

Lecture Notes in Computer Science
Edited by G. Goos, J. Hartmanis, and J. van Leeuwen

2297

Springer

Berlin

Heidelberg

New York

Barcelona

Hong Kong

London

Milan

Paris

Singapore

Tokyo

Roland Backhouse Roy Crole
Jeremy Gibbons (Eds.)

Algebraic and Coalgebraic Methods in the Mathematics of Program Construction

International Summer School and Workshop
Oxford, UK, April 10-14, 2000
Revised Lectures



Springer

Series Editors

Gerhard Goos, Karlsruhe University, Germany
Juris Hartmanis, Cornell University, NY, USA
Jan van Leeuwen, Utrecht University, The Netherlands

Volume Editors

Roland Backhouse
University of Nottingham, School of Computer Science and IT
Jubilee Campus, Wollaton Road, Nottingham NG8 1BB, United Kingdom
E-mail: rcb@cs.nott.ac.uk

Roy Crole
University of Leicester, Dept. of Mathematics and Computer Science
University Road, Leicester LE1 7RH, United Kingdom
E-mail: roy.crole@mcs.le.ac.uk

Jeremy Gibbons
Oxford University Computing Laboratory, Wolfson Building
Parks Road, Oxford OX1 3QD, United Kingdom
E-mail: jeremy.gibbons@comlab.ox.ac.uk

Cataloging-in-Publication Data applied for

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

Algebraic and coalgebraic methods in the mathematics of program construction ;
international summer school and workshop, Oxford, UK, April 10 - 14, 2000 ;
revised lectures / Roland Backhouse ... (ed.). - Berlin ; Heidelberg ; New York ;
Barcelona ; Hong Kong ; London ; Milan ; Paris ; Tokyo : Springer, 2002
(Lecture notes in computer science ; Vol. 2297)
ISBN 3-540-43613-8

CR Subject Classification (1998): D.2, D.3, F.3, F.4

ISSN 0302-9743

ISBN 3-540-43613-8 Springer-Verlag Berlin Heidelberg New York

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, re-use of illustrations, recitation, broadcasting, reproduction on microfilms 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. Violations are liable for prosecution under the German Copyright Law.

Springer-Verlag Berlin Heidelberg New York
a member of BertelsmannSpringer Science+Business Media GmbH

<http://www.springer.de>

© Springer-Verlag Berlin Heidelberg 2002
Printed in Germany

Typesetting: Camera-ready by author, data conversion by DA-TeX Gerd Blumenstein
Printed on acid-free paper SPIN 10846432 06/3142 5 4 3 2 1 0

Preface

Program construction is about turning specifications of computer software into implementations. Doing so in a way that guarantees correctness is an undertaking requiring deep understanding of the languages and tools being used, as well as of the application domain. Recent research aimed at improving the process of program construction exploits insights from abstract algebraic tools such as lattice theory, fixpoint calculus, universal algebra, category theory, and allegory theory. This book provides an introduction to these mathematical theories and how they are applied to practical problems.

The book is based on the School on *Algebraic and Co-algebraic Methods in the Mathematics of Program Construction* held in April 2000 at the University of Oxford. The School, which was sponsored under the Mathematics for Information Technology initiative of the Engineering and Physical Research Council in the UK, had the goal of introducing this research area to new PhD students in a way that would provide a sound and broad basis for their own research studies. The lecturers were chosen on the basis of a combination of a distinguished track record of research contributions in this area and an ability to present difficult material in a comprehensible way without compromising scientific rigor.

The students that attended the School had varied backgrounds to which due account was given by the lecturers in preparing their material. The experience and feedback gained during the School has been used in preparing this major revision of that material. The lecture material has also been augmented by an introductory chapter giving a detailed overview of the remaining chapters. We hope that this chapter will prove useful in allowing readers to select the chapters most relevant to their own research and to plan their further reading.

Our thanks go to all those involved in making the School a very enjoyable and successful event. Particular thanks go to the EPSRC and the London Mathematical Society for their generous financial support, to the students for their enthusiasm and positive feedback, to the lecturers for their unstinting hard work, and to Springer-Verlag for making the publication of this book possible.

January 2002

Roland Backhouse
Roy Crole
Jeremy Gibbons

Contributors

- Peter Aczel:** Departments of Mathematics and Computer Science,
University of Manchester, Manchester M13 9PL, England.
petera@cs.man.ac.uk, <http://www.cs.man.ac.uk/~petera/>.
- Roland Backhouse:** School of Computing and Information Technology,
University of Nottingham, Nottingham NG8 1BB, England.
rcb@cs.nott.ac.uk, <http://www.cs.nott.ac.uk/~rcb/>.
- Richard Bird:** Oxford University Computing Laboratory, Wolfson Building,
Parks Road, Oxford OX1 3QD, England.
richard.bird@comlab.ox.ac.uk,
<http://www.comlab.ox.ac.uk/oucl/work/richard.bird/>.
- Roy Crole:** Department of Mathematics and Computer Science,
University of Leicester, University Road, Leicester LE1 7RH, England.
roy.crole@mcs.le.ac.uk, <http://www.mcs.le.ac.uk/~rcrole/>.
- Henk Doornbos:** EverMind, Westerkade 15/4, 9718 AS Groningen,
The Netherlands.
henk.doornbos@evermind.com, <http://www.evermind.com>.
- Jeremy Gibbons:** Oxford University Computing Laboratory,
Wolfson Building, Parks Road, Oxford OX1 3QD, England.
jeremy.gibbons@comlab.ox.ac.uk,
<http://www.comlab.ox.ac.uk/oucl/work/jeremy.gibbons/>.
- Bart Jacobs:** Department of Computer Science, University of Nijmegen,
P.O. Box 9010, 6500 GL Nijmegen, The Netherlands.
bart@cs.kun.nl, <http://www.cs.kun.nl/~bart/>.
- Burghard von Karger:** Christian-Albrechts-University of Kiel,
Institute of Computer Science and Applied Mathematics,
Olshausenstrasse 40, D-24098 Kiel, Germany.
burghard.von.karger@gmx.de,
<http://www.informatik.uni-kiel.de/~bvk/>.
- Shin Cheng Mu:** Oxford University Computing Laboratory,
Wolfson Building, Parks Road, Oxford OX1 3QD, England.
shin-cheng.mu@comlab.ox.ac.uk,
<http://www.comlab.ox.ac.uk/oucl/work/shin-cheng.mu/>.
- Hilary Priestley:** Oxford University Mathematical Institute,
24–29 St Giles', Oxford OX1 3LB, England.
hap@maths.ox.ac.uk,
<http://www.stannes.ox.ac.uk/college/members/priestley.ha.html>.

Table of Contents

Chapter 1 Introduction	1
<i>Roy Crole</i>	
1. Preliminaries	1
1.1. Assumed Knowledge	1
1.2. Volume Overview	2
2. A Mathematical History Tour	3
2.1. Fixed Points	3
2.2. Induction and Coinduction	7
2.3. Types and Categories	8
2.4. Algebras and Coalgebras	9
3. Mathematics in ACMMP	13
3.1. Chapter 2: Ordered Sets and Complete Lattices	14
3.2. Chapter 3: Introducing Algebras and Coalgebras	14
3.3. Chapter 4: Galois Connections and Fixed Point Calculus	15
3.4. Chapter 5: Calculating Functional Programs	15
3.5. Chapter 6: Algebra of Program Termination	16
3.6. Chapter 7: Exercises in Coalgebraic Specification	17
3.7. Chapter 8: Algebraic Methods for Optimization Problems	17
3.8. Chapter 9: Temporal Algebra	18
Chapter 2 Ordered Sets and Complete Lattices	21
<i>Hilary A. Priestley</i>	
1. Introduction	21
2. From Binary Relations to Diagrams	23
2.1. A Fundamental Example: Powersets	23
2.2. Input-Output Relations Pictorially	24
2.3. Exercise	25
2.4. Binary Relations and Their Polars	26
2.5. Exercise	27
2.6. Summing Up So Far	27
3. Order, Order, Order, ...	28
3.1. Partial Order	28
3.2. Information Orderings	29
3.3. Diagrams	30
3.4. Duality: Buy One, Get One Free	31
3.5. Bottom and Top	31
3.6. Lifting	32
3.7. New Posets from Old: Sums and Products	32
3.8. Maps between Posets	33
3.9. Pointwise Ordering of Maps	34
3.10. Up-Sets: An Inbred Example	35

VIII Table of Contents

3.11. Monotone Maps and Up-Sets	35
3.12. Exercise (More on Monotone Maps and Up-Sets)	36
3.13. Down Is Nice Too	36
3.14. Exercise (Turning Things Upside Down)	36
3.15. The Down-Set Operator, $\downarrow$, and the Up-Set Operator, $\uparrow$	37
3.16. Exercise (A Context Explored)	38
3.17. Maximal and Minimal Elements	38
3.18. Stocktaking	38
4. Lattices in General and Complete Lattices in Particular	39
4.1. Lattices	39
4.2. Examples of Lattices	41
4.3. Distributive Lattices	41
4.4. Boolean Algebras	42
4.5. Lattices in Logic	42
4.6. Upper Bounds and Sups, Lower Bounds and Infs	43
4.7. Much Ado about Nothing, and about Everything	44
4.8. Complete Lattices	45
4.9. Completeness on the Cheap	45
4.10. A Special Class of Complete Lattices	45
4.11. Suprema, Infima, and Monotone Maps	46
5. Complete Lattices, Concretely: Closure Systems and Closure Operators	47
5.1. A Useful Technical Remark	47
5.2. Complete Lattices of Sets	47
5.3. Closure Systems	47
5.4. Closure Systems	48
5.5. Examples	49
5.6. From a Complete Lattice to a Closure System	49
5.7. Defining Closure Operators	49
5.8. New Complete Lattices from Old:	
From a Closure Operator to a Complete Lattice	50
5.9. Closure Operators more Concretely	51
6. Galois Connections: Basics	51
6.1. Introduction	51
6.2. Lattice Representation via Galois Connections	52
6.3. Galois Connections from Binary Relations: Method I	53
6.4. Galois Connections and Algebras—A Fleeting Glimpse	54
6.5. Galois Connections by Sectioning	54
6.6. Galois Connections from Binary Relations: Method II	55
6.7. Galois Connections: Basic Properties	55
6.8. $\triangleright$ and $\triangleleft$ Have Isomorphic Images	56
6.9. Equivalent Definitions for Galois Connections	57
6.10. The Good (and Less Good) Behaviour of Galois Maps	58
6.11. Uniqueness of Adjoints: $\triangleright$ from $\triangleleft$ and $\triangleleft$ from $\triangleright$	59
6.12. Exercise (Surjective and Injective Galois Maps)	60
6.13. A Look Ahead	60

6.14. Existence of Adjoints: A Technical Lemma	61
6.15. Existence Theorem for Adjoints	61
6.16. Postscript	62
7. Making Connections, Conceptually	62
7.1. From a Galois Connection to a Closure Operator	62
7.2. From a Closure Operator to a Galois Connection	63
7.3. Contexts and Concepts: Re-setting the Scene	63
7.4. Ordering Concepts	63
7.5. Three for the Price of One: A Trinity of Complete Lattices	64
7.6. Manufacturing Concepts	65
7.7. Density: Generating all Concepts via γ or μ	65
7.8. From a Complete Lattice to a Concept Lattice	66
7.9. The Case for the Defence	66
7.10. Summing Up	67
8. The Existence of Fixed Points	67
8.1. Fixed Points and Least Fixed Points: Definitions	68
8.2. Characterizing Least Fixed Points via Least Prefix Points	68
8.3. The Knaster–Tarski Fixed Point Theorem	69
8.4. Exercise	69
8.5. Exercise	69
8.6. From Complete Lattices to CPOs	69
8.7. A Sense of Direction	70
8.8. Exercise (Sups and Directed Sups Related)	70
8.9. CPOs	71
8.10. Directed Sets and Continuity	71
8.11. New CPOs from Old	72
8.12. Fixed Point Theorem for a Continuous Function on a CPO	72
8.13. From Continuity to Monotonicity	72
8.14. An Assumption: Zorn’s Lemma (ZL), CPO Form	73
8.15. The Fixed Point Theorem for a Monotone Endofunction on a CPO	73
8.16. Exercise	74
8.17. Exercise	74
8.18. Concluding Remarks	74
9. Speaking Categorically	75
9.1. Categories	75
Chapter 3 Algebras and Coalgebras	79
<i>Peter Aczel</i>	
1. Introduction	79
2. Terms	80
2.1. Variable-Free Terms	80
2.2. Terms as Set Theoretical Objects	81
2.3. Terms with Variables	82
2.4. Initial (F, X) -Algebras	83
2.5. Substitution	83

3. Trees	84
3.1. Variable-Free Trees	84
3.2. Representing Terms as Well-Founded Trees	84
3.3. Corecursion	85
3.4. Set Theoretical Representation of Trees	85
3.5. Trees with Variables	87
3.6. Substitution	87
3.7. Solution Property	87
4. The Monad of a Substitution System	88
Chapter 4 Galois Connections and Fixed Point Calculus	89
<i>Roland Backhouse</i>	
1. Introduction	89
1.1. Fixed Point Equations	89
1.2. Languages	89
1.3. Functions	90
1.4. Datatypes	91
1.5. Galois Connections	91
1.6. Basic Assumptions	92
1.7. Issues and Applications	93
2. Galois Connections — Introductory Examples	93
2.1. Simple Examples	94
2.2. The Floor Function	96
3. Identifying Galois Connections	100
3.1. Symmetric Definitions	101
3.2. Universal Property	102
3.3. Commutativity Properties	103
4. Pair Algebras	105
4.1. Infima and Suprema	107
4.2. Extremum Preservation Properties	109
4.3. Existence Theorem	111
5. Fixed Points	115
5.1. Prefix Points	115
5.2. A First Example	119
5.3. Kleene Algebra	122
6. Fixed Point Calculus	127
6.1. Basic Rules	128
6.2. Fusion	129
6.3. Uniqueness	134
6.4. Parameterised Prefix Points	136
6.5. Mutual Recursion	141
6.6. An Illustration — Arithmetic Expressions	143
7. Further Reading	146

Chapter 5 Calculating Functional Programs	149
<i>Jeremy Gibbons</i>	
1. Introduction	149
1.1. Why Calculate Programs?	149
1.2. Functional Programming	150
1.3. Universal Properties	150
1.4. The Categorical Approach to Datatypes	153
1.5. The Pair Calculus	155
1.6. Bibliographic Notes	158
1.7. Exercises	159
2. Recursive Datatypes in the Category <i>Set</i>	160
2.1. Overview	160
2.2. Monomorphic Datatypes	160
2.3. Folds	161
2.4. Polymorphic Datatypes	163
2.5. Properties of Folds	165
2.6. Co-datatypes and Unfolds	166
2.7. ... and Never the Twain Shall Meet	169
2.8. Bibliographic Notes	169
2.9. Exercises	170
3. Recursive Datatypes in the Category <i>Cpo</i>	173
3.1. The Category <i>Cpo</i>	173
3.2. Continuous Algebras	175
3.3. The Pair Calculus Again	176
3.4. Hylomorphisms	177
3.5. Bibliographic Notes	180
3.6. Exercises	180
4. Applications	183
4.1. A Simple Compiler	183
4.2. Monads and Comonads	186
4.3. Breadth-First Traversal	191
4.4. Bibliographic Notes	193
4.5. Exercises	193
5. Appendix: Implementation in Haskell	197
5.1. Products	197
5.2. Sums	197
5.3. Functors	198
5.4. Datatypes	198
5.5. Folds and Unfolds	198
5.6. Lists	198
5.7. Trees	200
5.8. Quicksort	200

Chapter 6 Algebra of Program Termination	203
<i>Henk Doornbos and Roland Backhouse</i>	
1. Introduction	203
2. Imperative Programming and Well-Founded Relations	203
2.1. Relation Algebra	204
2.2. Imperative Programming	206
2.3. Domains and Division	208
2.4. Well-Foundedness Defined	210
2.5. Totality of While Statements	214
2.6. Induction Principles	215
2.7. Admits-Induction Implies Well-Founded	217
3. Hylo Equations	218
3.1. Relators and Hylos	219
3.2. Hylo Programs	220
3.3. Intermediate Data Structures	227
3.4. The Hylo Theorem	229
3.5. Reducing Problem Size	230
3.6. A Calculus of F -Reductivity	232
Chapter 7 Exercises in Coalgebraic Specification	237
<i>Bart Jacobs</i>	
1. Introduction	237
2. Mathematical Preliminaries	239
3. Specification of Groups and Vector Spaces	241
4. A First Coalgebraic Specification: Binary Trees	244
4.1. Elements of Binary Trees	247
5. Bisimulations and Bisimilarity	248
6. Invariants	253
7. Temporal Logic for Coalgebras	255
7.1. A Concrete Description of $\Box$ and $\Diamond$ for Binary Trees	256
7.2. Using $\Box$ and $\Diamond$ for Specification and Verification of Binary Trees	259
8. Towards a μ -Calculus for Coalgebras	261
9. A Case Study: Peterson's Mutual Exclusion Algorithm	265
9.1. Peterson's Solution for Mutual Exclusion	265
9.2. Dealing with Time in Coalgebraic Specification	267
9.3. Class-Valued Methods	269
9.4. Peterson's Algorithm in Coalgebraic Specification	270
10. Refinements between Coalgebraic Specifications	275
11. Conclusion	277
Chapter 8 Algebraic Methods for Optimization Problems	281
<i>Richard Bird, Jeremy Gibbons, and Shin-Cheng Mu</i>	
1. Introduction	281
1.1. Bibliographic Notes	283

2. The Algebra of Relations	283
2.1. Relations	283
2.2. Special Kinds of Relation	286
2.3. Breadth	287
2.4. Folds	287
2.5. Bibliographic Notes	290
2.6. Exercises	290
3. Optimization Problems	291
3.1. The Eilenberg-Wright Lemma	292
3.2. Preorders	292
3.3. Monotonicity	293
3.4. Minimum	293
3.5. The Greedy Theorem	294
3.6. Thinning	295
3.7. Bibliographic Notes	297
3.8. Exercises	297
4. Optimal Bracketing	299
4.1. Representation	300
4.2. The Converse-of-a-Function Theorem	300
4.3. Spines	301
4.4. An Application of Thinning	302
4.5. An Application of Greediness	303
4.6. Refinement of the Greedy Algorithm to a Program	304
4.7. Summary	304
4.8. The Haskell Program	305
4.9. Bibliographic Notes	305
4.10. Exercises	305
Chapter 9 Temporal Algebra	309
<i>Burghard von Karger</i>	
1. Introduction	309
2. Preliminaries	311
2.1. Lattices	311
2.2. Galois Connections	312
2.3. Boolean Algebra	314
2.4. Fixed Points	318
2.5. Regular Algebra	320
2.6. Iteration	320
2.7. Repetition	322
3. Galois Algebra	322
3.1. Definition and Basic Properties	323
3.2. New Algebras from Old	327
3.3. Diamonds and Boxes	327
3.4. ‘Until’ and ‘Since’	330
3.5. Confluence	332
3.6. Linearity	335

3.7. Infinity	337
4. Sequential Algebra	337
4.1. Observations	339
4.2. Lifting to Sets	343
4.3. Sequential Set Algebras	346
4.4. Abstract Sequential Algebras	352
4.5. Time-Wise Duality	354
5. Relational Laws of Sequential Algebra	355
5.1. Basic Laws	356
5.2. Predicates	359
5.3. Left and Right Domain	361
6. Interval Calculi	364
6.1. Somewhere and Everywhere	365
6.2. Importability	368
6.3. Engineer's Induction	371
6.4. Finite and Infinite Observations	373
6.5. Measuring Time	375
6.6. Phase Calculus	377
6.7. Duration Calculus	381
7. Conclusion	382
Author Index	387