

Contents

1	Introduction	1
1.1	Goals of the Work	2
1.2	An Overview of Achieved Results	3
1.3	Plan of the Rest of the Text	6
2	Preliminaries	7
2.1	Relations	7
2.2	Labelled Transition Systems and Finite Automata	7
2.3	Forward and Backward Simulations	8
2.4	Trees and Tree Automata	8
2.5	Regular Tree Model Checking	9
3	Computing Simulations over Labelled Transition Systems	13
3.1	Preliminaries	14
3.2	The LTS Simulation Algorithm	15
3.2.1	Correctness of the Algorithm	16
3.2.2	Implementation and Complexity of the Algorithm	22
3.3	Conclusions and Future Work	25
4	Simulation-based Reduction of Tree Automata	27
4.1	Tree Automata Simulations and Bisimulations	30
4.1.1	Downward and Upward Simulation	30
4.1.2	Downward and Upward Bisimulation	32
4.2	Combined Relations for Quotienting	33
4.2.1	Runs and Simulations	34
4.2.2	Mediated Preorder	36
4.2.3	Quotienting with Mediated Equivalence	38
4.3	Variants of the Combined Relation	40
4.4	Computing the Proposed Relations	46
4.4.1	Computing Downward Simulation	46
4.4.2	Complexity of Computing Downward Simulation	48

4.4.3	Computing Upward Simulation	51
4.4.4	Complexity of Computing Upward Simulation	53
4.4.5	Computing Downward Bisimulation Equivalences	58
4.4.6	Computing Upward Bisimulation Equivalences	59
4.4.7	Computing the Combined Relations	61
4.5	Experiments	61
4.6	Conclusions and Future Work	63
5	Language Inclusion and Universality of Finite (Tree)	
	Automata	65
5.1	Universality and Language Inclusion of FA	68
5.1.1	Universality of FA	68
5.1.2	Correctness of the Optimised Universality Checking	71
5.1.3	The FA Language Inclusion Problem	73
5.2	Universality and Language Inclusion of Tree Automata	76
5.2.1	The Role of Upward Simulation	77
5.2.2	Tree Automata Universality Checking	77
5.2.3	Correctness of the TA Universality Checking	78
5.2.4	Downward Universality Checking with Antichains	80
5.2.5	Tree Automata Language Inclusion Checking	81
5.3	Experiments with Classical versus Pure Antichain Algorithms for Tree Automata	83
5.3.1	Experiments with Antichain-based Universality Checking	84
5.3.2	Experiments with Antichain-based Inclusion Checking	84
5.3.3	Experiments with Regular Tree Model Checking	86
5.4	Experiments with Pure versus Simulation Enhanced Antichain Algorithms	90
5.4.1	Experiments on FA	90
5.4.2	Experiments on TA	92
5.5	Conclusions and Future Work	94
6	Simulation-based Reduction of Alternating Büchi Automata	97
6.1	Basic Definitions	98
6.2	Simulation Relations	99
6.2.1	Runs and Simulations	101
6.3	Mediated Equivalence and Quotienting	103
6.3.1	The Notion and Intuition of Mediated Equivalence	103
6.3.2	Quotienting Automata According to Mediated Equivalence Preserves Language	107
6.4	Computing the Relations	118
6.4.1	Computing Backward Simulation	119
6.4.2	Complexity of Computing Backward Simulation	121
6.5	Experimental Results	123
6.6	Conclusion and Future Work	125

7	Conclusions and Future Directions	127
7.1	A Summary of the Contributions	127
7.2	Further Directions	128
7.3	Publications Related to this Work	129
References		131

Finite automata on finite words (FA) are one of the basic concepts of computer science. Besides classical applications of FA such as compiler construction or text searching, FA are widely used in modeling and verification, which are the application domains of our interest. Tree automata (TA) are a natural generalisation of FA that accepts ordered trees/terms. TA share most of the good properties of FA, from closure to decidability and complexity (even though complexities of many tree automata problems are higher, they are still comparable with the complexity of the corresponding FA ones). This makes tree automata a convenient tool for modelling and reasoning about various kinds of structured objects such as syntactical trees, structured documents, configurations of complex systems, algebraic term representations of data or computations, etc. (see, e.g., [JEG⁺97]). One of the main motivations for this work is in particular the use of tree automata in verification, mainly in the method of regular tree model checking [Sha91, BT02, Ald90, BHRV06], an infinite-state system verification method where tree automata are used for representing sets of reachable states of a system.

In the above context, checking language equivalence/inclusion and reducing size of automata while preserving the language are fundamental issues, and performing these operations efficiently is crucial in practice. The language inclusion problem and the minimisation problem for (nondeterministic) automata are PSPACE-complete for FA and even EXPTIME-complete for TA. A classical approach to cope with these problems is determinisation. Both FA as well as TA can be determinised and minimised in a canonical way. Testing language inclusion of deterministic minimal automata is then easy. However, since even the canonical minimal deterministic automaton can still be exponentially larger than the original nondeterministic one, its computation easily becomes a major bottleneck of any automata-based method.

A reasonable and pragmatic approach to the size reduction and language inclusion problem is to consider some relation on states of an automaton that respects language inclusion on states, but which can be checked efficiently, using a polynomial algorithm. Such a relation can then be used for approx-