

Table of Contents

1. Preface	1
1.1 Summary	1
1.2 Motivation	1
1.2.1 The Framework	3
1.2.2 The Application	7
1.2.3 Related Work	8
1.2.4 Organization of the Monograph	10
<hr/>	
Part I. = Introduction	
<hr/>	
2. The Intraprocedural Framework	15
2.1 Intraprocedural Program Optimization	15
2.1.1 Procedures and Flow Graphs	16
2.1.2 Provably Optimal Program Transformations	17
2.1.3 Provably Precise Data Flow Analyses	18
2.2 Abstract Interpretation	18
2.2.1 Data Flow Information	19
2.2.2 Local Abstract Semantics	19
2.2.3 Global Abstract Semantics	19
2.2.4 <i>MOP</i> -Correctness and <i>MOP</i> -Precision	21
2.2.5 The Generic Fixed Point Algorithm	21
2.2.6 Formal Specification of DFA-Algorithms	23
2.2.7 Forward, Backward, and Bidirectional DFA-Algorithms	25
2.3 A Cookbook for “Intraprocedural Optimization”	26
2.3.1 Optimal Program Optimization	26
2.3.2 Precise Data Flow Analysis	27
3. Intraprocedural Code Motion: The Transformations	31
3.1 Preliminaries	31
3.2 Intraprocedural Code Motion Transformations	33
3.2.1 Admissible Transformations	34
3.2.2 Computationally Optimal Transformations	36
3.2.3 Lifetime Optimal Transformations	37

3.3	The <i>BCM</i> -Transformation	38
3.3.1	Specification	38
3.3.2	Proving Computational Optimality	39
3.4	The <i>LCM</i> -Transformation	40
3.4.1	Specification	40
3.4.2	Proving Lifetime Optimality	41
3.5	An Illustrating Example	42
4.	Intraprocedural Code Motion: The DFA-Algorithms	49
4.1	DFA-Algorithm A_{ds} : Down-Safety	49
4.1.1	Specification	50
4.1.2	Proving Precision	50
4.2	DFA-Algorithm A_{ea} : Earliestness	54
4.2.1	Specification	54
4.2.2	Proving Precision	55
4.3	DFA-Algorithm A_{dl} : Delayability	58
4.3.1	Specification	58
4.3.2	Proving Precision	59
4.4	DFA-Algorithm A_{un} : Unusability	63
4.4.1	Specification	63
4.4.2	Proving Precision	64

Part II. = The Framework

5.	The Programming Language	71
5.1	Programs	71
5.1.1	Syntax	71
5.1.2	Notations	73
5.1.3	Conventions	74
5.2	Sublanguages	75
5.3	Separate Compilation and Separate Optimization	76
6.	Higher Order Data Flow Analysis	79
6.1	Formal Callability	80
6.2	Formal Passability	84
6.3	Potential Passability	86
6.3.1	Computing Potential Passability	91
6.3.2	An Efficient Variant	94
6.4	Main Results on Potential Passability	95
6.4.1	Correctness and Precision	95
6.4.2	Complexity	97

7. The Interprocedural Setting	99
7.1 Flow Graph Systems	99
7.1.1 The Functions <i>fg</i> , <i>callee</i> , <i>caller</i> , <i>start</i> , and <i>end</i>	100
7.1.2 The Interface Between HO-DFA and IDFA	101
7.2 Interprocedural Flow Graphs	101
7.2.1 Interprocedural Paths	103
7.2.2 Complete Interprocedural Paths	105
7.3 Interprocedural Program Optimization	106
7.3.1 Provably Optimal Interprocedural Program Transformations	107
7.3.2 Provably Precise Interprocedural Data Flow Analyses	108
8. Abstract Interpretation	109
8.1 Data Flow Analysis Stacks	109
8.2 Local Abstract Semantics	111
8.2.1 The Structure of the Semantic Functions	112
8.3 Global Abstract Semantics	114
8.3.1 The Interprocedural Meet Over All Paths Approach	114
8.3.2 The Interprocedural Maximal Fixed Point Approach	115
8.4 <i>IMOP</i> -Correctness and <i>IMOP</i> -Precision	117
8.4.1 The Main Lemma	117
8.4.2 The Interprocedural Correctness Theorem	121
8.4.3 The Interprocedural Coincidence Theorem	123
8.5 The Generic Fixed Point Algorithms	126
8.5.1 Computing the Semantics of Procedures	126
8.5.2 Computing the <i>IMFP</i> -Solution	131
8.5.3 An Efficient Variant for Computing the <i>IMFP</i> -Solution	135
8.6 Formal Specification of IDFA-Algorithms	137
8.7 Forward, Backward, Bidirectional IDFA-Algorithms	139
9. Cookbook “Interprocedural Optimization”	141
9.1 Optimal Interprocedural Program Optimization	141
9.1.1 Fixing the Program Transformations and the Optimality Criterion	141
9.1.2 Fixing the Optimal Program Transformation	142
9.2 Precise Interprocedural Data Flow Analysis	142
9.2.1 Specifying the IDFA A_φ	142
9.2.2 Proving Precision of A_φ	143

Part III. = The Application

10. Interprocedural Code Motion: The Transformations	147
10.1 Differences to the Intraprocedural Setting	148
10.1.1 Computational Optimality	148
10.1.2 Safety	153
10.1.3 Down-Safety	154
10.1.4 Canonicity	157
10.2 Preliminaries	157
10.2.1 Basic Definitions	160
10.2.2 First Results	164
10.3 Interprocedural Code Motion Transformations	166
10.3.1 Admissible Transformations	167
10.3.2 Computationally Optimal Transformations	170
10.3.3 Lifetime Optimal Transformations	171
10.4 The <i>IBCM</i> -Transformation	173
10.4.1 Specification	174
10.4.2 Proving Computational Optimality	178
10.5 The <i>ILCM</i> -Transformation	189
10.5.1 Specification	191
10.5.2 Proving Lifetime Optimality	196
10.6 An Example	201
11. Interprocedural Code Motion: The IDFA-Algorithms	209
11.1 IDFA-Algorithm A_{ds}	209
11.1.1 Specification	210
11.1.2 Proving Precision	212
11.2 IDFA-Algorithm A_{ea}	221
11.2.1 Specification	221
11.2.2 Proving Precision	223
11.3 IDFA-Algorithm A_{dl}	229
11.3.1 Specification	229
11.3.2 Proving Precision	230
11.4 IDFA-Algorithm A_{un}	238
11.4.1 Specification	239
11.4.2 Proving Precision	240

Part IV. = Conclusion

12. Perspectives	251
12.1 Reconsidering Code Motion Anomalies	251
12.2 Pragmatics	260
12.2.1 Static Procedure Nesting and Complexity of IDFA....	260
12.2.2 Aliasing: Call by Name and Call by Reference.....	264
12.3 Future Work	265
12.3.1 The Framework Side	265
12.3.2 The Application Side	267
12.3.3 Generators for IDFA and Optimization	269
Bibliography	271
Index	285



<http://www.springer.com/978-3-540-65123-9>

Optimal Interprocedural Program Optimization
A New Framework and Its Application

Knoop, J.

1998, XXV, 288 p., Softcover

ISBN: 978-3-540-65123-9