Contents

Preface xv

1 Introduction 1

1.1 What is this book about? The verification problem 1
1.2 Testing and verification for establishing system requirements 2
1.3 From systems to models of systems 3
1.4 Design automation ecosystem 7
1.5 Challenges and state of the art 7
1.6 Road map 11
1.7 Learning and teaching using this book 14

2 Modeling Computation 17

2.1 Quick introduction to automata 17
2.1.1 Example: JK flip-flop 17
2.1.2 Language for specifying automata 17
2.2 Specifying automata 20
2.2.1 State variables and valuations 20
2.2.2 Predicates 20
2.2.3 Transitions 21
2.2.4 Automata 22
2.3 Special automata classes 23
2.3.1 Finite and discrete automata 23
2.3.2 Nondeterminism 23
2.3.3 Discrete sequences and sampled time 23
2.4 Semantics: Executions, reachable states, and invariants 24
2.5 Example: Dijkstra's token ring algorithm 24
2.5.1 Legal states and invariants 26
2.5.2 Asynchronous and synchronous models 26
2.6 Example: Reasoning about impossibility 27

2.7 Exercises 28

3 Modeling Physics 31

3.1 Quick introduction to differential equations 31
3.1.1 Example: Vehicle speed control 31
3.1.2 Language for specifying differential equations 31
3.2 Specifying ordinary differential equations 33
3.2.1 State variables and valuations 34
3.2.2 Dense time and trajectories 34
3.2.3 Trajectories as solutions 35
3.3 Special classes of ODEs 37
3.3.1 Time-invariant and autonomous systems 37
3.3.2 Linear systems 38
3.4 Semantics: Reachable states, invariants, and stability 39
3.4.1 Example: Pendulum 40
3.4.2 Example: Kinematic vehicle model 42
3.5 Lyapunov's direct method for proving stability 44
3.5.1 Stability of linear dynamical systems 45
3.6 Differential equations as automata 46
3.7 Example: Simple economy 47
3.8 Numerical simulations for ordinary differential equations 48
3.9 Closing the loop and control synthesis 49
3.9.1 Proportional-integral-derivative controller 51
3.9.2 Controller synthesis problem 52
3.10 Exercises 53

4 Modeling Cyber-Physical Systems 55

4.1 Quick introduction to hybrid automata 55
4.1.1 Example: Rimless wheel 55
4.1.2 Language for specifying hybrid systems 56
4.2 Specifying hybrid automata 58
4.2.1 State variables and transitions 58
4.2.2 Trajectories and closures 58
4.2.3 Hybrid automata 60
4.2.4 A guide for hybrid modeling 61
4.3 Special classes of hybrid automata 61
4.3.1 Deterministic hybrid automata 61
4.3.2 Switched systems 62
4.3.3 Linear hybrid automata 63
4.3.4 Example: Thermostat 63

4.3.5 Rectangular hybrid automata 64
4.3.6 Timed automata 65
4.4 Semantics: Hybrid executions 65
4.4.1 Numerical simulation of hybrid executions 67
4.4.2 Reachable states, invariants, and stability 68
4.4.3 Time-abstract semantics 69
4.4.4 Execution zoo 71
4.5 Example: Spacecraft docking 72
4.6 Example: Small aircraft traffic management system 74
4.7 Exercises 75

5 Composing Models 79
5.1 Composing automata 79
5.2 Composing input/output automata 80
5.2.1 Input/output automata 80
5.2.2 Compatibility and composition of input/output automata 80
5.3 Example: Channels, logical clocks, and distributed systems 81
5.3.1 First-in, first-out channels 81
5.3.2 Logical time in distributed systems: Lamport clocks 83
5.3.3 Composed system: A network of processes communicating over channels 84
5.3.4 Traces and projections 85
5.4 Composing hybrid input/output automata 87
5.4.1 Hybrid input/output automata 88
5.4.2 Compatibility and composition of hybrid input/output automata 89
5.5 Example: Interconnecting flip-flops 90
5.6 Example: Timed channels 91
5.7 Example: Pulse generator and oscillator 93
5.8 Traces, untiming, and properties of compositions 93
5.9 Example: Emergency braking on highways 96
5.10 Exercises 98

6 Specifying Requirements 101
6.1 Requirements analysis 101
6.2 Safety standards 102
6.2.1 DO-178C 102
6.2.2 ISO 26262 104
6.2.3 Beyond current safety standards and requirements 105
6.3 From requirements to verification 105

6.3.1 Formal verification algorithms 106
6.3.2 Resources for verification and computational complexity 107
6.3.3 Invariants and safety requirements 109
6.3.4 Progress requirements 110
6.4 Linear temporal logic 111
6.4.1 Background definitions 112
6.4.2 LTL syntax 113
6.4.3 LTL semantics 113
6.5 Computational tree logic 116
6.5.1 Computational tree logic syntax 116
6.5.2 Computational tree logic semantics 116
6.5.3 Expressiveness of linear temporal logic and computational tree logic 117
6.6 Further reading 118
6.6.1 Checking temporal logic models 118
6.6.2 Planning and synthesis with temporal logics 118
6.6.3 Dense time, signal, and stochastic temporal logics 119
6.6.4 Runtime verification and monitoring 120
6.7 Exercises 121

7 Verifying Invariants 123
7.1 Quick introduction to proving invariants 123
7.2 Reasoning with inductive invariants 125
7.2.1 Invariance and composition 127
7.3 Proving timing-based mutual exclusion 127
7.3.1 Example: Fischer's mutual exclusion 127
7.3.2 Analysis of Fischer's mutual exclusion 130
7.4 Proving inductive invariants without solving ordinary differential equations 133
7.4.1 Example: Checking subtangential conditions 134
7.4.2 Barrier certificates 136
7.5 Satisfiability and satisfiability modulo theories 137
7.5.1 Satisfiability 137
7.5.2 Satisfiability modulo theory 138
7.5.3 Modeling for satisfiability and satisfiability modulo theory 140
7.6 Further reading 141
7.6.1 Finding and learning invariants 141
7.7 Exercises 143

8 Abstractions and Compositional Reasoning 145
8.1 Quick introduction to abstractions: Timing abstraction 145
8.2 Abstraction definitions 148
8.3 Proving abstractions: Simulation relations 149
8.4 Bisimulations and time-abstract bisimulations 152
8.4.1 Untiming and bisimulations 153
8.4.2 Example: Simulation and trace inclusion 154
8.4.3 Backward simulations 154
8.5 Hybridization 155
8.6 Substituting with abstractions 157
8.7 Designing a CEGAR-based cyber-physical verification system 158
8.7.1 Space of abstractions 159
8.7.2 Model checker 162
8.7.3 Counterexample validation 162
8.7.4 Refinement strategy 163
8.8 Further reading 163
8.9 Exercises 163

9 Reachability Analysis 165
9.1 Quick introduction to reachability analysis 165
9.2 Finite automata 166
9.2.1 Finite state reachability 166
9.3 Timed automata 168
9.3.1 Syntax for timed automata 168
9.3.2 Example: Timed light switch 170
9.3.3 Clock equivalence relation on states 170
9.3.4 Control state reachability and region automata 173
9.4 Integral timed automata to rectangular hybrid automata 176
9.4.1 Rational timed automata 176
9.4.2 Multirate automata 176
9.4.3 Rectangular hybrid automata 177
9.5 Undecidability of control state reachability for rectangular hybrid automata 178
9.5.1 Two-counter machines 178
9.5.2 Reduction of control state reachability of rectangular hybrid automata to the halting problem of two-counter machines 179
9.5.3 Initialized rectangular hybrid automata 182
9.6 Relaxing the verification problem 183
9.6.1 Bounded reachability analysis 184
9.7 Data structures for reachability analysis 186

9.7.1 Rectangles 186
9.7.2 Polytopes 189
9.7.3 Zonotopes 192
9.7.4 Ellipsoids 193
9.8 Exercises 194
10 Progress Analysis 197
10.1 Quick introduction to progress 197
10.2 Termination of discrete-time automata 198
10.2.1 Termination with well-founded relations 199
10.2.2 Example: UpDown counter 200
10.2.3 Termination with disjunctive well-founded relations 201
10.2.4 Example: UpDown revisited 202
10.3 Self-stabilization 202
10.3.1 Example: Distributed minimum spanning tree algorithm 203
10.3.2 Stabilization of the minimum spanning tree algorithm 205
10.4 Convergence and stability of asynchronous systems without metrics 206
10.4.1 Convergence for finite state systems 207
10.5 Stability proofs for dynamical systems 208
10.6 Stability of hybrid automata 210
10.6.1 Common Lyapunov functions 210
10.6.2 Multiple Lyapunov functions 211
10.6.3 Stability under slow switching: Average dwell time 212
10.7 Exercises 214
11 Data-Driven Verification 217
11.1 Quick introduction to data-driven safety verification 218
11.1.1 Discrepancy functions 218
11.1.2 BasicSimReach algorithm 219
11.1.3 Example: Moore-Greitzer jet engine 222
11.2 Computing discrepancy 222
11.2.1 Linear dynamical systems 223
11.2.2 Nonlinear dynamical systems: Optimization approaches 223
11.2.3 Nonlinear models: Local discrepancy 224
11.3 Hybrid system verification 226
11.3.1 C2E2 verification tool 227
11.3.2 Example: Reachability analysis for PulseGen || Oscillator with C2E2 228
11.4 Example: Powertrain control system 228
11.5 Verifying cyber-physical systems with incomplete models 229

11.5.1 Hybrid automata with black-box modules 230
11.5.2 Learning discrepancy from simulations 232
11.5.3 DryVR verification tool 235
11.6 Example: Analyzing risk in automatic emergency braking systems 236
11.7 Example: Autonomous spacecraft rendezvous 237
11.8 Further reading 241
11.8.1 Related approaches, software tools, and applications 241
11.8.2 Limitations and open problems 242
11.8.3 Falsification 243
11.8.4 Statistical model checking 243
11.8.5 Verification for machine learning modules 244
11.9 Exercises 244
Appendix A: Linear Algebra and Real Analysis 247
A.1 Sets and functions 247
A.1.1 Vectors and norms 247
A.1.2 Continuity and derivatives 248
A.1.3 Covers and partitions 248
A.2 Linear functions 249
A.3 Eigenvalues and eigenvectors 249
A.3.1 Positive definite matrices 250
A.3.2 Jordan normal form 250
A.3.3 Matrix norms 251
A.3.4 Interval matrices 251
A.4 Grönwall's inequality 252
Appendix B: Computability and Complexity 255
B.1 Computability 255
B.1.1 Turing machines 255
B.1.2 Configurations and computations 256
B.1.3 Language recognition and decidable languages 257
B.2 Complexity 258
B.2.1 Common complexity classes 258
B.3 Reasoning about the optimality of algorithms 260
B.3.1 Reductions 260
B.3.2 Completeness 261
Appendix C: Specification Language Reference 263
C.1 Conventions 263
C.2 Types 264
C.3 Formal arguments 264

C.4 Automaton declaration 265
C.5 Action declarations 265
C.6 Variables 266
C.7 Predicates and expressions 267
C.8 Transitions 267
C.9 Trajectories 268
C.10 Urgency 269
C.11 Modeling with nondeterminism 270

References 271

Index 291