
Contents

| | | |
|----------|--|-----------|
| 1 | Introduction and Basic Concepts | 1 |
| 1.1 | Two Approaches to Formal Reasoning | 3 |
| 1.1.1 | Proof by Deduction | 3 |
| 1.1.2 | Proof by Enumeration | 4 |
| 1.1.3 | Deduction and Enumeration | 5 |
| 1.2 | Basic Definitions | 5 |
| 1.3 | Normal Forms and Some of Their Properties | 8 |
| 1.4 | The Theoretical Point of View | 14 |
| 1.4.1 | The Problem We Solve | 17 |
| 1.4.2 | Our Presentation of Theories | 17 |
| 1.5 | Expressiveness vs. Decidability | 18 |
| 1.6 | Boolean Structure in Decision Problems | 19 |
| 1.7 | Problems | 21 |
| 1.8 | Glossary | 23 |
| 2 | Decision Procedures for Propositional Logic | 25 |
| 2.1 | Propositional Logic | 25 |
| 2.1.1 | Motivation | 25 |
| 2.2 | SAT Solvers | 27 |
| 2.2.1 | The Progress of SAT Solving | 27 |
| 2.2.2 | The DPLL Framework | 28 |
| 2.2.3 | BCP and the Implication Graph | 30 |
| 2.2.4 | Conflict Clauses and Resolution | 35 |
| 2.2.5 | Decision Heuristics | 39 |
| 2.2.6 | The Resolution Graph and the Unsatisfiable Core | 41 |
| 2.2.7 | SAT Solvers: Summary | 42 |
| 2.3 | Binary Decision Diagrams | 43 |
| 2.3.1 | From Binary Decision Trees to ROBDDs | 43 |
| 2.3.2 | Building BDDs from Formulas | 46 |
| 2.4 | Problems | 50 |
| 2.4.1 | Warm-up Exercises | 50 |

| | | |
|----------|---|-----------|
| 2.4.2 | Modeling | 50 |
| 2.4.3 | Complexity | 51 |
| 2.4.4 | DPLL SAT Solving | 52 |
| 2.4.5 | Related Problems | 52 |
| 2.4.6 | Binary Decision Diagrams | 53 |
| 2.5 | Bibliographic Notes | 54 |
| 2.6 | Glossary | 57 |
| 3 | Equality Logic and Uninterpreted Functions | 59 |
| 3.1 | Introduction | 59 |
| 3.1.1 | Complexity and Expressiveness | 59 |
| 3.1.2 | Boolean Variables | 60 |
| 3.1.3 | Removing the Constants: A Simplification | 60 |
| 3.2 | Uninterpreted Functions | 60 |
| 3.2.1 | How Uninterpreted Functions Are Used | 61 |
| 3.2.2 | An Example: Proving Equivalence of Programs | 63 |
| 3.3 | From Uninterpreted Functions to Equality Logic | 64 |
| 3.3.1 | Ackermann's Reduction | 66 |
| 3.3.2 | Bryant's Reduction | 69 |
| 3.4 | Functional Consistency Is Not Enough | 72 |
| 3.5 | Two Examples of the Use of Uninterpreted Functions | 74 |
| 3.5.1 | Proving Equivalence of Circuits | 75 |
| 3.5.2 | Verifying a Compilation Process with Translation Validation | 77 |
| 3.6 | Problems | 78 |
| 3.6.1 | Warm-up Exercises | 78 |
| 3.6.2 | Problems | 78 |
| 3.7 | Glossary | 79 |
| 4 | Decision Procedures for Equality Logic and Uninterpreted Functions | 81 |
| 4.1 | Congruence Closure | 81 |
| 4.2 | Basic Concepts | 83 |
| 4.3 | Simplifications of the Formula | 85 |
| 4.4 | A Graph-Based Reduction to Propositional Logic | 88 |
| 4.5 | Equalities and Small-Domain Instantiations | 92 |
| 4.5.1 | Some Simple Bounds | 93 |
| 4.5.2 | Graph-Based Domain Allocation | 94 |
| 4.5.3 | The Domain Allocation Algorithm | 96 |
| 4.5.4 | A Proof of Soundness | 98 |
| 4.5.5 | Summary | 101 |
| 4.6 | Ackermann's vs. Bryant's Reduction: Where Does It Matter? | 101 |
| 4.7 | Problems | 103 |
| 4.7.1 | Conjunctions of Equalities and Uninterpreted Functions | 103 |
| 4.7.2 | Reductions | 104 |

| | | |
|----------|--|------------|
| 4.7.3 | Complexity | 105 |
| 4.7.4 | Domain Allocation | 106 |
| 4.8 | Bibliographic Notes | 106 |
| 4.9 | Glossary | 108 |
| 5 | Linear Arithmetic | 111 |
| 5.1 | Introduction | 111 |
| 5.1.1 | Solvers for Linear Arithmetic | 112 |
| 5.2 | The Simplex Algorithm | 113 |
| 5.2.1 | Decision Problems and Linear Programs | 113 |
| 5.2.2 | Basics of the Simplex Algorithm | 114 |
| 5.2.3 | Simplex with Upper and Lower Bounds | 116 |
| 5.2.4 | Incremental Problems | 120 |
| 5.3 | The Branch and Bound Method | 120 |
| 5.3.1 | Cutting-Planes | 122 |
| 5.4 | Fourier–Motzkin Variable Elimination | 126 |
| 5.4.1 | Equality Constraints | 126 |
| 5.4.2 | Variable Elimination | 126 |
| 5.4.3 | Complexity | 129 |
| 5.5 | The Omega Test | 129 |
| 5.5.1 | Problem Description | 129 |
| 5.5.2 | Equality Constraints | 130 |
| 5.5.3 | Inequality Constraints | 132 |
| 5.6 | Preprocessing | 138 |
| 5.6.1 | Preprocessing of Linear Systems | 138 |
| 5.6.2 | Preprocessing of Integer Linear Systems | 139 |
| 5.7 | Difference Logic | 140 |
| 5.7.1 | Introduction | 140 |
| 5.7.2 | A Decision Procedure for Difference Logic | 142 |
| 5.8 | Problems | 142 |
| 5.8.1 | Warm-up Exercises | 142 |
| 5.8.2 | The Simplex Method | 143 |
| 5.8.3 | Integer Linear Systems | 143 |
| 5.8.4 | Omega Test | 144 |
| 5.8.5 | Difference Logic | 145 |
| 5.9 | Bibliographic Notes | 145 |
| 5.10 | Glossary | 146 |
| 6 | Bit Vectors | 149 |
| 6.1 | Bit-Vector Arithmetic | 149 |
| 6.1.1 | Syntax | 149 |
| 6.1.2 | Notation | 151 |
| 6.1.3 | Semantics | 152 |
| 6.2 | Deciding Bit-Vector Arithmetic with Flattening | 156 |
| 6.2.1 | Converting the Skeleton | 156 |

| | | |
|----------|--|------------|
| 6.2.2 | Arithmetic Operators | 157 |
| 6.3 | Incremental Bit Flattening | 160 |
| 6.3.1 | Some Operators Are Hard | 160 |
| 6.3.2 | Enforcing Functional Consistency | 162 |
| 6.4 | Using Solvers for Linear Arithmetic | 163 |
| 6.4.1 | Motivation | 163 |
| 6.4.2 | Integer Linear Arithmetic for Bit Vectors | 163 |
| 6.5 | Fixed-Point Arithmetic | 165 |
| 6.5.1 | Semantics | 165 |
| 6.5.2 | Flattening | 167 |
| 6.6 | Problems | 167 |
| 6.6.1 | Semantics | 167 |
| 6.6.2 | Bit-Level Encodings of Bit-Vector Arithmetic | 168 |
| 6.6.3 | Using Solvers for Linear Arithmetic | 169 |
| 6.7 | Bibliographic Notes | 169 |
| 6.8 | Glossary | 170 |
| 7 | Arrays | 171 |
| 7.1 | Introduction | 171 |
| 7.2 | Arrays as Uninterpreted Functions | 172 |
| 7.3 | A Reduction Algorithm for Array Logic | 175 |
| 7.3.1 | Array Properties | 175 |
| 7.3.2 | A Reduction Algorithm | 176 |
| 7.4 | Problems | 178 |
| 7.5 | Bibliographic Notes | 178 |
| 7.6 | Glossary | 179 |
| 8 | Pointer Logic | 181 |
| 8.1 | Introduction | 181 |
| 8.1.1 | Pointers and Their Applications | 181 |
| 8.1.2 | Dynamic Memory Allocation | 182 |
| 8.1.3 | Analysis of Programs with Pointers | 184 |
| 8.2 | A Simple Pointer Logic | 185 |
| 8.2.1 | Syntax | 185 |
| 8.2.2 | Semantics | 187 |
| 8.2.3 | Axiomatization of the Memory Model | 188 |
| 8.2.4 | Adding Structure Types | 189 |
| 8.3 | Modeling Heap-Allocated Data Structures | 190 |
| 8.3.1 | Lists | 190 |
| 8.3.2 | Trees | 191 |
| 8.4 | A Decision Procedure | 193 |
| 8.4.1 | Applying the Semantic Translation | 193 |
| 8.4.2 | Pure Variables | 195 |
| 8.4.3 | Partitioning the Memory | 196 |
| 8.5 | Rule-Based Decision Procedures | 197 |

| | | |
|-----------|--|------------|
| 8.5.1 | A Reachability Predicate for Linked Structures | 198 |
| 8.5.2 | Deciding Reachability Predicate Formulas | 199 |
| 8.6 | Problems | 202 |
| 8.6.1 | Pointer Formulas | 202 |
| 8.6.2 | Reachability Predicates | 203 |
| 8.7 | Bibliographic Notes | 204 |
| 8.8 | Glossary | 206 |
| 9 | Quantified Formulas | 207 |
| 9.1 | Introduction | 207 |
| 9.1.1 | Example: Quantified Boolean Formulas | 209 |
| 9.1.2 | Example: Quantified Disjunctive Linear Arithmetic | 211 |
| 9.2 | Quantifier Elimination | 211 |
| 9.2.1 | Prenex Normal Form | 211 |
| 9.2.2 | Quantifier Elimination Algorithms | 213 |
| 9.2.3 | Quantifier Elimination for Quantified Boolean Formulas | 214 |
| 9.2.4 | Quantifier Elimination for Quantified Disjunctive Linear Arithmetic | 217 |
| 9.3 | Search-Based Algorithms for QBF | 218 |
| 9.4 | Problems | 220 |
| 9.4.1 | Warm-up Exercises | 220 |
| 9.4.2 | QBF | 220 |
| 9.5 | Bibliographic Notes | 223 |
| 9.6 | Glossary | 224 |
| 10 | Deciding a Combination of Theories | 225 |
| 10.1 | Introduction | 225 |
| 10.2 | Preliminaries | 225 |
| 10.3 | The Nelson–Oppen Combination Procedure | 227 |
| 10.3.1 | Combining Convex Theories | 227 |
| 10.3.2 | Combining Nonconvex Theories | 230 |
| 10.3.3 | Proof of Correctness of the Nelson–Oppen Procedure | 233 |
| 10.4 | Problems | 236 |
| 10.5 | Bibliographic Notes | 236 |
| 10.6 | Glossary | 239 |
| 11 | Propositional Encodings | 241 |
| 11.1 | Overview | 241 |
| 11.2 | Lazy Encodings | 244 |
| 11.2.1 | Definitions and Notations | 244 |
| 11.2.2 | Building Propositional Encodings | 245 |
| 11.2.3 | Integration into DPLL | 246 |
| 11.2.4 | Theory Propagation and the DPLL(T) Framework | 246 |
| 11.2.5 | Some Implementation Details of DPLL(T) | 250 |
| 11.3 | Propositional Encodings with Proofs (Advanced) | 253 |

| | | |
|----------|---|-----|
| 11.3.1 | Encoding Proofs | 254 |
| 11.3.2 | Complete Proofs | 255 |
| 11.3.3 | Eager Encodings | 257 |
| 11.3.4 | Criteria for Complete Proofs | 258 |
| 11.3.5 | Algorithms for Generating Complete Proofs | 259 |
| 11.4 | Problems | 263 |
| 11.5 | Bibliographic Notes | 264 |
| 11.6 | Glossary | 267 |
| A | The SMT-LIB Initiative | 269 |
| B | A C++ Library for Developing Decision Procedures | 271 |
| B.1 | Introduction | 271 |
| B.2 | Graphs and Trees | 272 |
| B.2.1 | Adding “Payload” | 274 |
| B.3 | Parsing | 274 |
| B.3.1 | A Grammar for First-Order Logic | 274 |
| B.3.2 | The Problem File Format | 276 |
| B.3.3 | A Class for Storing Identifiers | 277 |
| B.3.4 | The Parse Tree | 277 |
| B.4 | CNF and SAT | 278 |
| B.4.1 | Generating CNF | 278 |
| B.4.2 | Converting the Propositional Skeleton | 281 |
| B.5 | A Template for a Lazy Decision Procedure | 281 |
| | References | 285 |
| | Index | 299 |



<http://www.springer.com/978-3-540-74104-6>

Decision Procedures

An Algorithmic Point of View

Kroening, D.; Strichman, O.

2008, XVI, 306 p., Hardcover

ISBN: 978-3-540-74104-6