

---

# Contents

<b>1</b>	<b>Introduction</b>	1
1.1	Simplicity, Expressiveness, and Performance	1
1.1.1	Simplicity	2
1.1.2	Expressiveness	5
1.1.3	Performance	9
1.2	The Logical Foundations of Maude	11
1.3	Programming, Specification, and Verification	14
1.4	A High-Performance Logical Framework	17
1.5	Core Maude vs. Full Maude	20
1.6	Book Structure	21
1.7	How to Read This Book	23

---

## Part I Core Maude

---

<b>2</b>	<b>Using Maude</b>	31
2.1	Getting Maude	31
2.2	Running Maude	31
2.3	Getting Support and More Information	35
2.4	Reporting Bugs in Maude	36
<b>3</b>	<b>Syntax and Basic Parsing</b>	39
3.1	Identifiers	39
3.2	Modules	40
3.3	Sorts and Subsorts	41
3.4	Operator Declarations	44
3.5	Kinds	46
3.6	Operator Overloading	48
3.7	Variables	49
3.8	Terms and Preregularity	49
3.9	Parsing	51
3.9.1	Default Precedence Values	53

3.9.2	Default Gathering Patterns . . . . .	54
3.9.3	The Extended Signature of a Module . . . . .	55
3.9.4	Parsing Examples . . . . .	56
<b>4</b>	<b>Functional Modules . . . . .</b>	<b>61</b>
4.1	Unconditional Equations . . . . .	62
4.2	Unconditional Memberships . . . . .	63
4.3	Conditional Equations and Memberships . . . . .	64
4.4	Operator Attributes . . . . .	68
4.4.1	Equational Attributes . . . . .	68
4.4.2	The <code>iter</code> Attribute . . . . .	71
4.4.3	Constructors . . . . .	71
4.4.4	Polymorphic Operators . . . . .	75
4.4.5	Format . . . . .	76
4.4.6	Ditto . . . . .	79
4.4.7	Operator Evaluation Strategies . . . . .	80
4.4.8	Memo . . . . .	84
4.4.9	Frozen Arguments . . . . .	87
4.4.10	Special . . . . .	88
4.5	Statement Attributes . . . . .	89
4.5.1	Labels . . . . .	89
4.5.2	Metadata . . . . .	89
4.5.3	Nonexec . . . . .	90
4.5.4	Otherwise . . . . .	90
4.6	Admissible Functional Modules . . . . .	95
4.7	Matching and Equational Simplification . . . . .	96
4.8	More on Matching and Simplification Modulo . . . . .	100
4.9	The <code>reduce</code> , <code>match</code> , <code>trace</code> , and <code>show</code> Commands . . . . .	106
4.10	A Number Hierarchy . . . . .	111
4.11	Partial Operations, Subsorting, and Errors . . . . .	115
<b>5</b>	<b>A Hierarchy of Data Types: From Trees to Sets . . . . .</b>	<b>119</b>
5.1	Nonempty Binary Trees . . . . .	120
5.2	Nonempty Lists . . . . .	121
5.3	Lists . . . . .	122
5.4	Multisets . . . . .	124
5.5	Sets . . . . .	126
5.6	Idempotent Semigroups . . . . .	127
<b>6</b>	<b>System Modules . . . . .</b>	<b>131</b>
6.1	Unconditional Rules . . . . .	132
6.2	Conditional Rules . . . . .	134
6.3	Admissible System Modules . . . . .	135
6.4	The <code>rewrite</code> , <code>frewrite</code> , and <code>search</code> Commands . . . . .	139
6.4.1	The <code>rewrite</code> Command . . . . .	139

6.4.2	The <b>frewrite</b> Command .....	142
6.4.3	The <b>search</b> Command .....	143
6.5	More Examples of System Modules .....	148
6.5.1	Petri Nets .....	148
6.5.2	Blocks World .....	154
<b>7</b>	<b>Playing with Maude</b> .....	159
7.1	Writing on a Blackboard .....	160
7.2	The Hopping Rabbits Game .....	163
7.3	The Josephus Problem .....	165
7.4	The Three Basins Puzzle .....	167
7.5	Crossing the Bridge .....	169
7.6	The Looping Chips Game .....	172
7.7	The Khun Phan Puzzle .....	173
7.8	Crossing the River .....	176
7.9	Dominoes on a Chessboard .....	179
7.10	Black or White .....	182
7.11	The Game Is Not Over .....	183
<b>8</b>	<b>Module Operations</b> .....	185
8.1	Module Importation .....	186
8.1.1	Protecting .....	187
8.1.2	Extending .....	188
8.1.3	Including .....	189
8.1.4	Default Conventions in Module Importations .....	190
8.1.5	Some Module Hierarchy Examples .....	191
8.2	Module Summation and Renaming .....	193
8.2.1	The Summation Module Expression .....	193
8.2.2	Module Renaming .....	194
8.3	Parameterized Programming .....	198
8.3.1	Theories .....	198
8.3.2	Views .....	204
8.3.3	Parameterized Modules .....	209
8.3.4	Module Instantiation .....	214
8.3.5	A Specification of Sorted Lists .....	221
8.3.6	The Lambda Calculus .....	224
<b>9</b>	<b>Predefined Data Modules</b> .....	231
9.1	Boolean Values .....	232
9.2	Natural Numbers .....	237
9.3	Random Numbers and Counters .....	241
9.4	Integer Numbers .....	244
9.5	Machine Integers .....	247
9.6	Rational Numbers .....	251
9.7	Floating-Point Numbers .....	256

9.8	Strings .....	260
9.9	String and Number Conversions .....	264
9.10	Quoted Identifiers .....	266
9.11	Basic Theories and Standard Views .....	267
9.11.1	TRIV .....	267
9.11.2	DEFAULT .....	268
9.11.3	STRICT-WEAK-ORDER and STRICT-TOTAL-ORDER .....	269
9.11.4	TOTAL-PREORDER and TOTAL-ORDER .....	272
9.12	Containers: Lists and Sets .....	274
9.12.1	Lists .....	274
9.12.2	Sets .....	277
9.12.3	Relating Lists and Sets .....	280
9.12.4	Generalized Lists .....	281
9.12.5	Generalized Sets .....	284
9.12.6	Sortable Lists .....	287
9.12.7	Making Lists out of Sets .....	294
9.13	Maps and Arrays .....	296
9.13.1	Maps .....	297
9.13.2	Arrays .....	299
9.14	A Linear Diophantine Equation Solver .....	301
<b>10</b>	<b>Specifying Parameterized Data Structures in Maude .....</b>	<b>307</b>
10.1	Stacks .....	308
10.2	Queues .....	309
10.3	Priority Queues .....	310
10.4	Lists .....	313
10.5	Sorted Lists .....	315
10.6	Multisets .....	318
10.7	Binary Trees .....	320
10.8	General Trees .....	322
10.9	Binary Search Trees .....	324
10.10	AVL Trees .....	328
10.11	2-3-4 Trees .....	332
10.12	Red-Black Trees .....	336
10.13	Efficiency Concerns .....	337
<b>11</b>	<b>Object-Based Programming .....</b>	<b>339</b>
11.1	Configurations .....	340
11.2	Object-Message Fair Rewriting .....	351
11.3	Example: Data Agents .....	353
11.4	External Objects .....	361
11.4.1	Sockets .....	361
11.4.2	Buffered Sockets .....	369

<b>12 Model Checking Invariants Through Search</b> .....	373
12.1 Invariants .....	373
12.2 Model Checking of Invariants .....	374
12.3 Bounded Model Checking of Invariants .....	378
12.4 Verifying Infinite-State Systems Through Abstractions .....	380
<b>13 LTL Model Checking</b> .....	385
13.1 LTL Formulas and the LTL Module .....	385
13.2 Associating Kripke Structures to Rewrite Theories .....	387
13.3 LTL Model Checking .....	392
13.4 Equational Abstractions .....	399
13.5 The LTL Satisfiability and Tautology Checker .....	408
13.6 Verifying LTL Properties of Imperative Concurrent Programs .....	410
13.6.1 The Semantics of a Simple Parallel Language .....	410
13.6.2 Model Checking Dekker's Algorithm .....	412
13.7 Crossing the River (Revisited) .....	416
13.8 Other Model-Checking Examples .....	418
<b>14 Reflection, Metalevel Computation, and Strategies</b> .....	419
14.1 Reflection and Metalevel Computation .....	419
14.2 The <b>META-TERM</b> Module .....	421
14.2.1 Metarepresenting Sorts and Kinds .....	421
14.2.2 Metarepresenting Terms .....	422
14.3 The <b>META-MODULE</b> Module: Metarepresenting Modules .....	423
14.4 The <b>META-LEVEL</b> Module: Metalevel Operations .....	428
14.4.1 Moving Between Reflection Levels: <b>upModule</b> , <b>upTerm</b> , <b>downTerm</b> , and Others .....	428
14.4.2 Simplifying: <b>metaReduce</b> and <b>metaNormalize</b> .....	432
14.4.3 Rewriting: <b>metaRewrite</b> and <b>metaFrewrite</b> .....	434
14.4.4 Applying Rules: <b>metaApply</b> and <b>metaXapply</b> .....	435
14.4.5 Matching: <b>metaMatch</b> and <b>metaXmatch</b> .....	439
14.4.6 Searching: <b>metaSearch</b> and <b>metaSearchPath</b> .....	442
14.4.7 Parsing and Pretty-Printing: <b>metaParse</b> and <b>metaPrettyPrint</b> .....	444
14.4.8 Sort Operations .....	447
14.4.9 Other Metalevel Operations: <b>wellFormed</b> .....	454
14.5 Internal Strategies .....	455
<b>15 Metaprogramming Applications</b> .....	459
15.1 Commutative Order-Sorted Unification .....	460
15.2 Rule Instrumentation .....	472
15.3 A Deadlock-Freedom Transformation .....	478

<b>16 Mobile Maude</b>	485
16.1 Processes and Mobile Objects	486
16.1.1 Processes and Root Objects	486
16.1.2 Mobile Objects	488
16.1.3 Message Forwarding	489
16.2 Mobile Maude Additional Definitions	491
16.3 Mobile Maude's Rewriting Semantics	492
16.3.1 Letting Mobile Objects Do Something	493
16.3.2 Object Communication	494
16.3.3 Object Mobility	498
16.3.4 The Creation of Mobile Objects	503
16.3.5 Mobile Object Destruction	505
16.4 Mobile Maude Architecture	506
16.5 A Buying Printers Example	510
16.6 Model Checking Mobile Maude Applications	518
16.6.1 Redefinition of the <code>SOCKET</code> Module	519
16.6.2 Two-Level Atomic Propositions for the Buying Printers Example	520
<b>17 User Interfaces and Metalanguage Applications</b>	523
17.1 The <code>LOOP-MODE</code> Module	523
17.2 User Interfaces	524
17.3 Using the Loop	528
17.4 Metalanguage Applications: Tokens, Bubbles, and Metaparsing	529
17.5 Interactive Maude	537
17.5.1 <code>IMaude</code>	538
17.5.2 <code>IOP</code>	548

---

## Part II Full Maude

---

<b>18 Full Maude: Extending Core Maude</b>	559
18.1 Running Full Maude	560
18.2 Using Core Maude Modules in Full Maude	565
18.3 Additional Module Operations in Full Maude	567
18.3.1 The Tuple and Power Module Expressions	569
18.3.2 Parameterized Views	571
18.3.3 Example: Leftist Trees	574
18.4 Moving Up and Down Between Reflection Levels	578
18.4.1 Up	578
18.4.2 Down	580
18.5 Differences Between Full Maude and Core Maude	581
18.6 Adding New Features to Full Maude	583
18.6.1 A Unification Command	583
18.6.2 A New Module Expression: <code>DEADLOCK-FREE</code>	591

<b>19 Object-Oriented Modules</b> .....	599
19.1 Object-Oriented Systems.....	600
19.1.1 Objects and Messages .....	600
19.1.2 Classes .....	601
19.1.3 Inheritance.....	602
19.1.4 Object-Oriented Rules .....	603
19.2 More Examples of Object-Oriented Modules .....	607
19.2.1 A Puzzle.....	607
19.2.2 A Simple Spreadsheet .....	609
19.2.3 Blocks World .....	612
19.3 A Bigger Example: A Rent-a-Car Store .....	614
19.4 Object-Oriented Parameterized Programming .....	618
19.4.1 Theories .....	618
19.4.2 Views .....	619
19.4.3 Parameterized Object-Oriented Modules .....	619
19.5 Module Operations on Object-Oriented Modules .....	622
19.5.1 Module Summation and Renaming .....	622
19.5.2 Module Instantiation .....	623
19.6 Example: Extended Rent-a-Car Store .....	624
19.7 A Strategy for Sequential Rule Execution .....	630
19.8 Model Checking a Round-Robin Scheduling Algorithm .....	634
19.9 From Object-Oriented Modules to System Modules.....	638

---

## Part III Applications and Tools

---

<b>20 A Sampler of Application Areas</b> .....	645
20.1 Models of Computation .....	645
20.2 Semantics of Programming Languages and Software Analysis .....	647
20.3 Maude as a Metalanguage.....	650
20.3.1 Representing, Mapping, and Reasoning About Logics .....	650
20.3.2 Specifying and Building Formal Tools .....	652
20.4 Modeling and Analysis of Networks and Distributed Systems .....	653
20.4.1 Distributed Architectures and Components.....	653
20.4.2 Specification and Analysis of Communication Protocols .....	655
20.4.3 Modeling and Analysis of Security Protocols .....	656
20.5 Real-Time Systems.....	657
20.6 Probabilistic Systems .....	658
20.7 Modeling and Analysis of Biological Systems .....	661
<b>21 Some Tools</b> .....	667
21.1 Maude Tools .....	667
21.1.1 The ITP: An Inductive Theorem Prover .....	667
21.1.2 The Maude Termination Tool .....	669
21.1.3 The Church-Rosser Checker .....	670

21.1.4	The Maude Coherence Checker .....	672
21.1.5	The Sufficient Completeness Checker .....	673
21.1.6	The Real-Time Maude Tool .....	675
21.1.7	Predicate Abstraction in Maude .....	676
21.2	Other Tools .....	677
21.2.1	The Open Calculus of Constructions .....	677
21.2.2	The Maude MSOS Tool .....	682
21.2.3	The CCS and LOTOS Tools .....	683
21.2.4	The MSR Cryptoprotocol Specification Language ....	684
21.2.5	JavaFAN .....	687
21.2.6	Java+ITP .....	688
21.2.7	The ITP/OCL Tool .....	690
21.2.8	The Pathway Logic Assistant .....	692

---

## Part IV Reference

---

<b>22</b>	<b>Debugging and Troubleshooting .....</b>	<b>697</b>
22.1	Debugging Approaches .....	697
22.1.1	Tracing .....	697
22.1.2	Term Coloring .....	706
22.1.3	The Debugger .....	708
22.1.4	The Profiler .....	711
22.1.5	Performance Note .....	722
22.2	Traps and Known Problems .....	723
22.2.1	Associativity and Idempotency .....	723
22.2.2	Segmentation Fault (Core Dumped) .....	724
22.2.3	Bare Variable Lefthand Sides .....	725
22.2.4	Operator Overloading and Associativity .....	725
22.2.5	Preregularity and Equational Attributes .....	726
22.2.6	Collapse Theories .....	728
22.2.7	One-Sided Identities and Associativity .....	729
22.2.8	Memberships for Associative Operators .....	731
22.2.9	Memberships for Iterated Operators .....	734
<b>23</b>	<b>Complete List of Maude Commands .....</b>	<b>737</b>
23.1	Command Line Flags .....	737
23.2	Rewriting Commands .....	738
23.3	Matching Commands .....	740
23.4	Searching Commands .....	741
23.5	Tracing Commands .....	742
23.6	Print Option Commands .....	743
23.7	Show Option Commands .....	744
23.8	Show Commands .....	745
23.9	Profiler Commands .....	746



23.10	Debugger Commands .....	746
23.11	Miscellaneous Commands .....	747
23.12	System Level Commands .....	748
<b>24</b>	<b>Core Maude Grammar .....</b>	<b>751</b>
24.1	The Grammar .....	751
24.2	Synonyms .....	755
24.3	Lexical Issues .....	756
	<b>References .....</b>	<b>757</b>
	<b>Subject Index .....</b>	<b>783</b>
	<b>Index of Maude Modules .....</b>	<b>791</b>
	<b>Index of Maude Theories .....</b>	<b>795</b>
	<b>Index of Maude Views .....</b>	<b>797</b>

All About Maude - A High-Performance Logical  
Framework

How to Specify, Program, and Verify Systems in  
Rewriting Logic

Clavel, M.; Durán, F.; Eker, S.; Lincoln, P.; Martí-Oliet, N.;  
Meseguer, J.; Talcott, C.

2007, XXII, 802 p. With CD-ROM., Softcover

ISBN: 978-3-540-71940-3