

Table of contents

Preface	11
1 Introduction	13
1.1 Parsing as a craft	14
1.2 The approach used	14
1.3 Outline of the contents	15
1.4 The annotated bibliography	15
2 Grammars as a generating device	16
2.1 Languages as infinite sets	16
2.1.1 Language	16
2.1.2 Grammars	17
2.1.3 Problems	18
2.1.4 Describing a language through a finite recipe	22
2.2 Formal grammars	24
2.2.1 Generating sentences from a formal grammar	25
2.2.2 The expressive power of formal grammars	27
2.3 The Chomsky hierarchy of grammars and languages	28
2.3.1 Type 1 grammars	28
2.3.2 Type 2 grammars	32
2.3.3 Type 3 grammars	37
2.3.4 Type 4 grammars	40
2.4 VW grammars	41
2.4.1 The human inadequacy of CS and PS grammars	41
2.4.2 VW grammars	42
2.4.3 Infinite symbol sets	45
2.4.4 BNF notation for VW grammars	45
2.4.5 Affix grammars	46
2.5 Actually generating sentences from a grammar	47
2.5.1 The general case	47
2.5.2 The CF case	49
2.6 To shrink or not to shrink	51

2.7	A characterization of the limitations of CF and FS grammars	54
2.7.1	The <i>uvwxy</i> theorem	54
2.7.2	The <i>uvw</i> theorem	56
2.8	Hygiene in grammars	56
2.8.1	Undefined non-terminals	56
2.8.2	Unused non-terminals	57
2.8.3	Non-productive non-terminals	57
2.8.4	Loops	57
2.9	The semantic connection	57
2.9.1	Attribute grammars	58
2.9.2	Transduction grammars	59
2.10	A metaphorical comparison of grammar types	60
3	Introduction to parsing	62
3.1	Various kinds of ambiguity	62
3.2	Linearization of the parse tree	64
3.3	Two ways to parse a sentence	64
3.3.1	Top-down parsing	65
3.3.2	Bottom-up parsing	66
3.3.3	Applicability	67
3.4	Non-deterministic automata	68
3.4.1	Constructing the NDA	69
3.4.2	Constructing the control mechanism	69
3.5	Recognition and parsing for Type 0 to Type 4 grammars	70
3.5.1	Time requirements	70
3.5.2	Type 0 and Type 1 grammars	70
3.5.3	Type 2 grammars	72
3.5.4	Type 3 grammars	73
3.5.5	Type 4 grammars	74
3.6	An overview of parsing methods	74
3.6.1	Directionality	74
3.6.2	Search techniques	75
3.6.3	General directional methods	76
3.6.4	Linear methods	76
3.6.5	Linear top-down and bottom-up methods	78
3.6.6	Almost deterministic methods	79
3.6.7	Left-corner parsing	79
3.6.8	Conclusion	79
4	General non-directional methods	81
4.1	Unger's parsing method	82
4.1.1	Unger's method without ϵ -rules or loops	82
4.1.2	Unger's method with ϵ -rules	85
4.2	The CYK parsing method	88
4.2.1	CYK recognition with general CF grammars	89
4.2.2	CYK recognition with a grammar in Chomsky Normal Form	92
4.2.3	Transforming a CF grammar into Chomsky Normal Form	94

4.2.4	The example revisited	99
4.2.5	CYK parsing with Chomsky Normal Form	99
4.2.6	Undoing the effect of the CNF transformation	101
4.2.7	A short retrospective of CYK	104
4.2.8	Chart parsing	105
5	Regular grammars and finite-state automata	106
5.1	Applications of regular grammars	106
5.1.1	CF parsing	106
5.1.2	Systems with finite memory	107
5.1.3	Pattern searching	108
5.2	Producing from a regular grammar	109
5.3	Parsing with a regular grammar	110
5.3.1	Replacing sets by states	111
5.3.2	Non-standard notation	113
5.3.3	DFA's from regular expressions	114
5.3.4	Fast text search using finite-state automata	116
6	General directional top-down methods	119
6.1	Imitating left-most productions	119
6.2	The pushdown automaton	121
6.3	Breadth-first top-down parsing	125
6.3.1	An example	125
6.3.2	A counterexample: left-recursion	127
6.4	Eliminating left-recursion	128
6.5	Depth-first (backtracking) parsers	130
6.6	Recursive descent	131
6.6.1	A naive approach	133
6.6.2	Exhaustive backtracking recursive descent	136
6.7	Definite Clause grammars	139
7	General bottom-up parsing	144
7.1	Parsing by searching	146
7.1.1	Depth-first (backtracking) parsing	146
7.1.2	Breadth-first (on-line) parsing	147
7.1.3	A combined representation	148
7.1.4	A slightly more realistic example	148
7.2	Top-down restricted breadth-first bottom-up parsing	149
7.2.1	The Earley parser without look-ahead	149
7.2.2	The relation between the Earley and CYK algorithms	155
7.2.3	Ambiguous sentences	156
7.2.4	Handling ϵ -rules	157
7.2.5	Prediction look-ahead	159
7.2.6	Reduction look-ahead	161
8	Deterministic top-down methods	164
8.1	Replacing search by table look-up	165

8.2	LL(1) grammars	168
8.2.1	LL(1) grammars without ϵ -rules	168
8.2.2	LL(1) grammars with ϵ -rules	170
8.2.3	LL(1) versus strong-LL(1)	174
8.2.4	Full LL(1) parsing	175
8.2.5	Solving LL(1) conflicts	178
8.2.6	LL(1) and recursive descent	180
8.3	LL(k) grammars	181
8.4	Extended LL(1) grammars	183
9	Deterministic bottom-up parsing	184
9.1	Simple handle-isolating techniques	185
9.1.1	Fully parenthesized expressions	186
9.2	Precedence parsing	187
9.2.1	Constructing the operator-precedence table	190
9.2.2	Precedence functions	192
9.2.3	Simple-precedence parsing	194
9.2.4	Weak-precedence parsing	196
9.2.5	Extended precedence and mixed-strategy precedence	197
9.2.6	Actually finding the correct right-hand side	198
9.3	Bounded-context parsing	198
9.3.1	Floyd productions	199
9.4	LR methods	200
9.4.1	LR(0)	201
9.4.2	LR(0) grammars	205
9.5	LR(1)	205
9.5.1	LR(1) with ϵ -rules	210
9.5.2	Some properties of LR(k) parsing	211
9.6	LALR(1) parsing	213
9.6.1	Constructing the LALR(1) parsing tables	214
9.6.2	LALR(1) with ϵ -rules	216
9.6.3	Identifying LALR(1) conflicts	217
9.6.4	SLR(1)	218
9.6.5	Conflict resolvers	219
9.7	Further developments of LR methods	219
9.7.1	Elimination of unit rules	220
9.7.2	Regular right part grammars	220
9.7.3	Improved LALR(1) table construction	220
9.7.4	Incremental parsing	221
9.7.5	Incremental parser generation	221
9.7.6	LR-regular	221
9.7.7	Recursive ascent	221
9.8	Tomita's parser	222
9.8.1	Stack duplication	223
9.8.2	Combining equal states	223
9.8.3	Combining equal stack prefixes	226
9.8.4	Discussion	226

9.9	Non-canonical parsers	227
9.10	LR(k) as an ambiguity test	228
10	Error handling	229
10.1	Detection versus recovery versus correction	229
10.2	Parsing techniques and error detection	230
10.2.1	Error detection in non-directional parsing methods	230
10.2.2	Error detection in finite-state automata	231
10.2.3	Error detection in general directional top-down parsers	231
10.2.4	Error detection in general directional bottom-up parsers	232
10.2.5	Error detection in deterministic top-down parsers	232
10.2.6	Error detection in deterministic bottom-up parsers	232
10.3	Recovering from errors	233
10.4	Global error handling	233
10.5	Ad hoc methods	237
10.5.1	Error productions	237
10.5.2	Empty table slots	237
10.5.3	Error tokens	238
10.6	Regional error handling	238
10.6.1	Backward/forward move	238
10.7	Local error handling	240
10.7.1	Panic mode	240
10.7.2	FOLLOW set error recovery	241
10.7.3	Acceptable-sets derived from continuations	241
10.7.4	Insertion-only error correction	244
10.7.5	Locally least-cost error recovery	246
10.8	Suffix parsing	246
11	Comparative survey	249
11.1	Considerations	249
11.2	General parsers	250
11.2.1	Unger	250
11.2.2	Earley	250
11.2.3	Tomita	250
11.2.4	Notes	251
11.3	Linear-time parsers	251
11.3.1	Requirements	251
11.3.2	Strong-LL(1) versus LALR(1)	251
11.3.3	Table size	252
12	A simple general context-free parser	253
12.1	Principles of the parser	253
12.2	The program	258
12.2.1	Handling left recursion	260
12.3	Parsing in polynomial time	260
13	Annotated bibliography	264

13.1	Miscellaneous literature	265
13.2	Unrestricted PS and CS grammars	269
13.3	Van Wijngaarden grammars and affix grammars	271
13.4	General context-free parsers	273
13.5	LL parsing	279
13.6	LR parsing	282
13.7	Left-corner parsing	292
13.8	Precedence and bounded-context parsing	294
13.9	Finite-state automata	299
13.10	Natural language handling	300
13.11	Error handling	302
13.12	Transformations on grammars	310
13.13	General books on parsing	310
13.14	Some books on computer science	312
Author index		313
Index		317