

Cambridge University Press
0521018293 - Programs, Recursion and Unbounded Choice: Predicate-Transformation Semantics and
Transformation Rules
Wim H. Hesselink
Frontmatter
[More information](#)

PROGRAMS, RECURSION AND UNBOUNDED CHOICE

Cambridge University Press

0521018293 - Programs, Recursion and Unbounded Choice: Predicate-Transformation Semantics and Transformation Rules

Wim H. Hesselink

Frontmatter

[More information](#)

Cambridge Tracts in Theoretical Computer Science

Managing Editor Professor C.J. van Rijsbergen,
Department of Computing Science, University of Glasgow

Editorial Board

S. Abramsky, Department of Computing Science, Imperial College of Science
and Technology

P.H. Aczel, Department of Computer Science, University of Manchester

J.W. de Bakker, Centrum voor Wiskunde en Informatica, Amsterdam

J.A. Goguen, Programming Research Group, University of Oxford

J.V. Tucker, Department of Mathematics and Computer Science,
University College of Swansea

Titles in the series

1. G. Chaitin *Algorithmic Information Theory*
2. L.C. Paulson *Logic and Computation*
3. M. Spivey *Understanding Z*
4. G. Revesz *Lambda Calculus, Combinators and Functional Programming*
5. A. Ramsay *Formal Methods in Artificial Intelligence*
6. S. Vickers *Topology via Logic*
7. J.-Y. Girard, Y. Lafont & P. Taylor *Proofs and Types*
8. J. Clifford *Formal Semantics & Pragmatics for Natural Language Processing*
9. M. Winslett *Updating Logical Databases*
10. K. McEvoy & J.V. Tucker (eds) *Theoretical Foundations of VLSI Design*
11. T.H. Tse *A Unifying Framework for Structured Analysis and Design Models*
12. G. Brewka *Nonmonotonic Reasoning*
13. G. Smolka *Logic Programming over Polymorphically Order-Sorted Types*
15. S. Dasgupta *Design Theory and Computer Science*
17. J.C.M. Baeten (ed) *Applications of Process Algebra*
18. J.C.M. Baeten & W. P. Weijland *Process Algebra*
23. E.-R. Olderog *Nets, Terms and Formulas*
27. W.H. Hesselink *Programs, Recursion and Unbounded Choice*
29. P. Gärdenfors (ed) *Belief Revision*
30. M. Anthony & N. Biggs *Computational Learning Theory*

Cambridge University Press
0521018293 - Programs, Recursion and Unbounded Choice: Predicate-Transformation Semantics and Transformation Rules
Wim H. Hesselink
Frontmatter
[More information](#)

PROGRAMS, RECURSION AND UNBOUNDED CHOICE

Predicate-Transformation Semantics and Transformation Rules

WIM H. HESSELINK
Department of Computer Science
University of Groningen



Cambridge University Press

0521018293 - Programs, Recursion and Unbounded Choice: Predicate-Transformation Semantics and Transformation Rules

Wim H. Hesselink

Frontmatter

[More information](#)

CAMBRIDGE UNIVERSITY PRESS

Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore, São Paulo

Cambridge University Press

The Edinburgh Building, Cambridge CB2 2RU, UK

Published in the United States of America by Cambridge University Press, New York

www.cambridge.org

Information on this title: www.cambridge.org/9780521404365

© Cambridge University Press 1992

This book is in copyright. Subject to statutory exception and to the provisions of relevant collective licensing agreements, no reproduction of any part may take place without the written permission of Cambridge University Press.

First published 1992

This digitally printed first paperback version 2005

A catalogue record for this publication is available from the British Library

ISBN-13 978-0-521-40436-5 hardback

ISBN-10 0-521-40436-3 hardback

ISBN-13 978-0-521-01829-6 paperback

ISBN-10 0-521-01829-3 paperback

CONTENTS

List of symbols	xiii
0 Introduction	1
0.1 Semantics of imperative sequential programs	1
0.2 Predicate-transformation semantics	3
0.3 Program transformations	4
0.4 Overview	5
0.5 The chapters in detail	6
0.6 Notation	8
0.7 Design decisions	10
1 Weakest preconditions	12
1.1 Predicates and predicate transformers	13
1.2 Weakest preconditions	15
1.3 Guards, assertions, termination and totality	17
1.4 Composition and nondeterminate choice	19
1.5 Intermezzo on the conditional combination	20
1.6 Program variables, state functions and localized relations	21
1.7 The assignment	23
1.8 Deterministic choice	26
1.9 Appendix on predicate calculus	27
1.10 Exercises	29
2 Annotation, recursion and repetition	32
2.1 Specification with Hoare triples	32
2.2 Proofs by annotation	35
2.3 Specification and invocation of procedures	37
2.4 Correctness of recursive declarations	40
2.5 An abstract version of recursive procedures	42

<i>vi</i>	<i>CONTENTS</i>
2.6 Homomorphisms and simple commands	44
2.7 Induction rules	45
2.8 The repetition	48
2.9 Exercises	52
3 Healthiness laws	58
3.1 Conjunctivity properties of predicate transformers	58
3.2 Two laws	59
3.3 Some important implications	60
3.4 Guards, assertions and assignments	62
3.5 The termination law and repetitions	63
3.6 Exercises	64
4 Semantics of recursion	66
4.1 Complete lattices and predicate transformers	66
4.2 Fixpoints in complete lattices	70
4.3 A syntax for commands with unbounded choice	71
4.4 The interpretation of recursion	73
4.5 Healthiness laws: the universal conjunctivity of <i>wlp</i>	75
4.6 The termination law	76
4.7 The syntactic algebra	79
4.8 The semantic homomorphisms	80
4.9 The induction rules	82
4.10 Conclusion	83
4.11 Exercises	84
5 Ramifications	87
5.1 Refinement and relative refinement	87
5.2 Refinement of procedures	89
5.3 Insertion of guards, and calculus	90
5.4 The commutation problem	91
5.5 Strongest postconditions	93
5.6 Termination and well-founded triples	94
5.7 Two new recursion theorems	97
5.8 Exercises	99

<i>CONTENTS</i>	<i>vii</i>
6 Relational semantics	101
6.1 Relations as an alternative specification method	102
6.2 The relational view of guards, composition and choice	103
6.3 Termination and totality	105
6.4 From commands to relations	105
6.5 Exercises	107
7 Determinacy and disjunctivity	109
7.1 Determinacy	109
7.2 Disjunctivity properties	111
7.3 Determinacy and disjunctivity	112
7.4 Disjunctivity of <i>wp</i>	113
7.5 Finite nondeterminacy	113
7.6 Exercises	115
8 Syntactic criteria	116
8.1 Syntactic reflexion of semantic properties	116
8.2 Membership of the syntactic reflexion	120
8.3 Ψ -disjunctivity	122
8.4 Totality, disjunctivity and finite nondeterminacy	125
8.5 Exercises	129
9 Operational semantics of recursion	130
9.1 The interpreter	131
9.2 The proof of the faithful interpreter	131
9.3 The operational interpretation of tail recursion	135
9.4 General operational semantics	141
9.5 Exercises	142
10 Procedure substitutions	143
10.1 Substitutions	143
10.2 Substitution commutes with extension	144
10.3 Procedure abstraction is allowed	144
10.4 A classical example	146
10.5 Exercises	147

<i>viii</i>	<i>CONTENTS</i>
11 Induction and semantic equality	149
11.1 Congruences	150
11.2 The set <i>Lia</i>	150
11.3 The strong congruence	151
11.4 Semantic default rules	153
11.5 An application: the storage of a parameter	153
11.6 Compositionality of the strong congruence	155
11.7 The necessity of <i>Lia</i>	156
11.8 Exercises	157
12 Induction and refinement	160
12.1 Admissible preorders	160
12.2 The strong preorder	161
12.3 Construction of the strong congruence	161
12.4 Commutation up to refinement	163
12.5 Exercises	166
13 The strong preorder	167
13.1 Intermezzo: an extension of the theorem of Knaster–Tarski	167
13.2 Unfolding	169
13.3 The construction of the strong preorder	171
13.4 The abortive interpretations	173
13.5 Inf–safety	174
13.6 Sup–safety	176
13.7 Linear approximation	176
13.8 The set <i>Lia</i> and sequential composition	178
13.9 Exercises	179
14 Temporal operators	180
14.1 Stability and the function ‘always’	181
14.2 Termination and unfolding	184
14.3 The temporal predicate transformer for always	185
14.4 Temporal functions for eventually	187
14.5 Possible termination	189
14.6 Exercises	192

Cambridge University Press
0521018293 - Programs, Recursion and Unbounded Choice: Predicate-Transformation Semantics and Transformation Rules
Wim H. Hesselink
Frontmatter
[More information](#)

<i>CONTENTS</i>	<i>ix</i>
15 Predicative fairness	193
15.1 Starvation of predicates	194
15.2 An abstract syntax and weak fairness	196
15.3 A general fairness definition	197
15.4 Examples	200
15.5 The operational meaning	203
15.6 A proposal for strong fairness	206
16 Solutions of exercises	208
References	217
Index of concepts and identifiers	221

PREFACE

This book is about programs as mathematical objects. We focus on one of the aspects of programs, namely their functionality, their meaning or semantics. Following Dijkstra we express the semantics of a program by the weakest precondition of the program as a function of the postcondition. Of course, programs have other aspects, like syntactic structure, executability and (if they are executable) efficiency. In fact, perhaps surprisingly, for programming methodology it is useful to allow a large class of programs, many of which are not executable but serve as partially implemented specifications.

Weakest preconditions are used to define the meanings of programs in a clean and uniform way, without the need to introduce operational arguments. This formalism allows an effortless incorporation of unbounded nondeterminacy. Now programming methodology poses two questions. The first question is, given a specification, to design a general program that is proved to meet the specification but need not be executable or efficient, and the second question is to transform such a program into a more suitable one that also meets the specification.

We do not address the methodological question how to design, but we concentrate on the mathematical questions concerning semantic properties of programs, semantic equality of programs and the refinement relation between programs. We provide a single formal theory that supports a number of different extensions of the basic theory of computation.

The correctness of a program with respect to a specification is for us only one of its semantic properties. We are equally interested in its incorrectness or its equivalence to other programs. For example, we provide formal rules to prove that a recursive procedure does not meet its specification.

The book can be used for courses on predicate–transformation semantics of various sizes. For an introductory course, I prefer to use Chapters 1, 2, 3 and 4, but Chapter 4 can be replaced by subjects from Chapter 5 or by Chapter 6, possibly followed by 7. The more powerful methods of program transformation are contained in Chapters 10, 11 and 12; these chapters should be accessible after 4, 6

Cambridge University Press

0521018293 - Programs, Recursion and Unbounded Choice: Predicate-Transformation Semantics and Transformation Rules

Wim H. Hesselink

Frontmatter

[More information](#)

xii

PREFACE

and 7 have been covered. Chapter 14 on temporal predicate transformers can be treated directly after Chapter 4.

Some of the material grew out of courses of lectures on programming or programming theory delivered at the University of Groningen. Other parts of the book grew out of my need for a non-operational understanding of standard implementation techniques like the use of stacks for the implementation of recursive procedures. Chapters 14 and 15 were directly inspired by [Morris 1990].

I offer sincere thanks to A. de Bruin, J.J. Lukkien, J. Morris, R. Reinds, J.C.S.P. van der Woude and J. von Wright for critically reading drafts of the book. I also want to express here my deepest gratitude to the three persons who have stimulated and guided my transition from mathematics to computer science: J.W. de Bakker, E.W. Dijkstra and J.L.A. van de Snepscheut. Van de Snepscheut was for five years the inspiring leader of our department. Under his direction I got the opportunity to spend a sabbatical year at the University of Texas at Austin, in 1986/1987. There I worked with Dijkstra, who converted me to a new mathematical discipline, ten years after I had obtained my doctorate in mathematics. Finally, De Bakker showed continuous interest in my computer science papers and encouraged me to write this book. Of course, there were many more persons who stimulated my development but no enumeration can do justice to all of them.

Groningen, the Netherlands
June 1991

Wim H. Hesselink

LIST OF SYMBOLS

Symbols are listed by section number.

Postfix operators

- $-^*$ 4.3
- $-\odot$ 4.3, 10.1
- $-^e$ (where $e = 0$ or 1) 4.4
- $\heartsuit$ 0.0

Special parentheses

- $[\]$ 0.0, 1.1
- $[\ / \]$ 10.1
- $\{ \ }$ 0.6, 2.1
- $\llbracket \ \rrbracket$ 6.4

Programming operators

- $!$ 1.3
- $?$ 1.3
- $;$ 1.4, 4.7, 6.2
- $\parallel$ 1.4, 4.7, 6.2
- $\parallel_f$ 15.2

Relational operators

- $\cong$ 0.3, 1.3
- $=_V$ 1.6
- $\neq_V$ 1.6
- $\vdash$ 2.7
- $\leq$ 4.1
- $\sqsubseteq$ 5.1
- $<$ 5.6
- $\approx$ 11.3, 12.3
- $\ll$ 12.2, 13.3