HEURISTICS

Intelligent Search Strategies for Computer Problem Solving

Judea Pearl

Department of Computer Science
University of California
Los Angeles, California

ADDISON-WESLEY PUBLISHING COMPANY
Reading, Massachusetts • Menlo Park, California
London • Amsterdam • Don Mills, Ontario • Sydney
PART I
Problem-Solving Strategies and the Nature of Heuristic Information

1. Heuristics and Problem Representations
   1.1 Typical Uses of Heuristics in Problem Solving 3
       1.1.1 The 8-Queens Problem 4 / 1.1.2 The 8-Puzzle 6 / 1.1.3 The Road Map Problem 9 / 1.1.4 The Traveling Salesman Problem (TSP) 10 / 1.1.5 The Counterfeit Coin Problem 12
   1.2 Search Spaces and Problem Representations 14
       1.2.1 Optimizing, Satisficing, and Semi-Optimizing Tasks 14 / 1.2.2 Systematic Search and the Split-and-Prune Paradigm 15 / 1.2.3 State-Space Representation 20 / 1.2.4 Problem-Reduction Representations and AND/OR Graphs 21 / 1.2.5 Selecting a Representation 26
   1.3 Bibliographical and Historical Remarks 31
   Exercises 32

2. Basic Heuristic-Search Procedures
   2.1 Hill-Climbing: An Irrevocable Strategy 35
   2.2 Uninformed Systematic Search: Tentative Control Strategies 36
       2.2.1 Depth-First and Backtracking: LIFO Search Strategies 36 / 2.2.2 Breadth-First: A FIFO Search Strategy 42 / 2.2.3 Uninformed Search of AND/OR Graphs 44
   2.3 Informed, Best-First Search: A Way of Using Heuristic Information 46
       2.3.1 A Basic Best-First (BF) Strategy for State-Space Search 48 / 2.3.2 A General Best-First Strategy for AND/OR Graphs (GBF) 49
   2.4 Specialized Best-First Algorithms: Z*, A*, AO, and AO* 56
       2.4.1 Why Restrict the Evaluation Functions? 56 / 2.4.2 Recursive Weight Functions 57 / 2.4.3 Identifying G0, The Most Promising Solution-Base Graph 59 / 2.4.4 Specialized Best-First Strategies 61
# Contents

2.5 Hybrid Strategies 65

2.5.1 BF-BT Combinations 66 / 2.5.2 Introducing Irrevocable Decisions 68

2.6 Bibliographical and Historical Remarks 69

Exercises 71

3. Formal Properties of Heuristic Methods 73

3.1 $A^*$—Optimal Search for an Optimal Solution 75

3.1.1 Properties of $f^*$ 75 / 3.1.2 Termination and Completeness 76 / 3.1.3 Admissibility—A Guarantee for an Optimal Solution 77 / 3.1.4 Comparing the Pruning Power of Several Heuristics 79 / 3.1.5 Monotone (Consistent) Heuristics 82

3.2 Relaxing the Optimality Requirement 86

3.2.1 Adjusting the Weights of $g$ and $h$ 86 / 3.2.2 Two $\epsilon$-Admissible Speedup Versions of $A^*$ 88 / 3.2.3 $R_\epsilon^*$—A Limited Risk Algorithm Using Information about the Uncertainty of $h$ 90 / 3.2.4 $R_{\epsilon,n}$—A Speedup Version of $R_\epsilon^*$ 97

3.3 Some Extensions to Nonadditive Evaluation Functions ($BF^*$ and $GBF^* )$ 99

3.3.1 Notation and Preliminaries 100 / 3.3.2 Algorithmic Properties of Best-First Search $BF^*$ 103

3.4 Bibliographical and Historical Remarks 110

Exercises 112

4. Heuristics Viewed as Information Provided by Simplified Models 113

4.1 The Use of Relaxed Models 113

4.1.1 Where Do These Heuristics Come From? 113 / 4.1.2 Consistency of Relaxation-Based Heuristics 115 / 4.1.3 Overconstrained, Analogical, and Other Types of Auxiliary Models 116

4.2 Mechanical Generation of Admissible Heuristics 118

4.2.1 Systematic Relaxation 118 / 4.2.2 Can a Program Tell an Easy Problem When It Sees One? 121 / 4.2.3 Summary 123

4.3 Probability-Based Heuristics 124

4.3.1 Heuristics Based on the Most Likely Outcome 125 / 4.3.2 Heuristics Based on Sampling 126 / 4.3.3 Probability-Based Heuristics in the Service of Semi-Optimization Problems 128

4.4 Bibliographical and Historical Remarks 131

Exercises 133

PART II

Performance Analysis of Heuristic Methods

5. Abstract Models for Quantitative Performance Analysis 137

5.1 Mathematical Performance Analysis, or Test Tubes versus Fruit Flies in the Design of Gothic Cathedrals 137
PART III
Game-Playing Programs

8. Strategies and Models for Game-Playing Programs 221

8.1 Solving and Evaluating Games 222
  8.1.1 Game Trees and Game-Playing Strategies 222
  8.1.2 Bounded Look-Ahead and the Use of Evaluation Functions 226
  8.1.3 MIN-MAX versus NEG-MAX Notations 228

8.2 Basic Game-Searching Strategies 229
  8.2.1 Exhaustive Minimaxing and the Potential for Pruning 229
  8.2.2 The $\alpha$-$\beta$ Pruning Procedure: A Backtracking Strategy 231
  8.2.3 SSS*—A Best-First Search for an Optimal Playing Strategy 240
  8.2.4 SCOUT—A Cautious Test-Before-Evaluate Strategy 246

8.3 A Standard Probabilistic Model for Studying the Performance of Game-Searching Strategies 251
  8.3.1 The Probability of Winning a Standard Game with Random Win Positions 251
  8.3.2 Game Trees with an Arbitrary Distribution of Terminal Values 254
  8.3.3 The Mean Complexity of Solving a Standard $(d,b,P_0)$-game 259
  8.3.4 The Mean Complexity of Testing and Evaluating Multivalued Game Trees 268

8.4 Recreational Diversions 270
  8.4.1 The Board-Splitting Game—A Physical Embodiment of the Standard Game Tree 270
  8.4.2 Other Applications of the Minimax Convergence Theorem 273
  8.4.3 Games as Mazes with Hidden Paths: A Useful Metaphor 277

8.5 Bibliographical and Historical Remarks 285
Exercises 287

9. Performance Analysis for Game-Searching Strategies 288

9.1 The Expected Performance of SCOUT 289
  9.1.1 Games with Continuous Terminal Values 289
  9.1.2 Games with Discrete Terminal Values 292

9.2 The Expected Performance of $\alpha$-$\beta$ 293
  9.2.1 Historical Background 293
  9.2.2 An Integral Formula for $J_{\alpha-\beta}(d,b)$ 295
  9.2.3 The Branching Factor of $\alpha$-$\beta$ and Its Optimality 296
  9.2.4 How Powerful Is the $\alpha$-$\beta$ Pruning? 298

9.3 The Expected Performance of SSS* 300
  9.3.1 A Necessary and Sufficient Condition for Node Examination 300
  9.3.2 The Probability of Examining a Terminal Node 301
  9.3.3 The Expected Number of Terminal Nodes Examined by SSS* 303
  9.3.4 The Branching Factor of SSS* 304
  9.3.5 Numerical Comparison of the Performances of $\alpha$-$\beta$, SSS*, and SCOUT 300

9.4 The Benefit of Successor Ordering 310