

Contents

I	Introduction and Objectives	1
1	Introduction	2
1.1	The Erlang Language	2
1.2	Constraint-Based Testing	3
1.3	Reversible Debugging	5
1.4	Objectives and Contributions	6
1.5	Structure of this Thesis	9
1.6	Publications	9
1.7	Research Projects	11
1.8	Research Stays	12
II	Selected Papers	13
2	Reversible Computation in Term Rewriting	14
2.1	Introduction	15
2.2	Preliminaries	17
2.2.1	Terms and Substitutions	18
2.2.2	Term Rewriting Systems	18
2.2.3	Conditional Term Rewrite Systems	19
2.3	Reversible Term Rewriting	20
2.3.1	Unconditional Term Rewrite Systems	21
2.3.2	Conditional Term Rewrite Systems	25
2.4	Removing Positions from Traces	31

2.5	Reversibilization	38
2.5.1	Injectivization	39
2.5.2	Inversion	42
2.5.3	Improving the transformation for injective functions	44
2.6	Bidirectional Program Transformation	47
2.7	Related Work	50
2.8	Discussion and Future Work	52
3	A Theory of Reversibility for Erlang	54
3.1	Introduction	55
3.2	Language Syntax	57
3.3	The Language Semantics	59
3.3.1	Erlang Concurrency	65
3.4	A Reversible Semantics for Erlang	69
3.4.1	Properties of the Uncontrolled Reversible Semantics	76
3.5	Rollback Semantics	88
3.6	Proof-of-concept Implementation of the Reversible Semantics	97
3.7	Related Work	99
3.8	Conclusion and Future Work	101
4	CauDER: A Causal-Consistent Reversible Debugger for Erlang	102
4.1	Introduction	103
4.2	The Language	104
4.3	Causal-Consistent Reversible Debugging	108
4.4	CauDER: A Causal-Consistent Reversible Debugger	112
4.4.1	The CauDER Workflow	115
4.4.2	Finding Concurrency Bugs with CauDER	115
4.5	Related Work	118
4.6	Discussion	119
5	Causal-Consistent Replay Debugging for Message Passing Programs	121
5.1	Introduction	122
5.2	The Language	123
5.3	Logging Computations.	130
5.4	A Causal-Consistent Replay Semantics	132
5.5	Controlled Replay Semantics	136
5.6	Related Work and Conclusion	139

6	Concolic Execution in Functional Programming by Program Instrumentation	141
6.1	Introduction	142
6.2	The Language	143
6.3	Instrumented Semantics	146
6.4	Program Instrumentation	151
6.5	Concolic Execution	156
6.6	Discussion	158
7	Property-Based Test Case Generators for Free	159
7.1	Introduction	160
7.2	Preliminaries	161
7.3	A Framework for PBT of Erlang Programs	163
7.4	Type-Based Value Generation	167
7.5	The Interpreter of Filter Functions	169
7.6	Coroutining the Type-Based Generator and the Filter Interpreter . .	171
7.7	Experimental evaluation	175
7.8	Related Work	177
7.9	Conclusions	180
III	General Discussion of Results	183
8	Discussion	184
8.1	Reversible Term Rewriting	184
8.2	Reversible Debugging	186
8.3	Constraint-based Testing	191
IV	Conclusions	195
9	Conclusions and Future Work	196
9.1	Conclusions	196
9.2	Future Work	198
9.2.1	Reversible Term Rewriting	199
9.2.2	Reversible Debugging	199
9.2.3	Constraint-Based Testing	200