

Contents

Table of Contents	6
List of Figures	9
List of Tables	11
I Introduction	12
1 Preamble	13
1.1 Motivation	13
1.2 Contributions of the thesis	15
1.3 Structure of the thesis	16
2 Debugging Techniques	17
2.1 Trace Debugging	17
2.2 Omniscient Debugging	18
2.3 Abstract Diagnosis	18
2.4 Abstract Debugging	19
2.5 Delta Debugging	19
II Foundations	22
3 Algorithmic Debugging from the User’s Perspective	23
4 Preliminary Definitions and Notation	27
4.1 Execution Tree Nodes	28
4.2 Execution Tree, Marked Execution Tree	28
4.3 Algorithmic Debugging search strategies	31
4.3.1 Top-Down strategies	31
4.3.2 Divide & Query strategies	31
III Algorithmic Debugging	32
5 Algorithmic Debugging Techniques	33
5.1 Virtual Execution Trees	33
5.1.1 Introduction	33
5.1.2 A new architecture for Algorithmic Debugging	34

5.1.3	Implementation	39
5.2	Balancing Execution Trees	41
5.2.1	Introduction	41
5.2.2	Preliminary definitions	42
5.2.3	Collapsing and projecting nodes	43
5.2.4	Correctness	49
5.2.5	Implementation	51
5.2.6	Related work	52
5.2.7	Proofs of technical results	52
5.2.8	Case of study: Mergesort	55
5.3	Loops to recursion	59
5.3.1	Introduction	59
5.3.2	Transforming loops into recursive methods	62
5.3.3	Treatment of <i>break</i> and <i>continue</i> statements	70
5.3.4	Handling exceptions	73
5.3.5	Treatment of recursion and the return and goto statements	75
5.3.6	Implementation as a Java library, optimizations and empirical evaluation	76
5.3.7	Related work	81
5.3.8	Proofs of technical results	83
5.4	Loop Expansion	89
5.4.1	Introduction	89
5.4.2	Preliminaries	90
5.4.3	Loop Expansion optimization	90
5.4.4	Correctness	91
5.4.5	Related Work	93
5.5	Tree Compression	95
5.5.1	Introduction	95
5.5.2	Preliminaries	96
5.5.3	Tree Compression Optimization	96
5.6	Combining Loop Expansion and Tree Compression	100
6	Algorithmic Debugging Search Strategies	102
6.1	Divide & Query	102
6.1.1	Shapiro vs Hirunkitty	102
6.1.2	Limitations of Divide & Query	103
6.1.3	Divide & Query is not an optimal search strategy	104
6.2	Optimal Divide & Query	108
6.2.1	Introduction	108
6.2.2	Preliminary definitions	109
6.2.3	Debugging METs where all nodes have the same individual weight	109
6.2.4	Debugging METs where nodes can have different individual weights (including zero)	113
6.2.5	Debugging METs where nodes can have different individual weights (excluding zero)	113
6.2.6	Proofs of technical results	115
6.3	Implementation of Optimal Divide & Query	126
6.3.1	Introduction	126
6.3.2	Optimal Divide & Query	126
6.3.3	Implementation of Optimal Divide & Query	127
6.3.4	Standard architecture	130
6.3.5	Static MET architecture	130

6.3.6	Dynamic MET architecture	132
6.4	Divide by Queries	134
6.4.1	Introduction	134
6.4.2	Decidability	134
6.4.3	Valid sequence of questions	135
6.4.4	Optimal sequence of questions	136
7	Implementations	139
7.1	Declarative Debugger for Java (DDJ)	139
7.1.1	Introduction	139
7.1.2	Tool description	140
7.1.3	Usage scenario	143
7.1.4	Tool information	146
7.2	Hybrid Debugger for Java (HDJ)	147
7.2.1	Introduction	147
7.2.2	Debugging techniques	147
7.2.3	Hybrid Debugging	148
7.2.4	Implementation and empirical evaluation	152
7.2.5	Related work	153
8	Reformulation of Algorithmic Debugging	155
8.1	Some problems identified in current algorithmic debuggers	155
8.2	Paradigm-independent redefinition of Algorithmic Debugging	156
8.2.1	The Execution Tree	157
8.2.2	The Routine Tree	160
8.2.3	Search strategies for Algorithmic Debugging	161
8.2.4	Algorithmic Debugging transformations	161
8.2.5	An Algorithmic Debugging scheme	163
8.3	Related work	164
8.3.1	A little bit of history	164
8.3.2	Modern implementations	164
IV	Conclusions and Future Work	165
9	Conclusions	166
10	Open Lines of Research	169
	Bibliography	171