CONTENTS

Preface xv

Part I Fundamentals

Chapter 1 Fundamentals of Computation 1

1.1 Problems and Algorithms 2
1.2 Languages: Syntax and Semantics 5
1.3 Semantic Models 7
1.4 The von Neumann Model of Computation 10
1.5 Objects 13
1.6 Sets 13
1.7 Relations 16
1.8 Functions 19
1.9 Problems 25

Chapter 2 Expressions and Their Notation 27

2.1 Expressions 27
2.2 Function Definitions 33
2.3 Formal Syntax Notation 36
2.4 Problems 40

Chapter 3 Symbolic Expressions and Abstract Programs 41

3.1 Symbolic Expressions 42
3.2 Abstract Programs 52
3.3 Recursion 55
3.4 Abstract Syntax 62
3.5 Problems 65

Part II Function-Based Computing

Chapter 4 Lambda Calculus 67

4.1 Syntax of Lambda Calculus 68
4.2 General Model of Computation 69
4.3 Standard Simplifications 71
4.4 Identifiers 72
4.5 Substitution Rules 74
4.6 Conversion Rules, Reduction, and Normal Order 77
4.7 The Church-Rosser Theorem 80
4.8 Order of Evaluation 81
4.9 Multiple Arguments, Currying, and Naming Functions 83
4.10 Basic Arithmetic in Lambda Calculus 86
4.11 Boolean Operations in Lambda Calculus 88
4.12 Recursion in Lambda Calculus 90
4.13 Problems 92

Chapter 5 A Formal Basis for Abstract Programming 94

5.1 A Basic Syntax 95
5.2 Constants 96
5.3 Function Applications 97
5.4 Conditional Expressions 98
5.5 Let Expressions-Local Definitions 99
5.6 Recursive Definitions 103
5.7 Global Definitions 106
5.8 Higher-Order Functions 106
5.9 An Example-Symbolic Differentiation 108
5.10 Problems 111

Chapter 6 Self-Interpretation 113

6.1 Abstract Interpreters 115
6.2 Lambda Expressions as S-Expressions 119
6.3 An Expanded Interpreter 124
6.4 Association Lists 127
6.5 Closures 133
6.6 Recursive Closures 135
6.7 Closing the Loop—Read-Eval-Print 137
6.8 Problems 138

Chapter 7 The SECD Abstract Machine 140

7.1 List Memory 142
7.2 Basic Data Structures 143
7.3 The SECD Machine Structures 150

7.4 The Basic Instruction Set 153
7.5 Compiling Simple S-expressions 163
7.6 Sample Compilation 168
7.7 The Funarg Problem 171
7.8 Optimizations 173
7.9 Problems 177

Chapter 8 Memory Management for S-expressions 179

8.1 Allocation 180
8.2 Mark-Sweep Collection 183
8.3 Reference Counts 192
8.4 Garbage Compaction 195
8.5 Alternative List Representations 204
8.6 Problems 211

Chapter 9 Demand-Drive Evaluation 213

9.1 Explicit Delays and Forces 215
9.2 Program Examples 217
9.3 Recipes, Promises, Futures 219
9.4 Lazy Evaluation 224
9.5 Streams 229
9.6 Continuations 233
9.7 Problems 241

Chapter 10 LISP: Variations and Implementations 243

10.1 The Original LISP 244
10.2 Scheme—A Pure LISP 257
10.3 Common LISP—A Modern Standard 265
10.4 MultiLISP—A Parallel LISP 270
10.5 The CADR Machine 273
10.6 CADR Derivatives 278
10.7 Other LISP Machines 285
10.8 Benchmarking 291
10.9 Problems 295

Chapter 11 Combinators and Graph Reduction 296

11.1 Basic Combinators 297
11.2 Bracket Abstraction 300
11.3 Additional Combinators 303
11.4 Optimizations 309
11.5 Graph Reduction 312
11.6 Supercombinators 322
11.7 Combinator Machines 326
11.8 The G-Machine 332
11.9 Problems 336

Chapter 12 Other Function-Based Computing Systems 338
12.1 Logo 340
12.2 FP 343
12.3 Hope 351
12.4 Tree Machines 357
12.5 Alice 359
12.6 Rediflow 363
12.7 Function Caching 365
12.8 Problems 368

Part III Logic-Based Computing

Chapter 13 A Logic Overview 369
13.1 Informal Overview 370
13.2 Formal Logic Systems 376
13.3 Propositional Logic 384
13.4 A Simple Inference Engine 387
13.5 A Simple Problem 390
13.6 Problems 393

Chapter 14 Predicate Logic and the First Inference Engine 394
14.1 Basic Syntax 396
14.2 Basic Interpretations 398
14.3 Standard Expression Forms 400
14.4 Horn Clauses 407
14.5 Decidability Issues—The Herbrand Result 410
14.6 The Herbrand Inference Engine 415
14.7 Problems 417

Chapter 15 Fundamentals of Practical Inference Engines 419
15.1 Pattern Matching and Unification 420
15.2 Common Inference Rules 426
15.3 Resolution-Based Inference Rules 428
15.4 Refutation Completeness 431
15.5 Decision Procedures 432
15.6 Strategies—An Overview 438
15.7 Example 443
15.8 Deduction Trees 444
15.9 Problems 446

Chapter 16 The PROLOG Inference Engine 449
16.1 Structure of a PROLOG Program 450
16.2 Procedural Interpretation 455
16.3 A PROLOG-Like Propositional Inference Engine 457

16.4 The PROLOG Inference Engine 463
16.5 Special Features of PROLOG 471
16.6 Some Examples 480
16.7 Problems 483

Chapter 17 The Warren Abstract Machine 486
17.1 Program Structure 488
17.2 Major Data Structures and State Registers 491
17.3 Memory Word Format 494
17.4 Simplified Choice Point 496
17.5 Simplified WAM Instruction Set 497
17.6 PROLOG-to-WAM Compiler Overview 512
17.7 Supporting Built-ins 516
17.8 Detailed Example 520
17.9 Problems 525

Chapter 18 Optimizations and Extensions 526
18.1 An Optimized WAM Summary 527
18.2 Optimizing Single Clauses 527
18.3 Optimizing Multiple Clauses 539
18.4 Other Optimizations 544
18.5 Annotations and Backtrack Control 550
18.6 Problems 562

Chapter 19 PROLOG Implementations 564
19.1 Measuring PROLOG Performance 565
19.2 Compiling to A Conventional Computer 570
19.3 The Original Tick and Warren Machine 578
19.4 Fifth-Generation PROLOG Machines 580
19.5 The PLM and its Derivatives 585
19.6 The Low RISC Approach 589
19.7 Using Associative Memory 596
19.8 Problems 605

Chapter 20 All-Solutions Inference Engine 607
20.1 Database Overview 608
20.2 The Relational Model 614
20.3 Computationally Expensive Operators: Joins 619
20.4 Relational Algebra 623
20.5 Implementations 624
20.6 Production-Rule Systems 633
20.7 The Rete Algorithm 639
20.8 An Abstract Machine 645
20.9 Machines 651
20.10 Problems 655

Chapter 21 Parallel Inference Engines 657

21.1 Taxonomy of Logic Parallelism 658
21.2 Variations of OR Parallelism 665
21.3 Variations of AND Parallelism 676
21.4 AND/OR Process Model 685
21.5 Committed-Choice Languages 694
21.6 Implementations 702
21.7 Problems 706

References 707

Index 721

PREFACE

There is a revolution looming in computer science, not just in the speed and memory capacity of the computing equipment we use, but in the very way we think about, specify, and perform computations. Essentially, the basic von Neumann model of computing, which has ruled unchallenged for the last 40 years, is encountering not one but several potentially heavyweight challengers. This text represents an attempt to provide some insight into the conceptual seeds of the revolution, and the directions in which they are growing. As such, it was written with two overall goals in mind. First, we wish to develop in the reader a firm understanding of the mathematical roots of two trunks of these new computational models. Second, we want to demonstrate in a very concrete fashion how such models can and have been put into practice as real programming languages running on real processors, often with revolutionary design characteristics. Achieving these two goals should give the reader the ability to follow, or better yet participate in, construction of the languages and computing systems that will become the everyday computational tools of tomorrow.

VIEWPOINT

The viewpoint taken in this text is that of a practicing computer architect, where ***architecture*** is taken in its broadest meaning as "the art or science of building,..., esp. habitable structures" (*Webster's Third International Dictionary*). In our case the structures are computing systems, and their inhabitants are the sophisticated, largely nonnumeric programs found today in prototype form in highly interactive environments, artificial intelligence, intelligent databases, and similar advanced applications, but moving rapidly into tomorrow's mainstream.

When building such structures, a true architect must consider the interplay between both the hardware (machine organization) and the software (compilers, interpreters, and runtime routines) needed to sustain the inhabitants. In

Wilensky, Robert, 1984. *LISPcraft,* W. W. Norton, New York.
Winston, Patrick, and Berthold Horn, 1984. *LISP,* Addison-Wesley, Reading, MA.
Wirth, Niklaus, 1977. "What Can We Do about the Unnecessary Diversity of Notation for Syntatic Definitions?" *Communications of the ACM,* Vol. 20, No. 11, Nov., pp. 822–823.
Wolfe, Alexander, 1987. "TI Puts Its LISP Chip into a System for Military AI," *Electronics,* Mar., pp. 95–96.
Wong, Kam-Fai, and M. Howard Williams, 1989. "A Type Driven Hardware Engine for PROLOG Clause Retrieval over a Large Knowledge Base," *Proceedings 16th International Symposium on Computer Architecture,* Jerusalem, May, pp. 211–222.
Woo, N. S., 1985. "A Hardware Unification Unit: Design and Analysis," *12th International Symposium on Computer Architecture,* Boston, June, pp. 198–207.
Wos, Larry, R. Overbeek, E. Lusk, and J. Boyle, 1984. *Automated Reasoning: Introduction and Applications,* Prentice-Hall, Englewood Cliffs, NJ.
Yuhara, Masanobu, Aikara Hattori, Mitsuhiro Kishimoto, and Hiromu Hayashi, 1986. "Evaluation of the FACOM ALPHA LISP Machine," *Proceedings 13th Symposium on Computer Architecture,* Tokyo, Japan, June, pp. 184–190.
Yung, Chao-Chih, 1986. *Relational Databases,* Prentice-Hall, Englewood Cliffs, NJ.
Zloof, M., 1977. "Query-by-Example: A Data Base Language," *IBM Systems Journal,* Vol. 16, No. 4, pp. 324–343.

INDEX

A-list (*see* Association list)
A memory, 279, 281
AA, the instruction, 175
Abelson, 257
Abstract creators, 63
Abstract interpreter, 10, 62, 115
Abstract machine, 2, 8, 140, 271, 297, 332, 340, 486, 645
Abstract predicates, 63
Abstract program, 10, 41, 52, 94
  syntax of, 53, 95, 122
Abstract selector, 63
Abstract syntax, 10, 42, 62, 115
Abstract syntax function, 116
  for s-expressions, 122
Accumulating parameter, 53, 57, 108, 350
Ackerman, William B., 340
Ackerman's function, 57, 327, 340, 356
Action parallelism, 652
Action step in OPS, 639
Actions, 637
Active complement, 618, 624
Active domain, 18, 616
Active range, 18
Actual arguments, 455
ADD, the instruction, 154, 156, 251
Addition, the function, 87, 481, 513
Aggregate operators, 627
Akimoto, Maruo, 287
Algol 68, 9
Algorithm, 4
ALICE machine, 359, 702
All solutions, 665
All-solutions inference engines, 607
Allen, Donald C., 273
Allen, John, 244
Allocate:
  the function, 180
  the instruction, 502, 503, 530, 537
Allocated memory, 179
Allocation process, 180
Alpha conversion, 78
Alphabet, 36
Alternating cycles of garbage collection, 191
ALU (arithmetic logic unit), 81, 279, 579
AMPS machine, 357
Ancestor clause, 428
Ancestor stack, 317
And:
  the function, 89, 386, 404
  the instruction, 53, 101
AND node:
  in a deduction tree, 663
  in OPS, 643
AND/OR parallelism, 657, 662, 676, 702
AND/OR process, 686, 691
AND/OR Process Model, 657, 685
AND/OR trees, 662
Annotation, 526, 550, 681
Anonymous function, 54
Anonymous variable, 471, 532

Antecedent, 386, 407
Antecedent reasoning, 392
Antisymmetric relation, 18
Any node, 645
AP, the instruction, 238, 334
APF, the instruction, 222
APL language, 31, 35
AP0, the instruction, 217, 220, 239
Appel, Andrew W., 292, 340
Append, the function, 50, 58, 346, 472, 520, 526, 529, 535, 567, 570, 590, 594
Application, 22, 68
Application, the function, 345
Application Language Processor, 609
Applicative caching, 366
Applicative-order reduction, 82, 117, 249, 317
Apply, the function, 115, 117, 125, 131, 318
APVAL, the LISP property, 248
Areas, in memory, 276, 284
Argonne model of OR parallelism, 670
Argument, 15, 20
Argument copying, 582
Argument registers, 487, 490, 541
Arithmetic, 87, 475, 481
  syntax for expressions, 109
Arithmetic logic unit, 81, 279, 579
Arity, 20, 494
Array, 246
Arvind, 340
Assert, the instruction, 480, 519, 597, 603
Assertion, 407
Assignment, 12, 248
Assignment relation, 702
Assignment statement, 6, 12, 628
Assoc, the function, 126
Association list, 114, 127, 247, 366, 466
Associative function, 22, 349
Associative memory, 209, 596, 653, 702
Astrahan, M. M., 626
Atom, 38, 52, 245, 345, 384, 397, 450
Atom set, 413
Atomic constant, 345
Atomic data type, 245
Atomic node, 38
Atomic symbol, 245, 397
Attribute, 609, 611, 614
Attribute domain, 611
Attribute list, 627
Attribute name, 615
Attribute-value pair, 617, 635
Augustsson, Lennart, 340
Axiom, 378
Axiom schema, 387
Axiomatic semantics, 8, 9, 141
B combinator, 299
B register, 240, 493
Back up, 674
Backtrack:
  in functional languages, 240
  the instruction, 491, 514, 530, 536
  in logic languages, 440, 456, 524
  optimization, 550, 554, 598
Backtrack register, 240
Backtrack Table, 559
Backtrack trail, 497
Backus, John, 339, 343, 344, 351
Backus Naur form, 27, 36–40, 95
  expression, 38
  program, 38
  (*See also* Syntax)
Backward chaining, 392, 434, 437
Backward execution, 693
Baer, Jean-Loup, 274
Bag-of, the function, 478, 626
Bailey, Roger, 351
Baker, Henry G., 195
Baker's algorithm, 180, 196, 276
Barklund, Jonas, 548
Bartley, David H., 265
Basic composition combinator, 304
Basic type, 550
Basis case, 55, 90
Basis test, 55, 90
Bath Concurrent LISP Machine, 339
Bavel, Zamir, 13, 36, 55
BB register, 497, 548
BCE register, 497
BCP register, 497
Begin, the instruction, 258
Begin-end block, 254
Ben-ari, Mordechai, 190
Benchmarks, 291, 293, 567, 701
Benker, H., 565, 571
Berkling, Klaus, 301, 339
Beta conversion, 79
BH register, 497, 529
BIBOP (Big Bag Of Pages), 207, 265
Big Bag Of Pages, 207, 265
Big-num, 246
Bijective function, 21
Binary relation, 17
Binary resolution, 428, 430, 431, 443
Binary tree, 45
Binding, 2, 149, 395, 596
Binding array, 597, 702
Binding depth, 597
Binding stack, 284
Binding variable, 72, 398
Blasgen, M. W., 626
Block definitions, 101

Blocked literal, 692
Bloss, Adrienne, 224
BNF (*see* Backus Naur form)
Bobrow, D., 171, 173
Body:
  of a guarded clause, 697
  of a rule, 453
Boolean constants, 354
Boolean functions, 88
Bootstrap, 114
Borriello, Gaetano, 585, 591
Bottom element, 345
Bottom preserving, 345
Bound instance, 73
Bound variable, 2, 493
Bounded depth first, 442
Bowen, K. A., 570
Boyer-Moore theorem prover, 273
Bracket abstraction, 296, 300
Breadth first search, 441, 444
British Museum search, 380
British Nationality Act, 450
Broadcast, 596, 668
Brooks, Rodney, 203
Browning, S. A., 357
Brownston, Lee, 633
Bruynooghe, M., 556
BTR (backtrack trail) register, 497
Built-in functions, 120, 151, 165, 250, 298, 474
  in PROLOG, 516
Burge, W. H., 55, 68, 142, 215, 229, 298, 303, 339
Burstall, R. M., 351
Burton, F. W., 339
Busy-wait loop, 224
BUTFIRST, 341
  (*See also* Cdr)
Butterfly LISP, 273
Butterfly multiprocessor, 273
Bytecode interpreter, 571

C register, 152
Cache functional, 367
Cactus stack, 173, 247
CADR machine, 273
Caf (constant applicative form), 297, 345
Calculus of logic, 376
Call, the instruction, 278, 502, 513, 538
Call-by-name, 99, 224, 551
Call-by-need, 215, 224
Call-by-reference, 551
Call-by-result, 551
Call-by-value, 224, 551
Call/cc, the instruction, 234, 263
Call-return optimization, 537
Call-return sequence, 153
Calling pattern, 553
Caltech Tree Machine, 357
Campbell, J. A., 457
Cancel message, 688, 691
Car:
  as a combinator expression, 308
  the function, 46, 308, 346
  the instruction, 154, 156, 200, 209, 298
Car coding, 204, 550
Car field, 143, 494
Car field packing, 274
Carlton, M., 702
Carmel-2 machine, 702
Cartesian product, 15, 621
Catch, the instruction, 268, 341, 343, 583
CC, the instruction, 238
CCP field, 536
Cdr:
  as a combinator expression, 308
  the function, 46
  the instruction, 154, 156, 308
Cdr coding, 204, 274, 586
Cdr field, 143, 494
CE field, 536
Cells, 45, 48, 142, 494
Cellular Tree Machine, 357, 652
Central processing unit, 11
Chang, Chin-Liang, 376, 395, 400, 411, 420, 426, 432
Chang, Jung-Herng, 559, 561, 682
Chang, Si-en, 750
Character, 5, 36
Character string, 354
Characteristics, 13
Chare Kernal, 705
Checker goal, 681
CHI (Cooperative High-Speed Inference Machine, 582
Chiang, Y. P., 705
Chikayama, Takashi, 702, 703
Children nodes, 43
Choice point, 457, 461, 493, 496, 550
Choice point stack, 589, 646
  (*See also* Stack)
Choice point table, 560
Chunks, 533
Church, Alonso, 68
CHURCH langauge, 331
Church-Rosser theorem, 80
  lemma, 81
Ciepielewski, Andrezej, 675
Circular lists, 252
CISC (Complex Instruction Set Computer), 589
Clark, Douglas W., 195, 205

Clark, Keith L., 700
Clarke, T. J., 326
Class of an OPS object, 635
Class hierarchy, 267
Clausal form, 400, 406
Clause, 400, 406, 450
Clause, the instruction, 480, 519
Clause code for the Warren Abstract Machine, 489
Clause filtering, 603
Clause optimization, 535
Cleanup bits, 283
Clinger, William, 257
Clocksin, W. F., 376, 450
Closed deduction tree, 663
Closed function, 22
Closely coupled multiprocessors, 661
Closure, 114, 133, 149, 215, 277, 694
Codd, E., 614
Cohen, Jacques, 180, 191, 195, 449
Cohen, Paul, 369
Collector process, 190
Colmerauer, Alain, 449
Colomb, R. M., 605
Color, the function, 184
Colors, 184
Combinator, 23, 91, 298, 305
  mutually recursive, 105
Combinator compiler, 302
Combinator theory, 34, 296
Commit, the instruction, 665, 695, 697
Committed-choice, nondeterministic languages, 657, 665, 694
Common LISP, 244, 266
Commutative function, 22, 386
Compact LISP Machine, 280
Compacting garbage collectors, 196
Compare-and-swap, the instruction, 271
Compile, 163, 487
Compile, the instruction, 256
Compiler, 115, 141, 163, 265, 302, 609
  for combinators, 303
  for SECD machine, 163
  for the WAM, 512, 570
Complement, the function, 618
Complete logic theory, 379, 426, 431, 697
Complex Instruction Set Computers, 589
Components, 15, 609
Composable function, 22
Composite data structure, 249
Composite expression, 28
Composition, 22
  via combinator, 305
  as a function, 22, 35, 108
  as an instruction, 108, 347
Computable expression, 67
Computation, 3
Computer, 6
Concat, 51
  (*See also* Concatenate)
Concatenate, the function, 51, 473, 519, 567
  (*See also* Append)
Concatenation of strings, 37
Concert, multiprocessor, 271
Conclusion, 379, 407
Concrete syntax, 119
Concurrent PROLOG, 700
Cond:
  the instruction, 123
  the special form, 254
Condemned memory, 196
Condition, 386, 407, 637
Condition box, 629
Condition codes, 592
Conditional bindings, 669
Conditional execution, 1
Conditional expressions, 88, 98, 123
Conditional form, 123, 165, 345, 347
Conery, John S., 685
Conflict resolution, 639
  parallelism in, 652
Conflict set, 638
Conjunctive normal form, 389, 400
Connectives, 378
Cons:
  cell in the SECD, 143
  cell in the WAM, 495
  as a combinator expression, 308
  as a complexity measure, 59
  the function, 50, 308, 360
  implementation, 192
  the instruction, 154, 156, 192, 194, 200, 206, 472
CONS machine, 273
Consequence:
  of a Horn clause, 407
  of an implication, 386
  of a wff, 380
Consequent reasoning, 392
Consistency checks, 677
Consistent logic theory, 380, 382, 434
Constant, 28, 95, 96, 120, 297, 378, 395, 396
Constant applicative form, 297, 345
Constant combinator, 299
Constant relation, 621
Constant tag, 494
Constructor functions, 345, 347, 355, 359, 360
Consumer, 229, 558
Content addressable memory (*see* Associative memory)
Context, 129, 134

Continuation, 214, 233, 263, 457, 492, 497
Continuation Pointer, 492
Contradiction, 380, 383, 384
Control list, 150
Control predicates, 476
Control stack, 282
Co-routines, 236, 457
Coupled variable, 551
Cox, P. T., 556
CP (continuation pointer) register, 187, 492
Cpi (cycles per inference or cycles per instruction), 292, 570
CPU (central processing unit), 11
Crammond, James, 665
Create-cons function, 199
Create-goal, the instruction, 703
Create-zzz, 63
Creation region, 201
Creator, the function, 64
Critical cycles of garbage collection, 191
Cross-product effect, 652
CRT (Church-Rosser theorem), 80, 81
CSET, the instruction, 251
Current instruction buffer, 580
Curried function, 2, 24, 67, 298, 313
Curried graph, 312
Curry, Haskell B., 24, 297, 307
CURRY chip, 329
Currying a function, 24, 83, 127
Cut, 477, 479, 517, 547, 669
Cut storage optimization, 547
Cycles per inference, 570
Cycles per instruction, 292

D register, 153, 287
DADO machine, 652
Dangling pointer, 202
Dangling reference, 546
DAP, the instruction, 174
Darlington, John, 359
Data dependency analysis, 681
Data dependency graph, 557
Data Description Processor, 609
Data Description Schemas, 609
Data stack, 284
Data type, 352, 354
Database, 608
Database Manager, 609
Dataflow AND parallelism, 680
Dataflow computing, 340, 363, 640
Datalog, 632
Date, C. J., 608, 614
Davis, A. L., 339
Davis, M., 416
Deallocate, 538, 548
Debray, Saumya K., 534, 553
Decdr, the function, 586
Decidable, 380, 410
Decision method, 410
Decision problem, 410
Decision procedure, 372, 380, 391, 410, 419, 432, 437
Declarative language, 5
Declarative sentence, 384
Declare, the instruction, 278
Deduction, 380
Deduction theorem, 384, 415, 434, 436, 454
Deduction tree, 444, 469
Deep backtrack, 457, 460, 491, 497
Deep binding, 247
Deep guards, 700
Define, the instruction, 261, 278, 343
Defstruct, the instruction, 266
Defvar, the instruction, 268
Degree of a schema, 616
DeGroot, Douglas, 682
Delay:
  the function, 215
  the instruction, 260, 270
Delayed branch, 289, 594
Delayed evaluation, 215
Delayed lists, 219
Delete, the function, 609, 617
Delta machine, 631
Demand-driven evaluation, 213, 340
Demodulation, 430, 442
Demodulator, 430
Dennis, J. B., 340
Denotational semantics, 8, 10, 62, 140
Depth first search, 440, 444
Dereference, 504, 596
Descendant list, 689
Designated key, 617
Despain, Alvin M., 357, 527, 559, 561, 568, 702
Determinate append, 567
Determinate concatenate, 473, 521
Deterministic clauses, 540, 552, 553
Deterministic computation, 695
Deterministic problem, 2
Deutsch-Schorr-Waite algorithm, 187
Dhrystone program, 291
Dietrich, S. W., 633
Dif, the function, 109
Difference:
  the function, 618, 624
  over sets, 15
Difference list, 51
Dijkstra, E. W., 190
Diller, Antoni, 297, 307, 322
Direct consequence, 379
Direction tag, 187

Disjoint, 15
Disjunction, 406
(*See also* Or)
Disjunction, the instruction, 477, 518
Disjunctive normal form, 400
Dispatch RAM (random-access memory), 280
Dispatch register, 279
Display, 171
Distributed application combinator, 300
Distributed processors, 661
Distribution System, 362
Divide, the function, 622, 624
Do loop, 176, 245, 305
Dobry, T. P., 486, 520, 527, 545, 548, 550, 568, 570, 571, 585, 587
Domain, 18, 20, 381, 611, 614
Domain set, 611
Domain value, 18
Don't care value, 210, 601
Dot notation, 44
Douglass, Robert J., 658
DUM, the instruction, 160
Dump, 150, 153, 333
Dwork, Cynthia, 661
Dybvig, R. Kent, 257
Dynamic area, 285
Dynamic binding, 343
Dynamically scoped variables, 247

E register, 151, 287, 493
Eager beaver evaluation, 214, 359
Effectively computable, 55
Eisenbach, Susan, 351, 359
Elemental cancellator, 299
Elemental compositor, 300
Elements, 13
Empty clause, 407, 434
Empty tree, 43
Empty set, 14
Emptyp, the instruction, 341
End, the instruction, 342
Engine, 263
Enqueue-goal, the instruction, 703
Entity, 610
(*See also* Object)
Entity-Relationship Model, 610
Entity set, 610
Entry, the function, 283
Entry table, 202
Environment, 129, 134, 150, 333, 466, 490, 497
Environment initialization, 530
Environment register in the Warren Abstract Machine, 493
Environment-stacking inference engine, 465
Environment trimming, 544, 586
EP (evacuation pointer) register, 197
Ephemeral garbage collection, 203, 284
Eq, the function, 52
Eqlist, the function, 228
Equal, the function, 15, 58
Equal leaves function, 228
Equality, 397, 430, 475
Equation statements, 353
Equational form, 33, 94
Equijoin, the function, 621, 629
Equivalence classes, 19
Equivalence relation, 19, 386
Equivalent wffs (well-formed formulas), 382
Escape, the instruction, 516, 587
Eta conversion, 79
Evacuate, the function, 199
Evacuation area, 197
Evacuation pointer, 197
Evacuation process, 196–199
Evacuation region, 201
Eval, 34
the function, 63, 113, 115, 117, 124, 131, 137
the instruction, 256, 269
Eval-when, the instruction, 269
Evalquote, the instruction, 256
Evaluating an expression, 28, 65
Evaluating a wff (well-formed formula), 381
Evaluation in parallel, 362, 399
Evaluation functions, 8
Evaluation order, 81
Exception table, 209
Exclusive or, 93, 386
Execute, the instruction, 538
Executing an algorithm, 4
Executing a logic program, 455
Exhaustive search, 380
Existential quantifier, 398
Expand, the instruction, 259
Expand-syntax, the instruction, 259
Expansion expression, 260
Explanation, 373
Export binding, 704
Expression, 27, 34, 39, 52, 67
Extension table, 633
Extent, 268
External value cell, 277

F register, 143, 150, 180, 205, 287
FA (failure address), 497
FACOM ALPHA, the machine, 287, 582
Fact, 370, 371, 390, 450, 452, 538
Fact in OPS (*see* Working memory elements)
Factor, the function, 622

Factorial function, 56, 58, 91, 92, 122, 233, 307, 311, 335, 348, 350, 376, 476, 481
Factoring, 430, 442
Fagin, Barry S., 702
Fail:
the function, 239
the instruction, 240, 476, 516, 530
the message, 688, 690
Fail sequence in the Warren Abstract Machine, 501, 510, 537
Failure, 455
in unification, 421
Failure address, 497
FAIM-1, the machine, 339
False, 88
Falsify, 383
Falsity, 371
Favored alternative, 667
Favored binding, 667
Favored processor, 673, 675
FCP (Flat Concurrent PROLOG), 700, 702
FEL, the language, 339, 364
Fexpr, the LISP property, 248, 256
Feys, R., 297
FFP Machine, 357
FGHC (flat guarded Horn clause), 700, 702
FGL, the language, 339
Fibonnachi function, 57, 365
Field, 609, 611
Fifth Generation Computer Project, 580, 631, 702
File Manager, 609
Files, 609
Filter, the function, 232
Fine-grain parallelism, 659
First, the function, 46, 219, 341
(*See also* Car)
First-class continuation, 263
First-order predicate calculus, 395
syntax of, 396
Fitch, J. P., 339
Fixed-point combinator, 91, 103, 133
Fixed points, 22
Flat Concurrent PROLOG, 700, 702
Flat guarded Horn clause, 700, 702
Flat guarded languages, 700
Flatten, the function, 228, 288, 356
Flip, the function, 199
Flipping memory, 196–199
Flop, 565
Fluid environment, 262
Fluid-let, the instruction, 262
Fluid variables, 262, 268
For all (*see* Universal quantifier)
Force, the function, 215, 221, 227, 262, 367
Foreign key, 617
Forgy, Charles L., 633, 639, 649, 651
Fork, 223
Formal argument, 33, 69, 455
Formal model of logic, 376
Formal parameters, 456
Formal statement, 3
Forth, 130
FORTRAN, 31, 243, 344
Forward chaining, 392, 436, 437, 608, 634
Forward execution, 692
Forward pointer, 202
(*See also* Invisible pointer)
Forwarding reference, 591
Foster, Ian, 700
FP:
the instruction, 343
the language, 345
FP (frame pointer) register, 283
Fput, the instruction, 341
(*See also* Cons)
Frame, 56, 274, 282, 457, 496
Franz LISP, 244
Free bit, 599
Free expressions, 323, 324
Free instance, 73
Free list, 183, 190
Free memory, 179
Free occurrences, 398
Free variables, 2, 173, 298, 374, 400
Friedman, Daniel, 257
Fromspace, 196
Fuchi, K., 700
Fun, the instruction, 333
Funarg problem, 132, 141, 171, 245, 248, 277
Function:
the instruction, 255
mathematically, 19, 378, 395, 405
Function application, 19, 97
Function-based computing, 1
Function caching, 365, 633
Function definitions, 33, 95, 352
Function evaluation, 19
Function objects, 255
Function theory, 1
Functional forms, 344, 345
Functor, 396
Future, 214, 219, 224, 262, 271, 272
Future instruction buffer, 580

G-Machine, 8, 297, 332
Gabriel, Richard P., 244, 291
Gamma optimization, 326
Garbage, 146, 179, 183
Garbage collection, 146, 179, 183, 276, 328, 361, 543, 548

Garbage compaction, 195
Garbage cut, 549
Garey, Michael S., 2
Gather state in AND/OR process model, 687
Gee, J., 565, 571
Gellert, W., 13
General registers, 487
Generalization, 394
Generate and test, 239
Generating rule, 55
Generation number, 201, 290
Generator, 558
Generator goals, 681
Genius-level backtracker, 557
George, Lal, 336
Georgeff, M., 171
Get-constant, the instruction, 507
Get function, 63
Get instructions, 498, 503, 515
Get-list, the instruction, 507
Get-next-free cell, 181
Get-structure, the instruction, 507, 549
Get-val, the instruction, 531
Get-var, the instruction, 531
Get-zzz, the instructions, 63
Getv, the instruction, 503, 508
Gibson, Jack C., 291, 565
Gibson mix, 291, 565
Gilmore, P. C., 415
Ginosar, Ran, 702
Global definitions, 106
Global stack, 581
Global variables, 247, 261, 341, 343
Globalize variables, 546
GMD machine, 339
Go, the instruction, 254, 343
Goal, 373, 450
Goal clause, 407
Goal-list, 458
Goal records, 703
Goal stacking model, 459, 460
Goldberg, Adele, 9
Gordon, Michael, 7, 10
Gostelow, Kim P., 340
Goto, A., 702
Goto, the instruction, 254
Grammar, 5, 37
Grandparent problem, 446, 453
Graph preorder traversal, 470
Graph reduction, 296, 312, 359
Graph traversal algorithm, 188, 319, 329
Green, C. C., 205
Greenblatt, Richard D., 273
Gregory, Steve, 700, 702
Griss, Martin, 244
Ground, the function, 683
Ground atom, 414
Ground expression, 28
Ground instances, 415
Ground literal, 414
Ground term, 413, 551, 681
Guard, 665, 697
Guard predicate, 339
Guarded Horn clause, 658, 696
Gupta, Anoop, 649, 651
Gurd, John, 340
Guttag, John, 345

H register, 493
Halstead, Robert H., 244, 270
Haridi, Seif, 668
Harrison, Peter G., 173, 343
Harsat, Arie, 702
Hash node, 645
Hash table, 541, 625, 670
Hashing function, 625
Hausman, Bogumil, 675
Haynes, Christopher T., 233, 262
HB register, 529
Head, the instruction, 46
(*See also* Car)
Head of a rule, 453
Head order reduction, 301
Heap, 333
in the WAM, 493, 541
Hemispaces, 196
Henderson, Peter, 42, 52, 68, 115, 142, 215, 221, 224, 228, 229, 244
Hennessy, John L., 208, 292, 293, 565
Herbrand base, 413
Herbrand inference engine, 415
Herbrand interpretation, 411, 413
Herbrand universe, 411, 624
Herbrand's Theorem, 415
Hermenegildo, Manuel V., 676, 705
Hewitt, Carl, 195
Hickey, Tim, 191, 195
Hierarchical Model, 613
Higher-order functions, 106
Higher-order logics, 395
Higher-order types, 355
Hindley, J. R., 23, 86, 297
Hole, the instruction, 333
Hook tag, 702
HOPE, 339, 351
Horn clause, 407, 444
guarded, 658, 696
Horowitz, Ellis, 7, 36
HPM (High-Speed PROLOG Machine), 582
Hudak, Paul, 340
Hughes, John, 300, 322
Huynh, Tien, 345

Hypercube machine, 668, 705
Hyperresolution, 429
Hypothesis, 373, 378, 380

I-allocate, the instruction, 561
I-call, the instruction, 561
I combinator, 299, 346
I-state (Instantiation state), 554
I-structures, 340
I-unit, 580
Ianucci, Robert A., 340
IBM System/370, 572
IC-PROLOG, 700
Ichiyoshi, N., 702
ICOT (Institute for New Generation Computer Technology), 581
Id, the instruction, 340, 346
Identifier, 72, 95, 374, 378, 396
in PROLOG, 451
Identity element, 22, 349
Identity function, 18, 299
Idle state in AND/OR process model, 687
If, the instruction, 256
If expression, 341
If-then-else form, 1, 52, 98, 123, 351
If-then rule, 371, 386, 390, 407
Iffalse, the instruction, 342
Iftrue, the instruction, 342
Imperative variable, 72
Implicit keys, 617
Implies, the function, 386
Import bindings, 704
Inclusive or, 381
Incomplete logic theory, 379
Inconsistent logic theory, 380, 383
Independent, the function, 683
Independent variable, 551
Index, the function, 163, 166, 324
Indexing, 540, 603, 639
Indexing instructions, 497, 500, 540, 625
Indirect address, 197
Inference, 372, 379, 383, 419, 566
Inference engine, 370, 373, 381, 387, 415, 419, 449, 460, 564
Inference rule, 372, 378, 419, 426
Inferencing, 370, 379
Inferring, 372
Infinite expressions, 218
Infix, 29, 375
Infix notation, 18, 30
Injective function, 21
Input/output operation, 269, 476
Input substitutions, 3, 551
Insert, the function, 609, 617
Instance, 2, 72, 638
Instantiated variable, 3, 474, 493
Instantiation state, 554
Instruction, 11
Instruction counter, 152
Instruction measure, 565
Instruction mixes, 291
Instruction set architecture (ISA), 141
for the G-Machine, 334
for the LOW RISC, 593
for the PRM, 647
for the SECD machine, 153
for the SPUR, 290
for the WAM, 528
Instructions per second, 12
Int, the instruction, 333
Integers, the function, 218, 233
Integrated PROLOG Processor, 565
Intelligent backtracker, 556
Interconnection Network, 362
InterLISP, 244
Internal pressure, 364
Interpretation, 381, 382, 398, 624
Interpretative semantics, 8, 36, 69, 140
Interpreter, 8, 113, 125, 130
for PROLOG, 581
Intersection, the function, 15, 58, 618, 624
Intranode parallelism, 652
Invalid logic theory, 382
Inverse relation, 18
Invisible pointer, 197, 322, 495
Invoke, the instruction, 364
Is, the instruction, 475, 533
Is-a, the function, 17
Is-a-zzz, the function, 63
Is-part-of, the function, 17
ISA (*see* Instruction set architecture)
Ishida, Toru, 652
Isomorphism, 21
ISWIM, 339
Iteration, 57
Ivory processor, 281

Jamsek, Damir, 340
Janssens, Gerta, 534
Jayasouriahn, 605
Jensen, John C., 265
Johnson, David S., 2
Johnsson, Thomas, 332, 336
Join:
the function, 608, 619, 634, 678
the instruction, 156, 166, 224
Join-based AND parallelism, 677
Jones, Neil, 297

K combinator, 299, 347
Kale, Laxmukant, 704
Kamiya, Shigeo, 631

Kanada, Yasusi, 702
Katevenis, M., 591
Keller, Robert M., 272, 339, 357, 363, 365
Kennaway, J. R., 340
Kernel Language 1, 700, 702
Key, 611, 613, 616
Keyword, 267
Keyword arguments, 267
Khoshnevisan, Hessam, 343
Kieburtz, Richard B., 332, 336, 351
Kimura, Yasunori, 702
Kip (kilo instructions per second), 565
Kitsuregawa, Masaru, 624
KL-0 (Kernel Language 0), 581
KL-1 (Kernel Language 1), 700, 702
Kleene closure, 37
Kluge, Werner E., 339
Knapsack problem, 3
Knowledge Crunching Machine, 565
Kogge, Peter M., 151, 265, 280, 340, 565, 578, 596, 653
Kowalski, Robert, 369, 455
KRC, the language, 339
Kumar, Vipin, 559
Kurosawa, K., 565, 571

Label:
  of an expression, 325
  the instruction, 343
  in LISP, 254
Lam, M., 702
Lambda:
  the instruction, 258, 261
  the symbol, 68
Lambda calculus, 34, 67
  substitution rules for, 76
  syntax of, 68
Lambda expressions, 166, 119
Lambda LISP processors, 279
Landin, P. J., 68, 114, 141
Language, 5, 36, 37
Large-grain parallelism, 660
Last, 46
  (*See also* Cdr)
Last-call optimizations, 539
Last-clause optimization, 542
Lazy evaluation, 214, 224, 320, 328, 335, 351, 359
Ld, the instruction, 155
Ldc, the instruction, 155
Ldf, the instruction, 167
Leaf node, 42
Leaf processor, 357
Least committment, 422
Lee, Richard C., 376, 395, 400, 411, 420, 426, 432
Left, 46
  (*See also* Car)
Left ancestor stack, 317, 327
Left recursion, 633
Leftmost reduction, 81
Length, the function, 50, 58
Let, the instruction, 258, 260
Let expression, 52, 99, 121, 166
Letrec, the instruction, 54
Letrec expression, 104, 121, 166
Level, 285, 325, 358
Lewis, P. M., 36
Lfu, the instruction, 222, 227
Li, Deyi, 625
Lieberman, Henry, 195
Lifetime-based scavenging, 202
Lin, F. C., 272, 363
Lin, Yow-Jian, 559
Linpack Loops, 565
Linpack subroutines, 292
Lips (logical inferences per second), 566, 569
LISP, 243, 340, 360, 589
  common, 244, 266
LISP compiler, 256
LISP 1.5, 244
LISP Machine LISP, 244
List, 1, 41, 44, 50
  PROLOG, 472
List compaction, 204
List memory, 142, 495
Listener, 138, 229, 235
Literal, 246, 397, 400, 450
Livermore Loops, 292, 565
Load register, the instruction, 510
Local, the instruction, 343
Local definition, 99
Local function, 100
Locate, the function, 148, 155
Location in memory, 10
Locative pointer, 276
  (*See also* Invisible pointer)
Lock, 272
Lock-out, 676
Logic, 369, 372
Logic-based computing, 1
Logic computing, 369, 373
Logic expressions, 395
Logic programs, 372
Logic variables, 371, 373, 374, 378, 419, 450, 455
Logical connectives, 88, 378, 384
Logical consequence, 379, 383
Logical expressions, 88, 377
Logical inferences, 569
  per second, 566, 569

LOGLISP, the language, 340
Logo, the language, 340
Lookup, the function, 126
Loop unrolling, 577
LOW RISC (Reduced Instruction Set Computer) Machine, 589
Lucas, P., 9
Lukasiewicz, Jan, 31

M memory, 279, 281
MA, the instruction, 176
McCarthy, John, 42, 62, 244
McDermott, J., 633
MacLISP, 244, 273
Macro, 256, 259
  as a special form, 257
Macro substitution, 54, 99
Macrocode, 277
Macroexpander, 8, 487, 571
Mago, Gyula, 357
Maier, David, 614
Make, the instruction, 343, 638
Make-engine, the instruction, 263
Many-to-many relationships, 611
Many-to-one relationships, 611
Map, the function, 107, 352, 665
Map cache, 580
Map coloring problem, 558
Map2, the function, 107
Mark:
  as a bit, 182, 184, 286
  the function, 184
  in garbage collection, 182
  the instruction, 500, 513, 529, 535, 549
Mark-sweep garbage collection, 180, 183, 286, 549
  complexity of, 186
  multicolor marking, 189
  parallel, 190
  the process of marking, 184
Marti, J., 339
Mask, 209
Match, 596
Match line, 209
Match parallelism, 651
Match step, 638
Matching, 34
Matthews, Gene, 280
Maximally free expressions, 322, 325
Mayfly machine, 339
Mcode, the instruction, 271
Meaning, 7, 381
Medium-grain parallelism, 659
Mellish, C. S., 376, 450, 550
Member, the function, 52, 58, 473
Membership, 14
Memo function, 366, 633
Memory:
  mapping, 572
  in the SECD, 142
  as a unit, 11
  in the WAM, 494, 530
Memory cells, states of usage, 180
Mendelson, Elliot, 13, 376, 384, 395, 400, 402
Merge, the function, 232, 624, 697
Metainterpreters, 114
Metalanguages, 114
Metasymbol, 37
Mgu (most general unifier), 422
Microprogramming, 277, 358
Microscopic parallelism, 659
Mierowsky, C., 700
Millions of instructions per second, 12, 291, 565
Millroth, Hakam, 548
Mills, J. W., 571, 589, 594, 596
Milner, Robin, 340
Minimal substitution, 422
Mips (*see* Millions of instructions per second)
MIPS RISC (Reduced Instruction Set Computers) machine, 293
Miranker, Daniel, 652
Mitchell, John C., 340
Mix, 565
  for RETE nets, 650
  for the WAM, 568
Mixed clause, 402
ML, the language, 339
Mode, 551, 681
Mode analysis, 553, 575
Mode annotations, 552
Mode declarations, 552
Mode flag, 493, 507, 510, 549
Model of computation, 10, 69
Model interpretation, 382
Modify, the function, 617
Modulator, 430
Module, 277
Modus ponens, 379, 388, 389, 421, 427, 634
Modus tollens, 389, 427
Monkey and Bananas problem, 638
Monotonic logic, 608
Moon, David A., 281, 284
Morioka, M., 568
Morris, James, 215, 228
Most general unifier, 422
Muchnick, Steven, 297
Mulder, Hans, 565
Multi-PSI machine, 702
Multicolor marking, 189
MultiLISP, 244, 270

Multiple arguments, 120, 132, 352
Multiple results, 267, 469
Multiple simultaneous substitution, 84
Multiplication, the function, 87, 481
Multisets, 619
Mutator process, 190
Mutually recursive functions, 105

N-place relation, 16
N register, 545
Naganuma, Jiro, 599
Naish, Lee, 476
Naive backtracking, 555
Naive model of OR parallelism, 668
Nakajima, K., 571, 581
Nakashima, H., 571, 581
Nakazaki, Ryosei, 571, 582
Name of an object, 13
Name conflict, 78
Name list, 129
Natural join, 620, 624
Navigation, 614
Nconc, the instruction, 252
Neches, Philip, 629
Negation, 372
Negation by failure, 476
Negative clause, 401
Negative hyperresolution, 429
Negative literal, 401
Nested definitions, 101, 134, 168
Nesting level, 358
Network model, 613
New-id, the function, 118
Newton, Michael O., 570–572
Nil, 267
  the instruction, 154
  the object, 46
  the pointer, 143
Nilsson, Nils J., 420
Niwa, Masashi, 287
Node, 42
Node parallelism, 652
Node sharing, 645
Nonapplicative caching, 366
Nondeterminate append, 567
Nondeterministic computation, 374, 665, 695, 697
Nondeterministic problem, 2, 239, 473
Nonground expressions, 28
Nonmonotonic logic, 608
Nonrelational database models, 612
Nonterminal, the object, 37
Nonterminal cell, 143
Nonterminal node, 43
Nonunit clause, 402
Normal order, 80
Normal-order reduction, 82, 117, 317
Not:
  the function, 80, 89, 476
  the instruction, 519
  a node in Rete nets, 644
Notation, 5
Nubus, 279
Null, 52
Null string, 36

Object, 1, 13, 381, 412, 610
Object-level functional programming, 344
Object oriented, 267
Occurs bound, 74
Occurs check, 424
Occurs free, 73
On-the-fly collectors, 190
One-input node, 641
One-place relation, 17
One-to-one property of functions, 21
One-to-one relationship, 611
Onto, 21
Opcode, 144
Open coding, 572, 591, 645
Operation, 565
Operational semantics, 8
OPS, 633
OPS inference engine, 638
Optimization:
  of combinators, 309
  in FP programs, 349
  of Rete nets, 645
  of SECD programs, 173
  in the WAM, 526
Or:
  the function, 89, 90, 381, 388, 478
  the instruction, 250
OR node in a deduction tree, 664
OR parallelism, 608, 657, 662, 665
OR process, 686, 688
Ordered n-tuples, 15
Ordered pairs, 15
Ordered-tree-search Parlog program, 700
Ordering strategies, 440
Output, the instruction, 342
Output mode, 681
Output substitutions, 3, 551, 553
Owner, 614

P-code, 131
P-list (*see* Property list)
Packages, 341
Packet, 359
Packet pool, 359
Packet Pool Segment, 362
Pages of memory, 201

Pairs, 15
Papert, Seymour, 340
Parallel garbage collector, 190
Parallel inference engines, 657
Parallel Inference Machine, 702
Parallel machines, 339, 357, 359, 702
Parallelism, 190, 223, 270, 336
  in a computation, 651, 657
  in a language, 657
Parameters, 2, 3
Paramodulation, 430, 442
Parent clauses, 428
Parent node, 43
Parse, 40, 62
Partial function, 20
Partial ordering, 267
Pascal, 67, 151
Path compression, 597
Patt, Yale, 662
Pattern, 353, 356, 617, 637
Pattern/expansion pairs, 259
Pattern expression, 260
Pattern matcher, 426
Pattern matching, 351, 420, 635
Pattern variables, 260
Patterson, David A., 289, 357, 565
PBA (Processor Binding Array), 231, 671
PC (Program Counter) register, 492
Pcall, the instruction, 270
PDL (push-down list), 493, 509
PDL buffer, 274, 281
Pending literal, 692
PEPSys machine, 703
Performance measures, 565
Periera, L. M., 556
Permanent variable, 531, 532
Permutations, 437
Personal Sequential Inference Machine, 581
Peyton-Jones, Simon L., 322, 332, 340
PFO (program-forming object), 345, 347
Picnic problem, 390, 443
PIM (Parallel Inference Machine), 702
Pipelined AND parallelism, 678
Pipelining, 578
Pitman, Kent M., 244
PL-1 language, 9
Placeholder, 14, 396
PLM (Programmed Logic Machine), 585, 702
Pointer avoidance, 208
Pointer fields, 45
Pointer stack, 333
Polymorphic functions, 245
Pop, the stack function, 145
Portable Standard LISP, 244
Positive clause, 400
Positive literal, 400
Postfix notation, 29, 31
Pountain, R., 359
Power set, 37
PPP machine, 702
Pragma, 704
Precedence, 31
Predecessor function, 308
Predicate, 20, 52, 378
  a function in abstract syntax, 64
Predicate-based logics, 394
Predicate expression, 88, 394
Predicate function form, 20
Prefetch, 580
Prefix, 17, 375
Prefix notation, 29, 120
Premisses, 373, 378, 380
Prenex form, 400
Prenex normal form, 402
Preorder graph traversal, 470
Pressure as a scheduling mechanism, 364
Primary key, 617
Primes, the function, 219, 226
Primitive recursion combinator, 305
Print name, 248, 276
Private variables, 668
PRM (Production-Rule Machine), 8, 634, 645
Problem, 2, 3
Procedural instructions, 498, 502, 587
Procedural interpretation, 455
Procedural language, 5
Procedure, 488, 491
  in logic, 455
Procedure call, 455
Proceed, the instruction, 538
Process, 698
  the function, 232, 367
Processor Binding Array, 231, 671
Producer-consumer problem, 229, 236, 696, 701
Product of sums, 400
Production, 636
Production parallelism, 652
Production rule, 38
Production-Rule Machine, 8, 634, 645
Production-rule systems, 607, 633
Prog:
  the instruction, 341
  the special form, 254
Program, 5, 11, 432, 488
  as lists, 144
Program code area in the Warren Abstract Machine, 491
Program Counter, 11, 152
Program-forming objects, 345, 347

Program variables, 247
Programmable Logic Arrays, 287
Programmed functions, 161
Programmed Logic Machine, 585, 702
Programming language, 5, 11
Project, the function, 618, 624
PROLOG, 408, 449, 486, 564
Promise, 210, 219
  (*See also* Future)
Proof, 379
  by contradiction, 434
  by refutation, 434
Proof tree, 444
Proper subset, 14
Properties:
  of an algorithm, 4
  of a function, 21
  of inference rules, 379
  of an object, 3, 17
Property, 248
Property list, 246, 253, 266, 277, 341
Property name, 248
Property value, 248
Proplog, 457, 525
Propogated pressure, 363
Proposition, 377, 384
Propositional letter, 384
Propositional logic, 370, 376, 384
Prove, the function, 459
Proxy function, 367
PSI (Personal Sequential Inference Machine), 581
PSI-II Machine, 581
Pure literal, 416
Push, the stack function, 145
Push-down list, 274, 281, 493, 509
Put-constant, the instruction, 510
Put instructions, 498, 510, 515
Put-list, the instruction, 510
Put-structure, the instruction, 510
Put-unsafe-value, the instruction, 546
Put-val, the instruction, 531
Put-value, the instruction, 546
Put-var, the instruction, 531
Putnam, H., 416
PutTval, the instruction, 532
Putv, the instruction, 510

QBE (query by example), 624
Quantifiers, 395, 398
Query, 450, 453, 609
  the function, 609
Query by example, 624
Query Processor, 609
Question, 407
Quick garbage, 191
Quicksort, 498, 570, 701
Quote, the instruction, 124, 174, 251, 341, 426

R combinator, 305, 331
Rabbit, a compiler, 265
Ramamohanarao, Kotagiri, 625
Ramesh, R., 662
Ramkumar, Balkrishna, 704
Ramsdell, John D., 329
Random function, 8
Random states, 266
Range, 18, 20
Range set, 611
Range value, 18
RAP, the instruction, 160
RAP-WAM, the machine, 705
Rate matching, 229
Read, the instruction, 476
Read-char, the instruction, 269
Read-eval-print loop, 137, 229, 245
Read mode, 493, 507, 508, 510, 667
Read-only variables, 701
READC, the instruction, 162
Reasoning, 372
Recipe, 214, 219
Record, 249, 608, 635
Recursion, 5, 55, 90, 103, 133, 311, 315
Recursion removal theorem, 349
Recursive call, 55, 160
Recursive closure, 135, 149
Recursive function, 41, 54, 55, 90
Recursive rule, 55
Rediflow system, 363
Redo message, 688, 690
Reduce, the function, 107
Reduce OR model, 704
Reduced Instruction Set Computers (RISC), 208, 265, 288, 589
Reduction, 28, 35, 77
  (*See also* Evaluation in parallel; Rewriting)
Reduction agent, 359, 362
Reeve, Michael, 359
Reference count, 180, 192, 329, 360, 644
Reference pointer, 197
  (*See also* Invisible pointer)
Reference tag, 494, 503
Referential transparency, 21, 29, 135, 215, 365
Reflexive relation, 18
Refutation complete, 431
Regions of memory, 201
Register allocation in the Warren Abstract Machine, 534
Register file, 579, 591

Register optimization in the Warren Abstract Machine, 532
Register windows, 289, 591
Registers, 11
  in the G-Machine, 334
  in the PRM, 646
  in the SECD, 150
  in the WAM, 492
Relation, 1, 16, 19, 370, 397, 607, 611, 614
Relational algebra, 615, 624
Relational DataBase Machine, 631
Relational expression, 624
Relational model, 607, 614
  in PROLOG, 625
Relational schema, 615
Relationship, 611
Remove, the instruction, 638
Rename, the database operation, 618, 624
Renaming, 21
  in lambda calculus, 77
  in logic, 403, 409
  in PROLOG, 465
Repeat, 478
Repeat combinator, 305
Replacement, 34
Representative name, 14
Resolution, 389, 420, 427, 428
  major variations, 429
Resolvant, 428
Restricted AND parallelism, 682, 705
Restriction strategies, 440, 441
Result, 386, 407
Result cache, 694
Rete Match algorithm, 639
Rete net, 634
Retract, the instruction, 480, 519, 597, 603
Retry, the instruction, 541
Retry-me-else, the instruction, 501, 513, 541, 542
Return, the instruction, 254, 502, 516, 537
Reversal combinator, 305
Reverse, the function, 51–60, 346, 473
  an imperative form, 60
Reverse Polish notation, 31, 144
Rewrite rule, 34, 37, 359
Rewrite units, 362
Rewriting, 312
RHS (right-hand side) parallelism, 652
Ribeiro, Joao, 702
Ribler, Randy L., 588
Right, the function, 46
  (*See also* Cdr)
RISC (*see* Reduced Instruction Set Computers)
Robert, P., 703
Robinson, J. Alan, 340, 428
Robison, A. D., 339, 345
Robison, S. V., 339
Robson, David, 19
Root node, 43, 645
Rosser, J. Barkley, 68
Rossi, Francesca, 676
Rotate, the function, 346
Row command, 629
Rplaca, 176, 201, 222, 252, 271
Rplacd, 186, 222, 252, 271
Rplacrc, 192
Rplactag, 189
Rtn, the instruction, 157
Rule, 450, 452
  in OPS, 636
Rules of thumb, 440

S combinator, 299
S-expression (*see* Symbolic expression)
S register, 151, 287, 493
Safe guards, 700
Sannella, D. T., 351
SASL, the language, 339
Satisfiable wff (well-formed formula), 382
Satisfying substitution, 374, 381, 382, 419
Scavenge, the function, 199
Scavenger, 196
Scavenger pointer, 198
Scavenging, 181, 196–204
Scheduling, 674
SCHEME, the language, 234, 257
SCHEME-79 chip, 285, 330
Schoenfinkel, M., 300
Schorr, H., 187
Schwartz, J. T., 339
Scope, 73, 99
  of quantifiers, 398
Search tree, 438
SECD Machine, 8, 33, 140, 238, 328, 466, 486
  nondeterministic extensions, 240
Sel, the instruction, 156, 166
Seldin, J. P., 23, 86, 297
Select:
  the instruction, 627
  the function, 618, 624, 642
Select line, 210
Selection, 614
Selector functions, 64, 301
Self-interpretation, 113, 482, 581
Self-recursion, 176
Semantic evaluation functions, 10, 62
Semantic invariance, 21
Semantic model, 7, 95
Semantics, 5, 7, 36, 378, 398
Semaphores, 271

Semidecidable logic theory, 411
Semiintelligent backtrack, 559, 693
Semispaces, 272
Sentence, 5
  the instruction, 341
  in logic, 377
Sequence, 231, 339, 345
Sequential Inference Machine, 631
Sergot, M. J., 450
Set(s), 13
  of functions, 21
  the instruction, 251, 343
Set!, the instruction, 259
Set of support axioms, 442
Set of support strategy, 442, 444
Set of Support theorem, 442
Set closure, 37
Set difference, 15
Set-fluid!, the instruction, 262
Set occurrence, 614
Set-of, the instruction, 626
Set theory, 1
Set variable, 14
Set-xxx, the instruction, 703
Set1, the instruction, 339
Shadowed binding, 247
Shadowed variable, 343
Shallow backtrack, 457, 460, 490, 497, 505, 705
Shallow binding, 248, 276, 284
Shankar, Subash, 605
Shapiro, Elud, 694, 700
Shared memory machine, 661, 705
Shemer, Jack, 629
Shen, Kish, 670
Shepard, John, 626
Shultis, J., 351
Shustek, L. J., 565
Sibert, E. E., 340
Side effect of a computation, 31, 221
Sidetracking, 589
SIMD (single-instruction, multiple-data) vector processors, 661
Simple expressions, 28
Simplification to lambda calculus, 71
Simplification strategies, 440, 442
Singhal, Ashok, 662
Single-instruction, multiple-data vector processors, 661
Size, 14
SKIM machines, 327
Skolem constant, 405
Skolem function, 405
Slolem standard form, 404
Skolemization, 405
Sleep, M. R., 339, 340, 365
Slot, 266
Slow garbage, 191
Smalltalk, 9
Smith, Sarah, 279
Sohi, Gurindar, 208, 601
Solution, 3
Solution capturing, 467
Solved literal, 692
Solving a problem, 2
Sort, the function, 9, 24
Sound, 379, 426
SP (scavenger pointer) register, 198, 283, 493
Spaghetti stacks, 173
Special, the instruction, 251
Special forms, 121, 126, 150. 216, 246, 252, 258
Special variables, 247, 268
Split, the function, 622
Spreadsheet, 628
Spring, George, 244, 257
SPUR, the machine, 288, 589
SQL (Structured Query Language), 626
SRI model of OR parallelism, 671
Stable cycles of garbage collection, 191
Stack, 56, 144, 150, 457
  S register, 151
  in the SECD machine, 144
  in the WAM, 493
Stack buffer, 579
Stack group, 282
Stack Machine, 32
Staging latches, 579
Start message, 688
Statement, 5, 377, 450
Static area, 245
Statically scoped variables, 247
Steele, Guy L., 244, 257, 265
Steenkiste, Peter, 208, 292, 293
Steinberg, S., 273
Stolfo, Salvatore J., 652
Stone, Harold S., 625
STOP, the instruction, 162
Stormon, Charles, 596, 599, 600, 603, 605
Stoy, Joseph, 7, 10, 23, 29, 68, 80, 86
Stoye, R. S., 327
Strand, 700, 701
Stream, 214, 229, 231, 268, 366, 658, 665
Stream AND parallelism, 694
Stream function, 694
Stream parallelism, 677
String, 36
String reduction, 312, 359
Strong typing, 345, 351
Structure, 494, 508, 551
Structure copying, 320, 705

Structure pointer, 496, 508
Structure pointer tag, 495
Structure sharing, 320, 328, 705
Structured data type, 245
Structured Query Language, 626
Subs, the function, 115, 117, 126
Subset, 14
Substitution, 2, 3, 6, 34, 35, 70, 74, 84, 398, 421, 425, 505
  the function, 423
Subsumption, 430, 442
Subtraction, 481
Succeed, 455, 699
Success message, 688, 690
Success pattern, 554
Successor function, 86, 306, 310
Sugaya, Masshiro, 702
Sum of products wff (well-formed formula) form, 400
Sumproduct, the function, 60
SUPER, the machine, 340
Supercombinators, 297, 309, 322
Superimposed code words, 605, 626
Superkey, 616
Surjective function, 21
Suspend, 697, 699
Suspend instruction, 703
Sussman, Gerald, J., 244, 257, 285
Sweep process, 184
Switch-on-constant, the instruction, 541
Switch-on-structure, the instruction, 541
Switch-on-term, the instruction, 540
Switch-on-type, the instruction, 517, 540, 549
Syllogism, 420, 428
Symbol, 5, 13, 36, 377
Symbol table, 177, 513
Symbolic differentiation, 108
Symbolic expression, 1, 42, 119, 143, 495
  syntax of, 44
Symbolics, 281
Symmetric relation, 18
Syntax, 5, 36, 37
  of abstract program, 53, 95, 122
  of clausal logic, 401
  of constant applicative forms, 295
  of FP, 346
  of HOPE, 353
  of infix expression, 39
  of lambda calculus, 68
  of logic, 377, 385, 396, 401
  of Logo, 342
  of OPS, 636
  of prefix s-expressions, 109
  of PROLOG, 451
  of Proplog, 458
Syntax (*Cont.*)
  of s-expressions, 44, 109
  of SQL, 627
System of logic, 376
System R, 626
System syntax expander, 259

T processors, 358
Table, 614
Table skeleton, 624
Tabling, 633
Tag field, 45, 142, 333, 494, 572, 591
Tag implementation, 206
Tagging a variable, 465
Tags, recursively defined, 340
Tail, 46
  (*See also* Cdr)
Tail recursion, 57, 174, 349, 460
Tail recursion optimization, 543
TAK, the function, 292, 294
Taki, K., 581, 583, 702
Tanake, Hidehiko, 624
Tarnlund, S. A., 700
Task queue, 272
Tasks, parallel, 270
Tautology, 378, 382, 384
Taylor, George S., 288
Taylor, Stephen, 700
Teitelman, Warren, 244
Temporary variable, 531, 532, 537, 703
Teradata DBC/1012, 629
Term, 39, 396, 412, 450
Term matching, 662
Terminal cells, 143
Terminal character, 37
Terminal node, 42, 645
Test, the instruction, 174, 342
Texas Instruments LISP Chip, 280
Theorem, 379, 411
Theory of logic, 376
There exists (*see* Existential quantifier)
Theta comparable, 623
Threaded code, 265, 596
Throw, the instruction, 268, 341, 343, 582
Thunk, 215
Tick, Evan, 565, 568, 570, 578, 586, 588
Time constraints, 229
To, the instruction, 342
Token Lists, 646
Token Queue, 646
Tokens, 581, 640
Tospace, 196, 285
Total function, 20
Touati, H., 527, 568, 570
Touch, the instruction, 271
Towers of Hanoi problem, 60, 481, 570

TR register, 493
Tracing a program, 568
Trail buffer, 579
Trail optimization, 527
Trail stack, 19–20, 493, 505, 527, 598
Trailing, 284, 505, 527
Transitive closure, 19
Transitive relation, 18
Transpose, 347
Trap, the instruction, 176
Trap mechanism, 289
Treat, the algorithm, 653
Tree, 42
Tree machines, 357, 630, 652
Treleaven, P. C., 326, 340
Triple, 110
Trit, 602
True under an interpretation, 381
Trust, the instruction, 542
Trust-me, the instruction, 542
Truth, 88, 299, 371, 476
Truth table, 384
Truth table analysis, 386, 387
Try:
  the function, 240
  the instruction, 240, 549
Try-me-else, the instruction, 549
Tuple, 15, 44, 345, 370, 609
  at a time processing, 621, 625
Turk, Andrew W., 527
Turner, D. A., 68, 297, 300, 303, 327, 339
Turtle graphics, 341
Two-input nodes, 642
Two nodes, 645
Two-place (binary) relation, 17
Type, 13, 18
Type definition, 353
Type variable, 354, 355
Type variable declarations, 354
Typing systems, 551

Uchida, S., 702
Ufr, the instruction, 222, 227
Ullman, Jeffrey, 608, 612, 614, 626, 628
Unconditional bindings, 668
Unconditional branch, 254
Undecidable logic theory, 380, 411
Undefined object, 345
Unfair scheduling policy, 272
Unification, 351, 420, 472, 661
Unification parallelism, 661
Unifier, 421
Unify, the function, 422, 464, 506, 551, 578, 600
Unify-constant, the instruction, 511
Unify instructions, 498, 510, 515
Unify-val, the instruction, 511, 531
Unify-var, the instruction, 511, 531
Unify-void, the instruction, 532
Unifying lists, 511, 601
Unifying substitution, 421, 428
Unifyv, the instruction, 511
Union, the function, 15, 58, 618, 624
Unit clause, 401
Unit preference strategy, 441, 444
Unit resolution, 429
Universal quantifier, 398
Universe, 412
Unreadable data objects, 266
Unsafe variables, 546
Unsatisfiable wff (well-formed formula), 383, 415, 434
Unsound logic theory, 379
Unspecial, the instruction, 251
Untyped languages, 351
Unwinding the trail, 501, 506, 598
Update, the function, 609
UR resolution, 429

VAL, the language, 340
Valid logic theory, 382
Validity, 5
Value, 13, 27
Value field, 45, 333, 494
Value list, 129, 144
Value stack, 333
Van Roy, Peter, 550, 702
Van Wijngaarden, 9
Variable, 2, 12, 72, 345, 395
  (*See also* Identifier)
Variable initialization environment, 530
Variable name, 597, 629
Variable optimization, 532
Variable tag, 494, 507, 531
Vector, the function, 107, 582, 661
Vector processors, 661, 702
Veen, Arthur H., 340
Vegdahl, S. R., 340
Version vectors model of OR parallelism, 675
Vienna Definition Language, 9
VLSI chip, 285, 329, 357, 358
Void variable, 532
Von Neumann bottleneck, 11
Von Neumann model, 10

W combinator, 305
Wadler, P. L., 340
Waite, W. M., 187
Waiter process, 138
Waiting list, 688
Waiting state, 687

Walk, K., 9
WAM (*see* Warren Abstract Machine)
Warren Abstract Machine, 8, 241, 340, 486, 497, 526, 528, 533, 702, 705
Warren, David H. D., 486, 544, 550, 578, 610, 671
Warren, David S., 486, 553, 663
Watson, Ian, 340
Waves of processing, 358
Wegbreit, B., 171, 173
Wegner, P., 9
Weissman, Clark, 244
Well-formed formulas, 377, 398
Westphal, H., 703
Wffs (well-formed formulas), 377, 398
Where, the instruction, 54, 103
Whererec, the instruction, 54, 106
Whetstone program, 291
While loop, 108, 254
White, Colin J., 614
Wild card, 471
Wilensky, Robert, 244
Williams, M. Howard, 626
Window, 19–33
Window pointer, 19–33
Winter abstract machine, 340
Wirth, Niklaus, 36
Wme (working memory element), 634
Wolfe, Alexander, 280
Wong, Kam-fai, 626
Woo, N. S., 565
Work file, 634
  (*See also* Register file)
Working memory, 634
Working memory elements, 634
Wos, Larry, 420, 426, 432
Write, the instruction, 476, 638
Write mode, 493, 507, 508, 510, 511, 703
WRITEC, the instruction, 162

X-1 machine, 588
X-tree project, 357
XWAM, 633

Y combinator, 304, 311, 315
Ynet, 630
Yuhara, Masanobu, 287
Yung, Chao-chih, 625

ZAPP machine, 339
Zero:
  the constant, 481
  the predicate, 89
ZetaLISP, 244, 281
Zloof, M., 628