots, x_n\}$ and $B = [\sigma_1/x_1, \dots, \sigma_n/x_n]$. If $N_i \in \Gamma$, $\text{Comp}_{\mathcal{V}}(B_i, \sigma_i, N_i)$ ($1 \leq i \leq n$) and $B \vdash_{\vee} M : \tau$, then

$$\text{Comp}_{\mathcal{V}}(B_1 \cup \dots \cup B_n, \tau, M[N_1/x_1, \dots, N_n/x_n]).$$

Proof. The proof is given by induction on the derivation of $B \vdash_{\vee} M : \tau$. The most interesting case is when the last applied rule is $(\rightarrow I)$.

Let $M \equiv \lambda x.M'$, $\tau \equiv \mu \rightarrow \nu$, $\mu \in I(C_{\vee})$ and

$$\frac{B[\mu/x] \vdash_{\vee} M' : \nu}{B \vdash_{\vee} \lambda x.M' : \mu \rightarrow \nu} (\rightarrow I).$$

If $N \in \Gamma$ and $\text{Comp}_{\mathcal{V}}(B', \mu, N)$, then by induction

$$\text{Comp}_{\mathcal{V}}(B' \cup B_1 \cup \dots \cup B_n, \nu, M'[N_1/x_1, \dots, N_n/x_n, N/x])$$

which implies

$$\text{Comp}_{\mathcal{V}}(B' \cup B_1 \cup \dots \cup B_n, \nu, (\lambda x.M'[N_1/x_1, \dots, N_n/x_n])N)$$

by Lemma 12.1.35. So $\text{Comp}_{\mathcal{V}}(B_1 \cup \dots \cup B_n, \tau, M[N_1/x_1, \dots, N_n/x_n])$ by definition of $\text{Comp}_{\mathcal{V}}$. All other cases follow directly from the inductive hypothesis. \square

Moreover, the following property holds.

Property 12.1.41. Let $M, N \in \Lambda\Omega$ such that $M \rightarrow_{\exists\ell\Omega} N$.

- (i) If $B \vdash M : \sigma$ then $B \vdash N : \sigma$.
- (ii) If $B \vdash N : \sigma$ then $B' \vdash M : \sigma$, for some B' such that $\forall x \in \text{Var}, B'(x) \leq_{\vee} B(x)$.

Proof. Easy, by Theorem 12.1.6, and by the fact that to the term Ω only the type ω can be assigned. \square

▼ Proof of the \mathcal{V} -Approximation Theorem (Theorem 12.1.15 pag. 186).

(i) (\Rightarrow) Clearly $\text{Comp}_{\mathcal{V}}([\tau/x], \tau, x)$ by Lemma 12.1.38.(i).

Let $\text{FV}(M) \subseteq \{x_1, \dots, x_n\}$; if $B \vdash_{\vee} M : \sigma$, $B = [\sigma_1/x_1, \dots, \sigma_n/x_n]$ and $\text{Comp}_{\mathcal{V}}([\sigma_i/x_i], \sigma_i, x_i)$ ($1 \leq i \leq n$) then $\text{Comp}_{\mathcal{V}}(B, \sigma, M)$ by Lemma 12.1.40. By Lemma 12.1.38.(ii) and definition of $\text{App}_{\mathcal{V}}$ the proof is done.

(\Leftarrow) By definition, there is M' such that $M =_{\Gamma} M'$ and A matches M' except at occurrences of Ω . A derivation of $B \vdash_{\vee} A : \sigma$ can be transformed into a derivation of $B \vdash_{\vee} M' : \sigma$, simply by replacing every subderivation

$$\overline{B \vdash_{\vee} \Omega : \omega}^{(\omega)} \quad \text{by} \quad \overline{B \vdash_{\vee} N : \omega}^{(\omega)},$$

where N is the subterm replaced by Ω in M' . $B \vdash_{\vee} M' : \sigma$ implies $B \vdash_{\vee} M : \sigma$, since the type assignment system is closed under $=_{\Gamma}$ on terms as consequence of the fact that it induces a $\lambda\Gamma$ -model, so the proof is given.

- (ii) It follows by point (i) of the \mathcal{V} -approximation theorem and Property 12.1.41. ■

12.1.4 Proof of Theorems 12.1.24 and 12.1.25

Let $V_0 \equiv \lambda x.(\lambda x_1 x_2.DD)(x(\lambda x_1.DD)(\lambda x_1.DD))$,

$V_1 \equiv \lambda x.(\lambda x_1 x_2 x_3.DD)(x(\lambda x_1.DD)(\lambda x_1 x_2.DD))(x(\lambda x_1 x_2.DD)(\lambda x_1.DD))$.

In Sect. 7.1.1 we proved that $V_0 \approx_{\mathcal{V}} V_1$; now we will prove $V_0 \not\sim_{\mathcal{V}} V_1$. Note that both V_0 and V_1 are Γ -unsolvable of order 2, so the model does not equate all Γ -unsolvable terms of the same order.

▼ Proof of V-Incompleteness Theorem (Theorem 12.1.24 pag. 190).

Let $\sigma \equiv \sigma_0 \wedge \sigma_1$, where $\sigma_0 \equiv (\omega \rightarrow \omega) \rightarrow (\omega \rightarrow \omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega$,

$$\sigma_1 \equiv (\omega \rightarrow \omega \rightarrow \omega) \rightarrow (\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega,$$

and let $D^1 \equiv \lambda x_1.DD$, $D^2 \equiv \lambda x_1 x_2.DD$ and $D^3 \equiv \lambda x_1 x_2 x_3.DD$.

We will show that $B \vdash_{\vee} V_1 : \sigma \rightarrow \omega \rightarrow \omega$ while $B \not\vdash_{\vee} V_0 : \sigma \rightarrow \omega \rightarrow \omega$, for all bases B . Let d_{12} be the derivation

$$\frac{\frac{\frac{\frac{\frac{\frac{\frac{B[\sigma/x] \vdash_{\vee} x : \sigma} \quad B[\sigma/x, \omega \rightarrow \omega/x_1 \vdash_{\vee} DD : \omega}{B[\sigma/x, \omega \rightarrow \omega/x_1, \omega \rightarrow \omega/x_2 \vdash_{\vee} DD : \omega}^{(\omega)} \quad (\var) \quad (\wedge E)} \quad \frac{B[\sigma/x] \vdash_{\vee} D^1 : (\omega \rightarrow \omega) \rightarrow \omega \quad B[\sigma/x, \omega \rightarrow \omega/x_1] \vdash_{\vee} D^1 : (\omega \rightarrow \omega) \rightarrow \omega}{B[\sigma/x] \vdash_{\vee} xD^1 : (\omega \rightarrow \omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega}^{(\rightarrow I)} \quad (\leq \vee)} \quad (\wedge E) \quad (\leq \vee)} \quad (\rightarrow E) \quad (\rightarrow E)}{B[\sigma/x] \vdash_{\vee} xD^1 D^2 : \omega \rightarrow \omega}^{(\rightarrow E)}$$

and let d_{21} be the derivation

$$\frac{\frac{\frac{\frac{\frac{\frac{\frac{B[\sigma/x, \omega \rightarrow \omega/x_1, \omega \rightarrow \omega/x_2] \vdash_{\vee} DD : \omega} \quad B[\sigma/x, \omega \rightarrow \omega/x_1] \vdash_{\vee} D^1 : (\omega \rightarrow \omega) \rightarrow \omega}{B[\sigma/x, \omega \rightarrow \omega/x_1, \omega \rightarrow \omega/x_2] \vdash_{\vee} xD^2 D^1 : (\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega}^{(\omega)} \quad (\var) \quad (\wedge E)} \quad (\rightarrow I) \quad (\leq \vee)} \quad (\leq \vee) \quad (\rightarrow E)} \quad (\rightarrow E) \quad (\rightarrow E)}{B[\sigma/x] \vdash_{\vee} xD^2 D^1 : \omega \rightarrow \omega}^{(\rightarrow E)}$$

It is easy to build a derivation d_3 proving the typing

$$B[\sigma/x] \vdash_{\vee} D^3 : (\omega \rightarrow \omega) \rightarrow (\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega.$$

Now, we can build the following derivation:

$$\frac{\begin{array}{c} d_3 \\ \hline B[\sigma/x] \vdash_{\vee} D^3 : (\omega \rightarrow \omega) \rightarrow (\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega \end{array} \cdots \quad \begin{array}{c} d_{12} \\ \hline B[\sigma/x] \vdash_{\vee} xD^1D^2 : \omega \rightarrow \omega \end{array} \cdots \quad \begin{array}{c} d_{21} \\ \hline B[\sigma/x] \vdash_{\vee} xD^2D^1 : \omega \rightarrow \omega \end{array} (\dots)}{\begin{array}{c} B[\sigma/x] \vdash_{\vee} D^3(xD^1D^2) : (\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega \\ \hline B[\sigma/x] \vdash_{\vee} (\lambda x_1x_2x_3.DD)(x(\lambda x_1.DD)(\lambda x_2.DD))(x(\lambda x_1x_2.DD)(\lambda x_1.DD)) : \omega \rightarrow \omega \end{array} (\rightarrow E) \quad (\rightarrow I)}$$

$$\frac{\hline B[\sigma/x] \vdash_{\vee} V_1 : \sigma \rightarrow \omega \rightarrow \omega}{}$$

Note that we can apply the rule $(\rightarrow E)$, since $\omega \rightarrow \omega \in I(C_{\vee})$; moreover, each type considered in the basis is an input type.

On the other hand, $B \vdash_{\vee} V_0 : \sigma \rightarrow \omega \rightarrow \omega$ implies, by Lemma 10.1.7.(vi), $B[\sigma/x] \vdash_{\vee} (\lambda x_1x_2.DD)(x(\lambda x_1.DD)(\lambda x_1.DD)) : \omega \rightarrow \omega$. Therefore there is $\mu \in I(C_{\vee})$ such that $B[\sigma/x] \vdash_{\vee} \lambda x_1x_2.DD : \mu \rightarrow \omega \rightarrow \omega$ and $B[\sigma/x] \vdash_{\vee} x(\lambda x_1.DD)(\lambda x_1.DD) : \mu$, by Lemma 10.1.7.(vii). Since $\mu \not\simeq_{\vee} \omega$, again by Lemma 10.1.7.(vii), there is $\tau \in I(C_{\vee})$ such that $B[\sigma/x] \vdash_{\vee} x(\lambda x_1.DD) : \tau \rightarrow \mu$ and

$$B[\sigma/x] \vdash_{\vee} \lambda x_1.DD : \tau. \tag{12.1}$$

Since $\tau \rightarrow \mu \not\simeq_{\vee} \omega$, again by Lemma 10.1.7.(vii), there is $\pi \in I(C_{\vee})$ such that $B[\sigma/x] \vdash_{\vee} x : \pi \rightarrow \tau \rightarrow \mu$ and

$$B[\sigma/x] \vdash_{\vee} \lambda x_1.DD : \pi. \tag{12.2}$$

By Lemma 10.1.7.(ii) $\sigma \leq_{\vee} \pi \rightarrow \tau \rightarrow \mu$, so, since \vee is legal, there are 3 possible cases.

1. $\pi \leq_{\vee} \omega \rightarrow \omega \rightarrow \omega$ is not possible; in fact, it is easy to see that the typing given in Eq. 12.2 implies $\omega \rightarrow \omega \leq_{\vee} \pi$, by Theorem 12.1.19.(ii). So, by Property 12.1.28.(ii), this would imply $\omega \rightarrow \omega \simeq_{\vee} \omega \rightarrow \omega \rightarrow \omega$, which is an absurd.
2. $\pi \leq_{\vee} (\omega \rightarrow \omega) \wedge (\omega \rightarrow \omega \rightarrow \omega)$ is not possible. In fact, by rules (c), $(\omega \rightarrow \omega) \wedge (\omega \rightarrow \omega \rightarrow \omega) \leq_{\vee} \omega \rightarrow \omega \rightarrow \omega$, so we can reason as in the previous case.
3. $\pi \leq_{\vee} \omega \rightarrow \omega$ and $(\omega \rightarrow \omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega \leq_{\vee} \tau \rightarrow \mu$, therefore $\tau \leq_{\vee} \omega \rightarrow \omega \rightarrow \omega$ by Property 10.1.6. Yet an absurd, by the typing given in Eq. 12.1 and by Theorem 12.1.19.(ii). ■

In order to prove Theorem 12.1.25, we need Lemma 12.1.42.

Lemma 12.1.42. *Let U be a closed Γ -unsolvable term of order 0 and let ∇ be a type system $< C, \leq_{\nabla}, I(C) >$ inducing a $\lambda\Gamma$ -model \mathcal{M} that is fully abstract with respect to the \mathbf{V} -operational semantics.*

- (i) All closed Γ -unsolvable terms of the same finite order n are equated in \mathcal{M} .
- (ii) $U \sqsubseteq_{\mathcal{M}} \lambda x.U$.
- (iii) There exists $\theta \in I(C)$ such that $B \vdash_{\nabla} \lambda x.U : \theta$ and $B \vdash_{\nabla} \lambda xy.U : \theta \rightarrow \theta$, while $B \not\vdash_{\nabla} U : \theta$ and $B \not\vdash_{\nabla} \lambda x.U : \theta \rightarrow \theta$, for all bases B ; moreover, $\theta \rightarrow \theta \in I(C)$.
- (iv) If θ is the type considered in the previous point then $(\theta \rightarrow \theta) \rightarrow \theta \rightarrow \theta, \theta \rightarrow (\theta \rightarrow \theta) \rightarrow \theta \in I(C)$.

Proof. (i) By the fact that all closed Γ -unsolvable terms of the same finite order are equated in \mathbf{V} (see Corollary 7.1.9) and by definition of full abstraction.

- (ii) By the fact that $U \prec_{\nabla} \lambda x.U$ and by definition of full abstraction.
- (iii) Note that $\lambda x.U$ is an input value for the $\lambda\Gamma$ -calculus, so by the definition of the $\lambda\Gamma$ -model and by the previous point of this lemma, there is $\theta \in I(C)$ such that $B \vdash_{\nabla} \lambda x.U : \theta$ and $B \not\vdash_{\nabla} U : \theta$, for all bases B (since U is closed). It is easy to build a derivation proving $B \vdash_{\nabla} \lambda xy.U : \theta \rightarrow \theta$. If $B \vdash_{\nabla} \lambda x.U : \theta \rightarrow \theta$ then $B \vdash_{\nabla} (\lambda x.U)\lambda x.U : \theta$, but $(\lambda x.U)\lambda x.U$ is a Λ -unsolvable term of order 0, so $B \not\vdash_{\nabla} \lambda x.U : \theta \rightarrow \theta$. If $\theta \rightarrow \theta \notin I(C)$ then $\theta \leq_{\nabla} \theta \rightarrow \theta$, so $B \vdash_{\nabla} \lambda x.U : \theta \rightarrow \theta$; hence, $\theta \rightarrow \theta \in I(C)$.
- (iv) If $(\theta \rightarrow \theta) \rightarrow \theta \rightarrow \theta \notin I(C)$ then $\theta \leq_{\nabla} (\theta \rightarrow \theta) \rightarrow \theta \rightarrow \theta$ and $B \vdash_{\nabla} \lambda x.U : (\theta \rightarrow \theta) \rightarrow \theta \rightarrow \theta$, so $B \vdash_{\nabla} (\lambda x.U)(\lambda xy.U)(\lambda x.U) : \theta$ which is an absurdum; hence, $(\theta \rightarrow \theta) \rightarrow \theta \rightarrow \theta \in I(C)$. In a similar way, $\theta \rightarrow (\theta \rightarrow \theta) \rightarrow \theta \in I(C)$. \square

▼ Proof of Theorem 12.1.25 (pag.190).

We prove that every Γ -model fully abstract with respect to the \mathbf{V} -operational semantics would equate the two terms V_0 and V_1 .

Let $< C, \leq_{\nabla}, I(C) >$ be a legal type system inducing a filter $\lambda\Gamma$ -model which is fully abstract with respect to the \mathbf{V} -operational semantics, and let θ be the input type considered in the Lemma 12.1.42. Note that $\theta \not\asymp_{\nabla} \omega$.

Let $\sigma \equiv \sigma_0 \wedge \sigma_1$ where $\sigma_0 \equiv \theta \rightarrow (\theta \rightarrow \theta) \rightarrow \theta$ and $\sigma_1 \equiv (\theta \rightarrow \theta) \rightarrow \theta \rightarrow \theta$; moreover let $D^1 \equiv (\lambda x_1.DD)$, $D^2 \equiv (\lambda x_1x_2.DD)$ and $D^3 \equiv (\lambda x_1x_2x_3.DD)$. We will show that $B \vdash_{\nabla} V_1 : \sigma \rightarrow \theta$ while $B \not\vdash_{\nabla} V_0 : \sigma \rightarrow \theta$, for all basis B . It is easy to build derivations proving the typings

$$\begin{aligned} B[\sigma/x] &\vdash_{\nabla} xD^1D^2 : \theta \\ B[\sigma/x] &\vdash_{\nabla} xD^2D^1 : \theta \\ B[\sigma/x] &\vdash_{\nabla} D^3 : \theta \rightarrow \theta \rightarrow \theta \end{aligned}$$

thus can build the following derivation

$$\frac{\vdots \quad \vdots \quad \vdots}{\frac{\frac{B[\sigma/x] \vdash_{\nabla} D^3 : \theta \rightarrow \theta \rightarrow \theta \quad (\dots) \quad B[\sigma/x] \vdash_{\nabla} xD^1D^2 : \theta \quad (\dots)}{B[\sigma/x] \vdash_{\nabla} D^3(xD^1D^2) : \theta \rightarrow \theta \quad (\rightarrow E)} \quad \frac{B[\sigma/x] \vdash_{\nabla} xD^2D^1 : \theta \quad (\dots)}{B[\sigma/x] \vdash_{\nabla} (\lambda x_1x_2x_3.DD)(x(\lambda x_1.DD)(\lambda x_1x_2.DD))(x(\lambda x_1x_2.DD)(\lambda x_1.DD)) : \theta \quad (\rightarrow E)}}{B \vdash_{\nabla} V_1 : \sigma \rightarrow \theta \quad (\rightarrow I)}$$

Note that we can apply the rule $(\rightarrow E)$, since $\theta \in I(C)$; moreover each type considered in the basis is an input type.

Since in Sect. 7.1.1 we proved that $V_0 \approx_V V_1$, by the full abstraction hypothesis it follows that $B \vdash_{\nabla} V_0 : \sigma \rightarrow \theta$; so, by Lemma 10.1.7.(vi), $B[\sigma/x] \vdash_{\nabla} (\lambda x_1x_2.DD)(x(\lambda x_1.DD)(\lambda x_1.DD)) : \theta$.

So there is $\mu \in I(C)$ such that $B[\sigma/x] \vdash_{\nabla} \lambda x_1x_2.DD : \mu \rightarrow \theta$ and $B[\sigma/x] \vdash_{\nabla} x(\lambda x_1.DD)(\lambda x_1.DD) : \mu$, by Lemma 10.1.7.(vii).

If $\mu \simeq_{\vee} \omega$ then $\omega \in I(C)$, so $I(C) = T(C)$ and $B[\sigma/x] \vdash_{\nabla} \lambda x_1x_2.DD : \omega \rightarrow \theta$; so $B[\sigma/x] \vdash_{\nabla} (\lambda x_1x_2.DD)(DD) : \theta$ by rule $(\rightarrow E)$, against Lemma 12.1.42.(iii), since $(\lambda x_1x_2.DD)(DD)$ is a Γ -unsolvable term of order 0.

Let $\mu \not\simeq_{\vee} \omega$; again by Lemma 10.1.7.(vii), there exists $\tau \in I(C)$ such that $B[\sigma/x] \vdash_{\nabla} x(\lambda x_1.DD) : \tau \rightarrow \mu$ and,

$$B[\sigma/x] \vdash_{\nabla} \lambda x_1.DD : \tau. \quad (12.3)$$

If $\tau \rightarrow \mu \simeq_{\vee} \omega$ then $\tau \rightarrow \omega \leq_{\vee} \omega \leq_{\vee} \tau \rightarrow \mu$, so by Property 10.1.6 $\omega \leq_{\vee} \mu$ and thus $\mu \simeq_{\vee} \omega$, which is not possible; hence, $\tau \rightarrow \mu \not\simeq_{\vee} \omega$.

Since $\tau \rightarrow \mu \not\simeq_{\vee} \omega$, again by Lemma 10.1.7.(vii), there is $\pi \in I(C)$ such that $B[\sigma/x] \vdash_{\nabla} x : \pi \rightarrow \tau \rightarrow \mu$ and,

$$B[\sigma/x] \vdash_{\nabla} \lambda x_1.DD : \pi. \quad (12.4)$$

By Lemma 10.1.7.(ii) $\sigma \leq_{\vee} \pi \rightarrow \tau \rightarrow \mu$, so, since ∇ is legal, there are three possible cases:

1. $\pi \leq_{\nabla} \theta \rightarrow \theta$ is not possible; otherwise the typing given in Eq. 12.4 would imply $B[\sigma/x] \vdash_{\nabla} \lambda x_1.DD : \theta \rightarrow \theta$ against Lemma 12.1.42.
2. $\pi \leq_{\nabla} \theta \wedge (\theta \rightarrow \theta)$ is not possible. In fact, by rule (c), $\theta \wedge (\theta \rightarrow \theta) \leq_{\nabla} \theta \rightarrow \theta$, so we can reason as in the previous case.
3. $\pi \leq_{\nabla} \theta$ and $(\theta \rightarrow \theta) \rightarrow \theta \leq_{\nabla} \tau \rightarrow \mu$, so by Property 10.1.6, $\tau \leq_{\vee} \theta \rightarrow \theta$; yet an absurdum, by the typing given in Eq. 12.3. ■

12.2 A Fully Abstract Model for the **V**-Operational Semantics

It was proved in Theorem 12.1.25 that there is not a filter $\lambda\Gamma$ -model that is fully abstract with respect to the **V**-operational semantics. But we will show that it is possible to build a fully abstract model starting from the model \mathcal{V} ,

in a way similar to that presented in the Sect. 11.4. We start by defining a preorder relation on terms.

Definition 12.2.1. (i) \trianglelefteq_σ is a relation on Λ^0 defined as follows:

- $M \trianglelefteq_\omega N$ is true;
- $M \trianglelefteq_{\sigma \rightarrow \tau} N$ where $\tau \simeq_\vee \omega$, if and only if
 $B \vdash_\vee M : \omega \rightarrow \omega$ implies $B \vdash_\vee N : \omega \rightarrow \omega$, for all basis B ;
- $M \trianglelefteq_{\sigma \rightarrow \tau} N$ where $\tau \not\simeq_\vee \omega$, if and only if
 $\forall P$ closed Γ -valuable term, $B \vdash_\vee P : \sigma$ implies $MP \trianglelefteq_\tau NP$;
- $M \trianglelefteq_{\sigma \wedge \tau} N$ if and only if both $M \trianglelefteq_\sigma N$ and $M \trianglelefteq_\tau N$.

(ii) $M \trianglelefteq N$ if and only if $M \trianglelefteq_\sigma N$, for all σ .

The previous definition is well posed, thanks to the following property.

Property 12.2.2. There is $P \in \Gamma^0$ such that $B \vdash_\vee P : \sigma$, for all B and σ .

Proof. By induction on σ , we will prove that there is P of the shape: $\lambda x_1 \dots x_n. DD$, for $n \geq 0$, to which σ can be assigned.

If $\sigma \equiv \omega$ then $B \vdash_\vee DD : \omega$, by rule (ω) . Let $\sigma \equiv \mu \rightarrow \nu$.

If $\nu \simeq_\vee \omega$ and $\mu \not\simeq_\vee \omega$ then, by rule (ω) , $B[\mu/x] \vdash_\vee DD : \omega$, and then, by rule $(\rightarrow I)$ and (\leq_\vee) , $B \vdash_\vee DD : \mu \rightarrow \nu$.

If $\mu, \nu \simeq_\vee \omega$, then, by rule (ω) , $B[\omega \rightarrow \omega/x] \vdash_\vee DD : \omega$, and the result follows by rules $(\rightarrow I)$ and (\leq_\vee) , taking into account Property 12.1.28.(i).

If $\mu, \nu \not\simeq_\vee \omega$, then by induction there is $P \in \Gamma^0$ such that $B \vdash_\vee P : \nu$ and, since $P \in \Gamma^0$, $B[\mu/x] \vdash_\vee P : \nu$, so the proof follows by rule $(\rightarrow I)$.

Let $\sigma \equiv \mu \wedge \nu$. By induction, there are $\lambda x_1 \dots x_p. DD$ and $\lambda x_1 \dots x_q. DD$ such that $B \vdash_\vee \lambda x_1 \dots x_p. DD : \mu$ and $B \vdash_\vee \lambda x_1 \dots x_q. DD : \nu$. Let $n = \max\{p, q\}$, so $\lambda x_1 \dots x_n. DD$ is the desired term. \square

Note that although in the model \mathcal{V} all types are inhabited, this does not imply that all filters are inhabited. Indeed, the filter $\uparrow \{\sigma\}$, where σ is $((\omega \rightarrow \omega) \rightarrow (\omega \rightarrow \omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega) \wedge ((\omega \rightarrow \omega \rightarrow \omega) \rightarrow (\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega)$ is not the interpretation of any term, since the reader can check that every term having type σ has also the type $(\omega \rightarrow \omega) \rightarrow (\omega \rightarrow \omega) \rightarrow \omega \rightarrow \omega$, which is not in the filter. If this filter were inhabited, then it would be $V_0 \not\leq_{\mathbf{V}} V_1$ (see Sect. 12.1.4).

Property 12.2.3. Let $M, N \in \Lambda^0$.

- (i) If $M \sqsubseteq_\nu N$ then $M \leq N$.
- (ii) \leq is reflexive.
- (iii) \leq is transitive.

Proof. (i) We will prove that $M \not\trianglelefteq N$ implies $M \not\leq_\nu N$. By definition, $M \not\trianglelefteq N$ means there is σ such that $M \not\trianglelefteq_\sigma N$. The proof is given by induction on σ .

Clearly $\sigma \not\simeq_{\vee} \omega$, since by definition $M \trianglelefteq_{\omega} N$ is true. If $\sigma \equiv \mu \rightarrow \nu$, there are two cases. If $\nu \simeq_{\vee} \omega$ then $B \vdash_{\vee} M : \omega \rightarrow \omega$ and $B \not\vdash_{\vee} N : \omega \rightarrow \omega$, so the proof is immediate by definition of \sqsubseteq_{ν} .

If $\nu \not\simeq_{\vee} \omega$ then there is a Γ -valuable $P \in \Lambda^0$ such that $MP \not\trianglelefteq_{\nu} NP$, by definition of \trianglelefteq . Hence, $MP \not\trianglelefteq_{\nu} NP$, by induction, so $M \not\trianglelefteq_{\nu} N$ by Lemma 10.1.13.(i). If $\sigma \equiv \mu \wedge \nu$ then the proof follows by induction.

- (ii) We will prove that $M \trianglelefteq_{\sigma} M$, for all σ , by induction on σ . The case ω is obvious. Let $\sigma \equiv \mu \rightarrow \nu$; the case $\nu \simeq_{\vee} \omega$ is obvious. Let $\nu \not\simeq_{\vee} \omega$ and let $P \in \Lambda^0$ be a closed Γ -valuable term such that $B \vdash_{\vee} P : \mu$. By induction $MP \trianglelefteq_{\nu} MP$, so the result follows by definition of \trianglelefteq .

The case $\sigma \equiv \mu \wedge \nu$ follows by induction.

- (iii) By induction on σ we prove that \trianglelefteq_{σ} is transitive. The only nontrivial case is $\sigma \equiv \pi \rightarrow \tau$, where $\tau \not\simeq_{\vee} \omega$. Let $M_0 \trianglelefteq_{\pi \rightarrow \tau} M_1$ and $M_1 \trianglelefteq_{\pi \rightarrow \tau} M_2$. If $P \in \Lambda^0$ is a Γ -valuable term and $B \vdash_{\vee} P : \pi$, then $M_0P \trianglelefteq_{\tau} M_1P$ and $M_1P \trianglelefteq_{\tau} M_2P$, by definition of \trianglelefteq . So $M_0P \trianglelefteq_{\tau} M_2P$ by induction; hence $M_0 \trianglelefteq_{\pi \rightarrow \tau} M_2$ by definition of \trianglelefteq . \square

Next two lemmas prove that the relation \trianglelefteq grasps exactly the behaviour of the **V**-operational semantics.

Lemma 12.2.4. *Let $M, N \in \Lambda^0$.*

$M \trianglelefteq N$ if and only if $M\vec{P} \trianglelefteq_{\omega \rightarrow \omega} N\vec{P}$, for each sequence of closed Γ -valuable terms \vec{P} .

Proof. (\Leftarrow) We will prove that $M \not\trianglelefteq N$ implies that there is a closed sequence of Γ -valuable terms \vec{P} such that $M\vec{P} \not\trianglelefteq_{\omega \rightarrow \omega} N\vec{P}$. By hypothesis there is a type σ such that $M \not\trianglelefteq_{\sigma} N$, so the proof is done by induction on σ .

If $\sigma \simeq_{\vee} \omega$ then $\sigma \equiv \underbrace{\omega \wedge \dots \wedge \omega}_{n \geq 1}$, by Theorem 12.1.26; but since

$M \trianglelefteq_{\omega} N$ by definition, this is not possible. If $\sigma \equiv \mu \rightarrow \nu$ and $\nu \simeq_{\vee} \omega$ then the proof is trivial. If $\sigma \equiv \mu \rightarrow \nu$ and $\nu \not\simeq_{\vee} \omega$ then there is a Γ -valuable term $P \in \Lambda^0$ such that $MP \not\trianglelefteq_{\nu} NP$, so the proof follows by induction. If $\sigma \equiv \mu \wedge \nu$ then the proof follows by induction.

(\Rightarrow) We will prove that, if there is a sequence of closed Γ -valuable terms \vec{P} and a type $\tau \not\simeq_{\vee} \omega$ such that $M\vec{P} \not\trianglelefteq_{\tau} N\vec{P}$, then $M \not\trianglelefteq N$. The proof will be given by induction on $\|\vec{P}\|$.

If $\|\vec{P}\| = 0$ then the proof is trivial, so let $\|\vec{P}\| \geq 1$ and $\vec{P} \equiv \vec{Q}\vec{Q}'$. Since Q' is a closed Γ -valuable term, $B \vdash_{\vee} Q' : \omega \rightarrow \omega$ by Property 12.1.8. This implies $M\vec{Q} \not\trianglelefteq_{(\omega \rightarrow \omega) \rightarrow \tau} N\vec{Q}$ by definition of \trianglelefteq ; so the proof follows by induction. \square

Note that $M \trianglelefteq_{\sigma} N$ and $\sigma \leq_{\angle} \tau$ do not imply $M \trianglelefteq_{\tau} N$. Nevertheless, $M \trianglelefteq_{\omega \rightarrow \sigma} N$ if and only if $M \trianglelefteq_{(\omega \rightarrow \omega) \rightarrow \sigma} N$.

Lemma 12.2.5. *Let $M, N \in \Lambda^0$.*

$M \preceq_{\mathbf{V}} N$ if and only if $M\vec{P} \trianglelefteq_{\omega \rightarrow \omega} N\vec{P}$, for each sequence of closed Γ -valuable terms \vec{P} .

Proof. Let Q be a closed Γ -valuable term. Then $Q \Downarrow_{\mathbf{V}}$ if and only if $B \vdash_{\sqrt{\cdot}} Q : \omega \rightarrow \omega$, by Property 12.1.8 and Lemma 12.1.17.

- ⇒ Let \vec{P} be a sequence of closed Γ -valuable terms, and let B be a basis.
If $M \preceq_{\mathbf{V}} N$ then $M\vec{P} \Downarrow_{\mathbf{V}}$ implies $N\vec{P} \Downarrow_{\mathbf{V}}$; thus $B \vdash_{\sqrt{\cdot}} M\vec{P} : \omega \rightarrow \omega$ implies $B \vdash_{\sqrt{\cdot}} N\vec{P} : \omega \rightarrow \omega$. So the proof is done, by definition of $\trianglelefteq_{\omega \rightarrow \omega}$.
- ⇐ Let $M\vec{P} \trianglelefteq_{\omega \rightarrow \omega} N\vec{P}$, for each sequence of closed terms \vec{P} . Let us recall the notion of weight of a term, defined in Definition 3.1.29 (pag. 43), and the fact, proved in Corollary 3.2.2, that the weight of a term is defined if and only if it has $\Xi\ell$ -normal form.

We will prove that, if $C[M], C[N] \in \Lambda^0$ and $\langle C[M] \rangle$ is defined then $\langle C[N] \rangle$ is defined, for all contexts $C[.]$. Hence the result follows from Theorem 7.1.3, taking into consideration that the set of closed Γ -lazy blocked normal forms coincides with the set of closed $\Xi\ell$ -normal forms.

The proof will be given by induction on $\langle C[M] \rangle$. There are two cases, according to the possible shape of $C[.]$.

- $C[.] \equiv [.]C_1[.]...C_m[.]$ ($m \in \mathbb{N}$).

If $m = 0$ then $\langle M \rangle$ defined implies M has $\Xi\ell$ -normal form, so $B \vdash_{\sqrt{\cdot}} M : \omega \rightarrow \omega$. But $B \vdash_{\sqrt{\cdot}} N : \omega \rightarrow \omega$ by definition of $\trianglelefteq_{\omega \rightarrow \omega}$, and the proof follows by Property 12.1.8 and Lemma 12.1.17.

Let $m \geq 1$ and let $M \equiv (\lambda x.M_0)M_1...M_p$. Pose $D[.] \equiv MC_1[.]...C_m[.]$, so $D[M] \equiv C[M]$ and $D[.] \equiv (\lambda x.M_0)M_1...M_pC_1[.]...C_m[.]$ ($m \in \mathbb{N}$).

If $p > 0$ then let $D^*[.] \equiv M_0[M_1/x]M_2...M_pC_1[.]...C_m[.]$, otherwise let $D^*[.] \equiv M_0[C_1[.]/x]C_2[.]...C_m[.]$; in both cases the weight of $D^*[M]$ is defined, since $\langle C[M] \rangle$ is defined. Moreover, $\langle D^*[M] \rangle < \langle C[M] \rangle$, so by induction $\langle D^*[N] \rangle$ is defined. But $D^*[N] \equiv MC_1[N]C_2[N]...C_m[N]$ has $\Xi\ell$ -normal form implies $B \vdash_{\sqrt{\cdot}} MC_1[N]...C_m[N] : \omega \rightarrow \omega$, so by hypothesis $B \vdash_{\sqrt{\cdot}} NC_1[N]...C_m[N] : \omega \rightarrow \omega$. Hence, $NC_1[N]...C_m[N]$ has $\Xi\ell$ -normal and the proof follows by Corollary 3.2.2.

- $C[.] \equiv (\lambda y.C_0[.])C_1[.]...C_m[.]$ ($m \in \mathbb{N}$).

The case $m = 0$ is trivial; otherwise the proof follows by induction on the weight of $C_0[M][C_1[M]/y]C_2[M]...C_m[M]$ and $C_1[M]$. □

So the desired result follows.

Theorem 12.2.6. $M \trianglelefteq N$ if and only if $M \preceq_{\mathbf{V}} N$, for all $M, N \in \Lambda^0$.

Proof. By Lemmas 12.2.4 and 12.2.5. □

The next definition overload the meaning of \trianglelefteq on a subset of filters, namely \trianglelefteq induces a preorder on $\mathcal{F}^0(\sqrt{\cdot})$, i.e. the set of filters of $\mathcal{F}(\sqrt{\cdot})$ that are interpretations of closed terms.

Definition 12.2.7. Let $f, g \in \mathcal{F}^0(\sqrt{\cdot})$ and let ρ be an environment.

$f \trianglelefteq g$ if and only if $M, N \in \Lambda^0$ such that $\llbracket M \rrbracket_{\rho}^{\mathcal{F}(\sqrt{\cdot})} = f$ and $\llbracket N \rrbracket_{\rho}^{\mathcal{F}(\sqrt{\cdot})} = g$ imply $M \trianglelefteq N$. Moreover, $f \triangleq g$ if and only if $f \trianglelefteq g$ and $g \trianglelefteq f$.

Note that if M is closed then $\llbracket M \rrbracket_{\rho}^{\mathcal{V}} = \llbracket M \rrbracket_{\rho'}^{\mathcal{V}}$, for all ρ, ρ' ; moreover, if M, N are closed then $\llbracket M \rrbracket_{\rho}^{\mathcal{V}} = \llbracket N \rrbracket_{\rho}^{\mathcal{V}}$ implies $M \trianglelefteq N$ and $N \trianglelefteq M$, by Property 12.2.3.(i). Note that \trianglelefteq is overloaded, since it denotes both a relation on Λ^0 and a relation on $\mathcal{F}^0(\vee)$.

Now we can define the new $\lambda\Gamma$ -model.

Definition 12.2.8. Let $f, g \in \mathcal{F}^0(\vee)$.

- (i) $[f]$ is the equivalence class of f with respect to the equivalence relation \trianglelefteq , while $\mathcal{F}_{\trianglelefteq}^0$ is the set of equivalence classes induced from \trianglelefteq on $\mathcal{F}^0(\vee)$.

Moreover, let $\mathcal{I}_{\trianglelefteq}^0 = \{[f] \in \mathcal{F}_{\trianglelefteq}^0 \mid \exists M \in \Lambda^0 \text{ such that } \llbracket M \rrbracket_{\rho}^{\mathcal{F}(\vee)} \in f\}$.

- (ii) $\circ_{\trianglelefteq} : \mathcal{F}_{\trianglelefteq}^0 \times \mathcal{F}_{\trianglelefteq}^0 \rightarrow \mathcal{F}_{\trianglelefteq}^0$ is defined as $[f] \circ_{\trianglelefteq} [g] = [f \circ_{\vee} g]$, for all $[f], [g] \in \mathcal{F}_{\trianglelefteq}^0$.
- (iii) The interpretation function $\llbracket . \rrbracket^{\mathcal{V}\mathcal{V}} : \Lambda \times (\text{Var} \rightarrow \mathcal{I}_{\trianglelefteq}^0) \rightarrow \mathcal{F}_{\trianglelefteq}^0$ is defined as:

$\llbracket M \rrbracket_{\zeta}^{\mathcal{V}\mathcal{V}} = [\llbracket M \rrbracket_{\rho}^{\mathcal{F}(\vee)}]$, where ρ is such that $\rho(x) \in \zeta(x)$ for all $x \in \text{Var}$.

- (iv) Let $\mathcal{V}\mathcal{V}$ be the quadruple: $\langle \mathcal{F}_{\trianglelefteq}^0, \mathcal{I}_{\trianglelefteq}^0, \circ, \llbracket . \rrbracket^{\mathcal{V}\mathcal{V}} \rangle$.

Note that the interpretation is defined for open terms too.

Property 12.2.9. Let $M, N, P, Q \in \Lambda^0$.

If $M \trianglelefteq N$ and $P \trianglelefteq Q$ then $MP \trianglelefteq NQ$.

Proof. Clearly $M \preceq_{\mathcal{V}} N$ and $P \preceq_{\mathcal{V}} Q$ imply $MP \preceq_{\mathcal{V}} NQ$, therefore the proof follows by Theorem 12.2.6. \square

Note that \circ_{\trianglelefteq} is well defined, by using the previous property. Furthermore, it is easy to see that $[f] \in \mathcal{I}_{\trianglelefteq}^0$ and $f' \in [f]$ imply that $f' \in \mathcal{I}(\vee)$.

Lemma 12.2.10. $\mathcal{V}\mathcal{V}$ is a $\lambda\Gamma$ -model.

Proof. We check that $\mathcal{V}\mathcal{V}$ satisfies the conditions of Definition 10.0.1.

If $\zeta \in (\text{Var} \rightarrow \mathcal{I}_{\trianglelefteq}^0)$ then let ρ be such that $\rho(x) \in \zeta(x)$ for all $x \in \text{Var}$.

1. $\llbracket x \rrbracket_{\zeta}^{\mathcal{V}\mathcal{V}} = [\llbracket x \rrbracket_{\rho}^{\mathcal{F}(\vee)}] = [\rho(x)] = \zeta(x)$.
2. $\llbracket MN \rrbracket_{\zeta}^{\mathcal{V}\mathcal{V}} = [\llbracket MN \rrbracket_{\rho}^{\mathcal{F}(\vee)}] = [\llbracket M \rrbracket_{\rho}^{\mathcal{F}(\vee)} \circ_{\vee} \llbracket N \rrbracket_{\rho}^{\mathcal{F}(\vee)}] = [\llbracket M \rrbracket_{\rho}^{\mathcal{F}(\vee)}] \circ_{\trianglelefteq} [\llbracket N \rrbracket_{\rho}^{\mathcal{F}(\vee)}] = \llbracket M \rrbracket_{\zeta}^{\mathcal{V}\mathcal{V}} \circ_{\trianglelefteq} \llbracket N \rrbracket_{\zeta}^{\mathcal{V}\mathcal{V}}$.
3. $\llbracket \lambda x. M \rrbracket_{\zeta}^{\mathcal{V}\mathcal{V}} \circ_{\trianglelefteq} d = [\llbracket \lambda x. M \rrbracket_{\rho}^{\mathcal{F}(\vee)}] \circ_{\trianglelefteq} d = [\llbracket \lambda x. M \rrbracket_{\rho}^{\mathcal{F}(\vee)} \circ_{\vee} f] = [\llbracket M \rrbracket_{\rho[f/x]}^{\mathcal{F}(\vee)}] = \llbracket M \rrbracket_{\zeta[d/x]}^{\mathcal{V}\mathcal{V}}$, for all $d \in \mathcal{I}_{\trianglelefteq}^0$ and $f \in d$.
4. Let $\llbracket M \rrbracket_{\zeta[[d]/x]}^{\mathcal{V}\mathcal{V}} = \llbracket N \rrbracket_{\zeta'[[d']/x']}^{\mathcal{V}\mathcal{V}}$, where $d, d' \in \mathcal{I}^0(\vee)$.
Thus $\llbracket M \rrbracket_{\rho[d/x]}^{\mathcal{F}(\vee)} = \llbracket N \rrbracket_{\rho[d'/x']}^{\mathcal{F}(\vee)}$, therefore $\llbracket \lambda x. M \rrbracket_{\rho} = \llbracket \lambda x'. N \rrbracket_{\rho'}$, and so $\llbracket \lambda x. M \rrbracket_{\zeta}^{\mathcal{V}\mathcal{V}} = \llbracket \lambda x'. N \rrbracket_{\zeta'}^{\mathcal{V}\mathcal{V}}$.
5. Trivial. \square

Since \trianglelefteq is a preorder on $\mathcal{F}^0(\vee)$ then it induces a partial order on $\mathcal{F}_{\trianglelefteq}^0$.

Definition 12.2.11. Let $M \sqsubseteq_{\mathcal{V}\mathcal{V}} N$ denote $\llbracket M \rrbracket_{\zeta}^{\mathcal{V}\mathcal{V}} \trianglelefteq \llbracket N \rrbracket_{\zeta}^{\mathcal{V}\mathcal{V}}$, for all $\zeta \in (\text{Var} \rightarrow \mathcal{I}_{\triangleleft}^0)$. Moreover, let $M \sim_{\mathcal{V}\mathcal{V}} N$ denote $M \sqsubseteq_{\mathcal{V}\mathcal{V}} N$ and $N \sqsubseteq_{\mathcal{V}\mathcal{V}} M$.

Consequently, the model $\mathcal{V}\mathcal{V}$ induces a partial order on the interpretation of terms (not only closed terms).

Lemma 12.2.12. Let $M, N \in \Lambda^0$. $M \sqsubseteq_{\mathcal{V}\mathcal{V}} N$ if and only if $M \trianglelefteq N$.

Proof. Let $\zeta \in (\text{Var} \rightarrow \mathcal{I}_{\triangleleft}^0)$, and let ρ be such that $\rho(x) \in \zeta(x)$ for all $x \in \text{Var}$. $M \sqsubseteq_{\mathcal{V}\mathcal{V}} N$ if and only if $\llbracket M \rrbracket_{\zeta}^{\mathcal{V}\mathcal{V}} \trianglelefteq \llbracket N \rrbracket_{\zeta}^{\mathcal{V}\mathcal{V}}$ if and only if $\llbracket \llbracket M \rrbracket_{\rho}^{\mathcal{F}(\checkmark)} \rrbracket \trianglelefteq \llbracket \llbracket N \rrbracket_{\rho}^{\mathcal{F}(\checkmark)} \rrbracket$ if and only if $\llbracket M \rrbracket_{\rho}^{\mathcal{F}(\checkmark)} \trianglelefteq \llbracket N \rrbracket_{\rho}^{\mathcal{F}(\checkmark)}$ if and only if $M \trianglelefteq N$. \square

The correctness is easy.

Theorem 12.2.13 ($\mathcal{V}\mathcal{V}$ -Correctness).

The model $\mathcal{V}\mathcal{V}$ is correct with respect to the \mathbf{V} -operational semantics.

Proof. $M \sqsubseteq_{\mathcal{V}\mathcal{V}} N$ implies $C[M] \sqsubseteq_{\mathcal{V}\mathcal{V}} C[N]$, for each closing context $C[.]$, by Property 10.0.2.(v). Hence $C[M] \trianglelefteq C[N]$ by Lemma 12.2.12; in particular, $C[M] \trianglelefteq_{\omega \rightarrow \omega} C[N]$, so $B \vdash_{\checkmark} C[M] : \omega \rightarrow \omega$ implies $B \vdash_{\checkmark} C[N] : \omega \rightarrow \omega$, for all bases B . Therefore, if $C[M]$ is Γ -valuable then $C[N]$ is Γ -valuable, by Property 12.1.8, since $C[M]$ and $C[N]$ are closed. Hence $M \preceq_{\mathbf{V}} N$. \square

The following theorem implies the full abstraction of $\mathcal{V}\mathcal{V}$ with respect to the \mathbf{V} -operational semantics.

Theorem 12.2.14 ($\mathcal{V}\mathcal{V}$ -Completeness).

The model $\mathcal{V}\mathcal{V}$ is complete with respect to the \mathbf{V} -operational semantics.

Proof. We will prove $\nexists_{\mathcal{V}\mathcal{V}}$ implies $\nexists_{\mathbf{V}}$.

$M \nexists_{\mathcal{V}\mathcal{V}} N$ means $\llbracket M \rrbracket_{\zeta}^{\mathcal{V}\mathcal{V}} \not\trianglelefteq \llbracket N \rrbracket_{\zeta}^{\mathcal{V}\mathcal{V}}$, for some $\zeta \in (\text{Var} \rightarrow \mathcal{I}_{\triangleleft}^0)$. Since the codomain of ζ is $\mathcal{I}_{\triangleleft}^0$, if $\text{FV}(M) \cup \text{FV}(N) = \{x_1, \dots, x_m\}$ then there are $P_i \in \Gamma^0$ such that $\zeta(x_i) = \llbracket \llbracket P_i \rrbracket_{\rho}^{\mathcal{F}(\checkmark)} \rrbracket$. Thus, let s be such that $s(x_i) = P_i$ ($1 \leq i \leq m$), hence $s(M), s(N) \in \Lambda^0$. By Property 10.0.2.(iv), $\llbracket s(M) \rrbracket_{\zeta'}^{\mathcal{V}\mathcal{V}} \not\trianglelefteq \llbracket s(N) \rrbracket_{\zeta'}^{\mathcal{V}\mathcal{V}}$, for all $\zeta' \in (\text{Var} \rightarrow \mathcal{I}_{\triangleleft}^0)$, so in particular $s(M) \nexists_{\mathcal{V}\mathcal{V}} s(N)$.

By Lemma 12.2.12, $s(M) \not\trianglelefteq s(N)$, so there is a sequence of closed Γ -valuable terms \vec{Q} such that $s(M)\vec{Q} \not\trianglelefteq_{\omega \rightarrow \omega} s(N)\vec{Q}$, by Lemma 12.2.4.

Let $C[.] \equiv (\lambda x_1 \dots x_m . [.]) s(x_1) \dots s(x_m) \vec{Q}$; clearly $C[M], C[N] \in \Lambda^0$, and moreover $C[M] \Downarrow_{\mathbf{V}}$ and $C[N] \Uparrow_{\mathbf{V}}$, so $M \not\preceq_{\mathbf{V}} N$. \square

Corollary 12.2.15. If $M \not\preceq_{\mathbf{V}} N$ then there is a head context separating M and N .

Proof. Immediate, by the proof of Theorem 12.2.14. \square

The technique used here for building the fully abstract model of the \mathbf{V} -operational semantics is similar to that used in [71] and [44], for different calculi. The use of intersection types and filter models allows for the application of such techniques to a wider class of models.

13. Filter $\lambda\Delta$ -Models and Domains

13.1 Domains

There is an analogy between $\lambda\Delta$ -filter models and $\lambda\Delta$ -models that are ω -algebraic lattices, which was first noticed in [28] and further developed in [1] and [3]. This analogy lies in the fact that type symbols in a $\lambda\Delta$ -filter model play the role of names for compact elements in the corresponding ω -algebraic lattice. It is out of the aim of this book to give a complete survey of the $\lambda\Delta$ -models based on ω -algebraic lattices. In case where $\Delta = \Lambda$, there are some textbooks giving a complete development of this topic, e.g. [5, 81, 87]. Here we will just give some basic informations in order to assure readability to those readers who are not expert in this topic, without developing the proofs for standard properties.

Let us recall the definition of a ω -algebraic complete lattice.

- Definition 13.1.1.** (i) A complete lattice $(\mathbb{L}, \sqsubseteq_{\mathbb{L}})$ is a set \mathbb{L} , equipped by a order relation $\sqsubseteq_{\mathbb{L}}$, such that for all $X \subseteq \mathbb{L}$ both $\sqcup X$ (the least upper bound of X) and $\sqcap X$ (the greatest lower bound of X) exist.
(ii) $X \subseteq \mathbb{L}$ is directed if and only if every two elements of X have an upper bound in X .
(iii) $x \in \mathbb{L}$ is compact if and only if every directed $X \subseteq \mathbb{L}$ is such that:

$$x \sqsubseteq_{\mathbb{L}} \sqcup X \text{ implies } x \sqsubseteq_{\mathbb{L}} y \text{ for some } y \in X.$$

Let $\mathbf{comp}(\mathbb{L})$ be the set of compact elements of \mathbb{L} .

- (iv) \mathbb{L} is ω -algebraic if and only if $x = \sqcup\{y \sqsubseteq x \mid y \text{ compact}\}$ and $\mathbf{comp}(\mathbb{L})$ is countable.

Let us use the word *domain* in order to denote a ω -algebraic complete lattice. It is easy to see that in a domain there is always a bottom (minimum) element, that as usual we denote by \perp .

- Definition 13.1.2.** (i) A function $h : \mathbb{L} \rightarrow \mathbb{L}'$ is monotone if and only if:

$$x \sqsubseteq_{\mathbb{L}} y \text{ implies } h(x) \sqsubseteq_{\mathbb{L}'} h(y).$$

- (ii) A function $h : \mathbb{L} \rightarrow \mathbb{L}'$ is continuous if and only if it is monotone, and moreover:

$$h(\sqcup X) = \sqcup\{h(x) \mid x \in X\} \text{ for all sets } X \subseteq \mathbb{L}.$$

(iii) A continuous function $h : \mathbb{L} \rightarrow \mathbb{L}'$ is strict if and only if:

$$h(\perp_{\mathbb{L}}) =_{\mathbb{L}'} \perp_{\mathbb{L}'}.$$

(iv) The pointwise order between two continuous functions $h, k : \mathbb{L} \rightarrow \mathbb{L}'$ is defined in the following way:

$$h \sqsubseteq_{\mathbb{L} \rightarrow \mathbb{L}'} k \text{ if and only if } \forall x \in \mathbb{L}. h(x) \sqsubseteq_{\mathbb{L}'} k(x).$$

(v) Two domains $(\mathbb{L}, \sqsubseteq_{\mathbb{L}})$ and $(\mathbb{L}', \sqsubseteq_{\mathbb{L}'})$ are isomorphic if and only if there are two continuous functions $h : \mathbb{L} \rightarrow \mathbb{L}'$ and $k : \mathbb{L}' \rightarrow \mathbb{L}$ such that:
 - $h \circ k = id_{\mathbb{L}'}$,
 - $k \circ h = id_{\mathbb{L}}$,
 where $id_{\mathbb{L}}$ and $id_{\mathbb{L}'}$ denote the identity function respectively on \mathbb{L} and \mathbb{L}' .

The notion of step function will play a key role in the construction of the isomorphism between filter spaces and domains.

Definition 13.1.3. (i) Let $a \in \mathbb{L}$ and $b \in \mathbb{L}'$.

The step function $s_{a,b} : \mathbb{L} \rightarrow \mathbb{L}'$ is defined as

$$\lambda x : \mathbb{L}. \text{ if } a \sqsubseteq_{\mathbb{L}} x \text{ then } b \text{ else } \perp_{\mathbb{L}'},$$

where λ denotes the metatheoretic abstraction.

- (ii) A step function $s_{a,b}$ is strict if and only if $s_{a,b}(\perp_{\mathbb{L}}) = \perp_{\mathbb{L}'}$.
 (iii) The partial order between step functions from \mathbb{L} to \mathbb{L}' is defined as follows:

$$s_{a,b} \sqsubseteq_{\mathbb{L} \rightarrow \mathbb{L}'} s_{c,d} \text{ if and only if } c \sqsubseteq_{\mathbb{L}} a \text{ and } b \sqsubseteq_{\mathbb{L}'} d.$$

Let $[\mathbb{L} \rightarrow \mathbb{L}'] = \{f \mid f : \mathbb{L} \rightarrow \mathbb{L}' \text{ is continuous}\}$, and let $[\mathbb{L} \rightarrow_{\perp} \mathbb{L}'] = \{f \mid f : \mathbb{L} \rightarrow \mathbb{L}' \text{ is continuous and strict}\}$. The following result holds.

Lemma 13.1.4. Let \mathbb{L} and \mathbb{L}' be domains.

- (i) $([\mathbb{L} \rightarrow \mathbb{L}'], \sqsubseteq_{\mathbb{L} \rightarrow \mathbb{L}'})$ is a domain whose compact elements are least upper bounds of finite sets of step functions.
 (ii) $([\mathbb{L} \rightarrow_{\perp} \mathbb{L}'], \sqsubseteq_{\mathbb{L} \rightarrow \mathbb{L}'})$ is a domain whose compact elements are least upper bounds of finite sets of strict step functions.

Proof. Define $\sqcup\{f, g\}(x) = \sqcup\{f(x), g(x)\}$ and $\sqcap\{f, g\}(x) = \sqcap\{f(x), g(x)\}$; then both $([\mathbb{L} \rightarrow \mathbb{L}'], \sqsubseteq_{\mathbb{L} \rightarrow \mathbb{L}'})$ and $([\mathbb{L} \rightarrow_{\perp} \mathbb{L}'], \sqsubseteq_{\mathbb{L} \rightarrow \mathbb{L}'})$ are complete lattices since $(\mathbb{L}', \sqsubseteq_{\mathbb{L}'})$ is a complete lattice.

The fact that both constructions give rise to an ω -algebraic lattice is an obvious consequence of the fact that both \mathbb{L} and \mathbb{L}' are ω -algebraic.

Moreover, note that if f is a continuous function from \mathbb{L} to \mathbb{L}' , such that $f(a) = b$, then $s_{a,b} \sqsubseteq_{\mathbb{L} \rightarrow \mathbb{L}'} f$. Then $f = \sqcup\{s_{a,b} \mid f(a) = b\}$. \square

A further operation on domains that will be useful is the *lifting*. Let $(\mathbb{L}, \sqsubseteq_{\mathbb{L}})$ be a domain and let $\mathbb{L}_{\perp} = \mathbb{L} \cup \{\perp\}$, where \perp is a fresh element not belonging to \mathbb{L} . Moreover, let $a \sqsubseteq_{\mathbb{L}_{\perp}} b$ if and only if either $a = \perp$ or $a \sqsubseteq_{\mathbb{L}} b$. The following lemma holds.

Lemma 13.1.5. *If $(\mathbb{L}, \sqsubseteq_{\mathbb{L}})$ is a domain then $(\mathbb{L}_{\perp}, \sqsubseteq_{\mathbb{L}_{\perp}})$ is a domain (the lifting of $(\mathbb{L}, \sqsubseteq_{\mathbb{L}})$).*

Proof. Easy. □

Let us call *domain constructor* an operation on domains. We will consider in this section a restricted set of domain constructors, namely

$$\mathcal{C} = \{[.\rightarrow .], [.\rightarrow_{\perp} .], (.)_{\perp}\}.$$

Let \mathbf{c} denote an element of \mathcal{C} . We will use all constructors in \mathcal{C} as being unary. It is possible to compose domain constructors, in order to obtain further domain constructors.

Property 13.1.6. Let $(\mathbb{L}, \sqsubseteq_{\mathbb{L}})$ be a domain.

If $\mathbf{c}_1 \in \{[.\rightarrow .], [.\rightarrow_{\perp} .]\}$ and $\mathbf{c}_2 \equiv (.)_{\perp}$ then $(\mathbf{c}_2(\mathbf{c}_1(\mathbb{L})), \sqsubseteq_{\mathbf{c}_2(\mathbf{c}_1(\mathbb{L}))})$ is a domain.

Proof. Easy. □

Definition 13.1.7. *Let $(\mathbb{L}, \sqsubseteq_{\mathbb{L}})$ and $(\mathbb{L}', \sqsubseteq_{\mathbb{L}'})$ be domains. A retraction pair is a pair of continuous functions $(i : \mathbb{L} \rightarrow \mathbb{L}', j : \mathbb{L}' \rightarrow \mathbb{L})$ such that:*

- $j \circ i = id_{\mathbb{L}}$,
- $i \circ j \sqsubseteq id_{\mathbb{L}'}$.

If (i, j) is a retraction pair from \mathbb{L} to \mathbb{L}' , i is called the embedding and j is called the projection.

Recalling the notion of isomorphism between domains, given in Definition 13.1.2.(v), if there is a retraction pair from \mathbb{L} to \mathbb{L}' , then sometimes \mathbb{L} is called a *subdomain* of \mathbb{L}' .

- Property 13.1.8.* (i) Let (i_1, j_1) be a retraction pair from \mathbb{L} to \mathbb{L}' and (i_2, j_2) be a retraction pair from \mathbb{L}' to \mathbb{L}'' . Then $(i_2 \circ i_1, j_1 \circ j_2)$ is a retraction pair from \mathbb{L} to \mathbb{L}'' .
(ii) An embedding (projection) function has a unique corresponding projection (embedding).
(iii) If (i, j) is a retraction pair from \mathbb{L} to \mathbb{L}' then both i and j are strict.

Domain constructors can be extended to retraction pairs. Let us show how the extension can be made in the particular cases we are interested in.

- Let (i, j) be a retraction pair between \mathbb{L} and $\mathbf{c}(\mathbb{L})$, where either $\mathbf{c} = [.\rightarrow.]$ or $\mathbf{c} = [.\rightarrow_{\perp}.]$. Let

- $\mathbf{c}(i) = \lambda x : \mathbf{c}(\mathbb{L}). i \circ x \circ j,$
- $\mathbf{c}(j) = \lambda x : \mathbf{c}^2(\mathbb{L}). j \circ x \circ i.$

It is easy to check that $(\mathbf{c}(i), \mathbf{c}(j))$ is a retraction pair between $\mathbf{c}(\mathbb{L})$ and $\mathbf{c}^2(\mathbb{L})$.

- In case of lifting, let (i, j) be a retraction pair between \mathbb{L} and $\mathbf{c}(\mathbb{L})$, where $\mathbf{c} = (.)_\perp$. Let
 - $\mathbf{c}(i) = \lambda x : \mathbf{c}(\mathbb{L}). \text{ if } x = \perp_{\mathbf{c}(\mathbb{L})} \text{ then } \perp_{\mathbf{c}^2(\mathbb{L})} \text{ else } i(x),$
 - $\mathbf{c}(j) = \lambda x : \mathbf{c}^2(\mathbb{L}). \text{ if } x = \perp_{\mathbf{c}^2(\mathbb{L})} \text{ then } \perp_{\mathbf{c}(\mathbb{L})} \text{ else } j(x).$
 Then $(\mathbf{c}(i), \mathbf{c}(j))$ is a retraction pair between $\mathbf{c}(\mathbb{L})$ and $\mathbf{c}^2(\mathbb{L})$.

In case \mathbf{c} is a compound domain constructor, the extension of \mathbf{c} to retraction pairs can be made starting from the previous defined extension and then using Property 13.1.8.(i).

Definition 13.1.9. Let $\mathbb{L}_0, \mathbb{L}_1, \dots, \mathbb{L}_n, \dots$ be domains.

- (i) A retraction sequence is a pair whose first component is the set

$$\{\mathbb{L}_i \mid i \geq 0\},$$

and whose second component is the set

$$\{(i_i, j_i) \mid (i_i, j_i) \text{ is a retraction pair from } \mathbb{L}_i \text{ to } \mathbb{L}_{i+1}, i \geq 0\}.$$

- (ii) The inverse limit of a retraction sequence is the set

$$\mathbb{L}_\infty = \{(a_0, a_1, \dots, a_n, \dots) \mid n \geq 0, a_n \in \mathbb{L}_n, a_n = j_n(a_{n+1})\},$$

partially ordered by the relation $\sqsubseteq_{\mathbb{L}_\infty}$, defined as follows:

$$(a_0, a_1, \dots, a_n, \dots) \sqsubseteq_{\mathbb{L}_\infty} (b_0, b_1, \dots, b_n, \dots) \text{ if and only if } a_n \sqsubseteq_{\mathbb{L}_n} b_n, \text{ for all } n.$$

The following property holds.

Property 13.1.10. The inverse limit \mathbb{L}_∞ of retraction sequence is a domain.

A domain equation is an equation of the shape

$$\mathbb{X} = \mathbf{c}(\mathbb{X}),$$

where \mathbf{c} is a domain constructor, and $=$ denotes the isomorphism between domains.

Theorem 13.1.11. Let $(\mathbb{L}, \sqsubseteq_{\mathbb{L}})$ be a domain, and let (i, j) be a retraction pair between \mathbb{L} and $\mathbf{c}(\mathbb{L})$. The inverse limit

$$\mathbb{L}_\infty = \{(a_0, a_1, \dots, a_n, \dots) \mid n \geq 0, a_n \in \mathbf{c}^n(\mathbb{L}), a_n = \mathbf{c}^n(j)(a_{n+1})\}$$

is a solution of the domain equation $\mathbb{X} = \mathbf{c}(\mathbb{X})$, i.e. $\mathbb{L}_\infty = \mathbf{c}(\mathbb{L}_\infty)$.

Proof. Let $\mathbf{c}^0(x) = x$ and $\mathbf{c}^{n+1}(x) = \mathbf{c}(\mathbf{c}^n(x))$ for all $n \in \mathbb{N}$.

$$\mathbf{c}^0(\mathbb{L}) \xrightarrow{i} \mathbf{c}^1(\mathbb{L}) \xleftarrow{j} \cdots \xleftarrow{\mathbf{c}^{k-1}(i) \circ \dots \circ \mathbf{c}^1(i)} \mathbf{c}^k(\mathbb{L}) \xleftarrow{\mathbf{c}^k(i)} \mathbf{c}^{k+1}(\mathbb{L}) \xleftarrow{\mathbf{c}^k(j)} \cdots$$

Let $r_{m,n} : \mathbf{c}^m(\mathbb{L}) \rightarrow \mathbf{c}^n(\mathbb{L})$ be the following function:

$$r_{m,n} = \begin{cases} \text{id}_{\mathbf{c}^m(\mathbb{L})} & \text{if } m = n, \\ \mathbf{c}^{n-1}(i) \circ \dots \circ \mathbf{c}^m(i) & \text{if } m < n, \\ \mathbf{c}^n(j) \circ \dots \circ \mathbf{c}^{m-1}(j) & \text{if } n < m \end{cases}$$

It is easy to check that if $m \leq n$ then $(r_{m,n}, r_{n,m})$ is a retraction pair between $\mathbf{c}^m(\mathbb{L})$ and $\mathbf{c}^n(\mathbb{L})$, by Property 13.1.8.(i). Let

- $i_{n,\infty} : \mathbf{c}^n(\mathbb{L}) \rightarrow \mathbb{L}_\infty$ be $\lambda x : \mathbf{c}^n(\mathbb{L}). (r_{n,0}(x), r_{n,1}(x), \dots, r_{n,n}(x), r_{n,n+1}(x), \dots)$;
- $j_{n,\infty} : \mathbb{L}_\infty \rightarrow \mathbf{c}^n(\mathbb{L})$ be $\lambda x : \mathbb{L}_\infty. (x)_n$,
where $(.)_n$ denotes the n -th element of a sequence;
- $I : \mathbb{L}_\infty \rightarrow \mathbf{c}(\mathbb{L}_\infty)$ be $\bigsqcup_{(n \geq 0)} (i_{n+1,\infty} \circ r_{n,n+1} \circ j_{n,\infty})$;
- $J : \mathbf{c}(\mathbb{L}_\infty) \rightarrow \mathbb{L}_\infty$ be $\bigsqcup_{(n \geq 0)} (i_{n,\infty} \circ r_{n+1,n} \circ j_{n+1,\infty})$.

Then (I, J) is a retraction pair between \mathbb{L}_∞ and $\mathbf{c}(\mathbb{L}_\infty)$, such that $I \circ J = \text{id}_{\mathbf{c}(\mathbb{L}_\infty)}$, so the two domains are isomorphic. \square

By Property 13.1.8.(ii), the solution of a domain equation $\mathbb{X} = \mathbf{c}(\mathbb{X})$ is completely determined by the initial domain $(\mathbb{L}, \sqsubseteq_{\mathbb{L}})$ and the embedding function i between \mathbb{L} and $\mathbf{c}(\mathbb{L})$.

Definition 13.1.12. A solution of a domain equation is minimal, if the initial domain $(\mathbb{L}, \sqsubseteq_{\mathbb{L}})$ is isomorphic to the domain $(\{\perp\}, \text{id}_{\{\perp\}})$.

Now we have all the ingredients in order to show the correspondence between a $\lambda\Delta$ -model that is an inverse limit solution of a domain equation of a given shape and a filter model.

Let us assume that the initial domain \mathbb{L} always has a finite number of elements, which implies that all the elements of \mathbb{L} are compact. Let us define the following procedure, in order to build from \mathbb{L}_∞ the filter space $\mathcal{F}(\mathbb{L}_\infty)$.

Let \mathbb{L}_∞ be a solution of the domain equation $\mathbb{X} = \mathbf{c}(\mathbb{X})$, where $\mathbf{c} \in \{[. \rightarrow .], [. \rightarrow .]_\perp, [. \rightarrow_\perp .]_\perp\}$, starting from the initial domain $(\mathbb{L}, \sqsubseteq_{\mathbb{L}})$ and from the embedding function i between \mathbb{L} and $\mathbf{c}(\mathbb{L})$. Note that by the particular set of constructors we chosen and by the fact that domain equations are defined modulo isomorphisms, by Lemma 13.1.4 we can consider the compact elements of \mathbb{L}_∞ to be either $\perp_{\mathbb{L}_\infty}$ or least upper bounds of finite sets of step functions from \mathbb{L}_∞ to \mathbb{L}_∞ . Moreover, $\mathbf{comp}(\mathbb{L}_\infty) = \bigsqcup_{n \geq 0} i_{n,\infty}(\mathbf{comp}(\mathbf{c}^n(\mathbb{L})))$. The set of type constants $C_{\mathbb{L}}$ and the inclusion relation $\leq_{\nabla_{\mathbb{L}}}$ can be built according to the following procedure **compact-as-types**(.).

Procedure compact-as-types(\mathbb{L}_∞)

1. Choose a set of type constants $C_{\mathbb{L}}$ such that there is a bijection $(.)^+$ between $C_{\mathbb{L}}$ and the compact elements of \mathbb{L} , such that $(\omega)^+ = \perp_{\mathbb{L}}$.
2. Define an intersection relation $\leq_{\nabla_{\mathbb{L}}}$ such that $\sigma \leq_{\nabla_{\mathbb{L}}} \tau$ if and only if $(\tau)^+ \sqsubseteq_{\mathbb{L}} (\sigma)^+$, for all $\sigma, \tau \in C_{\mathbb{L}}$.
3. Let $T(C_{\mathbb{L}})$ be the set of types built from the set of constants $C_{\mathbb{L}}$.
Let $(.)^*$ be the function from $T(C_{\mathbb{L}})$ to $\text{comp}(\mathbb{L}_\infty)$ defined as follows.
 - (3.1) If $\sigma \in C_{\mathbb{L}}$ then $(\sigma)^* = i_{0,\infty}((\sigma)^+)$.
 - (3.2) If $\mathbf{c} \in \{[.\rightarrow .], [.\rightarrow .]_\perp, \}$ then $(\sigma \rightarrow \tau)^* = s_{(\sigma)^*, (\tau)^*}$.
Otherwise, in case $\mathbf{c} = [.\rightarrow_\perp .]_\perp$, if $(\sigma)^* \neq \perp_{\mathbb{L}_\infty}$ then $(\sigma \rightarrow \tau)^* = s_{(\sigma)^*, (\tau)^*}$, while if $(\sigma)^* = \perp_{\mathbb{L}_\infty}$ then $(\sigma \rightarrow \tau)^* = s_{a, (\tau)^*}$ where $a = s_{\perp_{\mathbb{L}_\infty}, \perp_{\mathbb{L}_\infty}}$.
 - (3.3) $(\sigma \wedge \tau)^* = (\sigma)^* \sqcup (\tau)^*$.
4. Extend the intersection relation $\leq_{\nabla_{\mathbb{L}}}$ as follows.
 - (4.1) If $\sigma, \mu_i, \nu_i \in C_{\mathbb{L}}$ ($1 \leq i \leq n$) and $i((\sigma)^+) = \bigsqcup_{1 \leq i \leq n} s_{(\mu_i)^+, (\nu_i)^+}$ where $n \in \mathbb{N}$, then both $\sigma \leq_{\nabla_{\mathbb{L}}} (\mu_1 \rightarrow \nu_1) \wedge \dots \wedge (\mu_n \rightarrow \nu_n)$ and $(\mu_1 \rightarrow \nu_1) \wedge \dots \wedge (\mu_n \rightarrow \nu_n) \leq_{\nabla_{\mathbb{L}}} \sigma$.
 - (4.2) If $\mathbf{c} = [.\rightarrow_\perp .]_\perp$ then $(\omega \rightarrow \omega) \rightarrow \tau \leq_{\nabla_{\mathbb{L}}} \omega \rightarrow \tau$,

The definition of the mapping $(.)^*$ need some comments. Point (3.2) maps every arrow type into a step function. Note that if $\mathbf{c} = [.\rightarrow_\perp .]_\perp$, by definition of strict function, $s_{a,b} \in \mathbb{L}_\infty$ implies either $a \neq \perp$ or $a, b = \perp$. So types of the shape $\omega \rightarrow \tau$, where $(\tau)^* \neq \perp$, are in some sense redundant, and they are mapped into a step function that is the maximum one less to the step function $s_{\perp, (\tau)^*}$.

Point (4.1) takes into account the initial retraction pair. Point (4.2) reflects point (3.2). Moreover, if $C_{\mathbb{L}}$ is finite (so \mathbb{L} and $\mathbf{c}(\mathbb{L})$ are finite) then the number of rules to be joined to the intersection relation $\leq_{\nabla_{\mathbb{L}}}$ can be transformed in a finite number.

The following lemma proves that the procedure is correct, in the sense that $(.)^*$ is a surjection, and the inclusion relation between types respects the order relation of the domain.

Lemma 13.1.13. *Let \mathbb{L}_∞ be a solution of the domain equation $\mathbb{X} = \mathbf{c}(\mathbb{X})$, where $\mathbf{c} \in \{[.\rightarrow .], [.\rightarrow .]_\perp, [.\rightarrow_\perp .]_\perp\}$, starting from the initial domain $(\mathbb{L}, \sqsubseteq_{\mathbb{L}})$ and from the embedding function i between \mathbb{L} and $\mathbf{c}(\mathbb{L})$.*

- (i) $(.)^* : T(C_{\mathbb{L}}) \rightarrow \text{comp}(\mathbb{L}_\infty)$ is a surjection.
- (ii) For all $\sigma, \tau \in T(C_{\mathbb{L}})$, $\sigma \leq_{\nabla_{\mathbb{L}}} \tau$ if and only if $(\tau)^* \sqsubseteq_{\mathbb{L}_\infty} (\sigma)^*$.

Proof. (i) Let $(\mathbb{L}_\infty, \sqsubseteq_{\mathbb{L}_\infty})$ be the solution of the given domain equation.

The set $\text{comp}(\mathbb{L}_\infty)$ can be viewed as $\bigsqcup_{n \geq 0} i_{n,\infty}(\text{comp}(\mathbf{c}^n(\mathbb{L})))$. We will prove that, if $a \in i_{n,\infty}(\text{comp}(\mathbf{c}^n(\mathbb{L})))$ then there is σ such that $(\sigma)^* = a$, by induction on n . If $n = 0$, then the proof follows by construction.

Let $n \geq 1$. Since the set of constructors \mathbf{c} we are taking into consideration

is restricted, a compact element d of $\mathbf{c}^n(\mathbb{L})$ meets one of the following constraints.

- d is the bottom, then $\perp = (\omega)^*$, by construction.
- d is a step function $s_{a,b} : \mathbf{c}^{n-1}(\mathbb{L}) \rightarrow \mathbf{c}^{n-1}(\mathbb{L})$, where $a, b \in \mathbf{c}^{n-1}(\mathbb{L})$. Note that $\mathbf{c}^{n-1}(\mathbb{L})$ has a finite number of elements, for all $n \geq 1$; hence, if $a \in \mathbf{c}^{n-1}(\mathbb{L})$ then $i_{n-1,\infty}(a)$ is compact. So, by induction, there are σ, τ such that $i_{n-1,\infty}(a) = (\sigma)^*$ and $i_{n-1,\infty}(b) = (\tau)^*$, and therefore $s_{i_{n-1,\infty}(a), i_{n-1,\infty}(b)} = (\sigma \rightarrow \tau)^*$.
- d is the least upper bound of a finite set of step functions, namely $\sqcup\{s_{a_i,b_i} \mid 1 \leq i \leq m\}$; then by induction there are σ_i, τ_i such that $i_{n-1,\infty}(a_i) = (\sigma_i)^*$ and $i_{n-1,\infty}(b_i) = (\tau_i)^*$, and

$$\sqcup\{s_{a_i,b_i} \mid 1 \leq i \leq m\} = ((\sigma_1 \rightarrow \tau_1) \wedge \dots \wedge (\sigma_m \rightarrow \tau_m))^*.$$

(ii) (\Rightarrow) By induction on the definition of \leq_{∇_L} , taking into account the last used rule.

- (a) Then the proof follows since $\perp \sqsubseteq_{\mathbb{L}_\infty} a$, for all a , and since $(\omega)^* = \perp$.
- (b), (c), (c'), (e) By the definition of least upper bound.
- (f), (d) By the Definition 13.1.3.iii.
- (g) By the fact that $(\omega \rightarrow \omega)^* = \mathbb{A}x.\perp$, that is the smallest step function.
- (r) Obvious.
- (t) By induction.

Let $\sigma \leq_{\nabla_L} (\mu_1 \rightarrow \nu_1) \wedge \dots \wedge (\mu_n \rightarrow \nu_n)$ be the conclusion of a rule added by the point (4.1) of the procedure compact-as-types. Clearly also the rule $(\mu_1 \rightarrow \nu_1) \wedge \dots \wedge (\mu_n \rightarrow \nu_n) \leq_{\nabla_L} \sigma$ was added by construction, i.e. $\sigma \simeq_{\nabla_L} (\mu_1 \rightarrow \nu_1) \wedge \dots \wedge (\mu_n \rightarrow \nu_n)$. Hence $(\sigma)^*$ and $\sqcup_{1 \leq i \leq n} s_{(\mu_i)^*, (\nu_i)^*}$ denote the same compact element.

The case $(\mu_1 \rightarrow \nu_1) \wedge \dots \wedge (\mu_n \rightarrow \nu_n) \leq_{\nabla_L} \sigma$ is symmetric.

In both cases, $\sigma, \mu_i, \nu_i \in C_{\mathbb{L}}$ ($1 \leq i \leq n$), $i((\sigma)^+) = \sqcup_{1 \leq i \leq n} s_{(\mu_i)^+, (\nu_i)^+}$ and $(\sigma)^+ = j(\sqcup_{1 \leq i \leq n} s_{(\mu_i)^+, (\nu_i)^+})$.

Let $(\omega \rightarrow \omega) \rightarrow \tau \leq_{\nabla_L} \omega \rightarrow \tau$ be the conclusion of a rule added by the point (4.2) of the procedure compact-as-types. Hence $\mathbf{c} = [\cdot \rightarrow \perp]_{\perp}$ and if $(\tau)^* \neq (\omega)^*$ then $s_{(\omega)^*, (\tau)^*}$ does not belong to \mathbb{L}_∞ . In such case, by the point (3.2) of the procedure compact-as-types,

$$((\omega \rightarrow \omega) \rightarrow \tau)^* = (\omega \rightarrow \tau)^*.$$

(\Leftarrow) Let $(\tau)^* \sqsubseteq_{\mathbb{L}_\infty} (\sigma)^*$. The proof is given by induction on the total number of symbols of σ and τ .

If $(\tau)^* = \perp$ then the proof is trivial, since $\perp = (\omega)^*$, and ω is the biggest type. If τ or σ are type constants then the proof follows by construction.

If $\sigma \equiv \sigma_1 \rightarrow \sigma_2$ and $\tau \equiv \tau_1 \rightarrow \tau_2$, then by the Definition 13.1.3.(iii) and the definition of $(\cdot)^*$, $(\tau_2)^* \sqsubseteq_{\mathbb{L}_\infty} (\sigma_2)^*$ and $(\sigma_1)^* \sqsubseteq_{\mathbb{L}_\infty} (\tau_1)^*$. Thus by induction $\tau_1 \leq_{\nabla_L} \sigma_1$ and $\sigma_2 \leq_{\nabla_L} \tau_2$, and the proof follows by rule (f) of Definition 10.1.1.(ii).

If $\tau \equiv \tau_1 \rightarrow \tau_2$ and $(\sigma)^* \equiv \sqcup\{(\mu_i \rightarrow \nu_i)^* \mid 1 \leq i \leq m\}$, then there is $\{i_1, \dots, i_k\} \subseteq \{1, \dots, m\}$ such that $(\mu_{i_1} \wedge \dots \wedge \mu_{i_k})^* \sqsubseteq_{\mathbb{L}_\infty} (\tau_1)^*$ and $(\tau_2)^* \sqsubseteq_{\mathbb{L}_\infty} (\nu_{i_1} \wedge \dots \wedge \nu_{i_k})^*$, and so $(\tau_1 \rightarrow \tau_2)^* \sqsubseteq_{\mathbb{L}_\infty} ((\mu_{i_1} \wedge \dots \wedge \mu_{i_k}) \rightarrow (\nu_{i_1} \wedge \dots \wedge \nu_{i_k}))^*$. So, by induction, $\tau_1 \leq_{\nabla_L} \mu_{i_1} \wedge \dots \wedge \mu_{i_k}$ and $\nu_{i_1} \wedge \dots \wedge \nu_{i_k} \leq_{\nabla_L} \tau_2$, and, by rule (f) of Definition 10.1.1.(ii), $\tau_1 \rightarrow \tau_2 \geq_{\nabla_L} (\mu_{i_1} \wedge \dots \wedge \mu_{i_k}) \rightarrow \nu_{i_1} \wedge \dots \wedge \nu_{i_k} \simeq_{\nabla_L} ((\mu_{i_1} \wedge \dots \wedge \mu_{i_k}) \rightarrow \nu_{i_1}) \wedge \dots \wedge (\mu_{i_1} \wedge \dots \wedge \mu_{i_k}) \rightarrow \nu_{i_k} \geq_{\nabla_L} (\mu_{i_1} \rightarrow \nu_{i_1}) \wedge \dots \wedge (\mu_{i_k} \rightarrow \nu_{i_k}) \geq_{\nabla_L} \sigma$, by applying respectively Lemma 10.1.4.(iii), and the rules (c) and (f) of Definition 10.1.1.(ii).

If $(\tau)^* \equiv \sqcup\{(\mu_i \rightarrow \nu_i)^* \mid 1 \leq i \leq m\}$, then $(\mu_i \rightarrow \nu_i)^* \sqsubseteq_{\mathbb{L}_\infty} (\sigma)^*$, for all i ($1 \leq i \leq m$), and the proof follows the same lines as the previous point. \square

Note that the proof of part (\Leftarrow) of the Lemma 13.1.13 gives a justification of the legality condition (Definition 10.1.5) on a type system in order to induce a $\lambda\Delta$ -model. In fact, this condition reflects a semantic property on step functions.

Theorem 13.1.14. *Let \mathbb{L}_∞ be a solution of the domain equation $\mathbb{X} = \mathbf{c}(\mathbb{X})$, where $\mathbf{c} \in \{[\cdot \rightarrow \cdot], [\cdot \rightarrow \cdot]_\perp, [\cdot \rightarrow_\perp \cdot]_\perp\}$, starting from the initial domain $(\mathbb{L}, \sqsubseteq_{\mathbb{L}})$ and from the embedding function i between \mathbb{L} and $\mathbf{c}(\mathbb{L})$.*

*Let ∇ be a type system such that $C(\mathbb{L})$ and \leq_{∇_L} are built according to the procedure **compact-as-types** (\mathbb{L}_∞) .*

Then \mathbb{L}_∞ is isomorphic to the space of filters $\mathcal{F}(\nabla_L)$, ordered by set inclusion.

Proof. Let us define the functions $h : \mathbb{L}_\infty \rightarrow \mathcal{F}(\nabla_L)$ and $k : \mathcal{F}(\nabla_L) \rightarrow \mathbb{L}_\infty$ in the following way:

- $h(a) = \uparrow \{\sigma \mid (\sigma)^* \sqsubseteq a\}$,
- $k(f) = \sqcup\{(\sigma)^* \mid \sigma \in f\}$.

By Lemma 13.1.13, h and k realize the desired isomorphism. \square

Now we are ready to prove that each of the filter models we presented is isomorphic to a model built as the inverse limit solution of a domain equation.

13.1.1 \mathcal{H} as Domain

Take the domain equation:

$$\mathbb{X} = [\mathbb{X} \rightarrow \mathbb{X}].$$

Note that the minimal solution of this equation is the domain $(\{\perp\}, id_{\{\perp\}})$, since there is just one continuous function from $\{\perp\}$ to $\{\perp\}$, namely the function $\lambda x. \perp$. So take as initial domain $(\mathbb{I}, \sqsubseteq_{\mathbb{I}})$, where $\mathbb{I} = \{\perp_{\mathbb{I}}, \top_{\mathbb{I}}\}$, and $\sqsubseteq_{\mathbb{I}}$ is defined as $\perp_{\mathbb{I}} \sqsubseteq_{\mathbb{I}} \top_{\mathbb{I}}$. Take the domain $([\mathbb{I} \rightarrow \mathbb{I}], \sqsubseteq_{\mathbb{I} \rightarrow \mathbb{I}})$, which is a domain by Lemma 13.1.4.(i), and choose, as embedding function between $(\mathbb{I}, \sqsubseteq_{\mathbb{I}})$ and $([\mathbb{I} \rightarrow \mathbb{I}], \sqsubseteq_{\mathbb{I} \rightarrow \mathbb{I}})$, the function i so defined:

- $i(\perp_{\mathbb{I}}) = \perp_{[\mathbb{I} \rightarrow \mathbb{I}]}$,
- $i(\top_{\mathbb{I}}) = s_{\perp_{\mathbb{I}}, \top_{\mathbb{I}}}$,

and let \mathbb{I}_{∞} be the inverse limit solution so obtained.

The correspondence between C_{∞} and $\text{comp}(\mathbb{I})$ is defined in the following way:

- $(\omega)^+ = \perp_{\mathbb{I}}$,
- $(\phi)^+ = \top_{\mathbb{I}}$.

Let the function $(.)^* : T(C_{\infty}) \rightarrow \text{comp}(\mathbb{I}_{\infty})$ be the function made according to the procedure **compact-as-types**.

The procedure **compact-as-types**(\mathbb{I}_{∞}) generates, at point (4.1), the rules (h1), (h2) and (h3) of Fig. 11.1 (pag. 120) of the intersection relation \leq_{∞} . Note that point (4.2) is not applied in this case.

Let $\rho : \text{Var} \rightarrow \mathbb{I}_{\infty}$; the interpretation function $\llbracket . \rrbracket^{\mathbb{I}_{\infty}}$ (see [67]) is:

- $\llbracket x \rrbracket_{\rho}^{\mathbb{I}_{\infty}} = \rho(x)$,
- $\llbracket MN \rrbracket_{\rho}^{\mathbb{I}_{\infty}} = I(\llbracket M \rrbracket_{\rho}^{\mathbb{I}_{\infty}})(\llbracket N \rrbracket_{\rho}^{\mathbb{I}_{\infty}})$,
- $\llbracket \lambda x. M \rrbracket_{\rho}^{\mathbb{I}_{\infty}} = J(\mathbb{A}d. \llbracket M \rrbracket_{\rho[d/x]}^{\mathbb{I}_{\infty}})$,

where I and J are defined in the proof of Theorem 13.1.11.

Theorem 13.1.15. *Let $\mathcal{I} = < \mathbb{I}_{\infty}, \mathbb{I}_{\infty}, \circ_{\mathbb{I}_{\infty}}, \llbracket . \rrbracket^{\mathbb{I}_{\infty}} >$ where $\circ_{\mathbb{I}_{\infty}} \equiv \mathbb{A}xy. I(x)(y)$. \mathcal{I} is a $\lambda\Delta$ -calculus model, and it is isomorphic to \mathcal{H} .*

Proof. It is easy to check that \mathcal{I} satisfies the conditions of Definition 10.0.1, so it is a $\lambda\Delta$ -model. By Theorem 13.1.14, $(\mathbb{I}_{\infty}, \sqsubseteq_{\mathbb{I}_{\infty}})$ is isomorphic to $\mathcal{F}(\infty)$, ordered by set inclusion. Now we will prove that, if $\zeta : \text{Var} \rightarrow \mathbb{I}_{\infty}$ and $\rho : \text{Var} \rightarrow \mathcal{F}(\infty)$ are such that $\forall x \in \text{Var}$, $\zeta(x)$ and $\rho(x)$ are isomorphic elements, then $\forall M \in \Lambda \llbracket M \rrbracket_{\zeta}^{\mathbb{I}_{\infty}}$ is isomorphic to $\llbracket M \rrbracket_{\rho}^{\mathcal{F}(\infty)}$. Since an element in a domain is completely determined by the set of compact elements less equal to it, we need only to prove that:

$$B_{\rho} \vdash_{\infty} M : \sigma \quad \text{if and only if} \quad (\sigma)^* \sqsubseteq_{\mathcal{I}} \llbracket M \rrbracket_{\zeta}^{\mathbb{I}_{\infty}},$$

where $B_{\rho}(x) \in \rho(x)$, for all $x \in \text{Var}$.

(\Rightarrow) The proof follows by induction on M .

If $M \equiv x$ then $B_{\rho}(x) \leq_{\infty} \sigma$, so $(\sigma)^* \sqsubseteq_{\mathcal{I}} \zeta(x)$ and the proof is given.

Let $M \equiv PQ$. So $B_{\rho} \vdash_{\infty} PQ : \sigma$ implies that there is a type τ such that $B_{\rho} \vdash_{\infty} P : \tau \rightarrow \sigma$ and $B_{\rho} \vdash_{\infty} Q : \tau$ by Lemma 10.1.7.(vii). By induction both $(\tau \rightarrow \sigma)^* \sqsubseteq_{\mathcal{I}} \llbracket P \rrbracket_{\zeta}^{\mathbb{I}_{\infty}}$ and $(\tau)^* \sqsubseteq_{\mathcal{I}} \llbracket Q \rrbracket_{\zeta}^{\mathbb{I}_{\infty}}$; thus $s_{(\tau)^*, (\sigma)^*} \sqsubseteq_{\mathcal{I}} \llbracket P \rrbracket_{\zeta}^{\mathbb{I}_{\infty}} = I(\llbracket P \rrbracket_{\zeta}^{\mathbb{I}_{\infty}})$ by isomorphism. Hence $(\sigma)^* \sqsubseteq_{\mathcal{I}} I(\llbracket P \rrbracket_{\zeta}^{\mathbb{I}_{\infty}})(\llbracket Q \rrbracket_{\zeta}^{\mathbb{I}_{\infty}}) = (\llbracket PQ \rrbracket_{\zeta}^{\mathbb{I}_{\infty}})$.

Let $M \equiv \lambda x. N$. By Lemma 10.1.7.(iv), $B_{\rho} \vdash_{\infty} \lambda x. N : \sigma$ implies that there are μ_i, ν_i ($1 \leq i \leq n$) for some $n \in \mathbb{N}$, such that $(\mu_1 \rightarrow \nu_1) \wedge \dots \wedge (\mu_n \rightarrow \nu_n) \leq_{\infty} \sigma$, $B_{\rho}[\mu_i/x] \vdash_{\infty} N : \nu_i$ ($1 \leq i \leq n$). By induction $(\nu_i)^* \sqsubseteq_{\mathcal{I}} \llbracket N \rrbracket_{\zeta[d/x]}^{\mathbb{I}_{\infty}}$ where $(\mu_i)^* \sqsubseteq_{\mathcal{I}} d$. Therefore

$$(\sigma)^* \sqsubseteq_{\mathcal{I}} \bigsqcup_{1 \leq i \leq n} s_{(\mu_i)^*, (\nu_i)^*} \sqsubseteq_{\mathcal{I}} \bigsqcup_{\substack{1 \leq i \leq n \\ d \in \text{comp}(\mathbb{I}_\infty)}} s_{d, (\nu_i)^*} \sqsubseteq_{\mathcal{I}} J(\lambda d. \llbracket N \rrbracket_{\zeta[d/x]}^{\mathbb{I}_\infty}).$$

(\Leftarrow) The proof follows by induction on M . If $M \equiv x$ then the proof is easy. Let $M \equiv PQ$. So $(\sigma)^* \sqsubseteq_{\mathcal{I}} \llbracket PQ \rrbracket_{\zeta}^{\mathbb{I}_\infty} = I(\llbracket P \rrbracket_{\zeta}^{\mathbb{I}_\infty})(\llbracket Q \rrbracket_{\zeta}^{\mathbb{I}_\infty})$. Thus there is τ such that $s_{(\tau)^*, (\sigma)^*} \sqsubseteq_{\mathcal{I}} \llbracket P \rrbracket_{\zeta}^{\mathbb{I}_\infty}$ and $(\tau)^* \sqsubseteq_{\mathcal{I}} \llbracket Q \rrbracket_{\zeta}^{\mathbb{I}_\infty}$. The proof follows by induction. Let $M \equiv \lambda x.N$. So $(\sigma)^* \sqsubseteq_{\mathcal{I}} \llbracket M \rrbracket_{\zeta}^{\mathbb{I}_\infty} = J(\lambda d. \llbracket N \rrbracket_{\zeta[d/x]}^{\mathbb{I}_\infty})$ implies that there are μ_i, ν_i ($1 \leq i \leq n$) for some $n \in \mathbb{N}$, such that both $(\nu_i)^* \sqsubseteq_{\mathcal{I}} \llbracket N \rrbracket_{\zeta[d/x]}^{\mathbb{I}_\infty}$ and $(\sigma)^* \sqsubseteq_{\mathcal{I}} \bigsqcup_{1 \leq i \leq n} s_{(\mu_i)^*, (\nu_i)^*}$ and $(\mu_i)^* \sqsubseteq_{\mathcal{I}} d$. Hence $\sigma \geq_{\infty} \bigwedge_{1 \leq i \leq n} (\mu_i \rightarrow \nu_i)$ where $B_\rho[\mu_i/x] \vdash_{\infty} N : \nu_i$ ($1 \leq i \leq n$), so $B_\rho \vdash_{\infty} \lambda x.N : \mu_i \rightarrow \nu_i$ ($1 \leq i \leq n$) by rule $(\rightarrow I)$, and $B_\rho \vdash_{\infty} \lambda x.N : \sigma$ by rule (\leq_{∞}) . \square

The model \mathcal{I} was the first denotational $\lambda\Lambda$ -calculus model. It was built by Scott [89], and the induced Λ -theory was extensively studied in [54, 97], where the approximation theorem is proved by using the technique of indexed reductions. An analysis of the characterization of term in the \mathcal{I} -model can be found in [32]. By the structure of \mathbb{I} , it is possible to have a different inverse limit solution, by choosing as initial embedding function the function i' , such that $i'(\perp_{\mathbb{I}}) = \perp_{[\mathbb{I} \rightarrow \mathbb{I}]}$ and $i'(\top_{\mathbb{I}}) = s_{\top_{\mathbb{I}}, \top_{\mathbb{I}}}$. This model was first defined by Park [75], and it induce a Λ -theory quite different from \mathbf{H} , which was extensively studied in [53].

13.1.2 \mathcal{N} as Domain

\mathcal{N} is isomorphic to a $\lambda\Lambda$ -model, which arises from an inverse limit solution of the same domain equation as \mathcal{H} , i.e.

$$\mathbb{X} = [\mathbb{X} \rightarrow \mathbb{X}].$$

Take as initial domain $(\mathbb{N}, \sqsubseteq_{\mathbb{N}})$, where $\mathbb{N} = \{\perp_{\mathbb{N}}, \varkappa, \top_{\mathbb{N}}\}$, and $\sqsubseteq_{\mathbb{N}}$ is defined as $\perp_{\mathbb{N}} \sqsubseteq_{\mathbb{N}} b$, for all $b \in \mathbb{N}$ and $\varkappa \sqsubseteq_{\mathbb{N}} \top_{\mathbb{N}}$. Take the domain $([\mathbb{N} \rightarrow \mathbb{N}], \sqsubseteq_{\mathbb{N} \rightarrow \mathbb{N}})$, which is a domain by Lemma 13.1.4.(i), and choose, as embedding function between $(\mathbb{N}, \sqsubseteq_{\mathbb{N}})$ and $([\mathbb{N} \rightarrow \mathbb{N}], \sqsubseteq_{\mathbb{N} \rightarrow \mathbb{N}})$, the function i so defined:

- $i(\perp_{\mathbb{N}}) = \perp_{[\mathbb{N} \rightarrow \mathbb{N}]}$,
- $i(\varkappa) = s_{\top_{\mathbb{N}}, \varkappa}$,
- $i(\top_{\mathbb{N}}) = s_{\varkappa, \top_{\mathbb{N}}}$,

and let \mathbb{N}_{∞} be the inverse limit solution so obtained.

The correspondence between C_{\bowtie} and $\text{comp}(\mathbb{N})$ is defined in the following way:

- $(\omega)^+ = \perp_{\mathbb{N}}$,
- $(\psi)^+ = \varkappa$,
- $(\phi)^+ = \top_{\mathbb{N}}$.

Let the function $(.)^* : T(C_{\bowtie}) \rightarrow \mathbf{comp}(\mathbb{N}_{\infty})$ be the function made according to the procedure **compact-as-types**.

The procedure **compact-as-types**(\mathbb{N}_{∞}) generates, by point 2, the rule $(n0)$ of Fig. 11.3 (pag. 145). Furthermore, point (4.1) generates the rules $(n1), (n2), (n3), (n4)$ and $(n5)$ of Fig. 11.3 of the intersection relation \leq_{\bowtie} . Note that point (4.2) is not applied in this case.

Let the interpretation function $\llbracket . \rrbracket^{\mathbb{N}_{\infty}}$ and the composition $\circ_{\mathbb{N}_{\infty}}$ be defined as for the model \mathbb{I}_{∞} .

Theorem 13.1.16. *Let $\mathcal{J} = <\mathbb{N}_{\infty}, \mathbb{N}_{\infty}, \circ_{\mathbb{N}_{\infty}}, \llbracket . \rrbracket^{\mathbb{N}_{\infty}}>$ and $\circ_{\mathbb{N}_{\infty}} \equiv \lambda xy. I(x)(y)$. \mathcal{J} is a $\lambda\Lambda$ -calculus model, and it is isomorphic to \mathcal{N} .*

Proof. Similar to the proof of Theorem 13.1.15. \square

The model \mathcal{J} was first presented and studied in [30], as filter model. There the approximation theorem was proved using the computability technique we also used here. The notion of Λ -persistent normal form, on which the construction of the model is based, was first introduced in [17].

13.1.3 \mathcal{L} as Domain

Take the following domain equation:

$$\mathbb{X} = [\mathbb{X} \rightarrow \mathbb{X}]_{\perp}.$$

By the presence of the lifting domain constructor, this equation admits a minimal solution. In fact, take as initial domain $(\mathbb{E}, \sqsubseteq_{\mathbb{E}})$, where $\mathbb{E} = \{\perp_{\mathbb{E}}\}$, and $\sqsubseteq_{\mathbb{E}}$ is the identity relation. Take the domain $([\mathbb{E} \rightarrow \mathbb{E}]_{\perp}, \sqsubseteq_{[\mathbb{E} \rightarrow \mathbb{E}]_{\perp}})$, which is a domain by Lemma 13.1.4.(i) and Property 13.1.6, and choose, as embedding function between $(\mathbb{E}, \sqsubseteq_{\mathbb{E}})$ and $([\mathbb{E} \rightarrow \mathbb{E}], \sqsubseteq_{[\mathbb{E} \rightarrow \mathbb{E}]})$, the function i so defined:

- $i(\perp_{\mathbb{E}}) = \perp_{[\mathbb{E} \rightarrow \mathbb{E}]_{\perp}}$,

and let \mathbb{E}_{∞} be the inverse limit solution so obtained.

The correspondence between C_{\angle} and $\mathbf{comp}(\mathbb{E})$ is defined in the following way:

- $(\omega)^+ = \perp_{\mathbb{E}}$.

Let the function $(.)^* : T(C_{\bowtie}) \rightarrow \mathbf{comp}(\mathbb{E}_{\infty})$ be the function made according to the procedure **compact-as-types**.

The procedure **compact-as-types**(\mathbb{E}_{∞}) generates, at point (4.1), the trivial rule $\omega \leq \omega$ of the intersection relation \leq_{\angle} . Note that point (4.2) is not applied in this case.

Let the interpretation function $\llbracket . \rrbracket^{\mathbb{E}_{\infty}}$ and the composition $\circ_{\mathbb{E}_{\infty}}$ be defined as follows:

- $\llbracket x \rrbracket_{\rho}^{\mathbb{E}_{\infty}} = \rho(x)$,

- $\llbracket MN \rrbracket_{\rho}^{\mathbb{E}_{\infty}} = J'(I(\llbracket M \rrbracket_{\rho}^{\mathbb{E}_{\infty}}))(\llbracket N \rrbracket_{\rho}^{\mathbb{E}_{\infty}}),$
- $\llbracket \lambda x.M \rrbracket_{\rho}^{\mathbb{E}_{\infty}} = J(I'(\mathbf{A}d.\llbracket M \rrbracket_{\rho[d/x]}^{\mathbb{E}_{\infty}})),$

where I and J are defined in the proof of Theorem 13.1.11, I' and J' are the isomorphism pair between $[\mathbb{E}_{\infty} \rightarrow \mathbb{E}_{\infty}]$ and $[\mathbb{E}_{\infty} \rightarrow \mathbb{E}_{\infty}]_{\perp}$.

Theorem 13.1.17. *Let $\mathcal{E} = < \mathbb{E}_{\infty}, \mathbb{E}_{\infty}, \circ_{\mathbb{E}_{\infty}}, \llbracket \cdot \rrbracket^{\mathbb{E}_{\infty}} >$ where $\circ_{\mathbb{E}_{\infty}}$ is defined as $\mathbf{A}xy.I(x)(y)$.*

\mathcal{E} is a $\lambda\Lambda$ -calculus model, and it is isomorphic to \mathcal{L} .

Proof. Similar to the proof of Theorem 13.1.15, taking into account the different definition of interpretation. \square

The model \mathcal{E} was first presented and studied in [2].

13.1.4 \mathcal{V} as Domain

Take the following domain equation:

$$\mathbb{X} = [\mathbb{X} \rightarrow_{\perp} \mathbb{X}]_{\perp}.$$

This equation, like that one showed in the previous subsection, also admits a minimal solution. In fact, take as initial domain $(\mathbb{U}, \sqsubseteq_{\mathbb{U}})$, where $\mathbb{U} = \{\perp_{\mathbb{U}}\}$, and $\sqsubseteq_{\mathbb{U}}$ is the identity relation. Take the domain $([\mathbb{U} \rightarrow_{\perp} \mathbb{U}]_{\perp}, \sqsubseteq_{[\mathbb{U} \rightarrow_{\perp} \mathbb{U}]_{\perp}})$, which is a domain by Lemma 13.1.4.(i) and Property 13.1.6, and choose, as embedding function between $(\mathbb{U}, \sqsubseteq_{\mathbb{U}})$ and $([\mathbb{U} \rightarrow_{\perp} \mathbb{U}], \sqsubseteq_{[\mathbb{U} \rightarrow_{\perp} \mathbb{U}]})$, the function i so defined:

- $i(\perp_{\mathbb{U}}) = \perp_{[\mathbb{U} \rightarrow_{\perp} \mathbb{U}]_{\perp}}$

and let \mathbb{U}_{∞} be the inverse limit solution so obtained.

The function $(\cdot)^*$, generated by the procedure **compact-as-types**(\mathbb{U}_{∞}), has the following behaviour:

$$\begin{aligned} (\sigma \rightarrow \tau)^* &= s_{(\sigma)^*, (\tau)^*} \text{ if either } (\sigma)^*, (\tau)^* = \perp_{\mathbb{U}_{\infty}} \text{ or } (\sigma)^* \neq \perp_{\mathbb{U}_{\infty}}, \\ (\sigma \rightarrow \tau)^* &= s_{(\omega \rightarrow \omega)^*, (\tau)^*} \text{ if } (\sigma)^* = \perp_{\mathbb{U}_{\infty}}. \end{aligned}$$

Moreover, the intersection relation built by the procedure generates, at point (4.1), the trivial rule $\omega \leq \omega$, while at point (4.2) rule (v) of Fig. 12.1 (pag. 182) is generated.

Let $\rho : \text{Var} \rightarrow \mathbb{U}_{\infty}/\perp_{\mathbb{U}_{\infty}}$; the interpretation function $\llbracket \cdot \rrbracket^{\mathbb{U}_{\infty}}$ (see [44]) is:

- $\llbracket x \rrbracket_{\rho}^{\mathbb{U}_{\infty}} = \rho(x),$
- $\llbracket MN \rrbracket_{\rho}^{\mathbb{U}_{\infty}} = J'(I(\llbracket M \rrbracket_{\rho}^{\mathbb{U}_{\infty}}))(\llbracket N \rrbracket_{\rho}^{\mathbb{U}_{\infty}}),$
- $\llbracket \lambda x.M \rrbracket_{\rho}^{\mathbb{U}_{\infty}} = J(I'(\mathbf{strict}(\mathbf{A}d.\llbracket M \rrbracket_{\rho[d/x]}^{\mathbb{U}_{\infty}}))),$

where I and J are defined in the proof of Theorem 13.1.11, I' and J' are the isomorphism pair between $[\mathbb{U}_\infty \rightarrow_\perp \mathbb{U}_\infty]$ and $[\mathbb{U}_\infty \rightarrow_\perp \mathbb{U}_\infty]_\perp$, and strict is a function such that

$$\text{strict}(f)(x) = \begin{cases} \perp_{\mathbb{U}_\infty} & x = \perp_{\mathbb{U}_\infty}, \\ f(x) & \text{otherwise.} \end{cases}$$

Theorem 13.1.18. *Let $\mathcal{U} = <\mathbb{U}_\infty, \mathbb{U}_\infty / \perp_{\mathbb{U}_\infty}, \circ_{\mathbb{U}_\infty}, [\cdot]_{\mathbb{U}_\infty}^{\mathbb{U}_\infty}>$ where $\circ_{\mathbb{U}_\infty}$ is defined as $\lambda xy. I(x)(y)$.*

\mathcal{U} is a $\lambda\Gamma$ -calculus model, and it is isomorphic to \mathcal{V} .

Proof. Similar to the proof of Theorem 13.1.15, taking into account the fact that only the strict functions are present in the domain.

By Theorem 13.1.14, $(\mathbb{U}_\infty, \sqsubseteq_{\mathbb{U}_\infty})$ is isomorphic to $\mathcal{F}(\sqrt{})$, ordered by set inclusion. Then the proof follows from the definition of $[\cdot]_{\mathbb{U}_\infty}^{\mathbb{U}_\infty}$. \square

The model \mathcal{V} was first presented and studied in [44], both as an inverse limit solution of the previous domain equation and as a filter model. There the approximation model was proved using the indexed reduction technique.

13.1.5 Another Domain

Every solution of the domain equation

$$\mathbb{X} = [\mathbb{X} \rightarrow_\perp \mathbb{X}]$$

is a model for the Λ -NF⁰-calculus.

This fact was first noticed in [39]. We did not develop the study of such a calculus, since it does not seem to have interesting operational properties.

14. Further Reading

Other filters $\lambda\Lambda$ -models. In [37] two filter $\lambda\Lambda$ -models are designed which completely characterise sets of terms with similar computational behaviours. Moreover, in [4, 40] filter $\lambda\Lambda$ -models characterizing the easiness property of terms are proposed. Shortly, a term is easy when it can be consistently equated to every other term.

Other classes of $\lambda\Lambda$ -models. Berry [14] proposed a different class of domains based on the notion of stable functions. Starting from this notion, Girard [47] proposed qualitative domains as $\lambda\Lambda$ -models. Later qualitative domains were that were later refined in the coherence domains. The first denotational semantics of linear logics is based on this kind of domains[48]. The definition of intersection relation can be modified in order to describe this class of models using intersection types, as was proved in [52]. The notions of strongly stable functions and hypercoherence spaces, on which another class of $\lambda\Lambda$ -models is based, were introduced in [23]. Models based on the notion of bidomain, which is a space endowed with two notions of order (continuous order and stable order), were introduced in [99]. In this setting, a model correct with respect to the **L**-operational semantics was constructed in [61]. A quite complete presentation of the “webbed” $\lambda\Lambda$ -models, i.e. those whose domains are subdomains of some $(\mathbf{P}(D), \subseteq)$, can be found in [13]; clearly all the $\lambda\Lambda$ -models presented in this book belong to this class. $\lambda\Lambda$ -models based on game semantics were presented in [43]. Categorical presentations of $\lambda\Lambda$ -models in a typed setting can be found [8, 31, 62, 59, 90].

Incompleteness. The first incompleteness result for the $\lambda\Lambda$ -calculus semantics was proved in [53], where a λ -theory was shown for which there do not exist a correct and complete model in the class of Scott’s models built by an inverse limit construction. Further investigations on this topic, using topological tools, was made in [86]. Incompleteness results for the class of $\lambda\Lambda$ -models based on stable functions was proved in [11].

Lazy semantics. A general characterization of models that are correct with respect to the **L**-operational semantics was given in [12].

$\lambda\Gamma$ -Models. A general characterization of models that are correct with respect to the **V**-operational semantics was given in [82].

Semiseparability. An extension of the semiseparability algorithm to a finite set of approximants was introduced in [29].

Part IV

Computational Power

15. Preliminaries

In the Introduction we claimed that both the λA -calculus and the $\lambda \Gamma$ -calculus can be seen as paradigms for programming languages in the call-by-name and call-by-value settings respectively. In this chapter this claim will be justified. In fact, we will show that both the call-by-name and the call-by-value λ -calculi have the computational power of Turing machines, or equivalently, they are *computationally complete*. The completeness can be achieved without adding special constants to the language, but all data structures needed for computing, in particular booleans, natural numbers and functions, can be coded into A .

We will show how to code useful data structures. Moreover, we will prove that all the call-by-name and call-by-value reduction machines presented in Part II of this book can be effectively used for computing. In fact, computational completeness can be achieved by using each one of them.

15.1 Kleene's Recursive Functions

It is well known that not all the partial functions from natural numbers to natural numbers can be effectively computed. The most famous definition of the class of computable functions was given by Turing, by using the *Turing machines*. But Kleene's definition of partial recursive functions [56] makes it easier to define the coding of functions in a $\lambda \Delta$ -calculus [57].

The class of computable functions, or partial recursive functions, is given in two stages. First, the class of *primitive recursive* functions is defined. They are generated from a set of initial functions by closure under particular constructions (composition and primitive recursion). The primitive recursive functions include most functions ever encountered in practical mathematics and computer science. However, all primitive recursive functions are total, and hence the class must necessarily fall short of the full class of computable functions.

The second stage of Kleene's characterization extends the class of primitive recursive functions by adding an additional operator of minimization that introduces unbounded and possibly nonterminating searches. Therefore the class of all partial recursive functions is obtained.

Definition 15.1.1 (Primitive recursive functions).

- (i) *The following functions are primitive recursive functions:*
 1. *The function $Z : \mathbb{N} \rightarrow \mathbb{N}$ such that $Z(n) = 0$;*
 2. *The successor $S : \mathbb{N} \rightarrow \mathbb{N}$ such that $S(n) = n + 1$;*
 3. *The projection functions $\pi_i^m(x_1, \dots, x_m) = x_i$ ($1 \leq i \leq m \in \mathbb{N}$).*
- (ii) *If $h : \mathbb{N}^n \rightarrow \mathbb{N}$ and $g_1, \dots, g_n : \mathbb{N}^m \rightarrow \mathbb{N}$ are primitive recursive functions then the function f , defined as their composition in the following way:*

$$f(x_1, \dots, x_m) = h(g_1(x_1, \dots, x_m), \dots, g_n(x_1, \dots, x_m)),$$

is primitive recursive too ($n, m \in \mathbb{N}$).

- (iii) *If $h : \mathbb{N}^{m+2} \rightarrow \mathbb{N}$ and $g : \mathbb{N}^m \rightarrow \mathbb{N}$ are primitive recursive functions then f defined by primitive recursion in the following way:*

$$f(k, x_1, \dots, x_m) = \begin{cases} g(x_1, \dots, x_m) & \text{if } k = 0, \\ h(f(k-1, x_1, \dots, x_m), k-1, x_1, \dots, x_m) & \text{otherwise.} \end{cases}$$

is primitive recursive too ($m \in \mathbb{N}$).

By induction on the depth of nested instances of composition and primitive recursion, it is easy to check that each primitive recursive function is total. The next definition allows the construction of partial functions.

Definition 15.1.2. *Let $h : \mathbb{N}^2 \rightarrow \mathbb{N}$ be a total function, and let $x \in \mathbb{N}$. Then a function $f : \mathbb{N} \rightarrow \mathbb{N}$ can be defined by minimalization from h in the following way:*

$$f(x) = \mu y [h(x, y) = 0] = \begin{cases} \min\{k \in \mathbb{N} \mid h(x, k) = 0\} & \text{if such a } k \in \mathbb{N} \text{ exists,} \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Note that, in the previous definition, the function h is defined on all natural numbers, by hypothesis. Now, the full class of partial recursive functions can be defined as follows.

Definition 15.1.3 (Partial recursive functions).

A function $f : \mathbb{N}^m \rightarrow \mathbb{N}$ ($m \in \mathbb{N}$) is partial recursive if and only if one of the following conditions holds:

- (i) *f is a primitive recursive function;*
- (ii) *f is defined by composition of partial recursive functions;*
- (iii) *f is defined by minimalization starting from a total recursive function.*

It is important to notice that, in the previous definition, the minimalization construction must be applied only to *total* recursive functions, which form a nonrecursive class. By Church's thesis, the class of partial recursive functions coincides with the whole class of computable functions.

15.2 Representing Data Structures

The starting point for transforming a $\lambda\Delta$ -calculus in a programming language is to code some fundamental data structures in it, namely booleans and natural numbers.

Let us study the problem of representing booleans in a $\lambda\Delta$ -calculus whose operational behaviour is described by an evaluation relation $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$. In order to represent the truth values *True* and *False*, we need to define two terms having a suitable behaviour; in particular, they must be the basis for the definition of a further term having the behaviour of a conditional operator.

Definition 15.2.1. Let $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ be a evaluation relation. An \mathbf{O} -representation of booleans is any set $\{T, F\}$ such that:

- (i) $T, F \in \Delta \cap \Theta$;
- (ii) there is a term *Cond* such that, for every $M, N \in \Delta \cap \Theta$:

$$\text{Cond } TMN \Downarrow_{\mathbf{O}} M; \quad \text{Cond } FMN \Downarrow_{\mathbf{O}} N.$$

The next lemma shows that in order to represent the booleans, the choice of taking two $\Lambda\eta$ -different normal forms is correct in each one of the reduction machines we defined.

Lemma 15.2.2. Let $\mathbf{O} \in \{\mathbf{H}, \mathbf{N}, \mathbf{L}, \mathbf{V}\}$, and let $M, N \in \Lambda^0$ be two different $\Lambda\eta$ -normal forms. $\{M, N\}$ is an \mathbf{O} -representation of booleans.

Proof. If $\mathbf{O} \in \{\mathbf{H}, \mathbf{N}, \mathbf{L}\}$ then let $C[\cdot]$ be such that $M, N \Rightarrow_A C[\cdot]$ (see Fig. 2.1 pag. 32), otherwise let $C[\cdot]$ be such that $M, N \Rightarrow_F C[\cdot]$ (see Fig. 3.1 pag. 52). Then $C[M] \Downarrow_{\mathbf{O}} x$ and $C[M] \Downarrow_{\mathbf{O}} y$, for two different variables x and y . The term $\text{Cond} \equiv \lambda xy.y.C[u]$ plays the desired role in all the reduction machines under consideration. \square

It is a standard choice to define $T \equiv \lambda xy.x$ and $F \equiv \lambda xy.y$. In this case *Cond* can be taken as the identity term *I*, or simply omitted. In fact, if $M, N \in \Lambda\text{-NF}$ then $TMN \Downarrow_{\mathbf{O}} M$ and $FMN \Downarrow_{\mathbf{O}} N$, for all $\mathbf{O} \in \{\mathbf{H}, \mathbf{N}, \mathbf{L}, \mathbf{V}\}$. A boolean expression is every term B , such that $B \Downarrow_{\mathbf{O}}$ implies either $B \Downarrow_{\mathbf{O}} T$ or $B \Downarrow_{\mathbf{O}} F$.

Through the coding of the booleans it is possible to code more complex data structures, for example, the pairs. Let M, N be two Λ -normal forms; the pair $[M, N]$ can be coded as $\lambda x.xMN$. So projections can be built using booleans, by defining $\lambda x.xT$ and $\lambda x.xF$ as respectively the first and the second projections. In fact, $(\lambda x.xT)[M, N] \Downarrow_{\mathbf{O}} M$ and $(\lambda x.xF)[M, N] \Downarrow_{\mathbf{O}} N$. We will denote $\lambda x.xMN$ by $[M, N]$.

The coding of the natural numbers can be based on Peano's axioms, recalled in the following definition.

Definition 15.2.3 (Peano's natural numbers).

1. There is a natural number, called zero.
2. Given a natural number n , there is a unique natural number m that is its successor.
3. Two different natural numbers have different successors.
4. If n is a natural number then its successor is different from zero.
5. If A is a subset of natural numbers satisfying:
 - zero belongs to A ,
 - if n belongs to A then its successor belongs to A too,
 then A is the set of all natural numbers.

The notion of an \mathbf{O} -numeral system, given in the next definition, gives the conditions for building the coding of natural numbers with respect to an evaluation relation \mathbf{O} . Following the lines of Peano's axioms, the infinite set of natural numbers can be generated by two suitable terms playing the role of zero and successor. The other conditions assure that all the terms generated by iterating the application of successor to zero are different in the operational setting we are considering.

Definition 15.2.4. Let $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ be a deterministic evaluation relation. An \mathbf{O} -numeral system is a 5-tuple $\langle \mathbb{B}, \text{Zero}, \text{Succ}, \text{Test}, \text{Pred} \rangle$, where:

- (i) \mathbb{B} is an \mathbf{O} -representation of booleans.
- (ii) $\text{Zero}, \text{Succ}, \text{Test}, \text{Pred} \in \Delta \cap \Theta$ are such that, for all $n \in \mathbb{N}$:

1. $\underbrace{\text{Succ} (\dots (\text{Succ}}_n \text{Zero)...) \Downarrow_{\mathbf{O}}$.

Moreover, if $\underbrace{\text{Succ} (\dots (\text{Succ}}_n \text{Zero)...) \Downarrow_{\mathbf{O}} \lceil n \rceil$ then $\lceil n \rceil \in \Delta \cap \Theta$;
we will say that $\lceil n \rceil$ is the numeral representation of n .

2. $P \Downarrow_{\mathbf{O}} \lceil n \rceil$ implies $\text{Succ } P \Downarrow_{\mathbf{O}} \lceil n + 1 \rceil$.
3. $P \Downarrow_{\mathbf{O}} \text{Zero}$ implies $\text{Test } P \Downarrow_{\mathbf{O}} T$.
4. $Q \Downarrow_{\mathbf{O}} \lceil n + 1 \rceil$ implies $\text{Test } Q \Downarrow_{\mathbf{O}} F$.
5. $P \Downarrow_{\mathbf{O}} \lceil n + 1 \rceil$ implies $\text{Pred } P \Downarrow_{\mathbf{O}} \lceil n \rceil$.

This definition is well posed. According to it, the number $n \in \mathbb{N}$ is represented by the numerals $\lceil n \rceil$.

Property 15.2.5. Definition 15.2.4 respects the Peano constraints.

Proof. All points of Peano's definition are satisfied.

1. Immediate, by the definition.
2. Immediate, by the definition and since \mathbf{O} is deterministic.

3. Let $\lceil n \rceil \not\approx_O \lceil m \rceil$. Assume, by absurd, $Succ \lceil m \rceil \approx_O Succ \lceil n \rceil$. Then, by the contextual closure of \approx_O and point (ii).5 of Definition 15.2.4, $\lceil m \rceil \approx_O Pred(Succ \lceil m \rceil) \approx_O Pred(Succ \lceil n \rceil) \approx_O \lceil n \rceil$, against the hypothesis.
4. Trivial, by using *Test*.
5. $\left\{ \lceil n \rceil \in A \mid \underbrace{Succ (\dots (Succ}_{n} Zero) \dots) \Downarrow_{\mathbf{O}} \lceil n \rceil \text{ for some } n \in \mathbb{N} \right\}$ is the set of the numerals. \square

In the next definition a numeral system is presented that plays the desired role in all the operational semantics we have studied.

Definition 15.2.6. Let $\mathfrak{N} \equiv \langle \{T, F\}, Zero, Succ, Test, Pred \rangle$, where

- $T \equiv \lambda xy.x$ and $F \equiv \lambda xy.y$;
- $Zero \equiv [T, T]$;
- $Succ \equiv \lambda t.t(\lambda uvx.xF(\lambda y.yuv))$;
- $Test \equiv \lambda x.xT$;
- $Pred \equiv \lambda x.xF$.

We will check that \mathfrak{N} is a numeral system in the sense of Definition 15.2.4, for all $\mathbf{O} \in \{\mathbf{H}, \mathbf{N}, \mathbf{L}, \mathbf{V}\}$. It is easy to see that $\forall n \in \mathbb{N}$, the numeral $\lceil n \rceil$ in \mathfrak{N} is the same term for all \mathbf{O} -numeral system; in particular, $\lceil n \rceil \equiv \underbrace{[F, [F, \dots [F,}_{n} Zero] \dots]]$ and $\lceil n + 1 \rceil \equiv \lambda x.x\lceil n \rceil$.

Note that basic elements of \mathfrak{N} are A -normal forms, so they are both input and output values for all the machines. The proof can be done in the same manner for all the call-by-name reduction machines.

In all the formal systems presented in order to induce evaluation relations $\mathbf{H}, \mathbf{N}, \mathbf{L}, \mathbf{V}$, there is a rule named *(head)*. It is easy to see that, for each one of the given operational machines, *(head)* is reversible in the sense that in all considered cases when the conclusion is derivable then its premises are derivable. In order to simplify the proofs, in the call-by-name setting, i.e. when $\mathbf{O} \in \{\mathbf{H}, \mathbf{N}, \mathbf{L}\}$, we will denote by $(\text{head})^+$ a sequence of $n \geq 1$ applications of rule *(head)* in a derivation.

We need some properties of the operational semantics.

Property 15.2.7. Let $M, N, P, Q \in A^0$.

- (i) Let $\mathbf{O} \in \{\mathbf{H}, \mathbf{N}, \mathbf{L}, \mathbf{V}\}$. $M \Downarrow_{\mathbf{O}} N$ and $NP \Downarrow_{\mathbf{O}} Q$ if and only if $MP \Downarrow_{\mathbf{O}} Q$.
- (ii) If $M \uparrow_{\mathbf{H}}$, $M \uparrow_{\mathbf{N}}$, $M \uparrow_{\mathbf{L}}$ and $M \uparrow_{\mathbf{V}}$ then $MN \uparrow_{\mathbf{O}}$, for all $N \in A$ and $\mathbf{O} \in \{\mathbf{H}, \mathbf{N}, \mathbf{L}, \mathbf{V}\}$.

Proof. (i) In case $\mathbf{O} \in \{\mathbf{H}, \mathbf{N}, \mathbf{L}\}$, the proof follows respectively by Properties 6.1.4, 6.2.4 and 6.3.4. Let $\mathbf{O} \equiv \mathbf{V}$. By the confluence theorem we can assume $MP \Downarrow_{\mathbf{V}} R$ if and only if $NP \Downarrow_{\mathbf{V}} Q$. We show that $R \equiv Q$ by induction on the last applied rule in $M \Downarrow_{\mathbf{V}} N$. Rules *(var)* and *(block)* are not possible, since $M \in \Lambda^0$; while *(abs)* and *(head)* are trivial.
(ii) Since $M \uparrow_{\mathbf{O}}$, for all \mathbf{O} , implies that M is both a Λ and a Γ -unsolvable of order 0, then MN is an unsolvable of order 0 too. \square

Note that point (i) of the previous property is just a consequence of the fact that every reduction machine reduces at every step the head redex.

Theorem 15.2.8. *If $\mathbf{O} \in \{\mathbf{H}, \mathbf{N}, \mathbf{L}\}$ then \mathfrak{N} is an \mathbf{O} -numeral system.*

Proof. We prove that \mathfrak{N} satisfies all the conditions given in Definition 15.2.4.

- (i) $\{\mathsf{T}, \mathsf{F}\}$ is an \mathbf{O} -representation of booleans, by Lemma 15.2.2 and since in all cases $\approx_{\mathbf{O}}$ is a Λ -theory.
- (ii) Zero, Succ, Test, Pred are both input and output values.

1. $\text{Zero} \Downarrow_{\mathbf{O}} \ulcorner 0 \urcorner$, where $\ulcorner 0 \urcorner \equiv \text{Zero}$. By induction on n we will prove that the numeral $\ulcorner n + 1 \urcorner$ is the term $\lambda x.x\mathsf{F}\ulcorner n \urcorner \in \Delta \cap \Theta$.

Let $\underbrace{\text{Succ}(\dots(\text{Succ Zero})\dots)}_n \Downarrow_{\mathbf{O}} \ulcorner n \urcorner$; thus

$$\underbrace{\text{Succ}(\text{Succ}(\dots(\text{Succ Zero})\dots))}_{n+1} \Downarrow_{\mathbf{O}} R,$$

if and only if, since $\text{Succ} \equiv \lambda t.t(\lambda uvx.x\mathsf{F}(\lambda y.yuv))$,

$$\underbrace{\text{Succ}(\dots(\text{Succ Zero})\dots)(\lambda uvx.x\mathsf{F}(\lambda y.yuv))}_{n} \Downarrow_{\mathbf{O}} R$$

by *(head)*, if and only if $\ulcorner n \urcorner(\lambda uvx.x\mathsf{F}(\lambda y.yuv)) \Downarrow_{\mathbf{O}} R$, by Property 15.2.7.(i).

If $n = 0$, then this happens only if $(\lambda uvx.x\mathsf{F}(\lambda y.yuv))\mathsf{TT} \Downarrow_{\mathbf{O}} R$, by *(head)*, so $R \equiv \lambda x.x\mathsf{F}(\lambda y.y\mathsf{TT}) \equiv \ulcorner 1 \urcorner$.

Otherwise, $\ulcorner n \urcorner(\lambda uvx.x\mathsf{F}(\lambda y.yuv)) \Downarrow_{\mathbf{O}} R$ if and only if

$$(\lambda uvx.x\mathsf{F}(\lambda y.yuv))\mathsf{F}\ulcorner n - 1 \urcorner \Downarrow_{\mathbf{O}} R$$

by *(head)*, if and only if $\lambda x.x\mathsf{F}(\lambda y.y\mathsf{F}\ulcorner n - 1 \urcorner) \Downarrow_{\mathbf{O}} R$ by *(head)*⁺. But $\lambda x.x\mathsf{F}(\lambda y.y\mathsf{F}\ulcorner n - 1 \urcorner) \in \Lambda\text{-NF}$, so $R \equiv \lambda x.x\mathsf{F}(\lambda y.y\mathsf{F}\ulcorner n - 1 \urcorner) \equiv \ulcorner n + 1 \urcorner$.

- 2. If $P \in \Lambda$ is such that $P \Downarrow_{\mathbf{O}} \ulcorner n \urcorner$ then $\text{Succ } \ulcorner n \urcorner \Downarrow_{\mathbf{O}} \ulcorner n + 1 \urcorner$, reasoning as in the previous point.
- 3. Let $P \Downarrow_{\mathbf{O}} \text{Zero}$, so $\text{Test} \equiv \lambda x.x\mathsf{T}$ implies $\text{Test } P \Downarrow_{\mathbf{O}} R$ if and only if $P\mathsf{T} \Downarrow_{\mathbf{O}} R$ by *(head)*, if and only if $\text{Zero}\mathsf{T} \Downarrow_{\mathbf{O}} R$ by Property 15.2.7.(i), if and only if $\mathsf{T} \Downarrow_{\mathbf{O}} R$ (by *(head)*⁺ again); thus $R \equiv \mathsf{T}$.
- 4. The case $P \Downarrow_{\mathbf{O}} \ulcorner n + 1 \urcorner$ is similar to the previous point.
- 5. Let $P \Downarrow_{\mathbf{O}} \ulcorner n + 1 \urcorner$. Then $\text{Pred } P \Downarrow_{\mathbf{O}} R$ if and only if $P\mathsf{F} \Downarrow_{\mathbf{O}} R$ by *(head)*, if and only if $\ulcorner n + 1 \urcorner\mathsf{F} \Downarrow_{\mathbf{O}} R$ (by Property 15.2.7.(i)) if and only if $\ulcorner n \urcorner \Downarrow_{\mathbf{O}} R$ (by *(head)*⁺). So it must be $R \equiv \ulcorner n \urcorner$. \square

Now we check that \mathfrak{N} is also a numeral system in a call-by value setting.

Theorem 15.2.9. \mathfrak{N} is a \mathbf{V} -numeral system.

Proof. (i) $\{\mathsf{T}, \mathsf{F}\}$ is an \mathbf{V} -representation of booleans, by Lemma 15.2.2.

(ii) $\mathsf{Zero}, \mathsf{Succ}, \mathsf{Test}, \mathsf{Pred} \in \Gamma \cap \Gamma\text{-LBNF}$.

$\mathsf{Zero} \Downarrow_{\mathbf{V}} \ulcorner 0 \urcorner$, where $\ulcorner 0 \urcorner \equiv \mathsf{Zero}$. By induction on n we will prove that the numeral $\ulcorner n + 1 \urcorner$ is the term $\lambda x.x\mathsf{F}\ulcorner n \urcorner \in \Gamma \cap \Gamma\text{-LBNF}$.

Let us assume that $\underbrace{\mathsf{Succ}(\dots(\mathsf{Succ} \mathsf{Zero})\dots)}_{n+1} \Downarrow_{\mathbf{V}} \ulcorner n \urcorner$, by induction; there-

fore $\underbrace{\mathsf{Succ}(\mathsf{Succ}(\dots(\mathsf{Succ} \mathsf{Zero})\dots))}_{n+1} \Downarrow_{\mathbf{V}} R$ if and only if $\mathsf{Succ}\ulcorner n \urcorner \Downarrow_{\mathbf{V}} R$ by

induction and (head), if and only if $\ulcorner n \urcorner(\lambda uvx.x\mathsf{F}(\lambda y.yuv)) \Downarrow_{\mathbf{V}} R$ by rules (head) and (lazy), since $\mathsf{Succ} \equiv \lambda t.t(\lambda uvx.x\mathsf{F}(\lambda y.yuv))$.

If $n = 0$, then this happens only if $(\lambda uvx.x\mathsf{F}(\lambda y.yuv))\mathsf{TT} \Downarrow_{\mathbf{V}} R$ by rules (head) and (lazy), so it is easy to see that $R \equiv \lambda x.x\mathsf{F}(\lambda y.y\mathsf{TT}) \equiv \ulcorner 1 \urcorner$.

Otherwise, $\ulcorner n \urcorner(\lambda uvx.x\mathsf{F}(\lambda y.yuv)) \Downarrow_{\mathbf{V}} R$ if and only if

$$(\lambda uvx.x\mathsf{F}(\lambda y.yuv))\mathsf{F}\ulcorner n - 1 \urcorner \Downarrow_{\mathbf{V}} R,$$

if and only if $\lambda x.x\mathsf{F}(\lambda y.y\mathsf{F}\ulcorner n - 1 \urcorner) \Downarrow_{\mathbf{V}} R$. But $\lambda x.x\mathsf{F}(\lambda y.y\mathsf{F}\ulcorner n - 1 \urcorner) \in \Lambda\text{-NF}$, so $R \equiv \lambda x.x\mathsf{F}(\lambda y.y\mathsf{F}\ulcorner n - 1 \urcorner) \equiv \ulcorner n + 1 \urcorner$.

It is easy to check the remaining constraints given in Definition 15.2.4, since the proof follows the same lines as the Theorem 15.2.8. \square

16. Representing Functions

In order to represent a numerical function with respect to an evaluation relation \mathbf{O} , it is necessary to exhibit a term mimicking the behaviour of the function itself. More precisely the reduction machine, taken as input this term applied to a sequence of terms representing natural numbers, gives as output the term representing the result, if it exists, and does not stop otherwise.

The term representing the function ϕ will be denoted by $\lceil \phi \rceil$, extending the same notation used for natural numbers. The notion is defined in a formal way in the next definition.

Definition 16.0.10. Let $\mathbf{O} \in \mathcal{E}(\Delta, \Theta)$ be an evaluation relation, and let ϕ be a partial recursive function with arity $p \in \mathbb{N}$; let $\lceil n \rceil$ be the numeral representation of $n \in \mathbb{N}$ in an \mathbf{O} -numeral system.

ϕ is \mathbf{O} -representable if and only if there is a term $\lceil \phi \rceil \in \Lambda^0$ such that, for all terms N_i such that $N_i \Downarrow_{\mathbf{O}} \lceil n_i \rceil$ ($1 \leq i \leq p; n_1, \dots, n_p \in \mathbb{N}$):

- if $\phi(n_1, \dots, n_p)$ is defined then $\lceil \phi \rceil N_1 \dots N_p \Downarrow_{\mathbf{O}} \lceil \phi(n_1, \dots, n_p) \rceil$;
- if $\phi(n_1, \dots, n_p)$ is undefined then $\lceil \phi \rceil N_1 \dots N_p \uparrow_{\mathbf{O}}$.

We will prove that in all given reduction machines all partial recursive functions are representable. To do so, we need to consider separately the call-by-name cases and the call-by-value one.

16.1 Call-by-Name Computational Completeness

We will prove that each one of the studied call-by-name reduction machines can be used for computing all partial recursive functions. In all this section, \mathbf{O} will range over the set $\{\mathbf{H}, \mathbf{N}, \mathbf{L}\}$, i.e. it denotes each one of the call-by-name evaluation relations, and \mathfrak{N} is the \mathbf{O} -numeral system given in Definition 15.2.8.

As a first step, it must be checked that primitive recursive functions are \mathbf{O} -representable.

- Lemma 16.1.1.** (i) Z is \mathbf{O} -representable,
(ii) S is \mathbf{O} -representable,
(iii) Projections are \mathbf{O} -representable.

- Proof.* (i) $\lceil Z \rceil \equiv \lambda x. \text{Zero}$,
(ii) $\lceil S \rceil \equiv \text{Succ}$,
(iii) $\lceil \pi_m^i \rceil \equiv \lambda x_1 \dots x_m. x_i$ ($1 \leq i \leq m \in \mathbb{N}$).

□

Now let us consider the composition between primitive recursive functions.

Lemma 16.1.2. *Let $h : \mathbb{N}^m \rightarrow \mathbb{N}$ and $g_1, \dots, g_m : \mathbb{N}^p \rightarrow \mathbb{N}$ be \mathbf{O} -representable primitive recursive functions; so their composition:*

$$f(x_1, \dots, x_p) = h(g_1(x_1, \dots, x_p), \dots, g_m(x_1, \dots, x_p)),$$

is \mathbf{O} -representable.

Proof. By hypothesis there are terms $\lceil h \rceil, \lceil g_1 \rceil, \dots, \lceil g_m \rceil$ \mathbf{O} -representing functions h, g_1, \dots, g_m . Let

$$\lceil f \rceil \equiv \lambda x_1 \dots x_p. \lceil h \rceil(\lceil g_1 \rceil x_1 \dots x_p) \dots (\lceil g_m \rceil x_1 \dots x_p).$$

Let $N_i \in \Lambda$ be such that $N_i \Downarrow_{\mathbf{O}} \lceil n_i \rceil$, for some $n_i \in \mathbb{N}$ ($1 \leq i \leq p$); by hypothesis $\lceil g_j \rceil N_1 \dots N_p \Downarrow_{\mathbf{O}} \lceil g_j(n_1, \dots, n_p) \rceil$ ($1 \leq j \leq m$).

Let $R_i \in \Lambda$ be such that $R_i \Downarrow_{\mathbf{O}} \lceil n_i \rceil$, for some $n_i \in \mathbb{N}$ ($1 \leq i \leq m$); by hypothesis $\lceil h \rceil R_1 \dots R_m \Downarrow_{\mathbf{O}} \lceil h(n_1, \dots, n_m) \rceil$, so in particular

$$\lceil h \rceil(\lceil g_1 \rceil N_1 \dots N_p) \dots (\lceil g_m \rceil N_1 \dots N_p) \Downarrow_{\mathbf{O}} \lceil h(g_1(n_1, \dots, n_p), \dots, g_m(n_1, \dots, n_p)) \rceil.$$

But $\lceil f \rceil N_1 \dots N_p \Downarrow_{\mathbf{O}} R$ if and only if $\lceil h \rceil(\lceil g_1 \rceil N_1 \dots N_p) \dots (\lceil g_m \rceil N_1 \dots N_p) \Downarrow_{\mathbf{O}} R$ (by $(\text{head})^+$), so the proof is done since h, g_1, \dots, g_m are total. □

In order to represent the functions built by primitive recursion and by minimalization, a “fixed-point operator” is needed, that work well in all the call-by-name reduction machines. We already proved that in the $\lambda\Lambda$ -calculus every term has a fixed-point, and we showed, in the proof of Theorem 2.1.8, an operator building it, namely the term Y .

But, while $YM =_{\Lambda} M(YM)$, it does not hold that $YM \rightarrow_{\Lambda}^{*} M(YM)$, which is a necessary condition for using it as recursion operator in a call-by-name reduction machine.

So in the next theorem a further fixed-point operator, suitable for our purposes, is defined.

Theorem 16.1.3. *Let $Y_{\Lambda} \equiv (\lambda xy. y(xx))(\lambda xy. y(xx))$. If $M \in \Lambda$ then $Y_{\Lambda}M \rightarrow_{\Lambda}^{*} M(Y_{\Lambda}M)$; moreover, $Y_{\Lambda}M \Downarrow_{\mathbf{O}} R$ if and only if $M(Y_{\Lambda}M) \Downarrow_{\mathbf{O}} R$.*

Proof. Trivial. □

The following lemma shows how Y_{Λ} can be used for mimicking primitive recursion.

Lemma 16.1.4. Let $h : \mathbb{N}^{m+2} \rightarrow \mathbb{N}$ and $g : \mathbb{N}^m \rightarrow \mathbb{N}$ be \mathbf{O} -representable primitive recursive functions. The following function is \mathbf{O} -representable:

$$f(k, x_1, \dots, x_m) = \begin{cases} g(x_1, \dots, x_m) & \text{if } k = 0, \\ h(f(k - 1, x_1, \dots, x_m), k - 1, x_1, \dots, x_m) & \text{otherwise.} \end{cases}$$

Proof. By hypothesis there are terms $\ulcorner h \urcorner$ and $\ulcorner g \urcorner$ representing h and g . We will prove that $\ulcorner f \urcorner$ is \mathbf{O} -represented by $Y_A P$, where P is:

$$\lambda t y x_1 \dots x_m. \text{Test } y (\ulcorner g \urcorner x_1 \dots x_m) (\ulcorner h \urcorner (t(\text{Pred } y) x_1 \dots x_m) (\text{Pred } y) x_1 \dots x_m).$$

Let $N_i \Downarrow_{\mathbf{O}} \ulcorner n_i \urcorner$, $Q \Downarrow_{\mathbf{O}} \ulcorner k \urcorner$ for some $k, n_i \in \mathbb{N}$ ($1 \leq i \leq m$); the proof will be given by induction on k .

Let $k = 0$. $Y_A P Q N_1 \dots N_m \Downarrow_{\mathbf{O}} R$ if and only if $P(Y_A P) Q N_1 \dots N_m \Downarrow_{\mathbf{O}} R$ (by (head) $^+$), if and only if
 $\text{Test } Q (\ulcorner g \urcorner x_1 \dots x_m) (\ulcorner h \urcorner (Y_A P (\text{Pred } Q) x_1 \dots x_m) (\text{Pred } Q) x_1 \dots x_m) \Downarrow_{\mathbf{O}} R$
(by (head) $^+$), if and only if
 $\top (\ulcorner g \urcorner x_1 \dots x_m) (\ulcorner h \urcorner (Y_A P (\text{Pred } Q) N_1 \dots N_m) (\text{Pred } Q) N_1 \dots N_m) \Downarrow_{\mathbf{O}} R$
(by Property 15.2.7.(i), since $\text{Test } Q \Downarrow_{\mathbf{O}} \top$), if and only if $\ulcorner g \urcorner N_1 \dots N_m \Downarrow_{\mathbf{O}} R$.
But, by hypothesis $\ulcorner g \urcorner N_1 \dots N_m \Downarrow_{\mathbf{O}} \ulcorner g(n_1, \dots, n_m) \urcorner$.

Let $k > 0$. $Y_A P Q N_1 \dots N_m \Downarrow_{\mathbf{O}} R$ if and only if $P(Y_A P) Q N_1 \dots N_m \Downarrow_{\mathbf{O}} R$ (by (head) $^+$), if and only if
 $\text{Test } Q (\ulcorner g \urcorner N_1 \dots N_m) (\ulcorner h \urcorner (Y_A P (\text{Pred } Q) N_1 \dots N_m) (\text{Pred } Q) N_1 \dots N_m) \Downarrow_{\mathbf{O}} R$
(by (head) $^+$), if and only if
 $\mathsf{F} (\ulcorner g \urcorner N_1 \dots N_m) (\ulcorner h \urcorner (Y_A P (\text{Pred } Q) N_1 \dots N_m) (\text{Pred } Q) N_1 \dots N_m) \Downarrow_{\mathbf{O}} R$
(by Property 15.2.7.(i), since $\text{Test } Q \Downarrow_{\mathbf{O}} \mathsf{F}$), if and only if
 $\ulcorner h \urcorner (Y_A P (\text{Pred } Q) N_1 \dots N_m) (\text{Pred } Q) N_1 \dots N_m \Downarrow_{\mathbf{O}} R$ (by (head) $^+$).
But, by induction $(Y_A P) (\text{Pred } Q) N_1 \dots N_m \Downarrow_{\mathbf{O}} \ulcorner f(k - 1, n_1, \dots, n_m) \urcorner$; thus
 $R \equiv \ulcorner h(f(k - 1, x_1, \dots, x_m), k - 1, x_1, \dots, x_m) \urcorner$, since by hypothesis $\ulcorner h \urcorner$ is an \mathbf{O} -representation of h . \square

Thus all primitive recursive functions are representable in the considered settings.

In order to represent the composition of partial functions in a call-by-name setting, the main problem is to make the representation “strict”; namely, when a function is applied to an undefined argument then its evaluation must diverge. The proposed solution takes into account the fact that terms representing natural numbers are in A -head normal form and so are A -solvable.

Lemma 16.1.5. If $M \Downarrow_{\mathbf{O}} \ulcorner n \urcorner$ then $M K II \Downarrow_{\mathbf{O}} I$.

Proof. By Property 15.2.7.(i), $M K II \Downarrow_{\mathbf{O}} R$ if and only if $\ulcorner n \urcorner K II \Downarrow_{\mathbf{O}} R$. Thus by (head) $^+$ the proof follows by observing the shape of $\ulcorner n \urcorner$. \square

Then the representation of a function built by composition of partial functions is a term with the following operational behaviour: first it checks if all its arguments are defined and, in case at least one is undefined then it diverges; otherwise it computes the result.

Lemma 16.1.6. *Let $h : \mathbb{N}^n \rightarrow \mathbb{N}$ and $g_1, \dots, g_n : \mathbb{N}^m \rightarrow \mathbb{N}$ be \mathbf{O} -representable partial recursive functions. The function defined from them by composition, namely*

$$f(x_1, \dots, x_m) = h(g_1(x_1, \dots, x_m), \dots, g_n(x_1, \dots, x_m)),$$

is \mathbf{O} -representable.

Proof. Let

$$\begin{aligned} F &\equiv \lambda x_1 \dots x_m. \lceil h \rceil (\lceil g_1 \rceil x_1 \dots x_m) \dots (\lceil g_n \rceil x_1 \dots x_m) \quad \text{and} \\ \lceil f \rceil &\equiv \lambda x_1 \dots x_m. (\lceil g_1 \rceil x_1 \dots x_m KII) \dots (\lceil g_n \rceil x_1 \dots x_m KII) (Fx_1 \dots x_m). \end{aligned}$$

Let $N_i \Downarrow_{\mathbf{O}} \lceil n_i \rceil$ ($1 \leq i \leq m$). $\lceil f \rceil N_1 \dots N_m \Downarrow_{\mathbf{O}} R$ if and only if $(\lceil g_1 \rceil N_1 \dots N_m KII) \dots (\lceil g_n \rceil N_1 \dots N_m KII) (FN_1 \dots N_m) \Downarrow_{\mathbf{O}} R$ (by (head) $^+$). Let j be the minimum integer such that $\lceil g_j \rceil N_1 \dots N_m \uparrow_{\mathbf{O}}$ ($1 \leq j \leq m$). $(\lceil g_1 \rceil N_1 \dots N_m KII) \dots (\lceil g_n \rceil N_1 \dots N_m KII) (FN_1 \dots N_m) \Downarrow_{\mathbf{O}} R$ if and only if $I(\lceil g_2 \rceil N_1 \dots N_m KII) \dots (\lceil g_n \rceil N_1 \dots N_m KII) (FN_1 \dots N_m) \Downarrow_{\mathbf{O}} R$ (by (head) $^+$) if and only if $(\lceil g_2 \rceil N_1 \dots N_m KII) \dots (\lceil g_n \rceil N_1 \dots N_m KII) (FN_1 \dots N_m) \Downarrow_{\mathbf{O}} R$ (by (head)) if and only if $(\lceil g_j \rceil N_1 \dots N_m KII) \dots (\lceil g_n \rceil N_1 \dots N_m KII) (FN_1 \dots N_m) \Downarrow_{\mathbf{O}} R$. But $\lceil g_j \rceil N_1 \dots N_m \uparrow_{\mathbf{O}}$ implies $\lceil g_j \rceil N_1 \dots N_m$ is a Λ -unsolvable of order 0 in case $\mathbf{O} \equiv \mathbf{L}$. So $(\lceil g_j \rceil N_1 \dots N_m KII) \dots (\lceil g_n \rceil N_1 \dots N_m KII) (FN_1 \dots N_m)$ is a Λ -unsolvable of order 0, respectively, and so in all cases,

$$(\lceil g_j \rceil N_1 \dots N_m KII) \dots (\lceil g_n \rceil N_1 \dots N_m KII) (FN_1 \dots N_m) \uparrow_{\mathbf{O}}.$$

In case $\lceil g_i \rceil N_1 \dots N_m \Downarrow_{\mathbf{O}}$, for all i ($1 \leq i \leq n$), $\lceil f \rceil N_1 \dots N_m \Downarrow_{\mathbf{O}} R$ if and only if $(FN_1 \dots N_m) \Downarrow_{\mathbf{O}} R$, and the proof follows the same line as that of Lemma 16.1.2. \square

Finally, we check the computability of functions defined by minimalization. Let $P \equiv \lambda thxy. \text{Test}(hxy)y(thx(\text{Succ } y))$.

Lemma 16.1.7. *Let $h : \mathbb{N}^2 \rightarrow \mathbb{N}$ be an \mathbf{O} -representable total recursive function. Let N and Q be such that $N \Downarrow_{\mathbf{O}} \lceil n \rceil$ and $Q \Downarrow_{\mathbf{O}} \lceil k \rceil$.*

- (i) *If $h(n, k) = 0$ then $Y_{\Lambda} P \lceil h \rceil N Q \Downarrow_{\mathbf{O}} \lceil k \rceil$.*
- (ii) *Let $h(n, k) \neq 0$; so*

$$Y_{\Lambda} P \lceil h \rceil N Q \Downarrow_{\mathbf{O}} R \text{ if and only if } Y_{\Lambda} P \lceil h \rceil N (\text{Succ } Q) \Downarrow_{\mathbf{O}} R.$$

Proof. (i) $Y_{\Lambda} P \lceil h \rceil N Q \Downarrow_{\mathbf{O}} R$ if and only if $P(Y_{\Lambda} P \lceil h \rceil N Q) \Downarrow_{\mathbf{O}} R$ (by (head) $^+$) if and only if $\text{Test}(\lceil h \rceil N Q) Q(Y_{\Lambda} P \lceil h \rceil N (\text{Succ } Q)) \Downarrow_{\mathbf{O}} R$, (by (head) $^+$) if and only if $Q \Downarrow_{\mathbf{O}} R$ (since $\lceil h \rceil N Q \Downarrow_{\mathbf{O}} \lceil 0 \rceil$, always by (head) $^+$), but $Q \Downarrow_{\mathbf{O}} \lceil k \rceil$, so the proof is done.

- (ii) The proof is similar to that of the previous point, by using the fact that h is a total function. \square

Property 16.1.8. Let $h : \mathbb{N}^2 \rightarrow \mathbb{N}$ be an \mathbf{O} -representable total recursive function.

- (i) If $f(n) = \mu y[h(n, y) = 0]$ is defined then $Y_A P^\Gamma h \sqcap N \text{Zero} \Downarrow_{\mathbf{O}} \lceil f(n) \rceil$, for every N such that $N \Downarrow_{\mathbf{O}} \lceil n \rceil$.
- (ii) If $f(n) = \mu y[h(n, y) = 0]$ is undefined then $Y_A P^\Gamma h \sqcap N \text{Zero} \Uparrow_{\mathbf{O}}$, for all N such that $N \Downarrow_{\mathbf{O}} \lceil n \rceil$.

Proof. (i) Let $f(n) = k$, thus k is the minimum integer such that $h(n, k) = 0$.

By induction on k , the proof follows by Lemma 16.1.7.

- (ii) By Lemma 16.1.7. \square

Lemma 16.1.9. Let $h : \mathbb{N}^2 \rightarrow \mathbb{N}$ be an \mathbf{O} -representable total recursive function. $f(x) = \mu y[h(x, y) = 0]$ is \mathbf{O} -representable.

Proof. Let $\lceil f \rceil \equiv \lambda x. Y_A P^\Gamma h \sqcap x \text{Zero}$, where

$$P \equiv \lambda t hxy. \text{Test}(hxy)y(thx(\text{Succ } y)).$$

Then the proof follows directly by Property 16.1.8. \square

So the \mathbf{O} -representability of all partial recursive function follows.

Theorem 16.1.10. Let $\mathbf{O} \in \{\mathbf{H}, \mathbf{N}, \mathbf{L}\}$.

All partial recursive functions are \mathbf{O} -representable.

16.2 Call-by-Value Computational Completeness

Now let us prove that also the \mathbf{V} -reduction machine can compute all partial recursive functions. We will point out just the differences between call-by-name and call-by-value computability. The most interesting difference occurs in the coding of recursion and minimalization, which is done through a fixed-point operator in the call-by-name setting. We have seen that a call-by-value fixed-point operator is such that, when applied to a Γ -valuable term, it is operationally equal to a not Γ -valuable term. In fact, if we think to interpret terms as function, the undefined value is the fixed-point of every function, when parameters are passed by value. So in order to deal with both recursion and minimalization, we will use a call-by-value recursion operator, whose behaviour was been defined at the end of Sect. 3.1. Other quite small differences are in the composition of partial functions.

Let $Y_\Gamma \equiv (\lambda xf.f(\lambda z.xxfz))(\lambda xf.f(\lambda z.xxfz))$; clearly Y_Γ is a recursion operator. The following theorem shows its operational behaviour.

Theorem 16.2.1 (Recursion).

Let $Y_\Gamma \equiv (\lambda x f. f(\lambda z. x x f z))(\lambda x f. f(\lambda z. x x f z))$.

If $M \in \Gamma$ then $Y_\Gamma M \Downarrow_{\mathbf{V}}$ if and only if $M(\lambda z. Y_\Gamma M z) \Downarrow_{\mathbf{V}}$, where $z \notin FV(M)$.

Proof. Easy. \square

Now we can show that the $\lambda\Gamma$ -calculus is computationally complete.

Lemma 16.2.2. *Primitive recursive functions are \mathbf{V} -representable.*

Proof. The proof follows the same lines as Lemmas 16.1.1, 16.1.2 and 16.1.4, taking into account that Y_Γ must be used instead of Y_A , and moreover taking into account the behaviour of the \mathbf{V} -reduction machine. \square

The extension to partial functions is easier than in the call-by-name case. In fact the mathematical functions are naturally “strict”, in the sense that a function diverges if one of its arguments diverges, and this behaviour is exactly the behaviour of the \mathbf{V} -evaluation. In fact the following property holds.

Property 16.2.3. Let $M, N \in \Lambda^0$; $M \uparrow_{\mathbf{V}}$ implies $NM \uparrow_{\mathbf{V}}$.

Proof. Trivial, by the rule (*head*). \square

Lemma 16.2.4. Let $h : \mathbb{N}^n \rightarrow \mathbb{N}$ and $g_1, \dots, g_n : \mathbb{N}^m \rightarrow \mathbb{N}$ be \mathbf{V} -representable partial recursive functions. The function f defined by composition from them in the following way:

$$f(x_1, \dots, x_m) = h(g_1(x_1, \dots, x_m), \dots, g_n(x_1, \dots, x_m))$$

is \mathbf{V} -representable.

Proof. Let $H \equiv \lambda x_1 \dots x_m. \Gamma h \vdash (\Gamma g_1 \vdash x_1 \dots x_m) \dots (\Gamma g_n \vdash x_1 \dots x_m)$; it is easy to check that H is a \mathbf{V} -representation of f , by Property 16.2.3. \square

Finally, we check the \mathbf{V} -computability of functions defined by minimization. Let $F \equiv \lambda thxy. \text{Test}(hIxy)(\lambda v.y)(\lambda u.thx(\text{Succ } y))I$.

Lemma 16.2.5. Let $h : \mathbb{N}^2 \rightarrow \mathbb{N}$ be a \mathbf{V} -representable total recursive function such that $h(n, k) = m$ where $n, k, m \in \mathbb{N}$.

(i) If $m = 0$ then $(Y_\Gamma F)(\lambda z. \Gamma h \vdash) \Gamma n \vdash \Gamma k \vdash \Downarrow_{\mathbf{V}} \Gamma k \vdash$.

(ii) Let $m \neq 0$.

$(Y_\Gamma F)(\lambda z. \Gamma h \vdash) \Gamma n \vdash \Gamma k \vdash \Downarrow_{\mathbf{V}}$ if and only if $(Y_\Gamma F)(\lambda z. \Gamma h \vdash) \Gamma n \vdash \Gamma k + 1 \vdash \Downarrow_{\mathbf{V}}$.

Proof. (i) $(Y_\Gamma F)(\lambda z. \Gamma h \vdash) \Gamma n \vdash \Gamma k \vdash \Downarrow_{\mathbf{V}} R$ if and only if

$F(\lambda z. Y_\Gamma F z)(\lambda z. \Gamma h \vdash) \Gamma n \vdash \Gamma k \vdash \Downarrow_{\mathbf{V}} R$ if and only if.

$\text{Test}((\lambda z. \Gamma h \vdash) I \Gamma n \vdash \Gamma k \vdash)(\lambda v. \Gamma k \vdash)(\lambda u. (\lambda z. Y_\Gamma F z)(\lambda z. \Gamma h \vdash) \Gamma n \vdash (\text{Succ } \Gamma k \vdash)) I \Downarrow_{\mathbf{V}} R$ by two applications of rule (*head*), if and only if $(\lambda v. \Gamma k \vdash) I \Downarrow_{\mathbf{V}} R$ if and only if $\Gamma k \vdash \Downarrow_{\mathbf{V}} R$, always by (*head*), and this implies $R \equiv \Gamma k \vdash$.

- (ii) $(Y_F)(\lambda z.\Gamma h^\neg)\Gamma n^\neg\Gamma k^\neg \Downarrow_{\mathbf{V}}$ if and only if $F(\lambda z.Y_F Fz)(\lambda z.\Gamma h^\neg)\Gamma n^\neg\Gamma k^\neg \Downarrow_{\mathbf{V}}$
if and only if
 $\text{Test}((\lambda z.\Gamma h^\neg)I\Gamma n^\neg\Gamma k^\neg)(\lambda v.\Gamma k^\neg)(\lambda u.(\lambda z.Y_F Fz)(\lambda z.\Gamma h^\neg)\Gamma n^\neg(\text{Succ } \Gamma k^\neg))I \Downarrow_{\mathbf{V}}$
if and only if $\mathsf{F}(\lambda v.\Gamma k^\neg)(\lambda u.(\lambda z.Y_F Fz)(\lambda z.\Gamma h^\neg)\Gamma n^\neg(\text{Succ } \Gamma k^\neg))I \Downarrow_{\mathbf{V}}$ always by (head), if and only if $(\lambda u.(\lambda z.Y_F Fz)(\lambda z.\Gamma h^\neg)\Gamma n^\neg(\text{Succ } \Gamma k^\neg))I \Downarrow_{\mathbf{V}}$
if and only if $(\lambda z.Y_F Fz)(\lambda z.\Gamma h^\neg)\Gamma n^\neg(\text{Succ } \Gamma k^\neg) \Downarrow_{\mathbf{V}}$ And, again by (head),
if and only if $Y_F(\lambda z.\Gamma h^\neg)\Gamma n^\neg(\text{Succ } \Gamma k^\neg) \Downarrow_{\mathbf{V}}$. \square

The representation of a function defined by minimalization is different from the call-by-name case, since the term P , performing the iteration in the call-by-name case, does not work correctly in the call-by-value one. In fact, in P , the second argument of the function Test is not a value. The term F is a slight modification of P , playing the desired role in this particular setting.

Lemma 16.2.6. *Let $h : \mathbb{N}^2 \rightarrow \mathbb{N}$ be a \mathbf{V} -representable total recursive function. $f(x) = \mu y[h(x, y) = 0]$ is \mathbf{V} -representable.*

Proof. Let $\Gamma f^\neg \equiv \lambda x.Y_F F(\lambda z.\Gamma h^\neg)x\text{Zero}$, where

$$F \equiv \lambda t h x y. \text{Test}(h I x y)(\lambda v. y)(\lambda u. t(h x (\text{Succ } y)))I.$$

The proof follows by Lemma 16.2.5, in the same way as the proof of Lemma 16.1.9. \square

Theorem 16.2.7. *All partial recursive functions are \mathbf{V} -representable.*

16.3 Historical Remarks

In the literature, computability, for both the call-by-name and the call-by-value λ -calculi, was defined starting from an approach different from the present one. In fact, the representation of computable functions was developed inside a theory, while we have chosen an operational point of view. The next definition is the classical one for the λA -calculus.

Definition 16.3.1. *Let ϕ be a partial recursive function with arity $p \in \mathbb{N}$. ϕ is λ -definable if and only if there is a term $\Gamma \phi^\neg$ such that*

- $\phi(n_1, \dots, n_p)$ defined implies $\Gamma \phi^\neg \Gamma n_1^\neg \dots \Gamma n_p^\neg =_A \Gamma \phi(n_1, \dots, n_p)^\neg$;
- $\phi(n_1, \dots, n_p)$ undefined implies $\Gamma \phi^\neg \Gamma n_1^\neg \dots \Gamma n_p^\neg$ is an unsolvable term.

Inside the λA -calculus, the first numeral system was introduced by Church. In it, Γn^\neg is the term $\lambda f x. \underbrace{f(\dots(f x\dots))}_n$, which represents the n -th iteration of a function f applied to an argument x . This numeral system has been the starting point for the first λ -representation of partial recursive

functions, given by Kleene, in the λ -I-calculus [57]. The idea of using fixed-point operators to represent primitive recursion and minimalization comes from Turing [95].

The system we proposed in Section 15.2 is a slight modification of the numeral system of Barendregt [9], recalled in the following definition.

Definition 16.3.2 (Barendregt numeral system).

Let $\mathfrak{B} \equiv \langle \{T, F\}, \text{Zero}^{\mathfrak{B}}, \text{Succ}^{\mathfrak{B}}, \text{Test}^{\mathfrak{B}}, \text{Pred}^{\mathfrak{B}} \rangle$, where

- $T \equiv \lambda xy.x$ and $F \equiv \lambda xy.y$;
- $\text{Zero}^{\mathfrak{B}} \equiv \lambda x.x$;
- $\text{Succ}^{\mathfrak{B}} \equiv \lambda n.[F, n] \equiv \lambda nx.xFn$;
- $\text{Test}^{\mathfrak{B}} \equiv \lambda n.nT$;
- $\text{Pred}^{\mathfrak{B}} \equiv \lambda n.nF$.

\mathfrak{B} is a **N**-numeral system but is neither a **L**-numeral system nor a **H**-numeral system. In fact, the Barendregt representation of 1 is $\lceil 1 \rceil \equiv \lambda x.xF\text{Zero}^{\mathfrak{B}}$, and $\text{Succ}^{\mathfrak{B}}\text{Zero}^{\mathfrak{B}} \Downarrow_{\mathbf{O}} \lambda x.xF\text{Zero}^{\mathfrak{B}}$ where $\mathbf{O} \in \{\mathbf{L}, \mathbf{H}\}$; nevertheless

$$\begin{aligned} \text{Succ}^{\mathfrak{B}}(\text{Pred}^{\mathfrak{B}}\lceil 1 \rceil) &\Downarrow_{\mathbf{H}} \lambda x.x((\lambda n.nF)\lceil 1 \rceil) \\ \text{Succ}^{\mathfrak{B}}(\text{Pred}^{\mathfrak{B}}\lceil 1 \rceil)\lceil n \rceil &\Downarrow_{\mathbf{L}} \lambda x.x((\lambda n.nF)\lceil 1 \rceil) \end{aligned}$$

but $\lambda x.x((\lambda n.nF)\lceil 1 \rceil) \not\equiv \lceil 1 \rceil$, although

$$\lambda x.x((\lambda n.nF)\lceil 1 \rceil) =_{\Lambda} \lceil 1 \rceil.$$

As far as the call-by-value computability is concerned, Plotkin [78] was the first to point out the difference between call-by-name and call-by-value recursion. He proposed the following recursion operator:

$$\lambda f.(\lambda x.f(\lambda z.xxz))(\lambda x.f(\lambda z.xxz)),$$

which works similarly to Y_{Γ} . The computability in the theory $=_{\Gamma}$ was completely developed by Paolini [72], using Y_{Γ} and the Barendregt numeral system.

Bibliography

1. Samson Abramsky. Domain theory in logical form. *Annals of Pure and Applied Logic*, 51(1-2):1–77, 1991.
2. Samson Abramsky and Luke Ong. Full abstraction in the lazy lambda calculus. *Information and Computation*, 105(2):159–267, 1993.
3. Fabio Alessi. *Strutture di Tipi, Teorie dei Domini e Modelli del Lambda Calcolo*. Tesi di dottorato di ricerca in informatica, Università di Milano e Torino, 1990.
4. Fabio Alessi, Mariangiola Dezani-Ciancaglini, and Furio Honsell. Filter models and easy terms. In Antonio Restivo, Simona Ronchi Della Rocca, and Luca Roversi, editors, *Theoretical Computer Science, 7th Italian Conference, ICTCS 2001, Torino, Italy, October 4-6, 2001*, volume 2202 of *Lecture Notes in Computer Science*, pages 17–37. Springer-Verlag, 2001.
5. Roberto M. Amadio and Pierre-Louis Curien. *Domains and Lambda-Calculi*, volume 46 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, 1998.
6. Andrea Asperti, Cecilia Giovannetti, and Andrea Naletto. The bologna optimal higher-order machine. Technical Report UBLCS-95-9, University of Bologna, Department of Computer Science, March 1995.
7. Andrea Asperti and Stefano Guerrini. *The Optimal Implementation of Functional Programming Languages*. Cambridge University Press, Cambridge, 1998.
8. Andrea Asperti and Giuseppe Longo. *Categories, Types, and Structures: An Introduction to Category Theory for the Working Computer Scientist*. Foundations of Computing Series. The MIT Press, Cambridge, MA, 1991.
9. Henk Barendregt. *The Lambda Calculus: Its Syntax and Semantics* (2nd edition). North-Holland, Amsterdam, 1984.
10. Henk Barendregt, Mario Coppo, and Mariangiola Dezani-Ciancaglini. A filter lambda model and the completeness of type assignment. *The Journal of Symbolic Logic*, 48(4):931–940, December 1983.
11. Olivier Bastonero and Xavier Gouy. Strong stability and the incompleteness of stable models for lambda-calculus. *Annals of Pure and Applied Logic*, 100(1-3):247–277, 1999.
12. Olivier Bastonero, Alberto Pravato, and Simona Ronchi Della Rocca. Structures for lazy semantics. In Gries and de Roever, editors, *Programming Concepts and Methods*, pages 30–48. Chaptman & Hall, 1998.
13. Chantal Berline. From computation to foundations via functions and application: The λ -calculus and its webbed models. *Theoretical Computer Science*, 249(1):81–161, October 2000.
14. Gérard Berry. Stable models of typed lambda-calculi. In Giorgio Ausiello and Corrado Böhm, editors, *Automata, Languages and Programming, Fifth Colloquium, ICALP, Udine, Italy, July 17-21, 1978*, volume 62 of *Lecture Notes in Computer Science*, pages 72–89. Springer-Verlag, 1978.

15. Corrado Böhm. Alcune proprietà delle forme $\beta\eta$ -normali nel λK -calculus. Pubblicazione n. 696, Istituto per le Applicazioni del Calcolo, Roma, 1968.
16. Corrado Böhm and Mariangiola Dezani-Ciancaglini. A CUCH-machine: the automatic treatment of bound variables. *International Journal of Computer and Information Sciences*, 1(2):171–191, June 1972.
17. Corrado Böhm and Mariangiola Dezani-Ciancaglini. λ -terms as total or partial functions on normal forms. In G. Goos and J. Hartmanis, editors, *λ -Calculus and Computer Science Theory*, volume 37 of *Lecture Notes in Computer Science*, pages 96–121, Berlin, DE, 1975. Springer-Verlag.
18. Corrado Böhm, Mariangiola Dezani-Ciancaglini, P. Peretti, and Simona Ronchi Della Rocca. A discrimination algorithm inside λ -calculus. *Theoretical Computer Science*, 8(3):271–291, 1978.
19. Corrado Böhm and W. Gross. Introduction to the CUCH. In E. R. Caianiello, editor, *Automata Theory*, pages 35–65. Academic Press, New York, 1966.
20. Corrado Böhm and Adolfo Piperno. Characterizing X-separability and one-side invertibility in $\lambda\beta\Omega$ -calculus. In *Proceedings, Third Annual Symposium on Logic in Computer Science - LICS'88*, pages 91–103, Edinburgh, Scotland, 5–8 July 1988. IEEE Computer Society Press.
21. Corrado Böhm, Adolfo Piperno, and Stefano Guerrini. Lambda-definition of function(al)s by normal forms. In Donald Sannella, editor, *Programming Languages and Systems-ESOP'94, 5th European Symposium on Programming*, volume 788 of *Lecture Notes in Computer Science*, pages 135–149. Springer-Verlag, 1994.
22. Corrado Böhm, Adolfo Piperno, and Enrico Tronci. Solving equations in lambda-calculus. In *Logic Colloquium'88*, Amsterdam, 1988. North-Holland.
23. Antonio Bucciarelli and Thomas Ehrhard. A theory of sequentiality. *Theoretical Computer Science*, 113(2):273–291, 7 June 1993.
24. Rod Burstall and Furio Honsell. Operational semantics in a natural deduction setting. In Gérard Huet and Gordon Plotkin, editors, *Logical Frameworks*, pages 185–214, Cambridge, 1991. Cambridge University Press.
25. Alonzo Church. *The Calculi of Lambda Conversion*, volume 6 of *Annals of Mathematical Studies*. Princeton University Press, Princeton, 1941. Reprinted by University Microfilms Inc., Ann Arbor, MI in 1963 and by Klaus Reprint Corp., New York in 1965.
26. Alonzo Church and J. Barkley Rosser. Some properties of conversion. *Transactions of the American Mathematical Society*, 39:472–482, 1936.
27. Mario Coppo and Mariangiola Dezani-Ciancaglini. An extension of the basic functionality theory for the λ -calculus. *Notre Dame Journal of Formal Logic*, 21(4):685–693, October 1980.
28. Mario Coppo, Mariangiola Dezani-Ciancaglini, Furio Honsell, and Giuseppe Longo. Extended type structure and filter lambda models. In G. Lolli, G. Longo, and A. Marcja, editors, *Logic Colloquim'82*, pages 241–262. Elsevier Science Publishers B.V. (North-Holland), Amsterdam, 1984.
29. Mario Coppo, Mariangiola Dezani-Ciancaglini, and Simona Ronchi Della Rocca. (Semi)-separability of finite sets of terms in Scott's D_∞ -models of the λ -calculus. In Giorgio Ausiello and Corrado Böhm, editors, *Automata, Languages and Programming, Fifth Colloquium*, volume 62 of *Lecture Notes in Computer Science*, pages 142–164, Udine, Italy, 17–21 July 1978. Berlin, Springer-Verlag.
30. Mario Coppo, Mariangiola Dezani-Ciancaglini, and Maddalena Zacchi. Type theories, normal forms and D_∞ lambda models. *Information and Computation*, 72(2):85–116, 1987.
31. Roy L. Crole. *Categories for Types*. Cambridge University Press, Cambridge, 1993.

32. Pierre Louis Curien. Sur l'eta-expansion infinie. *Comptes Rendus de l'Académie des Sciences*, to appear.
33. Pierre-Louis Curien and Hugo Herbelin. The duality of computation. In *Proceedings of the ACM Sigplan International Conference on Functional Programming (ICFP-00)*, volume 35(9) of *ACM Sigplan Notices*, pages 233–243, Montréal, Canada, September 18–21 2000. ACM Press.
34. Haskell B. Curry and Robert Feys. *Combinatory Logic - Volume 1*. Studies in Logic and the Foundations of Mathematics. Elsevier, North-Holland, (Amsterdam, London, New York), L. E. J. Brouwer, E. W. Beth, A. Heyting editors, edition, 1958. With two sections by William Craig. Second edition, 1968.
35. Haskell B. Curry, J. Roger Hindley, and Jonathan P. Seldin. *Combinatory Logic - Volume 2*, volume 65 of *Studies in Logic and the Foundations of Mathematics*. Elsevier, North-Holland, (Amsterdam, London, New York), A. Heyting, H. J. Keisler, A. Mostowski, A. Robinson, P. Suppes editors, edition, 1972.
36. René David and Karim Nour. A syntactical proof of the operational equivalence of two λ -terms. *Theoretical Computer Science*, 180(1–2):371–375, 10 June 1997.
37. Mariangiola Dezani-Ciancaglini, Silvia Ghilezan, and Silvia Likavec. Behavioural inverse limit models. *Theoretical Computer Science*, 2003. To appear.
38. Mariangiola Dezani-Ciancaglini, Furio Honsell, and Fabio Alessi. A complete characterization of complete intersection-type preorders. *ACM Transactions on Computational Logic*, 4(1):120–147, January 2003.
39. Mariangiola Dezani-Ciancaglini, Furio Honsell, and Simona Ronchi Della Rocca. Models for theories of functions strictly depending on all their arguments. *The Journal of Symbolic Logic*, 51(3):845–846, 1986. (Abstract).
40. Mariangiola Dezani-Ciancaglini and Stefania Lusin. Intersection types and lambda theories. In *Electronic Proceedings of WIT'02* (<http://www.irit.fr/zeno/WIT2002/proceedings.shtml>), 2002.
41. Roberto Di Cosmo. A brief history of rewriting with extensionality. In Fairouz Kamareddine, editor, *International Summer School on Type Theory and Rewriting*, Glasgow, 1996. Kluwer.
42. Pietro Digianantonio. Game semantics for the pure lazy λ -calculus. In Samson Abramsky, editor, *Typed Lambda Calculi and Applications: 5th International Conference, TLCA 2001 Krakow, Poland, May 2–5, 2001*, volume 2044 of *Lecture Notes in Computer Science*, pages 106–120, Berlin, June 2003. Springer-Verlag.
43. Pietro Digianantonio, Gianluca Franco, and Furio Honsell. Game semantics for untyped $\lambda\beta\eta$ -calculus. In Jean-Yves Girard, editor, *Typed Lambda Calculi and Applications: 4th International Conference, TLCA'99, L'Aquila, Italy, April 1999*, volume 1581 of *Lecture Notes in Computer Science*, pages 114–128, Berlin, July 2003. Springer-Verlag.
44. Lavinia Egidi, Furio Honsell, and Simona Ronchi Della Rocca. Operational, denotational and logical descriptions: a case study. *Fundamenta Informaticæ*, 16(2):149–170, 1992.
45. Matthias Felleisen and Daniel P. Friedman. A syntactic theory of sequential state. *Theoretical Computer Science*, 69(3):243–287, 1989. Preliminary version in *Proc. 14th ACM Symp. Principles of Programming Languages* 1987, pages 314–325.
46. Matthias Felleisen, Daniel P. Friedman, Eugene E. Kohlbecker, and Bruce F. Duba. A syntactic theory of sequential control. *Theoretical Computer Science*, 52:205–237, 1987.
47. Jean-Yves Girard. The system F of variable types, fifteen years later. *Theoretical Computer Science*, 45(2):159–192, 1986.

48. Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.
49. J. Roger Hindley. *Basic Simple Type Theory*, volume 42 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, UK, 1997.
50. J. Roger Hindley and Giuseppe Longo. Lambda calculus models and extensibility. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 26:289–310, 1980.
51. J. Roger Hindley and Jonathan P. Seldin. *Introduction to Combinators and λ -Calculus*, volume 1 of *London Mathematical Society Student Texts*. Cambridge University Press, Cambridge, UK, 1986.
52. Furio Honsell and Simona Ronchi della Rocca. Reasoning about interpretation in qualitative lambda-models. In M. Broy and C.B. Jones, editors, *Proceeding of IFIP 2.2 Working Conference on Programming Concepts and Methods*, pages 505–521, Sea of Galilee, Israel, 1990. North Holland.
53. Furio Honsell and Simona Ronchi Della Rocca. An approximation theorem for topological lambda models and the topological incompleteness of lambda calculus. *Journal of Computer and System Sciences*, 45(1):49–75, August 1992.
54. J. Martin E. Hyland. A syntactic characterization of the equality in some models of the lambda calculus. *Journal of the London Mathematical Society*, 2(12):361–370, 1976.
55. Gilles Kahn. Natural semantics. In *Symposium on Theoretical Aspects of Computer Science*, volume 247 of *Lecture Notes in Computer Science*, pages 22–39, 1987.
56. A. J. Kfoury, Robert A. Moll, and Michael A. Arbib. *A Programming Approach to Computability*. Texts and Monographs in Computer Science. Springer-Verlag, Berlin, 1986. Second edition.
57. Stephen Cole Kleene. Lambda definability and recursiveness. *Duke Mathematical Journal*, 2:340–353, 1936.
58. Jan Willem Klop. *Combinatory Reduction Systems*, volume 127 of *Mathematical Centre Tracts*. Mathematisch Centrum, 413 Kruislaan, Amsterdam, 1980.
59. C. P. J. Koymans. Models of the lambda calculus. *Information and Computation*, 52(3):306–323, 1982.
60. Jean Louis Krivine. *Lambda-Calculus, Types and Models*. Ellis Horwood Series in Computers and Their Applications. Masson, Paris, and Ellis Horwood, Hemel Hempstead, 1993. Translation from French by René Cori, French orig. ed., Masson, Paris, 1990.
61. James Laird. A fully abstract bidomain model of unary PCF. In Martin Hofmann, editor, *Typed Lambda Calculi and Applications, 6th International Conference, TLCA 2003, Valencia, Spain, June 10–12, 2003, Proceedings*, volume 2701 of *Lecture Notes in Computer Science*, pages 211–225. Springer-Verlag, 2003.
62. Joachim Lambek. From lambda calculus to cartesian closed categories. In *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 375–402. Academic Press, 1980.
63. Peter J. Landin. The mechanical evaluation of expressions. *Computer Journal*, 6:308–320, January 1964.
64. Peter J. Landin. A correspondence between ALGOL 60 and Church's lambda-notation: Part I and Part II. *Communications of the ACM*, 8(2–3):89–101, 158–165, 1965.
65. Peter J. Landin. The next 700 programming languages. *Communications of the ACM*, 9(3):157–166, March 1966.
66. John McCarthy. *LISP 1.5 Programmer's Manual*. The MIT Press, Cambridge, Mass., 1962. (with Abrahams, Edwards, Hart, and Levin).

67. Albert Meyer. What is a model of the lambda calculus? *Information and Computation*, 52(1):87–122, 1982.
68. Robert Milner. Fully abstract models of typed lambda-calculus. *Theoretical Computer Science*, 4:1–22, 1977.
69. John C. Mitchell. *Foundations of Programming Languages*. The MIT Press, Cambridge, MA, 1996.
70. Eugenio Moggi. *The Partial Lambda-Calculus*. PhD thesis, Edinburgh University, February 1988. Report CST-53-88.
71. C.-H. Luke Ong. Fully abstract models of the lazy lambda calculus. In *29th Annual Symposium on Foundations of Computer Science*, pages 368–376, White Plains, New York, 24–26 October 1988. IEEE Computer Society Press.
72. Luca Paolini. Call-by-value separability and computability. In Antonio Restivo, Simona Ronchi Della Rocca, and Luca Roversi, editors, *Theoretical Computer Science, 7th Italian Conference, ICTCS 2001, Torino, Italy, October 4–6, 2001*, volume 2202 of *Lecture Notes in Computer Science*, pages 74–89. Springer-Verlag, 2001.
73. Luca Paolini and Simona Ronchi Della Rocca. Call by value solvability. *Theoretical Informatics and Applications*, 33(6):507–534, nov 1999.
74. Luca Paolini and Simona Ronchi Della Rocca. The parametric parameter passing λ -calculus. *Information and Computation*, 189(1):87–106, feb 2004.
75. D. M. R. Park. The Y-combinator in scott’s lambda-calculus models. Research Report CS-RR-013, Department of Computer Science, University of Warwick, Coventry, UK, June 1976.
76. Adolfo Piperno. An algebraic view of the Böhm-out technique. *Theoretical Computer Science*, 212(1–2):233–246, February 1999.
77. Andrew M. Pitts. Operational semantics and program equivalence. In G. Barthe, P. Dybjer, and J. Saraiva, editors, *Applied Semantics*, volume 2395 of *Lecture Notes in Computer Science*, pages 378–412. Springer-Verlag, 2002. (Revised version of lectures at the International Summer School On Applied Semantics, APPSEM 2000, Caminha, Minho, Portugal, 9–15 September 2000.).
78. Gordon D. Plotkin. Call-by-name, call-by-value and the λ -calculus. *Theoretical Computer Science*, 1:125–159, 1975.
79. Gordon D. Plotkin. LCF considered as a programming language. *Theoretical Computer Science*, 5:223–225, 1977.
80. Gordon D. Plotkin. A structural approach to operational semantics. DAIMI FN-19, Aarhus University, Aarhus, Denmark, September 1981.
81. Gordon D. Plotkin. Domains. Dept. of Computer Science, University of Edinburgh, 1983.
82. Alberto Pravato, Simona Ronchi Della Rocca, and Luca Roversi. The call by value λ -calculus: a semantic investigation. *Mathematical Structures in Computer Science*, 9(5):617–650, 1999.
83. G. E. Revesz. *Lambda-Calculus Combinators and Functional Programming*, volume 4 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, 1988.
84. Simona Ronchi Della Rocca. Discriminability of infinite sets of terms in the D_∞ -models of the λ -calculus. In Egidio Astesiano and Corrado Böhm, editors, *Proceedings of the 6th Colloquium on Trees in Algebra and Programming (CAAP’81)*, volume 112 of *Lecture Notes in Computer Science*, pages 350–364, Genova, Italy, March 1981. Springer-Verlag.
85. Simona Ronchi Della Rocca. Operational semantics and extensionality. In *Proceedings of the 2nd International ACM SIGPLAN Conference on Principles and Practice of Declarative Programming (PPDP-00)*, pages 24–31, Montréal, September 20–23 2000. ACM Press.

86. Antonino Salibra. Topological incompleteness and order incompleteness of the lambda calculus. *ACM Transactions on Computational Logic*, 4(3):379–401, July 2003.
87. David A. Schmidt. *Denotational Semantics: A Methodology for Language Development*. Allyn and Bacon, Boston, 1986.
88. Dana S. Scott. Continuous lattices. In F. William Lawvere, editor, *Toposes, Algebraic Geometry, and Logic*, volume 274 of *Lecture Notes in Mathematics*, pages 97–136. Springer-Verlag, Berlin, Heidelberg, and New York, 1972.
89. Dana S. Scott. Data types as lattices. *SIAM Journal of Computing*, 5:522–587, September 1976.
90. Dana S. Scott. Relating theories of the λ -calculus. In J. P. Seldin and J. R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 403–450. Academic Press, 1980.
91. Joseph E. Stoy. *Denotational Semantics of Programming Languages: The Scott-Strachey Approach to Programming Language Theory*. The MIT Press, Cambridge, USA, 1977.
92. Christopher Strachey. Fundamental concepts in programming languages. *Higher-Order and Symbolic Computation*, 13(1–2):11–49, April 2000. Notes for the International Summer School in Computer Programming, Copenhagen, 1967.
93. Masako Takahashi. Parallel reductions in lambda-calculus. *Information and Computation*, 118(1):120–127, April 1995.
94. Daniele Turi and Gordon Plotkin. Towards a mathematical operational semantics. In *Proceedings, Twelfth Annual IEEE Symposium on Logic in Computer Science LICS'97*, pages 280–291, Warsaw, Poland, 29 June–2 July 1997. IEEE Computer Society Press.
95. Alan M. Turing. The P -functions in λ - K -conversion. *The Journal of Symbolic Logic*, 2:164, 1937.
96. Philip Wadler. Call-by-value is dual to call-by-name. In Cindy Norris and Jr. James B. Fenwick, editors, *Proceedings of the Eighth ACM SIGPLAN International Conference on Functional Programming (ICFP-03)*, volume 38, 9 of *ACM SIGPLAN Notices*, pages 189–201, New York, August 25–29 2003. ACM Press.
97. Christopher P. Wadsworth. The relation between computational and denotational properties for scott's D_∞ -models of the lambda-calculus. *SIAM Journal of Computing*, 5(3):488–521, September 1976.
98. Glynn Winskel. *The Formal Semantics of Programming Languages: An Introduction*. Foundations of Computing Series. The MIT Press, February 1993.
99. Glynn Winskel. Stable bistructure models of PCF. In Igor Prívara, Branislav Rovan, and Peter Ruzicka, editors, *Mathematical Foundations of Computer Science 1994, 19th International Symposium*, volume 841 of *Lecture Notes in Computer Science*, pages 177–197, Kosice, Slovakia, 22–26 August 1994. Springer.

Index

$\rightarrow_{\Delta O_\eta}$, 97	nf_A , 29
$(.)^*$, 212	\Downarrow_O , 67
$(.)^+$, 212	Γ , 7
$=_\Delta$, 7	$\Gamma\text{-NF}$, 35
$=_{O_\eta}$, 97	$\Gamma\text{-hnf}$, 39
$=_{\Delta O_\eta}$, 97	$\Gamma\text{-lbnf}$, 66
$=_{\Delta \eta}$, 23	$\Gamma\text{-nf}$, 35, 66
$App_{\mathcal{H}}$, 133	Γ_{Ω} , 184
$App_{\mathcal{L}}$, 168	\bowtie , 145
$App_{\mathcal{N}}$, 154	L_0 , 85, 170
$App_{\mathcal{V}}$, 195	L_1 , 85, 170
$B \cup B'$, 133, 168, 195	Λ , 3
B , 189	$\Lambda\Omega$, 123
B^n , 29	$\Lambda\eta$, 26
C_{\bowtie} , 145	$\Lambda\text{-hnf}$, 25, 66
C_{\angle} , 163	$\Lambda\text{-lhnf}$, 25, 66
C_∞ , 120	$\Lambda\text{-nf}$, 25, 66
C_\vee , 182	$\Lambda\text{-pnf}$, 144
$Comp_{\mathcal{H}}$, 133	$\Lambda\Omega\text{-nf}$, 124
$Comp_{\mathcal{L}}$, 168	Λ_I , 7
$Comp_{\mathcal{N}}$, 154	\mathbf{N} , 77
$Comp_{\mathcal{V}}$, 195	Ω , 123
$Cond$, 227	$\Omega^{\mathcal{H}}$, 129
E_∞ , 128, 139	$\Phi^{\mathcal{H}}$, 130
$I(C)$, 109	Pred , 229
$I(C_{\bowtie})$, 145	Ψ , 36
$I(C_{\angle})$, 163	$\Psi\ell$, 36
$I(C_\infty)$, 120	$\Psi\ell\text{-nf}$, 36
$I(C_\vee)$, 182	\Rightarrow_D° , 138
O^n , 29	\Rightarrow_N° , 159, 160
$T(C)$, 109	\Rightarrow_Γ , 52
$T(C_{\bowtie})$, 145	\Rightarrow_A , 28
$T(C_{\angle})$, 163	Succ , 229
$T(C_\infty)$, 120	T , 229
$T(C_\vee)$, 182	Test , 229
U , 171	Θ , 65
U_n^i , 29	\Uparrow_O , 68
Y , 27	V_0 , 93, 198
Y_Γ , 238	V_1 , 93, 198
Y_A , 234	Var , 3
$[f]$, 205	Θ , 65

\rightarrow_{Δ} , 6	ψ , 145
$\rightarrow_{\Delta}^{\circ}$, 9	\rightarrow_{Ω} , 123
\swarrow , 39	$\rightarrow_{O\eta}^*$, 97
$\exists \ell$, 36	$\rightarrow_{\Delta O\eta}^*$, 97
$\exists \ell \Omega$ -nf, 185	\rightarrow_{Δ}^* , 7
Ξ , 36, 66	$\rightarrow_{\Gamma\Omega}$, 184
Zero, 229	$\rightarrow_{\sqrt{\Omega}}$, 184
\angle , 163	$\rightarrow_{\Lambda\angle\Omega}$, 165
\approx_N , 78	$\rightarrow_{\Lambda\Omega}$, 123
\approx_H , 73	$\rightarrow_{\angle\Omega}$, 165
\approx_V , 90	$\rightarrow_{O\eta}$, 97
\approx_L , 82	$\rightarrow_{\psi\ell}$, 36
\approx_O , 68	$\rightarrow_{\Xi\ell\Omega}$, 185
args, 29	\triangleleft , 202
H , 67, 73	\triangleleft_σ , 202
L , 67	$\Downarrow_{\mathbf{v}}$, 89
N , 67	\simeq_c , 29
O , 66	\smile , 96
V , 67	\smile_H , 97
$\rightarrow_{\Xi\ell}^{\circ}$, 42	\smile_L , 98
\circ , 105	\smile_N , 97
$\circ\nabla$, 114	\smile_v , 98
\sqsubseteq_L , 207	\sqcap , 207
c , 210	\sqsubseteq_k , 128
$\llbracket \cdot \rrbracket$, 105	$\sqsubseteq_{\mathcal{F}}$, 116
$\sim_{\mathcal{L}\mathcal{L}}$, 176	$\sqsubseteq_{\mathcal{H}}$, 125
$\sim_{\mathcal{V}\mathcal{V}}$, 206	$\sqsubseteq_{\mathcal{L}\mathcal{L}}$, 176
\equiv , 3	$\sqsubseteq_{\mathcal{N}}$, 146
$\mathcal{E}(\Delta, \Theta)$, 67	$\sqsubseteq_{\mathcal{V}\mathcal{V}}$, 206
\mathfrak{N} , 229	\checkmark , 182
\frown , 96	\dagger , 114
\Downarrow_H , 73	\vdash_{\angle} , 164
∞ , 120	\vdash_{∞} , 120
λ -term, 3	\vdash_{\vee} , 183
$\langle \cdot \rangle$, 44	\vdash_{\bowtie} , 146
\Downarrow_L , 82	\vec{M} , 3
\leq_{\bowtie} , 145	\mathcal{A} , 124
\leq_{\angle} , 163	\mathcal{A}^{\checkmark} , 185
\leq_{∞} , 120	\mathcal{A}^{\angle} , 165
\leq_{\vee} , 182	\mathcal{E} , 218
\leq_{∇_L} , 212	$\mathcal{F}^0(\checkmark)$, 204
\ll , 125	\mathcal{H} , 121
\ll_c° , 139	\mathcal{I} , 215
\ll_c , 136	\mathcal{J} , 217
λ , 208	$\mathcal{L}\mathcal{L}$, 176
∇ , 109	\mathcal{L} , 163
\Downarrow_N , 77	\mathcal{N} , 145
ϕ , 120	\mathcal{U} , 219
\preceq_V , 90	\mathcal{U}^{\checkmark} , 185
\preceq_N , 78	$\mathcal{V}\mathcal{V}$, 205
\preceq_H , 73	\mathcal{V} , 183
\preceq_L , 82	\mathbb{L} , 207
\preceq_O , 68	\mathbb{L}_∞ , 210

- abstraction, 3
- active, 9
 - $\Xi\ell$, 42
- algorithm
 - Γ -separability, 52
 - Λ -separability, 32
 - \mathcal{N} -Semiseparability, 160
 - semiseparability, 138
- application, 3
- approximant, 124
 - $\sqrt{\Omega}$, 185
 - $\angle\Omega$, 165
 - defined, 159
 - maximal along a path, 139
 - upper, 185
- arguments, 8
- basis, 109
 - agree, 114
 - finite, 133, 154, 168, 195
- binder, 4
- block
 - head, 8
- body, 8
- calculus
 - $\lambda\Lambda$, 25
 - $\lambda\Gamma$, 35
 - $\lambda\Delta$, 6
 - complete, 69
 - correct, 69
- call-by-name, 8, 25
- call-by-value, 8, 35
- closure
 - Var, 7
 - reduction, 7
 - substitution, 7
- compact, 207
- compact-as-types procedure, 212
- complete development, 15
- confluence, 8
 - $\Xi\ell$, 41
- congruence, 21
- context, 6
 - **H**-relevant, 75
 - **L**-relevant, 83
 - **N**-relevant, 79
 - **V**-relevant, 91
 - Δ -valuable, 12
 - discriminating, 69
 - head, 12
 - relevant, 69
- continuous, 207
- contractum
 - $\Psi\ell$, 36
 - Δ , 6
- correctness
 - **H**, 75
 - **L**, 83
 - **N**, 79
 - **V**, 90
- D, 4
- defined, 159
- degree
 - Γ , 39
 - Λ , 26
- derivation, 67, 110
 - size, 67
- discriminability
 - \mathcal{H} , 127
 - \mathcal{N} , 150
 - head, 99
- domain, 207
 - constructor, 209
- E, 4
- embedding, 209
- environment, 105
- equivalence
 - operational, 68
- evaluation relation, 66
 - deterministic, 68
 - nondeterministic, 68
 - uniform, 99
 - universal, 71
- filter, 113
- fixed point
 - Γ , 40
 - Λ , 27
 - operator, 27, 41
- function
 - Z , 226
 - **O**-representable, 233
 - composition, 226
 - continuous, 207
 - embedding, 209
 - minimalization, 226
 - monotone, 207
 - partial recursive, 226
 - primitive recursive, 226
 - projection, 209, 226
 - step, 208
 - strict, 207
 - successor, 226

- greatest lower bound, 207
- head
 - block, 8
 - context, 6
 - redex, 8
 - variable, 8
- head normal form
 - Γ , 39
 - Λ , 25, 66
 - Λ -lazy, 25, 66
 - $\Psi\ell$, 36
- hole, 6
- I, 4
- incompleteness
 - \mathbf{H} , 76
 - \mathbf{L} , 84
 - \mathbf{N} , 80
 - \mathbf{V} , 92
- inhabited, 173
- input value, 7
 - approximated, 184
 - standard set, 10
- interpretation, 105, 114
- intersection
 - relation, 109
 - \bowtie , 145
 - \angle , 163
 - ∞ , 120
 - \checkmark , 182
 - type assignment system, 108
 - types, 109
- inverse limit, 210
- isomorphic, 208
- K, 4
- language, 3
 - $\Lambda\Omega$, 123
- lattice
 - ω -algebraic, 207
 - complete, 207
- lazy
 - Ψ -contractum, 36
 - Ψ -redex, 36
 - Ψ -reduction, 36
- lazy blocked nf
 - Γ , 66
- least upper bound, 207
- legality, 112
 - \bowtie , 145
 - \angle , 163
 - ∞ , 121
- \checkmark , 183
- lifting, 209
- machine
 - reduction, 68
- minimalization, 226
- model
 - $\lambda\Delta$, 105
 - \mathcal{H} , 121
 - \mathcal{LL} , 176
 - \mathcal{L} , 163
 - \mathcal{N} , 145
 - \mathcal{VV} , 205
 - \mathcal{V} , 183
 - complete, 106
 - correct, 106
 - filter, 116
 - complete, 117
 - correct, 117
 - fully abstract, 117
 - fully abstract, 107
- monotone, 207
- normal form, 7
 - Γ , 35, 66
 - $\Gamma\Omega$, 184
 - Λ , 25, 66
 - $\Lambda\Omega$, 124
 - $\angle\Omega$, 165
 - $\Psi\ell$, 36
 - Δ , 7
 - $\Xi\ell\Omega$, 185
 - persistent, 144
- normalizing
 - $\Psi\ell$, 36
 - Δ , 7
- numeral system, 228
- O, 4
- occur, 4
- operational semantics
 - \mathbf{L} , 82
 - \mathbf{N} , 77
 - \mathbf{V} , 89
 - \mathbf{H} , 73
- order
 - Γ , 39
 - Λ , 26
- output value, 65
- pair, 227
- partial recursive, 226
- Peano's natural numbers, 228

- pointwise, 208
- primitive recursive, 226
- principality condition, 65
- principle
 - operational extensionality, 96
 - operational functionality, 96
- projection, 209
- recursion
 - call-by-value, 41
- redex
 - Γ , 35
 - $\Psi\ell$, 36
 - Δ , 6
 - $\Xi\ell$ -degree, 42
 - $\Xi\ell$ -principal, 42
 - degree, 9
 - head, 8
 - principal, 9
- reduction, 6
 - Γ_Ω , 184
 - $\sqrt{\Omega}$, 184
 - $\Lambda\angle\Omega$, 165
 - $\Lambda\Omega$, 123
 - $\angle\Omega$, 165
 - $\mathbf{O}\eta$, 97
 - Ω , 123
 - Ψ -lazy, 36
 - \hookrightarrow_Δ , 14
 - Δ , 6
 - \Rightarrow_Δ , 14
 - \Rightarrow_Δ° , 16
 - \Rightarrow_Δ^i , 19
 - \rightarrow_Δ^i , 19
 - \rightarrow_Δ^p , 9
 - $\Xi\ell\Omega$, 185
 - Ξ -lazy, 36
 - α , 5
 - η , 23
 - parallel deterministic, 14
 - parallel nondeterministic, 14
 - principal, 9
 - standard, 9
- replacement, 4
- simultaneous, 5
- retraction, 209
- retraction sequence, 210
- semantics
 - denotational, 106
 - operational, 68
- semiseparability
 - algorithm
- \mathcal{H} , 137
- \mathcal{N} , 159
- separability, 22
 - Γ , 40
 - Λ , 26
- separable, 22
- sequentialization, 9
- set of input types, 109
- size, 67
- solvability
 - Γ , 39
 - Λ , 26
- SOS, 65, 101
- standardization, 9
 - $\Xi\ell$, 43
 - theorem, 10
- strict, 207
- strongly normalizing
 - $\Psi\ell$, 36
 - Δ , 7
- subdomain, 209
- substitution, 5
- subterm, 4
- term
 - Γ -solvable, 38
 - Γ -valuable, 35
 - \mathbf{O} -comparable, 96
 - Δ -solvable, 12
 - Δ -unsolvable, 12
 - \mathcal{H} -computable, 133
 - \mathcal{L} -computable, 168
 - \mathcal{N} -computable, 154
 - \mathcal{V} -computable, 195
 - closed, 4
 - occurrence, 4
 - open, 4
 - openly Λ -solvable, 120
 - potentially Γ -valuable, 35
- term model, 107
- theory
 - $\Delta\mathbf{O}\eta$, 97
 - Δ , 21
 - consistent, 22
 - full extensional, 23
 - inconsistent, 22
 - input consistent, 22
 - input inconsistent, 22
 - maximal, 22
 - semisensible, 22
 - sensible, 22
- type
 - assignment system, 109
 - constant, 109

- input, 109
 - system, 109
 - \bowtie , 145
 - \angle , 163
 - ∞ , 120
 - \vee , 182
 - legal, 112
 - theory, 109
- typing, 110
 - variable
 - bound, 4
 - free, 4
 - head, 8
 - weight, 43

Monographs in Theoretical Computer Science · An EATCS Series

K. Jensen

Coloured Petri Nets

*Basic Concepts, Analysis Methods
and Practical Use, Vol. 1*
2nd ed.

K. Jensen

Coloured Petri Nets

*Basic Concepts, Analysis Methods
and Practical Use, Vol. 2*

K. Jensen

Coloured Petri Nets

*Basic Concepts, Analysis Methods
and Practical Use, Vol. 3*

A. Nait Abdallah

The Logic of Partial Information

Z. Fülop, H. Vogler

Syntax-Directed Semantics

Formal Models Based
on Tree Transducers

A. de Luca, S. Varricchio

**Finiteness and Regularity
in Semigroups and Formal Languages**

E. Best, R. Devillers, M. Koutny

Petri Net Algebra

S.P. Demri, E. S. Orłowska

**Incomplete Information:
Structure, Inference, Complexity**

J.C.M. Baeten, C.A. Middelburg

Process Algebra with Timing

L.A. Hemaspaandra, L. Torenvliet

Theory of Semi-Feasible Algorithms

E. Fink, D. Wood

Restricted-Orientation Convexity

Zhou Chaochen, M. R. Hansen

Duration Calculus

A Formal Approach to Real-Time
Systems

M. Große-Rhode

Semantic Integration

of Heterogeneous Software

Specifications

Texts in Theoretical Computer Science · An EATCS Series

J. L. Balcázar, J. Díaz, J. Gabarró
Structural Complexity I

M. Garzon
Models of Massive Parallelism
Analysis of Cellular Automata
and Neural Networks

J. Hromkovič
**Communication Complexity
and Parallel Computing**

A. Leitsch
The Resolution Calculus

G. Păun, G. Rozenberg, A. Salomaa
DNA Computing
New Computing Paradigms

A. Salomaa
Public-Key Cryptography
2nd ed.

K. Sikkel
Parsing Schemata
A Framework for Specification
and Analysis of Parsing Algorithms

H. Vollmer
Introduction to Circuit Complexity
A Uniform Approach

W. Fokkink
Introduction to Process Algebra

K. Weihrauch
Computable Analysis
An Introduction

J. Hromkovič
Algorithmics for Hard Problems
Introduction to Combinatorial
Optimization, Randomization,
Approximation, and Heuristics
2nd ed.

S. Jukna
Extremal Combinatorics
With Applications
in Computer Science

P. Clote, E. Kranakis
**Boolean Functions
and Computation Models**

L. A. Hemaspaandra, M. Ogiwara
The Complexity Theory Companion

C.S. Calude
Information and Randomness.
An Algorithmic Perspective
2nd ed.

J. Hromkovič
Theoretical Computer Science
Introduction to Automata,
Computability, Complexity,
Algorithmics, Randomization,
Communication and Cryptography

A. Schneider
Verification of Reactive Systems
Formal Methods and Algorithms

S. Ronchi Della Rocca, L. Paolini
The Parametric Lambda Calculus
A Metamodel for Computation

Y. Bertot, P. Castéran
**Interactive Theorem Proving
and Program Development**
Coq'Art: The Calculus
of Inductive Constructions

L. Libkin
Elements of Finite Model Theory