Dick Grune, Ceriel J.H. Jacobs

Parsing Techniques

2nd edition

— Monograph —

September 27, 2007

Springer
Berlin Heidelberg New York
Hong Kong London
Milan Paris Tokyo

Contents

Preface to the Second Edition v

Preface to the First Edition xi

1 Introduction 1
- 1.1 Parsing as a Craft 2
- 1.2 The Approach Used 2
- 1.3 Outline of the Contents 3
- 1.4 The Annotated Bibliography 4

2 Grammars as a Generating Device 5
- 2.1 Languages as Infinite Sets 5
  - 2.1.1 Language 5
  - 2.1.2 Grammars 7
  - 2.1.3 Problems with Infinite Sets 8
  - 2.1.4 Describing a Language through a Finite Recipe 12
- 2.2 Formal Grammars 14
  - 2.2.1 The Formalism of Formal Grammars 14
  - 2.2.2 Generating Sentences from a Formal Grammar 15
  - 2.2.3 The Expressive Power of Formal Grammars 17
- 2.3 The Chomsky Hierarchy of Grammars and Languages 19
  - 2.3.1 Type 1 Grammars 19
  - 2.3.2 Type 2 Grammars 23
  - 2.3.3 Type 3 Grammars 30
  - 2.3.4 Type 4 Grammars 33
  - 2.3.5 Conclusion 34
- 2.4 Actually Generating Sentences from a Grammar 34
  - 2.4.1 The Phrase-Structure Case 34
  - 2.4.2 The CS Case 36
  - 2.4.3 The CF Case 36
- 2.5 To Shrink or Not To Shrink 38

2.6 Grammars that Produce the Empty Language ... 41
2.7 The Limitations of CF and FS Grammars ... 42
2.7.1 The *uvwxy* Theorem ... 42
2.7.2 The *uvw* Theorem ... 45
2.8 CF and FS Grammars as Transition Graphs ... 45
2.9 Hygiene in Context-Free Grammars ... 47
2.9.1 Undefined Non-Terminals ... 48
2.9.2 Unreachable Non-Terminals ... 48
2.9.3 Non-Productive Rules and Non-Terminals ... 48
2.9.4 Loops ... 48
2.9.5 Cleaning up a Context-Free Grammar ... 49
2.10 Set Properties of Context-Free and Regular Languages ... 52
2.11 The Semantic Connection ... 54
2.11.1 Attribute Grammars ... 54
2.11.2 Transduction Grammars ... 55
2.11.3 Augmented Transition Networks ... 56
2.12 A Metaphorical Comparison of Grammar Types ... 56
2.13 Conclusion ... 59

3 Introduction to Parsing ... 61
3.1 The Parse Tree ... 61
3.1.1 The Size of a Parse Tree ... 62
3.1.2 Various Kinds of Ambiguity ... 63
3.1.3 Linearization of the Parse Tree ... 65
3.2 Two Ways to Parse a Sentence ... 65
3.2.1 Top-Down Parsing ... 66
3.2.2 Bottom-Up Parsing ... 67
3.2.3 Applicability ... 68
3.3 Non-Deterministic Automata ... 69
3.3.1 Constructing the NDA ... 70
3.3.2 Constructing the Control Mechanism ... 70
3.4 Recognition and Parsing for Type 0 to Type 4 Grammars ... 71
3.4.1 Time Requirements ... 71
3.4.2 Type 0 and Type 1 Grammars ... 72
3.4.3 Type 2 Grammars ... 73
3.4.4 Type 3 Grammars ... 75
3.4.5 Type 4 Grammars ... 75
3.5 An Overview of Context-Free Parsing Methods ... 76
3.5.1 Directionality ... 76
3.5.2 Search Techniques ... 77
3.5.3 General Directional Methods ... 78
3.5.4 Linear Methods ... 80
3.5.5 Deterministic Top-Down and Bottom-Up Methods ... 82
3.5.6 Non-Canonical Methods ... 83
3.5.7 Generalized Linear Methods ... 84

3.5.8 Conclusion 84
3.6 The “Strength” of a Parsing Technique 84
3.7 Representations of Parse Trees 85
3.7.1 Parse Trees in the Producer-Consumer Model 86
3.7.2 Parse Trees in the Data Structure Model 87
3.7.3 Parse Forests 87
3.7.4 Parse-Forest Grammars 91
3.8 When are we done Parsing? 93
3.9 Transitive Closure 95
3.10 The Relation between Parsing and Boolean Matrix Multiplication . . 97
3.11 Conclusion 100

4 General Non-Directional Parsing 103
4.1 Unger’s Parsing Method 104
4.1.1 Unger’s Method without ε-Rules or Loops 104
4.1.2 Unger’s Method with ε-Rules 107
4.1.3 Getting Parse-Forest Grammars from Unger Parsing 110
4.2 The CYK Parsing Method 112
4.2.1 CYK Recognition with General CF Grammars 112
4.2.2 CYK Recognition with a Grammar in Chomsky Normal Form 116
4.2.3 Transforming a CF Grammar into Chomsky Normal Form . . 119
4.2.4 The Example Revisited 122
4.2.5 CYK Parsing with Chomsky Normal Form 124
4.2.6 Undoing the Effect of the CNF Transformation 125
4.2.7 A Short Retrospective of CYK 128
4.2.8 Getting Parse-Forest Grammars from CYK Parsing 129
4.3 Tabular Parsing 129
4.3.1 Top-Down Tabular Parsing 131
4.3.2 Bottom-Up Tabular Parsing 133
4.4 Conclusion 134

5 Regular Grammars and Finite-State Automata 137
5.1 Applications of Regular Grammars 137
5.1.1 Regular Languages in CF Parsing 137
5.1.2 Systems with Finite Memory 139
5.1.3 Pattern Searching 141
5.1.4 SGML and XML Validation 141
5.2 Producing from a Regular Grammar 141
5.3 Parsing with a Regular Grammar 143
5.3.1 Replacing Sets by States 144
5.3.2 ε-Transitions and Non-Standard Notation 147
5.4 Manipulating Regular Grammars and Regular Expressions 148
5.4.1 Regular Grammars from Regular Expressions 149
5.4.2 Regular Expressions from Regular Grammars 151
5.5 Manipulating Regular Languages 152

5.6 Left-Regular Grammars 154
5.7 Minimizing Finite-State Automata 156
5.8 Top-Down Regular Expression Recognition 158
5.8.1 The Recognizer 158
5.8.2 Evaluation 159
5.9 Semantics in FS Systems 160
5.10 Fast Text Search Using Finite-State Automata 161
5.11 Conclusion 162

6 General Directional Top-Down Parsing 165
6.1 Imitating Leftmost Derivations 165
6.2 The Pushdown Automaton 167
6.3 Breadth-First Top-Down Parsing 171
6.3.1 An Example 173
6.3.2 A Counterexample: Left Recursion 173
6.4 Eliminating Left Recursion 175
6.5 Depth-First (Backtracking) Parsers 176
6.6 Recursive Descent 177
6.6.1 A Naive Approach 179
6.6.2 Exhaustive Backtracking Recursive Descent 183
6.6.3 Breadth-First Recursive Descent 185
6.7 Definite Clause Grammars 188
6.7.1 Prolog 188
6.7.2 The DCG Format 189
6.7.3 Getting Parse Tree Information 190
6.7.4 Running Definite Clause Grammar Programs 190
6.8 Cancellation Parsing 192
6.8.1 Cancellation Sets 192
6.8.2 The Transformation Scheme 193
6.8.3 Cancellation Parsing with ε-Rules 196
6.9 Conclusion 197

7 General Directional Bottom-Up Parsing 199
7.1 Parsing by Searching 201
7.1.1 Depth-First (Backtracking) Parsing 201
7.1.2 Breadth-First (On-Line) Parsing 202
7.1.3 A Combined Representation 203
7.1.4 A Slightly More Realistic Example 204
7.2 The Earley Parser 206
7.2.1 The Basic Earley Parser 206
7.2.2 The Relation between the Earley and CYK Algorithms 212
7.2.3 Handling ε-Rules 214
7.2.4 Exploiting Look-Ahead 219
7.2.5 Left and Right Recursion 224
7.3 Chart Parsing 226

7.3.1 Inference Rules 227
7.3.2 A Transitive Closure Algorithm 227
7.3.3 Completion 229
7.3.4 Bottom-Up (Actually Left-Corner) 229
7.3.5 The Agenda 229
7.3.6 Top-Down 231
7.3.7 Conclusion 232
7.4 Conclusion 233

8 Deterministic Top-Down Parsing 235
8.1 Replacing Search by Table Look-Up 236
8.2 LL(1) Parsing 239
8.2.1 LL(1) Parsing without ε-Rules 239
8.2.2 LL(1) Parsing with ε-Rules 242
8.2.3 LL(1) versus Strong-LL(1) 247
8.2.4 Full LL(1) Parsing 248
8.2.5 Solving LL(1) Conflicts 251
8.2.6 LL(1) and Recursive Descent 253
8.3 Increasing the Power of Deterministic LL Parsing 254
8.3.1 LL(*k*) Grammars 254
8.3.2 Linear-Approximate LL(*k*) 256
8.3.3 LL-Regular 257
8.4 Getting a Parse Tree Grammar from LL(1) Parsing 258
8.5 Extended LL(1) Grammars 259
8.6 Conclusion 260

9 Deterministic Bottom-Up Parsing 263
9.1 Simple Handle-Finding Techniques 265
9.2 Precedence Parsing 266
9.2.1 Parenthesis Generators 267
9.2.2 Constructing the Operator-Precedence Table 269
9.2.3 Precedence Functions 271
9.2.4 Further Precedence Methods 272
9.3 Bounded-Right-Context Parsing 275
9.3.1 Bounded-Context Techniques 276
9.3.2 Floyd Productions 277
9.4 LR Methods 278
9.5 LR(0) 280
9.5.1 The LR(0) Automaton 280
9.5.2 Using the LR(0) Automaton 283
9.5.3 LR(0) Conflicts 286
9.5.4 ε-LR(0) Parsing 287
9.5.5 Practical LR Parse Table Construction 289
9.6 LR(1) 290
9.6.1 LR(1) with ε-Rules 295

9.6.2 LR($k > 1$) Parsing ... 297
9.6.3 Some Properties of LR(k) Parsing ... 299
9.7 LALR(1) ... 300
9.7.1 Constructing the LALR(1) Parsing Tables ... 302
9.7.2 Identifying LALR(1) Conflicts ... 314
9.8 SLR(1) ... 314
9.9 Conflict Resolvers ... 315
9.10 Further Developments of LR Methods ... 316
9.10.1 Elimination of Unit Rules ... 316
9.10.2 Reducing the Stack Activity ... 317
9.10.3 Regular Right Part Grammars ... 318
9.10.4 Incremental Parsing ... 318
9.10.5 Incremental Parser Generation ... 318
9.10.6 Recursive Ascent ... 319
9.10.7 Regular Expressions of LR Languages ... 319
9.11 Getting a Parse Tree Grammar from LR Parsing ... 319
9.12 Left and Right Contexts of Parsing Decisions ... 320
9.12.1 The Left Context of a State ... 321
9.12.2 The Right Context of an Item ... 322
9.13 Exploiting the Left and Right Contexts ... 323
9.13.1 Discriminating-Reverse (DR) Parsing ... 324
9.13.2 LR-Regular ... 327
9.13.3 LAR(m) Parsing ... 333
9.14 LR(k) as an Ambiguity Test ... 338
9.15 Conclusion ... 338

10 Non-Canonical Parsers ... 343
10.1 Top-Down Non-Canonical Parsing ... 344
10.1.1 Left-Corner Parsing ... 344
10.1.2 Deterministic Cancellation Parsing ... 353
10.1.3 Partitioned LL ... 354
10.1.4 Discussion ... 357
10.2 Bottom-Up Non-Canonical Parsing ... 357
10.2.1 Total Precedence ... 358
10.2.2 NSLR(1) ... 359
10.2.3 LR(k,∞) ... 364
10.2.4 Partitioned LR ... 372
10.3 General Non-Canonical Parsing ... 377
10.4 Conclusion ... 379

11 Generalized Deterministic Parsers ... 381
11.1 Generalized LR Parsing ... 382
11.1.1 The Basic GLR Parsing Algorithm ... 382
11.1.2 Necessary Optimizations ... 383
11.1.3 Hidden Left Recursion and Loops ... 387

11.1.4 Extensions and Improvements 390
11.2 Generalized LL Parsing 391
11.2.1 Simple Generalized LL Parsing 391
11.2.2 Generalized LL Parsing with Left-Recursion 393
11.2.3 Generalized LL Parsing with ε-Rules 395
11.2.4 Generalized Cancellation and LC Parsing 397
11.3 Conclusion 398

12 Substring Parsing 399
12.1 The Suffix Grammar 401
12.2 General (Non-Linear) Methods 402
12.2.1 A Non-Directional Method 403
12.2.2 A Directional Method 407
12.3 Linear-Time Methods for LL and LR Grammars 408
12.3.1 Linear-Time Suffix Parsing for LL(1) Grammars 409
12.3.2 Linear-Time Suffix Parsing for LR(1) Grammars 414
12.3.3 Tabular Methods 418
12.3.4 Discussion 421
12.4 Conclusion 421

13 Parsing as Intersection 425
13.1 The Intersection Algorithm 426
13.1.1 The Rule Sets I_{rules}, I_{rough}, and I 427
13.1.2 The Languages of I_{rules}, I_{rough}, and I 429
13.1.3 An Example: Parsing Arithmetic Expressions 430
13.2 The Parsing of FSAs 431
13.2.1 Unknown Tokens 431
13.2.2 Substring Parsing by Intersection 431
13.2.3 Filtering 435
13.3 Time and Space Requirements 436
13.4 Reducing the Intermediate Size: Earley's Algorithm on FSAs 437
13.5 Error Handling Using Intersection Parsing 439
13.6 Conclusion 441

14 Parallel Parsing 443
14.1 The Reasons for Parallel Parsing 443
14.2 Multiple Serial Parsers 444
14.3 Process-Configuration Parsers 447
14.3.1 A Parallel Bottom-up GLR Parser 448
14.3.2 Some Other Process-Configuration Parsers 452
14.4 Connectionist Parsers 453
14.4.1 Boolean Circuits 453
14.4.2 A CYK Recognizer on a Boolean Circuit 454
14.4.3 Rytter's Algorithm 460
14.5 Conclusion 470

15 Non-Chomsky Grammars and Their Parsers 473
15.1 The Unsuitability of Context-Sensitive Grammars 473
15.1.1 Understanding Context-Sensitive Grammars 474
15.1.2 Parsing with Context-Sensitive Grammars 475
15.1.3 Expressing Semantics in Context-Sensitive Grammars 475
15.1.4 Error Handling in Context-Sensitive Grammars 475
15.1.5 Alternatives 476
15.2 Two-Level Grammars 476
15.2.1 VW Grammars 477
15.2.2 Expressing Semantics in a VW Grammar 480
15.2.3 Parsing with VW Grammars 482
15.2.4 Error Handling in VW Grammars 484
15.2.5 Infinite Symbol Sets 484
15.3 Attribute and Affix Grammars 485
15.3.1 Attribute Grammars 485
15.3.2 Affix Grammars 488
15.4 Tree-Adjoining Grammars 492
15.4.1 Cross-Dependencies 492
15.4.2 Parsing with TAGs 497
15.5 Coupled Grammars 500
15.5.1 Parsing with Coupled Grammars 501
15.6 Ordered Grammars 502
15.6.1 Rule Ordering by Control Grammar 502
15.6.2 Parsing with Rule-Ordered Grammars 503
15.6.3 Marked Ordered Grammars 504
15.6.4 Parsing with Marked Ordered Grammars 505
15.7 Recognition Systems 506
15.7.1 Properties of a Recognition System 507
15.7.2 Implementing a Recognition System 509
15.7.3 Parsing with Recognition Systems 512
15.7.4 Expressing Semantics in Recognition Systems 512
15.7.5 Error Handling in Recognition Systems 513
15.8 Boolean Grammars 514
15.8.1 Expressing Context Checks in Boolean Grammars 514
15.8.2 Parsing with Boolean Grammars 516
15.8.3 §-Calculus 516
15.9 Conclusion 517

16 Error Handling 521
16.1 Detection versus Recovery versus Correction 521
16.2 Parsing Techniques and Error Detection 523
16.2.1 Error Detection in Non-Directional Parsing Methods 523
16.2.2 Error Detection in Finite-State Automata 524
16.2.3 Error Detection in General Directional Top-Down Parsers . . 524
16.2.4 Error Detection in General Directional Bottom-Up Parsers . . 524

16.2.5 Error Detection in Deterministic Top-Down Parsers 525
16.2.6 Error Detection in Deterministic Bottom-Up Parsers 525
16.3 Recovering from Errors 526
16.4 Global Error Handling 526
16.5 Regional Error Handling 530
16.5.1 Backward/Forward Move Error Recovery 530
16.5.2 Error Recovery with Bounded-Context Grammars 532
16.6 Local Error Handling 533
16.6.1 Panic Mode 534
16.6.2 FOLLOW-Set Error Recovery 534
16.6.3 Acceptable-Sets Derived from Continuations 535
16.6.4 Insertion-Only Error Correction 537
16.6.5 Locally Least-Cost Error Recovery 539
16.7 Non-Correcting Error Recovery.............................. 540
16.7.1 Detection and Recovery.............................. 540
16.7.2 Locating the Error 541
16.8 Ad Hoc Methods 542
16.8.1 Error Productions 542
16.8.2 Empty Table Slots 543
16.8.3 Error Tokens 543
16.9 Conclusion 543

17 Practical Parser Writing and Usage 545
17.1 A Comparative Survey 545
17.1.1 Considerations 545
17.1.2 General Parsers...................................... 546
17.1.3 General Substring Parsers 547
17.1.4 Linear-Time Parsers 548
17.1.5 Linear-Time Substring Parsers 549
17.1.6 Obtaining and Using a Parser Generator 549
17.2 Parser Construction 550
17.2.1 Interpretive, Table-Based, and Compiled Parsers 550
17.2.2 Parsing Methods and Implementations 551
17.3 A Simple General Context-Free Parser 553
17.3.1 Principles of the Parser 553
17.3.2 The Program 554
17.3.3 Handling Left Recursion 559
17.3.4 Parsing in Polynomial Time 560
17.4 Programming Language Paradigms........................... 563
17.4.1 Imperative and Object-Oriented Programming 563
17.4.2 Functional Programming 564
17.4.3 Logic Programming 567
17.5 Alternative Uses of Parsing 567
17.5.1 Data Compression 567
17.5.2 Machine Code Generation............................. 570

17.5.3 Support of Logic Languages 573
17.6 Conclusion 573

18 Annotated Bibliography 575
18.1 Major Parsing Subjects 576
18.1.1 Unrestricted PS and CS Grammars 576
18.1.2 General Context-Free Parsing 576
18.1.3 LL Parsing 584
18.1.4 LR Parsing 585
18.1.5 Left-Corner Parsing 592
18.1.6 Precedence and Bounded-Right-Context Parsing 593
18.1.7 Finite-State Automata 596
18.1.8 General Books and Papers on Parsing 599
18.2 Advanced Parsing Subjects 601
18.2.1 Generalized Deterministic Parsing 601
18.2.2 Non-Canonical Parsing 605
18.2.3 Substring Parsing 609
18.2.4 Parsing as Intersection 611
18.2.5 Parallel Parsing Techniques 612
18.2.6 Non-Chomsky Systems 614
18.2.7 Error Handling 623
18.2.8 Incremental Parsing 629
18.3 Parsers and Applications 630
18.3.1 Parser Writing 630
18.3.2 Parser-Generating Systems 634
18.3.3 Applications 634
18.3.4 Parsing and Deduction 635
18.3.5 Parsing Issues in Natural Language Handling 636
18.4 Support Material 638
18.4.1 Formal Languages 638
18.4.2 Approximation Techniques 641
18.4.3 Transformations on Grammars 641
18.4.4 Miscellaneous Literature 642

A Hints and Solutions to Selected Problems 645

Author Index 651

Subject Index 655