

---

# Contents

<b>1</b>	<b>Introduction</b> .....	1
1.1	Formal Verification .....	1
1.2	Model Checking of Infinite State Space Systems .....	4
1.3	Goals and Contributions of the Thesis .....	6
1.4	Structure of the Thesis .....	7
<b>2</b>	<b>Regular Model Checking</b> .....	9
2.1	Finite Automata and Transducers .....	10
2.2	Regular Model Checking .....	11
2.2.1	Example: Token Passing .....	11
2.2.2	Acceleration techniques .....	12
2.3	Abstract Regular Model Checking .....	15
2.3.1	Abstraction Based on Automata State Equivalence .....	15
2.3.2	Configuration-Oriented Abstractions .....	17
2.3.3	Use of ARMC in Verification .....	18
2.4	Summary of RMC .....	18
<b>3</b>	<b>Regular Tree Model Checking</b> .....	21
3.1	Regular Tree Languages and Transducers .....	22
3.2	Basics of Regular Tree Model Checking .....	24
3.2.1	Example: Tree Token Passing .....	25
3.2.2	Acceleration Techniques .....	25
3.3	Abstract Regular Tree Model Checking .....	27
3.3.1	Abstraction Based on Automata State Equivalence .....	28
3.3.2	Abstraction Based on Languages of Finite Height .....	28
3.3.3	Abstraction Based on Predicate Languages .....	28
3.4	Implementation and Experimental Results .....	30
3.4.1	Algorithmic Decomposition of Bottom-up Tree Transducers .....	30
3.4.2	Verified protocols .....	31
3.5	Summary of RTMC .....	33

<b>4</b>	<b>Abstract Regular Tree Model Checking of Complex Dynamic Data Structures</b> . . . . .	35
4.1	Existing Alternative Approaches . . . . .	36
4.1.1	Techniques Based on Logics . . . . .	37
4.1.2	Automata-Based Techniques . . . . .	40
4.1.3	Other Techniques . . . . .	42
4.2	ARTMC of Programs with Recursive Data Structures: The Basic Framework . . . . .	44
4.2.1	The Considered Programs . . . . .	45
4.2.2	The Considered Properties . . . . .	46
4.2.3	The Verification Problem . . . . .	47
4.3	Logic of Bad Memory Patterns . . . . .	47
4.3.1	Syntax of LBMP . . . . .	47
4.3.2	Semantics of LBMP . . . . .	48
4.3.3	An Example: Doubly-Linked Lists . . . . .	49
4.3.4	Evaluating LBMP Specifications . . . . .	49
4.4	Tree Automata Encoding of Pointer Manipulating Programs . . . . .	50
4.4.1	Encoding of Sets of Memory Configurations . . . . .	50
4.4.2	Tree Memory Configurations in Mona . . . . .	53
4.4.3	Encoding Program Statements as Tree Transducers . . . . .	54
4.4.4	Verification of Programs with Pointers using ARTMC . . . . .	56
4.5	Implementation and Experimental Results . . . . .	58
4.5.1	An ARTMC Tool for Tree Automata Memory Encodings . . . . .	58
4.5.2	Case Studies . . . . .	59
4.5.3	Experimental Results . . . . .	64
4.6	Implementation Details . . . . .	64
4.6.1	Mona . . . . .	65
4.6.2	Transduction . . . . .	67
4.6.3	Encoding Memory Configuration in Binary . . . . .	67
4.6.4	Correctness of Tree Memory Encoding After an Abstraction . . . . .	69
4.6.5	Implementation of the Finite-Height Abstraction . . . . .	71
4.6.6	Implementation of Predicate-Based Abstraction . . . . .	72
4.6.7	Implementation of Routing Expressions . . . . .	74
4.7	Garbage Checking . . . . .	76
4.8	Summary of ARTMC for Dynamic Data Structures . . . . .	77
<b>5</b>	<b>Proving Termination of Tree Manipulating Programs</b> . . . . .	79
5.1	Existing Approaches . . . . .	81
5.2	The Notions Used . . . . .	83
5.3	Programs with Trees . . . . .	85
5.3.1	Program Preprocessing . . . . .	87
5.4	The Termination Analysis Loop . . . . .	89
5.5	Using ARTMC in the Proposed Framework . . . . .	91

5.6	Abstraction of Programs with Trees into Counter Automata ..	94
5.6.1	Abstract Control Flow Graphs .....	95
5.6.2	Translation to Counter Automata .....	96
5.7	Checking Spuriousness of Counterexamples .....	99
5.7.1	Deciding Spuriousness of Lassos without Destructive Updates .....	102
5.7.2	Checking Spuriousness of Lassos with Destructive Updates .....	104
5.8	Abstraction Refinement .....	105
5.9	Implementation and Experimental Results .....	108
5.9.1	Necessity of Refinement .....	110
5.10	A Support for the Insert and Delete Statements .....	111
5.11	Summary of Termination for Tree Manipulating Programs .....	113
<b>6</b>	<b>Conclusions and Future Directions .....</b>	<b>115</b>
	<b>References .....</b>	<b>119</b>
	<b>Proofs .....</b>	<b>129</b>
A.1	Proof of Lemma 1 .....	129
A.2	Proof of Theorem 2 .....	129
A.3	Proof of Theorem 4 .....	130
A.4	Proof of Theorem 5 .....	131
A.5	Proof of Corollary 1 .....	133