Contents

1 **Introduction** 1

2 **Boosting Distributed Constraint Networks** 5
2.1 Introduction 5
2.2 Previous Work 6
2.3 Technical Background 8
2.3.1 Distributed Constraint Satisfaction Problems 8
2.3.2 DisCSP Algorithms 9
2.3.3 Performance of DisCSP Algorithms 10
2.4 Risks in Search 11
2.4.1 Randomization Risk 11
2.4.2 Selection Risk 13
2.5 Boosting Distributed Constraint Satisfaction 14
2.5.1 Utilizing Competition with Portfolios 15
2.5.2 Utilizing Cooperation with Aggregation 16
2.5.3 Categories of Knowledge 16
2.5.4 Interpretation of Knowledge 17
2.5.5 Implementation of the Knowledge Sharing Policies . . . 17
2.5.6 Complexity 18
2.6 Empirical Evaluation 20
2.6.1 Basic Performance 20
2.6.2 Randomization Risk 21
2.6.3 Selection Risk 22
2.6.4 Performance with Aggregation 23
2.6.5 Scalability 24
2.6.6 Idle Time 25
2.7 Summary 26

3 **Parallel Tree Search for Satisfiability** 27
3.1 Introduction 27
3.2 Previous Work 28

Bibliografische Informationen
http://d-nb.info/1041796900

digitalisiert durch DEUTSCHE NATIONAL BIBLIOTHEK

3.3 Technical Background 29
3.3.1 DPLL Search 30
3.3.2 Modern SAT Solvers 30
3.3.3 Multicore Architectures 31
3.3.4 AIMD Feedback Control-Based Algorithm 31
3.4 ManySAT: A Parallel SAT Solver 32
3.4.1 Restart Policies 32
3.4.2 Heuristic 34
3.4.3 Polarity 34
3.4.4 Learning 35
3.4.5 Clause Sharing 37
3.4.6 Summary 39
3.5 Evaluation 39
3.5.1 Performance Against a Sequential Algorithm 39
3.5.2 Performance Against Other Parallel SAT Solvers 41
3.6 Control-Based Clause Sharing 42
3.6.1 Throughput and Quality-Based Control Policies 43
3.6.2 Experiments 45
3.7 Summary 46

4 Parallel Local Search for Satisfiability 49
4.1 Introduction 49
4.2 Previous Work 50
4.2.1 Incomplete Methods for Parallel SAT 50
4.2.2 Cooperative Sequential Algorithms 50
4.3 Technical Background 51
4.3.1 Local Search for SAT 51
4.3.2 Refinements 52
4.4 Knowledge Sharing in Parallel Local Search for SAT 55
4.4.1 Using Best Known Configurations 55
4.4.2 Weighting Best Known Configurations 55
4.4.3 Restart Policy 56
4.5 Experiments 57
4.5.1 Experimental Settings 57
4.5.2 Practical Performances with Four Cores 58
4.5.3 Practical Performances with Eight Cores 62
4.5.4 Analysis of the Diversification/Intensification Trade-off . 63
4.5.5 Analysis of the Limitations of the Hardware 68
4.6 Summary 69

5 Learning Variable Dependencies 71
5.1 Introduction 71
5.2 Previous Work 71
5.3 Technical Background 72
5.4 Exploiting Weak Dependencies in Tree-Based Search 74
5.4.1 Weak Dependencies 74

5.4.2 Example . . . 74
5.4.3 Computing Weak Dependencies . . . 75
5.4.4 The domFD Dynamic Variable Ordering . . . 75
5.4.5 Complexities of domFD . . . 76
5.5 Experiments . . . 77
5.5.1 The Problems . . . 78
5.5.2 Searching for All Solutions or for an Optimal Solution . . . 78
5.5.3 Searching for a Solution with a Classical Branch-and-Prune Strategy . . . 79
5.5.4 Searching for a Solution with a Restart-Based Branch-and-Prune Strategy . . . 80
5.5.5 Synthesis . . . 81
5.6 Summary . . . 82

6 Continuous Search . . . 83
6.1 Introduction . . . 83
6.2 Related Work . . . 84
6.3 Technical Background . . . 85
6.3.1 Constraint Satisfaction Problems . . . 85
6.3.2 Supervised Machine Learning . . . 86
6.4 Continuous Search in Constraint Programming . . . 87
6.5 Dynamic Continuous Search . . . 88
6.5.1 Representing Instances: Feature Definition . . . 88
6.5.2 Feature Pre-processing . . . 90
6.5.3 Learning and Using the Heuristics Model . . . 90
6.5.4 Generating Examples in Exploration Mode . . . 91
6.5.5 Imbalanced Examples . . . 91
6.6 Experimental Validation . . . 92
6.6.1 Experimental Settings . . . 92
6.6.2 Practical Performances . . . 93
6.6.3 The Power of Adaptation . . . 96
6.7 Summary . . . 97

7 Autonomous Search . . . 99
7.1 Introduction . . . 99
7.2 Solver Architecture . . . 101
7.2.1 Problem Modeling/Encoding . . . 102
7.2.2 The Evaluation Function . . . 103
7.2.3 The Solving Algorithm . . . 103
7.2.4 Configuration of the Solver: The Parameters . . . 104
7.2.5 Control . . . 105
7.2.6 Existing Classifications and Taxonomies . . . 105
7.3 Architecture of Autonomous Solvers . . . 107
7.3.1 Control by Self-adaptation . . . 107
7.3.2 Control by Supervised Adaptation . . . 108
7.3.3 Searching for a Solution vs. Solutions for Searching . . . 108

7.3.4 A Rule-Based Characterization of Solvers 109
7.4 Case Studies 115
7.4.1 Tuning Before Solving 115
7.4.2 Control During Solving 117
7.4.3 Control During Solving in Parallel Search 121
7.5 Summary 122

8 Conclusion and Perspectives 123

References 127