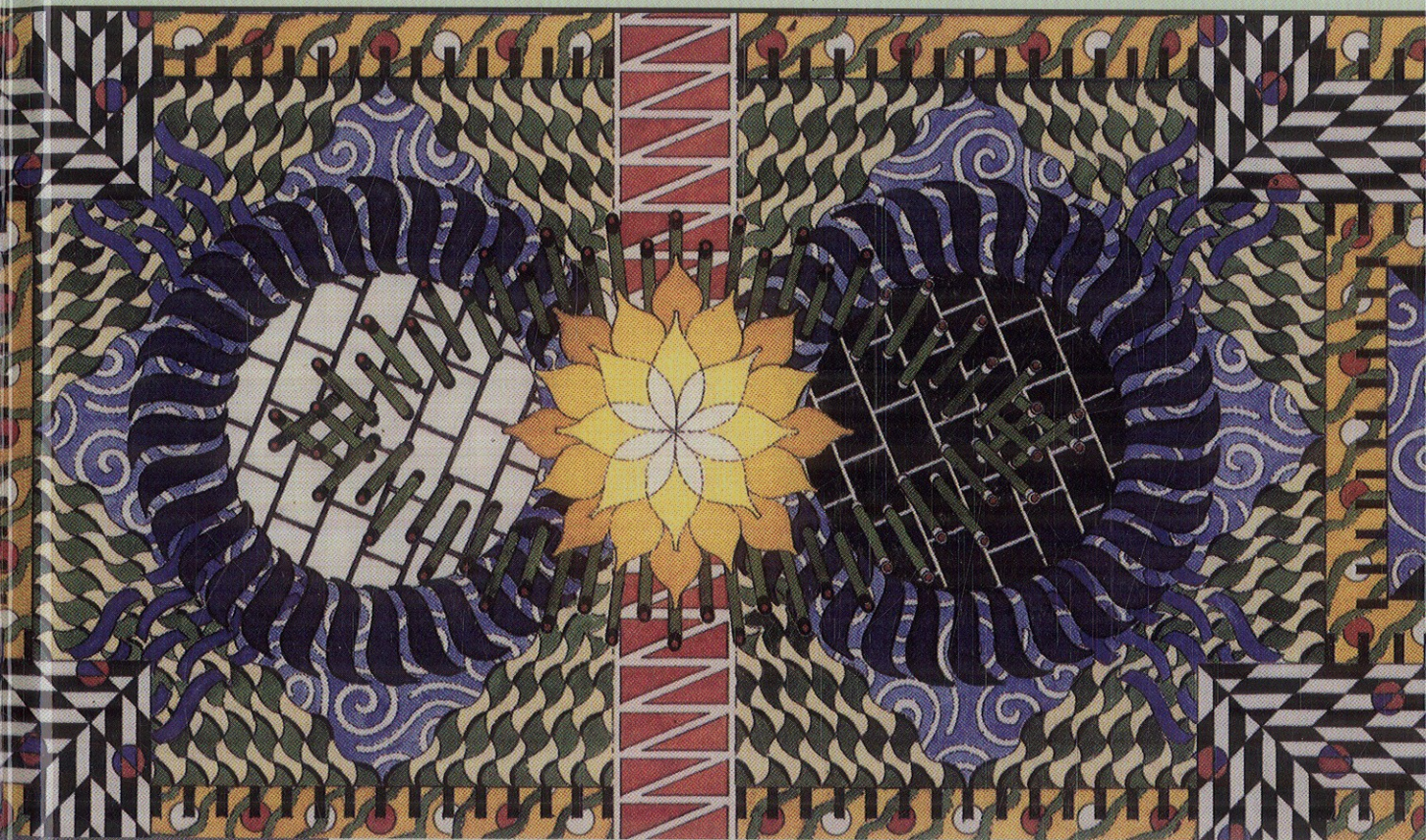


ENGINEERING A COMPILER

— SECOND EDITION —



MK
MORGAN KAUFMANN

Keith D. Cooper & Linda Torczon

Contents

About the Authors	iv
About the Cover	viii
Preface	xix

CHAPTER 1 Overview of Compilation **1**

1.1 Introduction	1
1.2 Compiler Structure	6
1.3 Overview of Translation	9
1.3.1 The Front End	10
1.3.2 The Optimizer	14
1.3.3 The Back End	15
1.4 Summary and Perspective	21
Chapter Notes	22
Exercises	23

CHAPTER 2 Scanners **25**

2.1 Introduction	25
2.2 Recognizing Words	27
2.2.1 A Formalism for Recognizers	29
2.2.2 Recognizing More Complex Words	31
2.3 Regular Expressions	34
2.3.1 Formalizing the Notation	35
2.3.2 Examples	36
2.3.3 Closure Properties of REs	39
2.4 From Regular Expression to Scanner	42
2.4.1 Nondeterministic Finite Automata	43
2.4.2 Regular Expression to NFA: Thompson's Construction	45
2.4.3 NFA to DFA: The Subset Construction	47
2.4.4 DFA to Minimal DFA: Hopcroft's Algorithm	53
2.4.5 Using a DFA as a Recognizer	57
2.5 Implementing Scanners	59
2.5.1 Table-Driven Scanners	60
2.5.2 Direct-Coded Scanners	65
2.5.3 Hand-Coded Scanners	69
2.5.4 Handling Keywords	72

2.6	Advanced Topics	74
2.6.1	DFA to Regular Expression	74
2.6.2	Another Approach to DFA Minimization: Brzozowski's Algorithm	75
2.6.3	Closure-Free Regular Expressions	77
2.7	Chapter Summary and Perspective	78
	Chapter Notes	78
	Exercises	80
CHAPTER 3 Parsers.....		83
3.1	Introduction	83
3.2	Expressing Syntax	85
3.2.1	Why Not Regular Expressions?	85
3.2.2	Context-Free Grammars	86
3.2.3	More Complex Examples	89
3.2.4	Encoding Meaning into Structure	92
3.2.5	Discovering a Derivation for an Input String	95
3.3	Top-Down Parsing	96
3.3.1	Transforming a Grammar for Top-Down Parsing	98
3.3.2	Top-Down Recursive-Descent Parsers	108
3.3.3	Table-Driven LL(1) Parsers	110
3.4	Bottom-Up Parsing	116
3.4.1	The LR(1) Parsing Algorithm	118
3.4.2	Building LR(1) Tables	124
3.4.3	Errors in the Table Construction	136
3.5	Practical Issues	141
3.5.1	Error Recovery	141
3.5.2	Unary Operators	142
3.5.3	Handling Context-Sensitive Ambiguity	143
3.5.4	Left versus Right Recursion	144
3.6	Advanced Topics	147
3.6.1	Optimizing a Grammar	148
3.6.2	Reducing the Size of LR(1) Tables	150
3.7	Summary and Perspective	155
	Chapter Notes	156
	Exercises	157

CHAPTER 4 Context-Sensitive Analysis	161
4.1 Introduction	161
4.2 An Introduction to Type Systems	164
4.2.1 The Purpose of Type Systems	165
4.2.2 Components of a Type System	170
4.3 The Attribute-Grammar Framework	182
4.3.1 Evaluation Methods	186
4.3.2 Circularity	187
4.3.3 Extended Examples	187
4.3.4 Problems with the Attribute-Grammar Approach	194
4.4 Ad Hoc Syntax-Directed Translation	198
4.4.1 Implementing Ad Hoc Syntax-Directed Translation	199
4.4.2 Examples	202
4.5 Advanced Topics	211
4.5.1 Harder Problems in Type Inference	211
4.5.2 Changing Associativity	213
4.6 Summary and Perspective	215
Chapter Notes	216
Exercises	217
 CHAPTER 5 Intermediate Representations	 221

5.1 Introduction	221
5.1.1 A Taxonomy of Intermediate Representations	223
5.2 Graphical IRs	226
5.2.1 Syntax-Related Trees	226
5.2.2 Graphs	230
5.3 Linear IRs	235
5.3.1 Stack-Machine Code	237
5.3.2 Three-Address Code	237
5.3.3 Representing Linear Codes	238
5.3.4 Building a Control-Flow Graph from a Linear Code	241
5.4 Mapping Values to Names	243
5.4.1 Naming Temporary Values	244
5.4.2 Static Single-Assignment Form	246
5.4.3 Memory Models	250

5.5	Symbol Tables	253
5.5.1	Hash Tables	254
5.5.2	Building a Symbol Table	255
5.5.3	Handling Nested Scopes	256
5.5.4	The Many Uses for Symbol Tables	261
5.5.5	Other Uses for Symbol Table Technology	263
5.6	Summary and Perspective	264
	Chapter Notes	264
	Exercises	265
CHAPTER 6 The Procedure Abstraction		269
6.1	Introduction	269
6.2	Procedure Calls	272
6.3	Name Spaces	276
6.3.1	Name Spaces of Algol-like Languages	276
6.3.2	Runtime Structures to Support Algol-like Languages	280
6.3.3	Name Spaces of Object-Oriented Languages	285
6.3.4	Runtime Structures to Support Object-Oriented Languages	290
6.4	Communicating Values Between Procedures	297
6.4.1	Passing Parameters	297
6.4.2	Returning Values	301
6.4.3	Establishing Addressability	301
6.5	Standardized Linkages	308
6.6	Advanced Topics	312
6.6.1	Explicit Heap Management	313
6.6.2	Implicit Deallocation	317
6.7	Summary and Perspective	322
	Chapter Notes	323
	Exercises	324
CHAPTER 7 Code Shape		331
7.1	Introduction	331
7.2	Assigning Storage Locations	334
7.2.1	Placing Runtime Data Structures	335
7.2.2	Layout for Data Areas	336
7.2.3	Keeping Values in Registers	340
7.3	Arithmetic Operators	342
7.3.1	Reducing Demand for Registers	344

7.3.2	Accessing Parameter Values	345
7.3.3	Function Calls in an Expression	347
7.3.4	Other Arithmetic Operators	348
7.3.5	Mixed-Type Expressions	348
7.3.6	Assignment as an Operator	349
7.4	Boolean and Relational Operators	350
7.4.1	Representations	351
7.4.2	Hardware Support for Relational Operations	353
7.5	Storing and Accessing Arrays	359
7.5.1	Referencing a Vector Element	359
7.5.2	Array Storage Layout	361
7.5.3	Referencing an Array Element	362
7.5.4	Range Checking	367
7.6	Character Strings	369
7.6.1	String Representations	370
7.6.2	String Assignment	370
7.6.3	String Concatenation	372
7.6.4	String Length	373
7.7	Structure References	374
7.7.1	Understanding Structure Layouts	375
7.7.2	Arrays of Structures	376
7.7.3	Unions and Runtime Tags	377
7.7.4	Pointers and Anonymous Values	378
7.8	Control-Flow Constructs	380
7.8.1	Conditional Execution	381
7.8.2	Loops and Iteration	384
7.8.3	Case Statements	388
7.9	Procedure Calls	392
7.9.1	Evaluating Actual Parameters	393
7.9.2	Saving and Restoring Registers	394
7.10	Summary and Perspective	396
	Chapter Notes	397
	Exercises	398

CHAPTER 8 Introduction to Optimization **405**

8.1	Introduction	405
8.2	Background	407
8.2.1	Examples	408
8.2.2	Considerations for Optimization	412
8.2.3	Opportunities for Optimization	415

8.3	Scope of Optimization	417
8.4	Local Optimization	420
8.4.1	Local Value Numbering	420
8.4.2	Tree-Height Balancing	428
8.5	Regional Optimization	437
8.5.1	Superlocal Value Numbering	437
8.5.2	Loop Unrolling	441
8.6	Global Optimization	445
8.6.1	Finding Uninitialized Variables with Live Information	445
8.6.2	Global Code Placement	451
8.7	Interprocedural Optimization	457
8.7.1	Inline Substitution	458
8.7.2	Procedure Placement	462
8.7.3	Compiler Organization for Interprocedural Optimization	467
8.8	Summary and Perspective	469
	Chapter Notes	470
	Exercises	471

CHAPTER 9 Data-Flow Analysis **475**

9.1	Introduction	475
9.2	Iterative Data-Flow Analysis	477
9.2.1	Dominance	478
9.2.2	Live-Variable Analysis	482
9.2.3	Limitations on Data-Flow Analysis	487
9.2.4	Other Data-Flow Problems	490
9.3	Static Single-Assignment Form	495
9.3.1	A Simple Method for Building SSA Form	496
9.3.2	Dominance Frontiers	497
9.3.3	Placing ϕ -Functions	500
9.3.4	Renaming	505
9.3.5	Translation Out of SSA Form	510
9.3.6	Using SSA Form	515
9.4	Interprocedural Analysis	519
9.4.1	Call-Graph Construction	520
9.4.2	Interprocedural Constant Propagation	522
9.5	Advanced Topics	526
9.5.1	Structural Data-Flow Algorithms and Reducibility	527
9.5.2	Speeding up the Iterative Dominance Framework	530

9.6 Summary and Perspective	533
Chapter Notes	534
Exercises	535
CHAPTER 10 Scalar Optimizations	539
10.1 Introduction	539
10.2 Eliminating Useless and Unreachable Code	544
10.2.1 Eliminating Useless Code	544
10.2.2 Eliminating Useless Control Flow	547
10.2.3 Eliminating Unreachable Code	550
10.3 Code Motion	551
10.3.1 Lazy Code Motion	551
10.3.2 Code Hoisting	559
10.4 Specialization	560
10.4.1 Tail-Call Optimization	561
10.4.2 Leaf-Call Optimization	562
10.4.3 Parameter Promotion	563
10.5 Redundancy Elimination	565
10.5.1 Value Identity versus Name Identity	565
10.5.2 Dominator-based Value Numbering	566
10.6 Enabling Other Transformations	569
10.6.1 Superblock Cloning	570
10.6.2 Procedure Cloning	571
10.6.3 Loop Unswitching	572
10.6.4 Renaming	573
10.7 Advanced Topics	575
10.7.1 Combining Optimizations	575
10.7.2 Strength Reduction	580
10.7.3 Choosing an Optimization Sequence	591
10.8 Summary and Perspective	592
Chapter Notes	593
Exercises	594
CHAPTER 11 Instruction Selection	597
11.1 Introduction	597
11.2 Code Generation	600
11.3 Extending the Simple Treewalk Scheme	603
11.4 Instruction Selection via Tree-Pattern Matching	610
11.4.1 Rewrite Rules	611
11.4.2 Finding a Tiling	616
11.4.3 Tools	620

11.5	Instruction Selection via Peephole Optimization	621
11.5.1	Peephole Optimization	622
11.5.2	Peephole Transformers	629
11.6	Advanced Topics	632
11.6.1	Learning Peephole Patterns	632
11.6.2	Generating Instruction Sequences	633
11.7	Summary and Perspective	634
	Chapter Notes	635
	Exercises	637

CHAPTER 12 Instruction Scheduling 639

12.1	Introduction	639
12.2	The Instruction-Scheduling Problem	643
12.2.1	Other Measures of Schedule Quality	648
12.2.2	What Makes Scheduling Hard?	649
12.3	Local List Scheduling	651
12.3.1	The Algorithm	651
12.3.2	Scheduling Operations with Variable Delays	654
12.3.3	Extending the Algorithm	655
12.3.4	Tie Breaking in the List-Scheduling Algorithm	655
12.3.5	Forward versus Backward List Scheduling	656
12.3.6	Improving the Efficiency of List Scheduling	660
12.4	Regional Scheduling	661
12.4.1	Scheduling Extended Basic Blocks	661
12.4.2	Trace Scheduling	663
12.4.3	Cloning for Context	664
12.5	Advanced Topics	666
12.5.1	The Strategy of Software Pipelining	666
12.5.2	An Algorithm for Software Pipelining	670
12.6	Summary and Perspective	673
	Chapter Notes	673
	Exercises	675

CHAPTER 13 Register Allocation 679

13.1	Introduction	679
13.2	Background Issues	681
13.2.1	Memory versus Registers	681
13.2.2	Allocation versus Assignment	682
13.2.3	Register Classes	683
13.3	Local Register Allocation and Assignment	684
13.3.1	Top-Down Local Register Allocation	685

13.3.2	Bottom-Up Local Register Allocation	686
13.3.3	Moving Beyond Single Blocks	689
13.4	Global Register Allocation and Assignment	693
13.4.1	Discovering Global Live Ranges	696
13.4.2	Estimating Global Spill Costs	697
13.4.3	Interferences and the Interference Graph	699
13.4.4	Top-Down Coloring	702
13.4.5	Bottom-Up Coloring	704
13.4.6	Coalescing Copies to Reduce Degree	706
13.4.7	Comparing Top-Down and Bottom-Up Global Allocators	708
13.4.8	Encoding Machine Constraints in the Interference Graph	711
13.5	Advanced Topics	713
13.5.1	Variations on Graph-Coloring Allocation	713
13.5.2	Global Register Allocation over SSA Form	717
13.6	Summary and Perspective	718
	Chapter Notes	719
	Exercises	720

APPENDIX A ILOC **725**

A.1	Introduction	725
A.2	Naming Conventions	727
A.3	Individual Operations	728
A.3.1	Arithmetic	728
A.3.2	Shifts	729
A.3.3	Memory Operations	729
A.3.4	Register-to-Register Copy Operations	730
A.4	Control-Flow Operations	731
A.4.1	Alternate Comparison and Branch Syntax	732
A.4.2	Jumps	732
A.5	Representing SSA Form	733

APPENDIX B Data Structures **737**

B.1	Introduction	737
B.2	Representing Sets	738
B.2.1	Representing Sets as Ordered Lists	739
B.2.2	Representing Sets as Bit Vectors	741
B.2.3	Representing Sparse Sets	741
B.3	Implementing Intermediate Representations	743
B.3.1	Graphical Intermediate Representations	743
B.3.2	Linear Intermediate Forms	748

B.4	Implementing Hash Tables	750
B.4.1	Choosing a Hash Function	750
B.4.2	Open Hashing	752
B.4.3	Open Addressing	754
B.4.4	Storing Symbol Records	756
B.4.5	Adding Nested Lexical Scopes	757
B.5	A Flexible Symbol-Table Design	760
 BIBLIOGRAPHY		765
INDEX		787