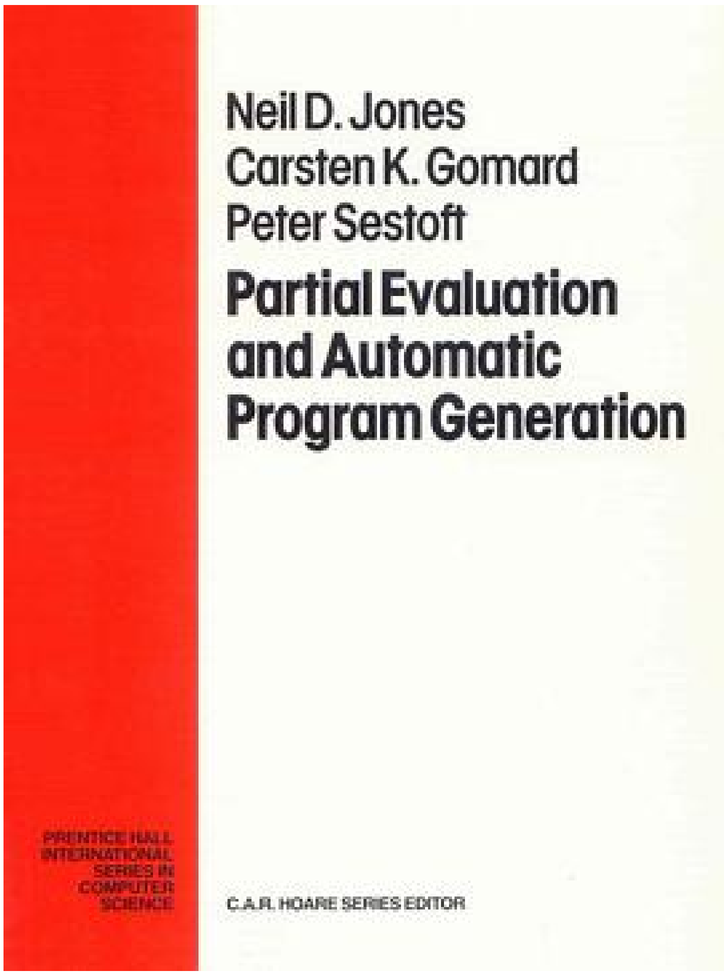
Partial Evaluation and Automatic Program Generation


Neil D. Jones
Carsten K. Gomard
Peter Sestoft
Partial Evaluation and Automatic Program Generation
PRENTICE HALL INTERNATIONAL SERIES IN COMPUTER SCIENCE
C.A.R. HOARE SERIES EDITOR

Partial Evaluation and Automatic Program Generation_下载链接1_

著者:Neil D. Jones

出版者:Prentice Hall

出版时间:1993-9

装帧:Paperback

isbn:9780130202499

The book Partial Evaluation and Automatic Program Generation gives a comprehensive presentation of partial evaluation: theory, techniques, and applications. It is suitable for self-study, and for graduate courses and advanced undergraduate courses on program transformation techniques.

作者介绍:

Neil D. Jones is now professor emeritus at DIKU, University of Copenhagen. Carsten Gomard is a partner and co-founder of Netcompany, a consultancy. Peter Sestoft is professor at the IT University of Copenhagen.

目录: Preface
1 Introduction
1.1 Partial evaluation = program specialization 1
1.2 Why do partial evaluation? 5
1.3 Computation in one stage or more 7
1.4 Partial evaluation and compilation 11
1.5 Automatic program generation 13
1.6 Critical assessment 15
1.7 Overview of the book 17
Part I: Fundamental Concepts in Programming Languages
2 Functions, Types, and Expressions
2.1 Functions 23
2.2 Types in programming languages 26
2.3 Recursive data types 32
2.4 Summary 36
2.5 Exercises 37
3 Programming Languages and Interpreters
3.1 Interpreters, compilers, and running times 38
3.2 The untyped lambda calculus: syntax and semantics 43
3.3 Three mini-languages 50
3.4 Compiling compilers 58
3.5 The central problems of compilation 60
3.6 Summary 61
3.7 Exercises 62
Part II: Principles of Partial Evaluation
4 Partial Evaluation for a Flow Chart Language
4.1 Introduction 68
4.2 What is partial evaluation? 69
4.3 Partial evaluation and compilation 73
4.4 Program specialization techniques 76
4.5 Algorithms used in mix 85
4.6 The second Futamura projection: compiler generation 86
4.7 Generating a compiler generator: mix^3 91
4.8 The tricks under the carpet 91
4.9 The granularity of binding-time analysis 94
4.10 Overview of mix performance 97
4.11 Summary and a more abstract perspective 98
4.12 Exercises 99
5 Partial Evaluation for a First-Order Functional Language
5.1 From flow charts to functions 101
5.2 Binding-time analysis by abstract interpretation 106
5.3 Adding annotations 110

5.4 Specialization algorithm for Scheme0 113
5.5 Call unfolding on the fly 118
5.6 Implementation 122
5.7 Using type rules for binding-time checking 123
5.8 Constructing the generating extension 125
5.9 Exercises 125
6 Efficiency, Speedup, and Optimality
6.1 Defining and measuring speedup 127
6.2 Flow chart mix gives linear speedup 130
6.3 Speedup analysis 132
6.4 Optimality of mix 138
6.5 Hierarchies of meta-languages 139
6.6 Exercises 141
7 Online, Offline, and Self-application
7.1 Decision making as a prephase? 145
7.2 Online and offline expression reduction 145
7.3 BTA and the taming of self-application 153
7.4 A recipe for self-application 157
7.5 Exercises 159
Part III: Partial Evaluation for Stronger Languages
8 Partial Evaluation for the Lambda Calculus 163
8.1 The lambda calculus and self-interpretation 164
8.2 Partial evaluation using a two-level lambda calculus 166
8.3 Congruence and consistency of annotations 169
8.4 Binding-time analysis 172
8.5 Simplicity versus power in Lambdamix 173
8.6 Binding-time analysis by type inference 175
8.7 BTA by solving constraints 175
8.8 Correctness of Lambdamix 183
8.9 Exercises 190
9 Partial Evaluation for Prolog (by Torben Mogensen)
9.1 An example 195
9.2 The structure of Logimix 196
9.3 Conclusion 200
9.4 Exercises 202
10 Aspects of Similix: A Partial Evaluator for a Subset of Scheme
10.1 An overview of Similix 204
10.2 Specialization with respect to functional values 210
10.3 Avoiding duplication 215
10.4 Call unfolding on the fly 217
10.5 Continuation-based reduction 218
10.6 Handling partially static structures 223
10.7 The Similix implementation 225
10.8 Exercises 225
11 Partial Evaluation for the C Language (by Lars Ole Andersen)
11.1 Introduction 229
11.2 Specialization of control flow 232
11.3 Function specialization 234
11.4 Data structures and their binding-time separation 239
11.5 Partial evaluation for C by two-level execution 245
11.6 Separation of the binding times 253
11.7 Self-application, types, and double encoding 256
11.8 C-mix: a partial evaluator for C programs 256
11.9 Towards partial evaluation for full Ansi C 258

11.10 Exercises 259
Part IV: Partial Evaluation in Practice
12 Binding-Time Improvements
12.1 A case study: Knuth, Morris, Pratt string matching 264
12.2 Bounded static variation 266
12.3 Conversion into continuation passing style 270
12.4 Eta conversion 273
12.5 Improvements derived from `free theorems' 274
12.6 Exercises 274
13 Applications of Partial Evaluation
13.1 Types of problems susceptible to partial evaluation 277
13.2 When can partial evaluation be of benefit? 285
13.3 Exercises 293
Part V: Advanced Topics
14 Termination of Partial Evaluation 297
14.1 Termination of online partial evaluators 297
14.2 Termination of offline partial evaluators 298
14.3 Binding-time analysis ensuring termination 301
14.4 Safety of BTA algorithm 305
14.5 Exercises 307
15 Program Analysis
15.1 Abstract interpretation 309
15.2 Closure analysis 314
15.3 Higher-order binding-time analysis 319
15.4 Projections and partially static data 323
15.5 Projection-based binding-time analysis 328
15.6 Describing the dynamic data 332
15.7 Summary 333
15.8 Exercises 333
16 Larger Perspectives
16.1 Relations to recursive function theory 335
16.2 Types for interpreters, compilers, and partial evaluators 337
16.3 Some research problems 346
17 Program Transformation
17.1 A language with pattern matching 347
17.2 Fold/unfold transformations 350
17.3 Partial evaluation by fold/unfold 355
17.4 Supercompilation and deforestation 358
17.5 Exercises 364
18 Guide to the Literature
18.1 A brief historical overview 366
18.2 Partial evaluation literature by subject language 368
18.3 Principles and techniques 370
18.4 Applications 372
18.5 Other topics related to partial evaluation 374
A The Self-Applicable Scheme0 Specializer
A.1 Using the Scheme0 specializer 376
A.2 Data structures in the Scheme0 specializer 378
A.3 The components of the Scheme0 specializer 380
Bibliography 389
Index 406
Back to book homepage.
· · · · · · (收起)

Partial Evaluation and Automatic Program Generation_下载链接1_

标签

计算机科学

编译原理

编程范式

理论计算机科学

pl

compiler

评论

Partial Evaluation and Automatic Program Generation_下载链接1_

书评

Partial Evaluation and Automatic Program Generation_下载链接1_