

University of Groningen

## Inductive types in constructive languages

Bruin, Peter Johan de

**IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.**

*Document Version*

Publisher's PDF, also known as Version of record

*Publication date:*

1995

[Link to publication in University of Groningen/UMCG research database](#)

*Citation for published version (APA):*

Bruin, P. J. D. (1995). *Inductive types in constructive languages*. s.n.

### Copyright

Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license. More information can be found on the University of Groningen website: <https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment>.

### Take-down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): <http://www.rug.nl/research/portal>. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

UNIVERSITY OF GRONINGEN

# Inductive Types in Constructive Languages

**Peter J. de Bruin**

March 24, 1995

# Abstract

Logic grammar is used to partly define a formal mathematical language “ADAM”, that keeps close to informal mathematics and yet is reducible to a foundation of Constructive Type Theory (or Generalized Typed Lambda Calculus). This language is employed in making a study of inductive types and related subjects, as they appear in languages for constructive mathematics and lambda calculi. The naturality property of objects with type parameters is described and employed.

## *Cover diagram*

Behold the mathematical universe,  
developing from original unity  
into categorical duality.

The central beam contains  
the initial and the final type,  
together with the remaining flat finite types.

It is flanked by the dual principles  
of generalized sum and product,  
and of initial and final fixed point construction.

RIJKSUNIVERSITEIT GRONINGEN

# Inductive Types in Constructive Languages

## Proefschrift

ter verkrijging van het doctoraat in de Wiskunde  
en Natuurwetenschappen aan de Rijksuniversiteit  
Groningen op gezag van de Rector Magnificus Dr.  
F. van der Woude in het openbaar te verdedigen op  
vrijdag 24 maart 1995 des namiddags te 4.00 uur

door

**Peter Johan de Bruin**

geboren op 20 september 1964 te Harderwijk

Promotor: Prof. dr. G. R. Renardel de Lavalette

## Preface

The fascination for mathematical truth has been the driving force for my research as it had been for my master's thesis written in Nijmegen. Being amazed at the lack of a general language for the irrefutable expression of mathematical argument, I looked for it in the direction of what is called Type Theory. I started my research in Groningen in 1988 under supervision of Roland Backhouse, and enjoyed his discussion club with Paul Chisholm, joyous Grant Malcolm, Albert Thijs, and broadly interested Ed Voermans. When Backhouse moved to Eindhoven in 1990, I and Thijs remained in Groningen and our roads parted.

The advent in 1991 of Gerard Renardel who accepted to take up my supervision with fresh interest, started a new period of seeking to assemble all available pieces. I owe him much for his constant trust and support in keeping up courage, his help in formulating ideas, his effort to curtail outgrowing branches, and for leaving the choice to carry this work through entirely to me. I also thank Jan Terlouw for his modest cooperation, and my fellow Ph.D. students for their good company and friendship.

Now my heart has turned from abstract truth to living Truth, and since 1993 I'm living in community *Agapè* of the Blessed Sacrament Fathers in Amsterdam. I am glad to thank its members, Aad, Eugène, Gerard, Herman, Jan, Paul, Pieter, and Theo, for their support while finishing this thesis. I thank the members of the Ph.D. committee, Roland C. Backhouse (Eindhoven), Wim H. Hesselink (Groningen), Gerard R. Renardel de Lavalette (Groningen), and Michel Sintzoff (Louvain-la-Neuve), for accepting this duty and providing kind comments on the manuscript, especially Wim who gave detailed comments which helped me to prepare the final text. Though it is tough material, my presentation sometimes being terse and not all ideas given sufficient scientific support, I hope you will get a catch of its beauty.

Peter de Bruin



2.12.3	Subsets	37
2.12.4	Relational notations	37
2.12.5	Currying	38
2.12.6	Pattern matching	39
2.12.7	Linear proof notation	39
2.13	Conclusion	39
<b>3</b>	<b>Common induction and recursion principles</b>	<b>40</b>
3.1	Examples of inductive types	40
3.2	More on natural numbers	43
3.3	Inductive subset definitions	44
3.3.1	Sets inductively defined by rules	44
3.3.2	The well-founded part of a relation	45
3.3.3	Inductive definitions as operators	47
3.3.4	Fixed points in a lattice	47
3.4	From induction to recursion	48
3.5	Conclusion	49
<b>4</b>	<b>Categories and algebra</b>	<b>50</b>
4.1	Categorical notions	50
4.2	Algebras and signatures	53
4.3	Initial algebras, catamorphisms	54
4.4	Algebras with equations	57
4.5	Initial algebras related to well-founded relations	60
4.6	An aside: monads	61
4.7	Algebraic Specification	63
4.8	Concluding remarks	64
<b>5</b>	<b>Specifying inductive types</b>	<b>65</b>
5.1	Single inductive types	65
5.1.1	Operator domains	65
5.1.2	Operators with arity	67
5.1.3	The wellordering of a single inductive type	68
5.2	Mutually inductive types	68
5.2.1	Using an exponential category	69
5.2.2	Plain algebra signatures	70
5.3	Production rules for polynomial functors	71
5.3.1	Positive type expressions	72
5.3.2	A type of polynomial functors	72
5.4	Adding equations	72
5.5	Conclusion	73
<b>6</b>	<b>Recursors in constructive type theories</b>	<b>74</b>
6.1	Algebraic recursion, or paramorphisms	74
6.2	Recursive dependent functions	76
6.3	Mendler's approach	78



# Contents

<b>Summary</b>	<b>5</b>
<b>1 Introduction</b>	<b>7</b>
1.1 Mathematical language . . . . .	7
1.2 Our approach to mathematical language . . . . .	8
1.3 Type Theory and Set Theory . . . . .	10
1.4 Related efforts . . . . .	11
1.5 Relational calculus . . . . .	12
1.6 ADAM's Type Theory . . . . .	12
1.7 Aspects of induction and recursion . . . . .	13
1.8 Frameworks for studying induction . . . . .	14
1.9 Our treatment of induction and recursion . . . . .	15
1.10 Other kinds of inductive types . . . . .	16
1.11 Original contributions . . . . .	16
<b>2 The language ADAM</b>	<b>18</b>
2.1 Language definition mechanism . . . . .	18
2.2 Basic grammar of ADAM . . . . .	20
2.3 Production rules . . . . .	23
2.3.1 Terms . . . . .	23
2.3.2 Patterns . . . . .	23
2.3.3 Definitions . . . . .	24
2.3.4 Declarations . . . . .	25
2.3.5 Coercion . . . . .	25
2.4 Types and universes . . . . .	26
2.5 Products and function spaces . . . . .	27
2.6 Sums and declaration types . . . . .	29
2.7 Families and quantifiers . . . . .	31
2.8 Finite types . . . . .	31
2.9 Infinite types . . . . .	33
2.10 Equality predicate . . . . .	33
2.11 The type of propositions . . . . .	34
2.12 More derived notions . . . . .	36
2.12.1 Predicates . . . . .	36
2.12.2 Subtypes . . . . .	36

6.4	Recursors for mutual induction and recursion	81
6.5	Summary	83
<b>7</b>	<b>Co-inductive types</b>	<b>85</b>
7.1	Dualizing $F$ -algebras	85
7.2	Anamorphism schemes	87
7.3	Dual recursion	89
7.4	Dual equations	90
7.5	Terminal interpretation of equations	90
7.6	Conclusion	91
<b>8</b>	<b>Existence of inductively defined sets</b>	<b>92</b>
8.1	Using transfinite ordinal induction	92
8.2	Kerckhoff's proof	94
8.3	Algebras with equations	96
<b>9</b>	<b>Partiality</b>	<b>98</b>
9.1	Domain theory	98
9.2	Optional objects	101
9.3	Building recursive cpo's by co-induction	103
9.4	Recursive object definitions	104
9.5	Conclusion	106
<b>10</b>	<b>Related subjects</b>	<b>107</b>
10.1	Impredicative type theories	107
10.1.1	Weak initial algebras	108
10.1.2	Weak final algebras	109
10.2	Using type-free values	109
10.2.1	Henson's calculus TK	109
10.3	Inductive universe formation	110
10.4	Bar recursion	112
<b>11</b>	<b>Reflections and conclusion</b>	<b>113</b>
11.1	Mathematical language	113
11.2	Constructive Type Theory	114
11.3	Language definition mechanism	115
11.4	Proofs and proof notation	116
11.5	Inductive types	117
11.6	Directions for further research	118
<b>A</b>	<b>Set theory</b>	<b>120</b>
A.1	ZFC axioms	120
A.2	Set encodings	121
A.3	Ordinals	122
A.4	Cardinals	122
A.5	A model of ZFC	123

A.6	An inductive model of ZFC	124
A.7	Anti-foundation	125
<b>B</b>	<b>ADAM's Type Theory</b>	<b>126</b>
B.1	Abstract syntax	126
B.2	Meta-predicates	127
B.3	Universes	129
B.4	Products	129
B.5	Sums	130
B.6	Finite types	130
B.7	Naturals	130
B.8	Equality	131
B.9	Existential propositions	131
B.10	Semantics	132
B.11	More derived notations	134
<b>C</b>	<b>Proof elimination in Type Theory</b>	<b>135</b>
C.1	Introduction	135
C.2	The basic system	137
C.3	Strong existence	137
C.3.1	New rules	137
C.3.2	Difficulties with reduction to canonical form	138
C.4	Applications	139
C.4.1	Iota	139
C.4.2	Quotient types	139
C.4.3	Inductive types	142
C.5	Conclusion	143
<b>D</b>	<b>Naturality of Polymorphism</b>	<b>144</b>
D.1	Introduction	144
D.2	Polymorphic typed lambda calculus	146
D.3	Turning type constructors into relation constructors	147
D.4	Naturality of expressions	148
D.5	Applications	151
D.6	Dinatural transformations	153
D.7	Second-order languages	154
D.8	Overloaded operators	155
	<b>Index</b>	<b>156</b>
	<b>Bibliography</b>	<b>160</b>
	<b>Samenvatting (Dutch summary)</b>	<b>165</b>