A Uniform (Bi-)Simulation-Based Framework for Reducing Tree Automata

FIT BUT Technical Report Series

Parosh A. Abdulla, Lukáš Holík, Lisa Kaati, and Tomáš Vojnar

Technical Report No. FIT-TR-2008-005
Faculty of Information Technology, Brno University of Technology

Last modified: July 14, 2008

A Uniform (Bi-)Simulation-Based Framework for Reducing Tree Automata*

Parosh A. Abdulla[1], Lukáš Holík[2], Lisa Kaati[1], and Tomáš Vojnar[2]

[1] Department of Information Technology, University of Uppsala, Sweden
email: {parosh,lisa.kaati}@it.uu.se
[2] Faculty of Information Technology, Brno University of Technology, Brno, Czech Republic
email: {holik,vojnar}@fit.vutbr.cz

Abstract. In this paper, we address the problem of reducing the size of nondeterministic (bottom-up) tree automata. We propose a uniform framework that allows for combining various upward and downward bisimulation and simulation relations in order to obtain a language-preserving combined relation suitable for reducing tree automata without a need to determinise them. The framework generalises and extends several previous works and provides a broad spectrum of different relations yielding a possibility of a fine choice between the amount of reduction and the computational demands. We, moreover, provide a significantly improved way of computing the various combined (bi-)simulation relations. We analyse properties of the considered relations both theoretically as well as through a series of experiments.

1 Introduction

Finite tree automata are a natural generalisation of word automata. Since trees (or terms) appear in many areas of computer science and engineering, tree automata are quite broadly applicable—including, for instance, applications in XML manipulation, natural language processing, or formal verification. In most of these applications, dealing with as small automata as possible is highly desirable. In order to reduce the size of a given tree automaton, one can always try to determinise and minimise it. However, the determinisation may lead to an exponential blow-up in the size, and even the minimal deterministic automaton might still be bigger than the original nondeterministic automaton. Moreover, even if the minimal deterministic automaton is really small, it might be impossible to compute it due to the very expensive determinisation step.

An alternative way to reduce a given (nondeterministic) tree automaton is to identify a suitable, language-preserving equivalence relation over its states and collapse those states that are equal according to this relation. As in the case of word automata, good candidates for such relations are various bisimulations and simulation equivalences. In particular, the so-called forward and backward bisimulations and simulation equivalences are well known to be useful when reducing the size of word automata. In this paper, we deal with their tree automata extensions—the so-called *downward* and *upward bisimulations* and *simulation equivalences*.

The *downward (bi-)simulations*, which straightforwardly generalise the appropriate backward (bi-)simulations from word automata to (bottom-up) tree automata, are compatible with the language inclusion preorder. That is, if a state r downward (bi-)simulates a state q, then the language accepted by q is a subset of the language accepted by r. Therefore, these relations are a natural choice for reducing the size of tree automata.

The *upward (bi-)simulations* are not compatible with the language inclusion preorder. Instead, they are compatible with the inclusion of the so-called *context languages*, where a context of a state q arises from a tree accepted at q by replacing some of its leaves by a "hole". It can, however, be shown that when we restrict ourselves to upward (bi-)simulations compatible with the set of final states of automata, the downward and upward (bi-)simulations can be *combined* in such a way that they yield

* The work was supported by the Czech Grant Agency (projects 102/07/0322 and 102/05/H050), the Czech-French Barrande project 2-06-27, and the Czech Ministry of Education by the project MSM 0021630528 *Security-Oriented Research in Information Technology*.

a language-compatible equivalence. In the worst case, the combined relation is as coarse as the appropriate downward (bi-)simulation equivalence, but according to our practical experiments, it usually leads to significantly better reductions of the automata.

Tree bisimulations can be computed efficiently in time $o(\hat{r}^2\ m\ log\ n)$ where $\hat{r}$ is the maximal rank of the input symbols, m the size of the transition table, and n the number of states of the given tree automaton [3, 10, 1]. However, the reduction obtained by using bisimulations is often limited. Tree simulations are weaker than bisimulations and hence offer a better reduction. On the other hand, despite the recent advances in algorithms devised for computing them [2], they are significantly more expensive to compute. The time complexity of computing simulation preorders is roughly in $o(\ell\ \hat{r}^2\ m^2)$ where ℓ is the size of the alphabet.

In this paper, we propose a novel notion of upward simulations and bisimulations *parameterised* by an *inducing* downward simulation or bisimulation (in any possible combination). Moreover, we introduce a new operator—called a *weakening combination operator*—for combining such upward and downward simulation and bisimulation relations on tree automata. This way, we obtain a *uniform tree (bi-)simulation framework* which brings in several significant advantages.

First, the proposed framework allows one to combine not only maximal downward and upward simulations or maximal downward and upward bisimulations as considered in previous works [2, 1], but to combine any inducing downward simulation (i.e., also a downward bisimulation or, e.g., the identity relation) with any induced upward simulation (i.e., also a bisimulation or identity). This way, we explain in a uniform way several previous results [10, 2, 1] and, moreover, we obtain several *new combined relations* suitable for reducing tree automata. The use of such relations mixing in various ways the advantages of simulations and bisimulations allows the user to *fine-tune* the ratio between the possible reductions and their cost.

We carefully analyse mutual relationships of the various considered relations. We establish a certain *partial ordering* between their reduction capabilities, but we also show that many of them have an *incomparable* reduction power.

Further, compared to the previous works, the newly proposed combination operator brings in also a significantly improved way of computing the combined (bi-)simulation relations. Before the combination was accomplished by randomly looking for some combined relation satisfying the needed requirements. The newly proposed combination operator computes a *maximal combined relation*, which we show to be unique. The algorithm that we propose for this purpose turns out to be quite simple and runs in time $O(n^3)$ (or, in fact, even in a slightly better time). The use of the maximal combined relations turns out to itself give much better results in our experiments than the previously used random combination algorithms. Let us also note that the notion of combined (bi-)simulation relations that we propose is applicable even for word automata as for a special case of tree automata.

In order to experimentally examine the broad spectrum of relations offered by our framework, we implemented a prototype tool in which we have performed thorough experiments with tree automata from the domain of formal verification of infinite-state systems based on the so-called regular tree model checking and abstract regular tree model checking [8, 4, 6, 7]. Our experimental results confirm that we have obtained a broad range of algorithms for reducing tree automata, differing in their computation complexity and reduction capabilities.

Related work. Several algorithms for reducing the size of non-deterministic tree automata while preserving their language have been proposed in the literature. The first attempt was done in [3] where an algorithm inspired by the partition refinement algorithm by Paige and Tarjan [12] was presented. In [10], two different types of bisimulations—a backward and forward bisimulation—were presented. These bisimulations turn out to be special cases of the relations arising in our framework.

Efficient algorithms for computing simulation equivalences over tree automata have then been discussed in [2] together with a proposal of combined simulation relations. In [1], the ideas from [2] were extended to work for bisimulations. In this paper, we combine these previous works in a framework which allows us to explain them in a uniform way and, moreover, to obtain multiple new relations applicable for reducing tree automata. The obtained framework allows the user to mix advantages of

simulation and bisimulation approaches in a degree suitable for a given scenario. The way we use for computing the upward and downward relations that we are dealing with is inspired by the approach of [2] and [1]. We, however, provide a new and significantly improved way of combining these relations via the newly proposed weakening combination operator.

Plan of the paper. The rest of the paper is organised as follows. We start with some preliminaries in Section 2. In Section 3, we introduce the notions of downward (bi-)simulations, and propose the notion of parameterised upward (bi-)simulations. Subsequently, in Section 4, we propose the weakening combination operator and its use for obtaining combined relations suitable for reducing tree automata. Next, Section 5 analyses properties of the most interesting possible combined relations. Section 6 overviews algorithms usable for computing downward and upward (bi-)simulations and proposes an algorithm for computing their combinations. In Section 7, we give an experimental evaluation of using the obtained spectrum of relations for reducing tree automata. Finally, we summarise the paper and discuss possible future work in Section 8.

2 Preliminaries

In this section, we introduce some preliminaries on relations, trees, and tree automata.

Trees. A *ranked alphabet* Σ is a set of symbols together with a function $\# : \Sigma \to \mathbb{N}$. For $f \in \Sigma$, the value $\#(f)$ is called the *rank* of f. For any $n \geq 0$, we denote by Σ_n the set of all symbols of rank n from Σ. Let ε denote the empty sequence. A *tree* t over a ranked alphabet Σ is a partial mapping $t : \mathbb{N}^* \to \Sigma$ that satisfies the following conditions:

- $dom(t)$ is a finite, prefix-closed subset of $\mathbb{N}^*$, and
- for each $p \in dom(t)$, if $\#(t(p)) = n \geq 0$, then $\{i \mid pi \in dom(t)\} = \{1, \ldots, n\}$.

Each sequence $p \in dom(t)$ is called a *node* of t. For a node p, we define the i^{th} *child* of p to be the node pi, and the i^{th} *subtree* of p to be the tree t' such that $t'(p') = t(pip')$ for all $p' \in \mathbb{N}^*$. A *leaf* of t is a node p which does not have any children, i.e., there is no $i \in \mathbb{N}$ with $pi \in dom(t)$. We denote by $T(\Sigma)$ the set of all trees over the alphabet Σ.

Tree Automata. A (finite, non-deterministic, bottom-up) *tree automaton* (abbreviated as TA in the following) is a quadruple $A = (Q, \Sigma, \Delta, F)$ where Q is a finite set of states, $F \subseteq Q$ is a set of final states, Σ is a ranked alphabet, and Δ is a set of transition rules. Each transition rule is a triple of the form $((q_1, \ldots, q_n), f, q)$ where $q_1, \ldots, q_n, q \in Q$, $f \in \Sigma$, and $\#(f) = n$. We use $(q_1, \ldots, q_n) \xrightarrow{f} q$ to denote that $((q_1, \ldots, q_n), f, q) \in \Delta$. In the special case where $n = 0$, we speak about the so-called *leaf rules*, which we sometimes abbreviate as $\xrightarrow{f} q$. We use $Lhs(A)$ to denote the set of *left-hand sides* of rules, i.e., the set of tuples of the form $(q_1, \ldots, q_n)$ where $(q_1, \ldots, q_n) \xrightarrow{f} q$ for some f and q. Finally, we denote by $\hat{r}(A)$ the smallest $n \in \mathbb{N}$ such that $n \geq m$ for each $m \in \mathbb{N}$ where $(q_1, \ldots, q_m) \in Lhs(A)$ for some $q_i \in Q$, $1 \leq i \leq m$. We omit the reference to A if no confusion may arise.

A *run* of A over a tree $t \in T(\Sigma)$ is a mapping $\pi : dom(t) \to Q$ such that, for each node $p \in dom(t)$ where $q = \pi(p)$, if $q_i = \pi(pi)$ for $1 \leq i \leq n$, then Δ has a rule $(q_1, \ldots, q_n) \xrightarrow{t(p)} q$. We write $t \xRightarrow{\pi} q$ to denote that π is a run of A over t such that $\pi(\varepsilon) = q$. We use $t \Longrightarrow q$ to denote that $t \xRightarrow{\pi} q$ for some run π. The *language* of a state q is defined by $L(q) = \{t \mid t \Longrightarrow q\}$, while the *language* of A is defined by $L(A) = \bigcup_{q \in F} L(q)$.

An *environment* is a tuple of the form $((q_1, \ldots, q_{i-1}, \Box, q_{i+1}, \ldots, q_n), f, q)$ obtained by removing a state q_i, $1 \leq i \leq n$, from the i^{th} position of the left hand side of a rule $((q_1, \ldots, q_{i-1}, q_i, q_{i+1}, \ldots, q_n), f, q)$, and by replacing it by a special symbol $\Box \notin Q$ (called a *hole* below). Like for transition rules, we write $(q_1, \ldots, \Box, \ldots, q_n) \xrightarrow{f} q$ provided $((q_1, \ldots, q_{i-1}, q_i, q_{i+1}, \ldots, q_n), f, q) \in \Delta$ for some $q_i \in Q$. Sometimes, we also write the environment as $(q_1, \ldots, \Box_i, \ldots, q_n) \xrightarrow{f} q$ to emphasise that the hole is at position i. We denote the set of all environments of A by $Env(A)$ and we will drop the reference to A if no confusion may arise.

Relations. For an equivalence relation $\equiv$ defined on a set Q, we call each equivalence class of $\equiv$ a *block*, and use $Q/\equiv$ to denote the set of blocks in $\equiv$. For a preorder P, we will denote $\equiv_P$ the maximal equivalence included in P.

Quotient Tree Automata. The idea of reducing the size of an automaton is to identify suitable equivalence relations on its states, and then collapse the sets of states which form equivalence classes. Consider a TA $A = (Q, \Sigma, \Delta, F)$ and an equivalence relation $\equiv$ on Q. The *quotient tree automaton* derived from A and $\equiv$ is $A_{\equiv} = (Q_{\equiv}, \Sigma, \Delta_{\equiv}, F_{\equiv})$ where:

- $Q_{\equiv}$ is the set of blocks in $\equiv$. In other words, we collapse all states which belong to the same block into one state of the quotient automaton.
- $(\mathsf{B}_1, \ldots, \mathsf{B}_n) \xrightarrow{f} \mathsf{B}$ iff $(q_1, \ldots, q_n) \xrightarrow{f} q$ for some $q_1 \in \mathsf{B}_1, \ldots, q_n \in \mathsf{B}_n, q \in \mathsf{B}$. This is, there is a transition in the quotient automaton iff there is a transition between states in the corresponding blocks in the original TA.
- $F_{\equiv}$ contains a block B iff $\mathsf{B} \cap F \neq \emptyset$. Intuitively, a block is accepting if it contains a state which is accepting.

3 (Bi-)Simulations on Tree Automata

We now present definitions of downward (bi-)simulations and subsequently, we propose a notion of upward (bi-)simulations *parameterised* by a downward simulation as one of the cornerstones of our framework. We will call a downward simulation that is used as a parameter of an upward (bi-)simulation an *inducing relation*, and the obtained upward (bi-)simulation will then be called an *induced relation*.

In the next section, we will show how a pair of an inducing and induced relation can be combined into a new equivalence suitable for reducing tree automata. Note that the inducing relation is thus used in two different ways: as a parameter of the upward (bi-)simulation and as a constituent of the combined relation. By considering various inducing relations, we obtain a wide spectrum of combined relations differing in their computational complexity and coarseness (which is usually better and never worse than that of the inducing relation).

3.1 Downward (Bi-)Simulations

For a tree automaton $A = (Q, \Sigma, \Delta, F)$, a *downward simulation* D is a binary relation on Q such that if qDr and $(q_1, \ldots, q_n) \xrightarrow{f} q$, then $(r_1, \ldots, r_n) \xrightarrow{f} r$ with $q_i D r_i$ for each $i : 1 \leq i \leq n$. A *downward bisimulation* D is a binary relation on Q such that if qDr, then $(q_1, \ldots, q_n) \xrightarrow{f} q$ if and only if $(r_1, \ldots, r_n) \xrightarrow{f} r$ with $q_i D r_i$ for each $i : 1 \leq i \leq n$.

The following two lemmas state some basic properties of downward (bi-)simulations.

Lemma 1. *Given a tree automaton A, the set of all downward simulations on A is closed under reflexive and transitive closure and under union.*

Lemma 2. *Given a tree automaton A, the set of all downward bisimulations on A is a subset of the set of all downward simulations on A and it is closed under symmetric, reflexive, and transitive closure and under union.*

Proofs of Lemmas 1 and Lemma 2 can be found in [5] (cf. also Appendix A.1). The lemmas imply that for a given tree automaton, there is a unique maximal downward bisimulation which is an equivalence and a unique maximal downward simulation which is a preorder. We note that the notion of downward bisimulations corresponds to that of backward bisimulations from [10].

The obvious fact that any downward bisimulation is a downward simulation allows us to simplify some further reasoning by considering just downward simulations and handling bisimulations as their special case.

3.2 Induced Upward Simulations

Given a tree automaton $A = (Q, \Sigma, \Delta, F)$ and an inducing downward simulation preorder D, an *upward simulation* U induced by D is a binary relation on Q such that if qUr, then

(i) if $(q_1, \ldots, q_n) \xrightarrow{f} q'$ with $q_i = q$, $1 \leq i \leq n$, then $(r_1, \ldots, r_n) \xrightarrow{f} r'$ with $r_i = r$, $q'Ur'$, and $q_j D r_j$ for each $j : 1 \leq j \neq i \leq n$;
(ii) $q \in F \implies r \in F$.

The following lemma subsumes basic properties of upward simulations. Note that it also implies that for any inducing downward simulation preorder D, there is always a unique maximal upward simulation induced by D which is a preorder.

Lemma 3. *Given a tree automaton A and a downward simulation preorder D, the set of all upward simulations induced by D is closed under reflexive and transitive closure and under union.*

3.3 Induced Upward Bisimulations

Let $A = (Q, \Sigma, \Delta, F)$ be a tree automaton and let D be a downward simulation preorder. An *upward bisimulation U on Q induced by D* is a binary relation on Q such that if qUr, then

(i) $(q_1, \ldots, q_n) \xrightarrow{f} q'$ with $q_i = q, 1 \leq i \leq n$, if and only if $(r_1, \ldots, r_n) \xrightarrow{f} r'$ with $r_i = r$, $q'Ur'$, and $q_j \equiv_D r_j$ for each $j : 1 \leq j \neq i \leq n$;
(ii) $q \in F \iff r \in F$.

As for upward simulations, it is not hard to prove the basic properties of upward bisimulations. Note that the following lemma implies that for any downward simulation preorder D, there is a unique maximal upward bisimulation induced by D that is an equivalence. It is also clear that any upward bisimulation induced by D is also an upward simulation induced by D. This will allow us to prove the main results just for upward simulations and maintain bisimulations as a special case.

Lemma 4. *Given a tree automaton A and a downward simulation preorder D, the set of all upward bisimulations induced by D is a subset of the set of all upward simulations on A induced by D and it is closed under symmetric, reflexive, and transitive closure and under union.*

Proofs of Lemmas 3 and 4 can be found in [5] (cf. also Appendix A.2). Let us note that the notion of an upward bisimulation induced by the identity relation corresponds to the notion of a forward bisimulation from [10].

4 Combined Relations for Reducing the Size of Tree Automata

Upward simulation equivalences and upward bisimulations alone cannot be used for reducing tree automata as they do not preserve their language. To circumvent this problem, we have to take into account the inducing relation and *combine* it with the induced upward (bi-)simulation—as we have already mentioned in the previous section, the induced relation is thus used in two different ways.

As one of the main contributions of this work, we are now going to define a new combination operator—we call it a *weakening combination operator*—over the inducing downward simulations and induced upward simulations. Unlike the operators used in previous works [2, 1], the new operator allows us to combine any inducing downward simulation (i.e., also a bisimulation or identity) with any induced upward simulation (bisimulation, identity) offering us a broad spectrum of the resulting relations.

Moreover, in the previous works, we were randomly computing one relation out of the set of possible language-preserving combined relations. Here, we first prove that there is always a *unique maximal combined preorder* for a given upward simulation and its inducing downward simulation. In Section 6, we then provide a simple algorithm for computing this maximal preorder. From a practical point of view, using the maximal preorder instead of a random one has in some cases a great impact on the size of the reduced automaton as witnessed by our practical experiments.

A weakening combination operator. We define the *weakening combination operator* $\oplus$, which is normally to be applied over an inducing downward simulation and an induced upward simulation, on arbitrary preorders as follows: Given two preorders H and S over a set Q, for $x,y \in Q$, $x(H \oplus S)y$ iff (i) $x(H \circ S)y$ and (ii) $\forall z \in Q : yHz \implies x(H \circ S)z$.[3] Intuitively, a pair $e = (x,y) \in Q \times Q$ that is in $H \circ S$ will also appear in $H \oplus S$ iff after adding it to the preorder H, there is a possibility to supplement the obtained relation $H \cup \{e\}$ by other pairs of elements of Q that are in $H \circ S$ to ensure transitivity and get this way a preorder again.

Lemma 5. *For any set Q and any preorders $H,S \subseteq Q \times Q$, $H \oplus S$ is a unique maximal preorder such that $H \subseteq H \oplus S \subseteq H \circ S$.*

Proof. Let $W = H \oplus S$ and $C = H \circ S$.[4] Keep in mind that $W,H,S \subseteq C$ and that H and S are reflexive and transitive. We first prove some auxiliary facts for any $x,y,z \in Q$ allowing us to derive existence of certain elements of the relations that we are dealing with:

I. $xCy \implies xHwSy$ for some $w \in Q$, which follows directly from the definition of C.
II. $xHyCz \implies xCz$. From yCz and (I), we have $yHwSz$ for some $w \in Q$. From $xHyHw$, we have xHw. From $xHwSz$ and from the definition of C, we have xCz.
III. $xWyHz \implies xCz$, which follows directly from the definition of W.
IV. $xCySz \implies xCz$. From xCy and (I), we have $xHwSy$ for some $w \in Q$. From $wSySz$, we have wSz. From $xHwSz$ and (II), we have xCz.
V. $xWyCz \implies xCz$. From yCz and (I), we have $yHwSz$ for some $w \in Q$. From $xWyHw$ and (III), we have xCw, which together with (I) gives $xHvSw$ for some $v \in Q$. From $vSwSz$, we have vSz and so vCz (as $S \subseteq C$), which together with xHv and (II) gives xCz.

Now, we can prove the claim of the lemma. First, we will argue that $H \subseteq W$. To do this, suppose that xHy for some $x,y \in Q$. We will show that xWy. As $H \subseteq C$, we have that xCy, which fulfils Condition (i) from the definition of $\oplus$. To satisfy Condition (ii), we have to show that for arbitrary $z \in Q$ such yHz, xCz holds. From transitivity of H and from $xHyHz$, we have xHz, which implies that xCz when we take into account that $H \subseteq C$. Thus, even Condition (ii) is fulfilled, and we are obtaining xWy from the definition of $\oplus$. Hence, we have proved that $H \subseteq W$. Moreover, the fact that $W \subseteq C$ is trivial as it is a part of the definition of $\oplus$.

We will now prove that W is a preorder. We first prove by contradiction that W is transitive. Suppose that there exist $x,y,z \in Q$ such that $xWyWz$, but not xWz. Recall that $W \subseteq C$. From (I), we have $xHwSyHvSz$ for some $v,w \in Q$. From $xWyHv$ and (III), we have xCv. From $xCvSz$ and (IV), we have xCz. From the definition of $\oplus$, xCz together with not xWz imply that there is a $q \in Q$ such that $xCzHq$, but not xCq. From $yWzHq$ and (III), we get yCq. Then $xWyCq$, and (V) gives xCq, which is a contradiction. We have proven that the relation W is transitive. Showing that W is also reflexive is immediate as we already know that $H \subseteq W$ and that H is reflexive. Thus, we have proven that W is a preorder.

Finally, we will show that W is a unique maximal preorder included in C and containing H. It can be easily seen from the definition of $\oplus$ that any pair $(x,y) \in C \setminus W$ cannot be contained in any preorder P such that $H \subseteq P \subseteq C$ as no relation P such that $H \subseteq P \subseteq C$ and $(x,y) \in P$ can be transitive. Thus, W contains all the pairs that can be elements of a preorder included in C and containing H, and therefore any such preorder P is a subset of W. As we have proven that W itself is a preorder, it has to be the unique maximal preorder that includes H and that is itself included in C. □

Lemma 5 is a key result that allows us to define combined preorders and equivalences applicable for reducing the size of tree automata as follows.

[3] Here, $H \circ S$ is the common composition of relations, i.e., $\forall x,y \in Q : x(H \circ S)y \iff \exists z \in Q : xHzSy$.

[4] For an easier orientation in the symbols, let us note that we use W for the result of applying the weakening combination operator, C to denote the composition of H and S, H is a preorder which is "hard" in the sense that it has to be included in W, whereas S is "soft" in the same sense.

Combined relations for reducing tree automata. Consider a tree automaton A, a downward simulation preorder D, and an upward simulation U induced by D. We call the relation $W = D \oplus U^{-1}$ a *combined preorder* and $\equiv_W$ a *combined equivalence*. Correctness of using the combined equivalence for reducing the size of tree automata is stated in the following theorem.[5]

Theorem 1. *$L(A_{\equiv_W}) = L(A)$ for any tree automaton A and each combined preorder W.*

A proof of Theorem 1 can be found in [5] (cf. also Appendix A.3). Note that the theorem also covers the case of reducing automata using downward simulations (and bisimulations) alone. Indeed, given any downward simulation D, the identity is always an upward simulation induced by D. Then, the combined preorder $D \oplus id^{-1}$ equals D, which means that we can reduce the automaton using $\equiv_D$. In particular, this elegantly covers as special cases the proofs of correctness of reducing automata using downward bisimulations and simulation equivalences stated in [2].

Corollary 1. *$L(A_{\equiv_D}) = L(A)$ for any tree automaton A and each downward simulation preorder D.*

5 Variants of Combined Relations and Their Properties

Theorem 1 and Lemmas 2 and 4 allow us to consider quite a large spectrum of relations suitable for reducing tree automata. We now examine properties of the relations from this spectrum that arise when we consider the identity, the maximal downward bisimulation, and the maximal downward simulation as the inducing relation D for both the maximal upward bisimulation and upward simulation.

Our notation for the various types of combined equivalences that we consider consists of two parts: a relation symbol and an additional symbol above the relation symbol. The relation symbol denotes the type of the inducing downward relation. Namely, $=$ denotes the identity, $\simeq$ denotes the maximal downward bisimulation, and $\sim$ the maximal downward simulation. The additional symbol then denotes the type of the upward relation. We use $\bullet$ for the maximal upward bisimulation and $\circ$ for the maximal upward simulation. No additional symbol corresponds to the maximum equivalence embedded in the downward relation itself—the downward (bi-)simulations can be viewed as compositions where the role of the upward relation is played by the identity. For example, $\overset{\circ}{\simeq}$ denotes the relation $\equiv_{D \oplus U^{-1}}$ where D is the maximal downward bisimulation and U is the maximal upward simulation induced by D. In what follows, we will implicitly consider all the downward and upward (bi-)simulations that we will be dealing with to be the maximal ones.

A partial ordering of the combined relations wrt. their coarseness. From the definition of a combined preorder, it clearly follows that, for a fixed inducing relation D, if we are choosing the type of the upward relations U from the strongest one to the coarsest one, i.e., starting from the identity and going through the upward bisimulation induced by D to the upward simulation induced by D, we obtain coarser and coarser combined preorders $D \oplus U^{-1}$.

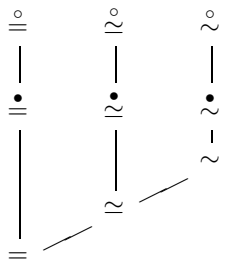
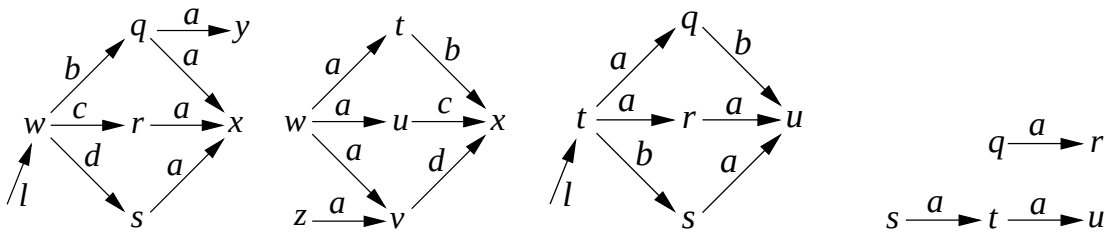


Fig. 1. Coarseness of various types of combined equivalences

On the other hand, if the inducing preorder D is growing, from the definition of the upward (bi-)simulation, we can see that the maximal upward (bi-)simulation U induced by D and thus also the relation $D \circ U^{-1}$ are growing too. But, when having $D \circ U^{-1}$ computed and then computing the preorder $D \oplus U^{-1}$ from it, the relation D acts as a restriction. A bigger relation D can cause that more pairs are violating Condition (ii) from the definition of $\oplus$. In general, having two downward simulation preorders D_1 and D_2, we are guaranteed that the maximal upward (bi-)simulation U_1 induced by D_1 is included in the maximal upward (bi-)simulation U_2 induced by D_2. Therefore, we

[5] Note that contrary to downward simulations, the combined preorders do not have to refine the language inclusion preorder, which is due to the fact that they strongly depend also on the upward simulations, which are not compatible with the language inclusion preorder. Nevertheless, Theorem 1 shows that the combined equivalences still preserve the language of the entire automaton when used for collapsing it.



(a) Δ_1 (b) Δ_2 (c) Δ_3

Fig. 2. Transition relations of automata proving the non-inclusion relationships from Figure 1, and of an automaton proving that one cannot use within $\oplus$ preorders included in the language inclusion preorder that are not downward simulations.

know that $D_1 \circ U_1^{-1} \subseteq D_2 \circ U_2^{-1}$, but the combined preorders $D_1 \oplus U_1^{-1}$ and $D_2 \oplus U_2^{-1}$ can be incomparable.[6]

Based on these observations, we obtain the partial ordering of all the considered types of combined equivalences according to inclusion which is depicted in Figure 1. For an automaton A, we denote by $\equiv(A)$ the combined equivalence of type $\equiv$ on A. In the figure, the line from $\equiv_1$ up to $\equiv_2$ means that for any automaton A, $\equiv_1(A) \subseteq \equiv_2(A)$. It is not hard to find an automaton A showing that all these relationships are strict, i.e., such that for each of the edges in the figure, $\equiv_1(A) \subsetneq \equiv_2(A)$. We construct such an automaton in Example 1.

Example 1. Let $Q = \{q,r,s,t,u,v,w,x,y,z\}$ be a set of states and let Σ be a ranked alphabet such that $\Sigma_0 = \{l\}$ and $\Sigma_1 = \{a,b,c\}$. The automaton $A = (Q,\Sigma,\Delta_1,\{x\})$ proves strictness of the relations in Figure 1. For each two types of relations from Figure 1 such that $\equiv_2$ is above $\equiv_1$, $\equiv_1(A) \subsetneq \equiv_2(A)$ holds. The transition relation Δ_1 is depicted in Figure 2(a). In the table below there are stated the appropriate combined equivalences for all the combinations of the considered types of inducing and induced relations. For each type of combination, we list nontrivial equivalence classes of the resulting combined equivalence:

$\overset{\circ}{=}$: $\{q,r,s\}$	$\overset{\circ}{\simeq}$: $\{t,u\},\{q,r,s\}$	$\overset{\circ}{\sim}$: $\{t,u,v\},\{q,r,s\},\{x,z\}$
$\overset{\bullet}{=}$: $\{r,s\}$	$\overset{\bullet}{\simeq}$: $\{t,u\},\{r,s\}$	$\overset{\bullet}{\sim}$: $\{t,u,v\},\{r,s\}$
$=$:	$\simeq$: $\{t,u\}$	$\sim$: $\{t,u,v\}$

It is now easy to check that all the inclusions from Figure 1 are strict for the automaton A. □

Incomparability of some of the combined relations. To complete the picture, we need to show that the types of combined relations that are not connected in Figure 1 are really incomparable. In other words, that for each pair $\equiv_1,\equiv_2$ of types of combined equivalences that are not connected in Figure 1 there exists an automaton A such that neither $\equiv_1(A) \not\subseteq \equiv_2(A)$ nor $\equiv_1(A) \not\supseteq \equiv_2(A)$. We construct such automata within Example 2.

Example 2. Let $Q = \{q,r,s,t,u,v\}$ be a set of states and let Σ be a ranked alphabet such that $\Sigma_0 = \{l\}$ and $\Sigma_1 = \{a,b,c\}$. All the incomparability results show up taking automata $A_1 = (Q \setminus \{v\},\Sigma,\Delta_2,\{u\})$ and $A_2 = (Q,\Sigma,\Delta_2 \cup \{v \xrightarrow{a} q\},\{u\})$ where the transition relation Δ_2 is depicted in Figure 2(b). One can easily check that $\overset{\bullet}{=}(A_1)$ and $\overset{\circ}{=}(A_1)$ define just one nontrivial equivalence class $\{r,s\}$ and thus they are incomparable with $\overset{\circ}{\simeq}(A_1), \overset{\bullet}{\sim}(A_1), \overset{\circ}{\sim}(A_1)$ that define only one nontrivial equivalence class $\{q,r\}$. In the case of the automaton A_2, the added transition $v \xrightarrow{a} q$ distinguishes the downward simulation from the downward bisimulation. Analogically as for A_1, we have that $\overset{\bullet}{\simeq}(A_2)$ and $\overset{\circ}{\simeq}(A_2)$ define just one nontrivial equivalence class $\{r,s\}$ and thus they are incomparable with $\overset{\bullet}{\sim}(A_2)$ and $\overset{\circ}{\sim}(A_2)$ that define only one nontrivial equivalence class $\{q,r\}$. This gives all the incomparability relationships. □

[6] Although, in our experiments, the former one usually *is* included in the latter one.

According to our experiments presented in Section 7, the reduction capabilities are rising when we move in Figure 1 not only in the bottom-up direction (according to the edges), but also in the left-right direction (as though within a full diamond). As a trade-off, the computational complexity of constructing the relations is rising in the same way from the bottom to the top and from the left to the right.

Impossibility of relaxing the need of downward simulations. It is easy to see that when not considering combined relations (and when not thinking of the computational complexity), one can replace the use of downward (bi-)simulations in reducing the size of tree automata by a use of any preorder which is included in the so called language inclusion preorder LP ($(q,r) \in LP \iff L(q) \subseteq L(r)$). A natural question comes forward: Is it also possible to induce (and combine by $\oplus$) an upward (bi-)simulation with any preorder included in LP (not only with downward simulations)? Here, we give a negative answer. Not all preorders included in LP can be used within the operator $\oplus$ for reducing automata. We prove this claim by the following counterexample.

Example 3. Consider an automaton $A = (Q, \Sigma, \Delta_3 \cup Leaves, F)$ where $Q = \{q,r,s,t,u\}$, $\Sigma_0 = \{l\}$, $\Sigma_1 = \{a\}$, Δ_3 is depicted in Figure 2(c), $Leaves = \{\xrightarrow{l} x \mid x \in Q\}$,[7] and $F = Q$. Let us choose the relation $R = id \cup \{(q,r),(r,t),(q,t)\}$, which is apparently contained in LP, as the inducing preorder.[8] We can choose the relation $U = id \cup \{(q,t)\}$ as the upward simulation preorder induced by R. Then, we obtain $R \circ U^{-1} = R \cup U^{-1} \cup \{(r,q)\}$. The pair (r,q) is present in $R \circ U^{-1}$ because of $(r,t) \in R$ and $(q,t) \in U$. Let $W = R \oplus U^{-1}$ be the combined preorder. $R \circ U^{-1}$ itself is already a preorder, and therefore $W = R \circ U^{-1}$. We see that we have obtained an equivalence class $\{q,r\}$ of $\equiv_W$ which is bad as it implies that the quotient automaton $A_{\equiv_W}$ contains the rule $\{q,r\} \xrightarrow{a} \{q,r\}$. This definitely changes the language of the original automaton A since no cycles were present in A.

Observe that if we take a downward simulation as the inducing preorder, such a situation does not arise. The problem above is caused by the presence of (r,q) in $R \circ U^{-1}$, which is enabled by $(r,t) \in R$. If R was a downward simulation containing (r,t), then R would have to contain even (q,s) from the definition of a downward simulation. So, we would get $r(R \oplus U^{-1})qRs$ which according to Condition (ii) of the definition of $\oplus$ enforces $r(R \circ U^{-1})s$. However, $r(R \circ U^{-1})s$ does not hold for any pair of an inducing downward simulation R and an induced upward simulation U (not even when one considers the maximal ones), and so the pair (r,q) is not present in any combined preorder, and we are never allowed to collapse q and r. □

A note on word automata. We note that all the above results carry over to word automata. The inclusion properties from Figure 1 hold for word automata too since they can be seen as a special case of tree automata. Moreover, our automata examples proving strictness of the relationships and incomparability relationships are built using just leaf and unary rules, and so they are valid for word automata as well.

6 Computing the Proposed Relations

Below, we first briefly discuss methods for computing downward and upward (bi-)simulations proposed in earlier works. Then, we propose an algorithm for computing the combined relations and analyse its complexity. For the rest of the section, let us fix a tree automaton $A = (Q, \Sigma, \Delta, F)$ and let $n = |Q|$, $m = |\Delta|$, $\ell = |\Sigma|$.

6.1 Computing Downward and Upward (Bi-)Simulations

The problem of computing (bi-)simulations over tree automata is addressed in [2, 1, 10]. In [2, 1], a quite general method for computing tree (bi-)simulations via transforming this problem to special instances

[7] The set of rules *Leaves* is present so that the language of the automaton is not be empty.

[8] As we deal here only with unary and leaf symbols, upward (bi-)simulation does not depend on the inducing relation.

of the classical problem of computing (bi-)simulations over labelled transition systems (LTS) is proposed. Classical (bi-)simulation algorithms like [12, 13] are then applied to the LTSs obtained from the translation.

As studied in [1], using the above approach, we obtain algorithms for computing the maximal downward and upward bisimulations in time $o(\hat{r}^3\, m \log n)$ and $o(m \log(n+\ell) + T(D))$, respectively, where $T(D)$ denotes the complexity of computing the inducing relation D.

The case of tree simulations is considered in [2], where we obtain algorithms with the following complexities when using the translation to LTS: Let D be the maximal downward simulation on A and let $|Lhs(A)/\equiv_D|$ be the size of the partitioning of the left-hand sides of the transition rules according to D. D can be computed in time $o((\ell+\hat{r}))\cdot|Lhs|\cdot|Lhs/\equiv_D| + m\cdot|Lhs/\equiv_D|)$, which can be roughly approximated by $o((\hat{r}+\ell)\, m^2)$. Let U be the maximal upward simulation on A induced by a preorder D, we denote the set of environments partitioned with respect to U as $Env/\equiv_U$. Assuming that $T(D)$ is the time of computing the inducing relation D, U can be computed in time $o((\ell\,|Env| + \hat{r}\, m)\,|Env/\equiv_U| + \hat{r}^2\, m \log |Env| + T(D))$, which can be roughly approximated by $o(\ell\,\hat{r}^2\, m^2 + T(D))$.

Let us add that specialised algorithms for computing the downward bisimulation and upward bisimulation induced by the identity were proposed in [10]. These algorithms run in time $o(\hat{r}^2\, m \log n)$ and $o(\hat{r}\, m \log n)$, respectively.

6.2 Computing the Combined Relations

Given an inducing downward simulation D and an upward simulation U induced by D, the combined preorder $W = D \oplus U^{-1}$ can be easily computed by simply following its definition. It is sufficient to start by computing the relation $C = D \circ U^{-1}$ and then just erase all the elements of C (which are pairs of elements of the base set Q) that break Condition (ii) from the definition of $\oplus$. Using suitable data structures, this computation starting from the relations U and D can be implemented in time $o(\min\{|D|\cdot|Q|, |U|\cdot|Q|\})$ as follows.

We encode a relation ρ on Q as an array indexed by elements of Q of lists of elements of Q. A state q is present in a list with index r iff $(r,q) \in \rho$. Note that given a boolean matrix representation of the relation, the "array of lists"-representation can be derived in time $o(|Q|^2)$. Note also that as U and D are reflexive, we have that $|U|, |D| \geq |Q|$ and thus $|Q|^2 \leq \min\{|D|\cdot|Q|, |U|\cdot|Q|\}$. Let arrays of lists $\mathsf{D}, \mathsf{U}^{-1}$ encode relations D, U^{-1}.

The relation $C = D \circ U^{-1}$ represented by a Boolean matrix C can be computed in the following way: (1) Initialise all entries of C to *false*. (2) For each $q \in Q$, pass through all elements of the list $\mathsf{D}[q]$, and for each $r \in \mathsf{D}[q]$, pass through all elements s of $\mathsf{U}^{-1}[r]$, and set $\mathsf{C}[q,s]$ to *true*. This procedure takes time $o(|\{(q,r,s) \mid (q,r) \in D \wedge (r,s) \in U^{-1}\}|) \subseteq o(\min\{|D|\cdot|Q|, |U|\cdot|Q|\})$.

Then we compute a Boolean matrix representation W of the relation $W = D \oplus U^{-1}$ as follows: (3) We initialise W as a copy of the matrix C (representing $D \circ U^{-1}$), and in the subsequent Step (4), we erase from W all the pairs of elements of Q that break Condition (ii) from the definition of $\oplus$. In Step (4), we proceed in the following way: For all $q \in Q$, for all $r \in \mathsf{D}[q]$, for all $s \in \mathsf{U}^{-1}[r]$, if not $\mathsf{C}[q,s]$ (i.e., $(q,s) \notin D \circ U^{-1}$), then $\mathsf{W}[q,s] = \mathit{false}$. This gives us the set $D \oplus U^{-1}$ represented by the matrix W. The complexity of Steps (3), (4) is in $o(|\{(q,r,s) \mid (q,r) \in D \wedge (r,s) \in U^{-1}\}| + |\{(q,r,s) \mid (q,r) \in U^{-1} \wedge (r,s) \in D\}|)$, which is again in $o(\min\{|D|\cdot|Q|, |U|\cdot|Q|\})$.

7 Experiments

We have implemented our algorithms in a prototype tool written in Java. We have used the tool on a number of tree automata from the frameworks of *regular tree model checking* (RTMC) and *abstract regular tree model checking* (ARTMC) [8, 4, 6, 7].

RTMC is the name of a family of techniques for analysing infinite-state systems such as parameterised networks of processes, systems with queues, stacks, unbounded integers, and/or dynamic linked data structures like lists or trees. In RTMC, states are represented by trees, sets of states by tree automata,

Table 1. The obtained reduction in percent and the computation time in seconds for the various considered relations applied for reducing TA obtained from RTMC and ARTMC case studies. The size of the TA is the number of their states plus the number of their transition rules.

TA		$\sim$		$\overset{\circ}{=}$		$\overset{\circ}{\simeq}$		$\overset{\circ}{\sim}$	
origin	size	reduction	time	reduction	time	reduction	time	reduction	time
ARTMC	195	18%	0.5 s	2%	0.5 s	23%	0.5 s	61%	1.0 s
RTMC	613	27%	3.5 s	19%	2.0 s	19 %	2.5 s	88%	5.1 s
RTMC	909	52%	3.6 s	72%	3.1 s	82%	3.4 s	89%	35.1 s
ARTMC	2029	10%	27.0 s	37%	26.0 s	33%	29.0 s	93%	39.0 s
RTMC	2403	26%	31.0 s	0%	25.0 s	0%	34.0 s	82%	37.1 s
TA		$\simeq$		$\overset{\bullet}{=}$		$\overset{\bullet}{\simeq}$		$\overset{\bullet}{\sim}$	
origin	size	reduction	time	reduction	time	reduction	time	reduction	time
ARTMC	195	18%	0.1 s	2%	0.5 s	23%	0.2 s	23%	0.6 s
RTMC	613	0%	0.3 s	0%	0.4 s	0%	0.8 s	27%	3.7 s
RTMC	909	14%	0.6 s	72%	0.4 s	82%	0.8 s	83%	4.1 s
ARTMC	2029	10%	1.7 s	14%	1.4 s	19%	3.1 s	44%	29.0 s
RTMC	2403	0%	0.3 s	0%	0.6 s	0%	0.7 s	27%	31.0 s

and transitions by tree transducers (or, sometimes, also by some specialised operations on tree automata). ARTMC is a combination of RTMC and the abstract-check-refine paradigm which usually greatly improves the efficiency of the technique. Most of the algorithms in the frameworks of both RTMC and ARTMC rely crucially on efficient automata reduction methods since the size of the generated automata often explodes, making computations infeasible without a reduction.

The tree automata that we have considered in our experiments arose within various computations within the frameworks of RTMC and ARTMC. Our experimental evaluation was carried out on an AMD Athlon 64 X2 2.19GHz PC with 2.0 GB RAM. We have compared the size of tree automata after reducing them with all the different reduction techniques considered in this paper.

Table 1 shows the computation time and the reduction (in percent) for the different relations within the considered framework and illustrates that we have really obtained a wide spectrum of relations differing in their reduction capabilities and computational complexity. As can be seen from the results, $\overset{\circ}{\sim}$ gives the best reduction in all experiments, but it also suffers from a high computation time. Combining simulations and bisimulations does not give the same amount of reduction as the combined simulation, but the computation time is lower and the reduction is better than $\overset{\bullet}{\simeq}$. Note that no attempt to optimise the implementation of any of the relations was done, and therefore the computation times could probably be much lower with an optimised implementation for all of them.

8 Conclusions and Future Work

We have presented a uniform framework for deriving equivalence relations suitable for reducing tree automata based on combining upward and downward simulations and bisimulations. The framework is based on two main ingredients: a new notion of upward (bi-)simulations parameterised by any downward (bi-)simulation and a new operator for combining upward and downward (bi-)simulations. The framework explains in a uniform way various previously obtained results [10, 2, 1] and also yields multiple new combined relations for reducing tree automata that mix in various degrees advantages of the various upward and downward simulations and bisimulations. This step is motivated by giving users of tree automata a finer choice between the reduction capabilities and computational costs of the relations to be used for reducing tree automata.

We have established a partial ordering of the obtained combined relations according to their reduction capabilities, and showed that some of them are also incomparable. Moreover, we have performed a number of experiments with automata from the area of (abstract) regular tree model checking that show a practical applicability of the obtained relations and allow us to conclude that the considered relations

really offer a fine choice of balance in the trade-off between reduction capabilities and computational requirements.

Furthermore, the proposed weakening combination operator on which our framework is based yields a significantly more efficient way of combining upward and downward (bi-)simulations than the previously used random combination algorithms.

The proposed framework is built on quite general principles and we believe that it can be extended to more advanced types of automata such as guided tree automata, nested word automata, or hedge automata that find their use in many applications in formal verification, decision procedures of various logics, structured document processing, or natural language processing. Reduction of automata from some of such classes has already been considered in the literature (e.g., in [9], the author proposes a bisimulation-based minimisation of weighted word automata, and a use of bisimulations for reducing weighted tree automata is considered in [11]).

From the practical point of view, it is also interesting to investigate more efficient techniques of computing the (bi-)simulation relations, e.g., by computing them in a symbolic way (for symbolically encoded automata). Furthermore, it can be interesting to explore more deeply the principles of the proposed combination of downward and upward (bi-)simulation relations. One can, for instance, think of defining still weaker types of relations preserving the language of tree automata by using the combined relations repeatedly as inducing relations.

Acknowledgement. We would like to thank Ahmed Bouajjani for fruitful discussions on the subject of the paper.

References

1. P. Abdulla, A. Bouajjani, L. Holík, L. Kaati, and T. Vojnar. Composed Bisimulation for Tree Automata., 2008. Accepted at CIAA'08.
2. P. Abdulla, A. Bouajjani, L. Holík, L. Kaati, and T. Vojnar. Computing Simulations over Tree Automata: Efficient Techniques for Reducing Tree Automata. In *Proc. of TACAS'08*, LNCS. Springer, 2008. An extended version appeared as the technical report FIT-TR-2007-001 of FIT, Brno University of Technology.
3. P. Abdulla, J. Högberg, and L. Kaati. Bisimulation Minimization of Tree Automata. In *Proc. of CIAA'06*, volume 4094 of *LNCS*, pages 173–185. Springer, 2006.
4. P. Abdulla, B. Jonsson, P. Mahata, and J. d'Orso. Regular Tree Model Checking. In *Proc. of CAV'02*, volume 2404 of *LNCS*. Springer, 2002.
5. P. Abdulla, A. Legay, J. d'Orso, and A. Rezine. Simulation-Based Iteration of Tree Transducers. In *Proc. of TACAS'05*, volume 3440 of *LNCS*. Springer, 2005.
6. A. Bouajjani, P. Habermehl, A. Rogalewicz, and T. Vojnar. Abstract Regular Tree Model Checking. In *Proc. of INFINITY'05*. Published in ENTCS 149(1), 2006.
7. A. Bouajjani, P. Habermehl, A. Rogalewicz, and T. Vojnar. Abstract Regular Tree Model Checking. *ENTCS*, 149:37–48, 2006.
8. A. Bouajjani and T. Touili. Extrapolating Tree Transformations. In *Proc. of CAV'02*, volume 2404 of *LNCS*. Springer, 2002.
9. P. Buchholz. Bisimulation relations for weighted automata. *Theor. Comput. Sci.*, 393(1-3):109–123, 2008.
10. J. Högberg, A. Maletti, and J. May. Backward and Forward Bisimulation Minimisation of Tree Automata. In *Proc. of CIAA'07*, volume 4094 of *LNCS*, pages 109–121. Springer, 2007.
11. J. Högberg, A. Maletti, and J. May. Bisimulation minimisation for weighted tree automata. In T. Harju, J. Karhumäki, and A. Lepistö, editors, *Proc. 11th Int. Conf. Developments in Language Theory*, volume 4588 of LNCS, pages 229–241. Springer, 2007.
12. R. Paige and R. Tarjan. Three Partition Refinement Algorithms. *SIAM Journal on Computing*, 16:973–989, 1987.
13. F. Ranzato and F. Tapparo. A New Efficient Simulation Equivalence Algorithm. In *Proc. of LICS'07*. IEEE CS, 2007.

A Appendix

A.1 Proofs of the Basic Properties of the Downward Relations

Proof (Lemma 1). We fix a tree automaton $A = (Q, \Sigma, \Delta, F)$.

Union: Given two downward simulations D_1 and D_2, we want to prove that $D = D_1 \cup D_2$ is also a downward simulation. Let qDr for some $q, r \in Q$, then either qD_1r or qD_2r. Assume without loss of generality that qD_1r. Then, from the definition of downward simulations, whenever $(q_1, \ldots, q_n) \xrightarrow{f} q$, then there is a rule $(r_1, \ldots, r_n) \xrightarrow{f} r$ with $q_iD_1r_i$ for all $i : 1 \leq i \leq n$. As $D_1 \subseteq D$ gives q_iDr_i for all the positions i, D fulfils the definition of a downward simulation.

Reflexive closure: It can be seen from the definition of downward simulations that the identity is a downward simulation. Thus, the union of the identity and any downward simulation is a downward simulation.

Transitive closure: Let D be a downward simulation and let D_T be its transitive closure. Let $q^1D_Tq^m$ and $(q_1^1, \ldots, q_n^1) \xrightarrow{f} q^1$. From $q^1D_Tq^m$, we have that there are states $q^1, \ldots, q^m$ such that $q^1Dq^2D \ldots Dq^m$. Therefore, from the definition of downward simulations, there are also rules $(q_1^1, \ldots, q_n^1) \xrightarrow{f} q^1, \ldots,$ $(q_1^m, \ldots, q_n^m) \xrightarrow{f} q^m$ with $q^1D \ldots Dq^m$, and $q_i^1D \ldots Dq_i^m$ for all $i : 1 \leq i \leq n$. Thus, as D_T is the transitive closure of D, we obtain $q_i^1D_Tq_i^m$ for all $i : 1 \leq i \leq n$. We have proven that D_T fulfils the definition of downward simulations.

□

Proof (Lemma 2). The fact that each downward bisimulation is also a downward simulation follows straight from the definitions of these relations. The closure under union, reflexivity, and transitivity can be proven analogically as in the case of downward simulations. What remains is the closure under symmetry. Let D be a downward bisimulation and let $D_S = D \cup D^{-1}$ be its symmetrical closure. It is sufficient to prove that D^{-1} is a downward bisimulation because then, from the closure under union, D_S is a downward bisimulation too.

Let $qD^{-1}r$. Then, from rDq, we have that $(r_1, \ldots, r_n) \xrightarrow{f} r$ if and only if $(q_1, \ldots, q_n) \xrightarrow{f} q$ with r_iDq_i for all $i : 1 \leq i \leq n$. As rDq is equivalent to $qD^{-1}r$, we can write that $(q_1, \ldots, q_n) \xrightarrow{f} q$ iff $(r_1, \ldots, r_n) \xrightarrow{f} r$ with $q_iD^{-1}r_i$ for all $i : 1 \leq i \leq n$. We directly see that D^{-1} matches the definition of downward bisimulations. □

A.2 Proofs of the Basic Properties of the Upward Relations

Lemmas 4 and 3 can be proven analogically as Lemmas 2 and 1.

Proof (Lemma 3). We fix a tree automaton $A = (Q, \Sigma, \Delta, F)$.

Union: Given a downward simulation preorder D and two upward simulations U_1 and U_2 induced by D, we want to prove that $U = U_1 \cup U_2$ is also an upward simulation induced by D. Let qUr for some $q, r \in Q$, then either qU_1r or qU_2r. Assume without loss of generality that qU_1r. Then, from the definition of upward simulations, whenever $(q_1, \ldots, q_n) \xrightarrow{f} q'$ with $q_i = q$, then there is a rule $(r_1, \ldots, r_n) \xrightarrow{f} r'$ with $q'U_1r'$, $q' \in F \implies r' \in F$, and q_jDr_j for all $j : 1 \leq j \neq i \leq n$. As $U_1 \subseteq U$ gives $q'Ur'$, U fulfils the definition of upward simulations induced by D.

Reflexive closure: It can be seen from the definition that the identity is an upward simulation induced by D for any downward simulation preorder D. Therefore, from the closure under union, the union of the identity and any upward simulation induced by D is an upward simulation induced by D.

Transitive closure: Let U be an upward simulation induced by a downward simulatoin preorder D and let U_T be its transitive closure. Let $q^1U_Tq^m$ and $(q_1^1, \ldots, q_n^1) \xrightarrow{f} r^1$ with $q^1 = q_i^1$. From $q^1U_Tq^m$, we have that there are states $q^1, \ldots, q^m$ such that $q^1Uq^2U \ldots Uq^m$. Therefore, there are also rules $(q_1^1, \ldots, q_n^1) \xrightarrow{f} r^1, \ldots, (q_1^m, \ldots, q_n^m) \xrightarrow{f} r^m$ with $q_i^1 = q_i^1, \ldots, q_i^m = q^m$, $r^1U \ldots Ur^m$, $r^1 \in F \implies$

$\ldots \implies r^m \in F$, and $q_j^1 D \ldots D q_j^m$ for all $j : 1 \leq j \neq i \leq n$. Thus, from the definition of U_T, we have $r^1 U_T r^m$, from the transitivity of $\implies$, we have $r^1 \in F \implies r^m \in F$, and from the transitivity of D, we have $q_j^1 D q_j^m$ for all $j : 1 \leq j \neq i \leq n$. We have thus proven that U_T fulfils the definition of an upward simulation induced by D.

□

Proof (Lemma 4). The fact that each upward bisimulation induced by a downward simulation D is also an upward simulation induced by D follows straight from the definitions of these relations. The closure under union, reflexivity, and transitivity can be proven analogically as in the case of upward simulations. What remains is the closure under symmetry. Let U be an upward bisimulation induced by a downward simulation preorder D and let $U_S = U \cup U^{-1}$ be its symmetrical closure. It is sufficient to prove that U^{-1} is an upward bisimulation induced by D because then, from the closure under union, U_S is an upward bisimulation induced by D too.

Let $qU^{-1}r$. Then, from rUq, we have that $(r_1, \ldots, r_n) \xrightarrow{f} r'$ with $r = r_i$ if and only if $(q_1, \ldots, q_n) \xrightarrow{f} q'$ with $q = q_i$, $r'Uq'$, and $r_j \equiv_D q_j$ for all $j : 1 \leq j \neq i \leq n$. As $\equiv_D$ is an equivalence and as $r'Uq'$ is equivalent to $q'U^{-1}r'$, we can write $(q_1, \ldots, q_n) \xrightarrow{f} q'$ with $q = q_i$ iff $(r_1, \ldots, r_n) \xrightarrow{f} r'$ with $r = r_i$, $q'U^{-1}r'$, and $q_j \equiv_D r_j$ for all $j : 1 \leq j \neq i \leq n$. We directly see that U^{-1} matches the definition of an upward bisimulation induced by D. □

A.3 Proof of Theorem 1

We fix a tree automaton A, a downward simulation preorder D, and an upward simulation U induced by D. Let $W = D \oplus (U^{-1})$.

We will relate the languages of A and $A_{\equiv_W}$. To do that, we first define the notion of a *context*. Intuitively, a context is a tree with "holes" instead of leaves. Formally, we consider a special symbol $\Box \notin \Sigma$ with rank 0. A *context* over Σ is a tree c over $\Sigma \cup \{\Box\}$ such that for all leaves $p \in c$, we have $c(p) = \Box$. For a context c with leaves $p_1, \ldots, p_n$ and for trees $t_1, \ldots, t_n$, we define $c[t_1, \ldots, t_n]$ to be the tree t, where

- $dom(t) = dom(c) \bigcup \{p_1 \cdot p' \mid p' \in dom(t_i)\} \bigcup \cdots \bigcup \{p_n \cdot p' \mid p' \in dom(t_n)\}$,
- for each $p = p_i \cdot p'$, we have $t(p) = t_i(p')$, and
- for each $p \in dom(c) \setminus \{p_1, \ldots, p_n\}$, we have $t(p) = c(p)$.

In other words, $c[t_1, \ldots, t_n]$ is the result of appending the trees $t_1, \ldots, t_k$ to the holes of c. We extend the notion of runs to contexts. Let c be a context with leaves $p_1, \ldots, p_n$. A *run* π of A on c from $(q_1, \ldots, q_n)$ is defined in a similar manner to a run on a tree except that for a leaf p_i, we have $\pi(p_i) = q_i$, $1 \leq i \leq n$. In other words, each leaf labelled with $\Box$ is annotated by one q_i. We use $c\,[q_1, \ldots, q_n] \xRightarrow{\pi} q$ to denote that π is a run of A on c from $(q_1, \ldots, q_n)$ such that $\pi(\varepsilon) = q$. The notation $c\,[q_1, \ldots, q_n] \Longrightarrow q$ is explained in a similar manner to runs on trees.

Lemma 6. *If $c[q_1, q_2, \ldots, q_n] \Longrightarrow q$ and $q_i U r_i$ for some $1 \leq i \leq n$, then there are states $r_1, \ldots, r_{i-1}$, $r_{i+1}, \ldots, r_n, r$ such that $q_j D r_j$ for each j such that $1 \leq j \neq i \leq n$, qUr, and $c[r_1, \ldots, r_n] \Longrightarrow r$.*

Proof. To simplify the notation, we assume (without loss of generality) that $i = 1$. We use induction on the structure of c. The base case is trivial since the context c consists of a single hole. For the induction step, we assume that c is not only a single hole. Suppose that $c[q_1, q_2, \ldots, q_n] \xRightarrow{\pi} q$ for some run π and that $q_1 U r_1$. Let $p_1, \ldots, p_j$ be the left-most leaves of c with a common parent. Let p be the parent of $p_1, \ldots, p_j$. Notice that $q_1 = \pi(p_1), \ldots, q_j = \pi(p_j)$. Let $q' = \pi(p)$ and let c' be the context c with the leaves $p_1, \ldots, p_j$ deleted. In other words, $dom(c') = dom(c) \setminus \{p_1, \ldots, p_j\}$, $c'(p') = c(p')$ if $p' \in dom(c') \setminus \{p, p_1, \ldots, p_j\}$, and $c'(p) = \Box$. Observe that $c'[q', q_{j+1}, \ldots, q_n] \Longrightarrow q$ and that $(q_1, \ldots, q_j) \xrightarrow{f} q'$ for some f. By the definition of the upward simulation and the premise $q_1 U r_1$, it follows that there are $r_2, \ldots, r_n, r'$ such that $q_2 D r_2 \in, \ldots, q_j D r_j, q' U r'$, and $(r_1, \ldots, r_j) \xrightarrow{f} r'$. Since c' is smaller than c, we can apply the induction hypothesis and conclude that there are $r_{j+1}, \ldots, r_n, r$ such that $q_{j+1} D r_{j+1}, \ldots, q_n D r_n, qUr$, and $c'[r', r_{j+1}, \ldots, r_n] \Longrightarrow r$. The claim follows immediately. □

For a state $r \in Q$, a set $\mathsf{B} \subseteq Q$ of states, and a relation $R \subseteq Q \times Q$, we write $\mathsf{B}Rq$ to denote that there is a $q \in \mathsf{B}$ with qRr.

Lemma 7. *For blocks $\mathsf{B}_1,\ldots,\mathsf{B}_n,\mathsf{B} \in Q_{\equiv_W}$ and a context c, if $c[\mathsf{B}_1,\ldots,\mathsf{B}_n] \Longrightarrow \mathsf{B}$, then there exist states $r_1,\ldots,r_n,r \in Q$ with $\mathsf{B}_1Dr_1,\ldots,\mathsf{B}_nDr_n,\mathsf{B}Ur$, and $c[r_1,\ldots,r_n] \Longrightarrow r$.*

Proof. The claim is shown by induction on the structure of c. In the base case, the context c consists of a single hole. We choose any $q \in \mathsf{B} \cap F$ provided that $\mathsf{B} \cap F \neq \emptyset$, and any $q \in \mathsf{B}$ otherwise. The claim holds obviously by reflexivity of D and U.

For the induction step, we assume that c is not only a single hole. Suppose that $c[\mathsf{B}_1,\ldots,\mathsf{B}_n] \stackrel{\pi}{\Longrightarrow} \mathsf{B}$ for some run π. Let $p_1,\ldots,p_j$ be the left-most leaves of c with a common parent. Let p be the parent of $p_1,\ldots,p_j$. Notice that $\mathsf{B}_1 = \pi(p_1),\ldots,\mathsf{B}_j = \pi(p_j)$. Let $\mathsf{B}' = \pi(p)$ and let c' be the context c with the leaves $p_1,\ldots,p_j$ deleted. In other words, $dom(c') = dom(c) \setminus \{p_1,\ldots,p_j\}$, $c'(p') = c(p')$ provided $p' \in dom(c') \setminus \{p,p_1,\ldots,p_j\}$, and $c'(p) = \Box$. Observe that $c'[\mathsf{B}',\mathsf{B}_{j+1},\ldots,\mathsf{B}_n] \Longrightarrow \mathsf{B}$. Since c' is smaller than c, we can apply the induction hypothesis and conclude that there are $v,q'_{j+1},\ldots,q'_n,q'$ such that $\mathsf{B}'Dv,\mathsf{B}_{j+1}Dq'_{j+1},\ldots,\mathsf{B}_nDq'_n,\mathsf{B}Uq'$, $c'[v,q'_{j+1},\ldots,q'_n] \Longrightarrow q'$. It follows that there are $u \in \mathsf{B}'$, $q_{j+1} \in \mathsf{B}_{j+1},\ldots,q_n \in \mathsf{B}_n,q \in \mathsf{B}$ such that uDv, qUq', and $q_{j+1}Dq'_{j+1},\ldots,q_nDq'_n$. By the definition of $A_{\equiv_W}$, there are states $q_1 \in \mathsf{B}_1,\ldots,q_j \in \mathsf{B}_j$, and $z \in \mathsf{B}'$ such that $(q_1,\ldots,q_j) \xrightarrow{f} z$ for some f. Since $D \subseteq W$ and uDv, we get uWv. Since $u,z \in \mathsf{B}'$, it follows that $u \equiv_W z$ and hence zWu. From transitivity of W, we get zWv. From the definition of W, there is a state w such that zDw and vUw. By the definition of the language inclusion preorder and premises zDw and $(q_1,\ldots,q_j) \xrightarrow{f} z$, there are states $r_1,\ldots,r_j$ with $q_1Dr_1,\ldots,q_jDr_j$, and $(r_1,\ldots,r_j) \xrightarrow{f} w$. By Lemma 6 and premises vUw and $c'[v,q'_{j+1},\ldots,q'_n] \Longrightarrow q'$, there are states $r_{j+1},\ldots,r_n$, and r with $q'_{j+1}Dr_{j+1},\ldots,q'_nDr_n,q'Ur$, and $c'[w,r_{j+1},\ldots,r_n] \Longrightarrow r$. Finally, by transitivity of D and U, we get $q_{j+1}Dr_{j+1},\ldots,q_nDr_n$, qUr. The claim thus holds. □

Lemma 8. *If $t \Longrightarrow \mathsf{B}$, then $t \Longrightarrow w$ for some w with $\mathsf{B}Uw$. Moreover, if $\mathsf{B} \in F_{\equiv_W}$, then also $w \in F$.*

Proof. Suppose that $t \stackrel{\pi}{\Longrightarrow} \mathsf{B}$ for some π. Let $p_1,\ldots,p_n$ be the leafs of t, and let $\pi(p_i) = \mathsf{B}_i$ for each $i : 1 \leq i \leq n$. Let c be the context that we get from t by deleting the leaves $p_1,\ldots,p_n$. Observe that $c[\mathsf{B}_1,\ldots,\mathsf{B}_n] \stackrel{\pi}{\Longrightarrow} \mathsf{B}$. It follows from Lemma 7 that there exist states $r_1,\ldots,r_n,r \in Q$ and $q_1 \in \mathsf{B}_1,\ldots,q_n \in \mathsf{B}_n,q \in \mathsf{B}$ such that $q_1Dr_1,\ldots,q_nDr_n,qUr$, $c[r_1,\ldots,r_n] \Longrightarrow r$, and if $\mathsf{B} \cap F \neq \emptyset$, then $r \in F$. By the definition of $A_{\equiv_W}$, it follows that there are $q'_1 \in \mathsf{B}_1,\ldots,q'_n \in \mathsf{B}_n$, and $f_1,\ldots,f_n$ such that $\xrightarrow{f_i} q'_i$ for each i such that $1 \leq i \leq n$. We show by induction on i that for each i such that $1 \leq i \leq n$ there are states $u^i_1,\ldots,u^i_i,v^i_{i+1},\ldots,v^i_n,w^i$ such that $q'_1Du^i_1,\ldots,q'_iDu^i_i,q_{i+1}Dv^i_{i+1},\ldots,q_nDv^i_n,rUw^i$, and $c[u^i_1,\ldots,u^i_i,v^i_{i+1},\ldots,v^i_n] \Longrightarrow w^i$. The base case where $i = 0$ is trivial. We consider the induction step. Since $D \subseteq W$ and $q_{i+1}Dv_{i+1}$, we get $q_{i+1}Wv_{i+1}$. Since $q_{i+1},q'_{i+1} \in \mathsf{B}_{i+1}$, we have that $q'_{i+1} \equiv_W q_{i+1}$ and hence $q'_{i+1}Wq_{i+1}$. By transitivity of W, it follows that $q'_{i+1}Wv_{i+1}$. By the definition of W, there is z_{i+1} such that $q'_{i+1}Dz_{i+1}$ and $v_{i+1}Uz_{i+1}$. By Lemma 6, there are $z_1,\ldots,z_i,z_{i+2},\ldots,z_n,z$ with $u^i_1Dz_1,\ldots,u^i_iDz_i$, $v^i_{i+2}Dz_{i+2},\ldots,v^i_nDz_n$, w^iUz, and $c[z_1,\ldots,z_n] \Longrightarrow z$. By transitivity of D and the premises $q'_jDu^i_j$ and $u^i_jDz_j$, we have q'_jDz_j for each $j : 1 \leq j \leq i$. By transitivity of D and the premises $q_jDv^i_j$ and $v^i_jDz_j$, we have q_jDz_j for each $j : i+2 \leq j \leq n$. Define $u^{i+1}_j = z_j$ for $j : 1 \leq j \leq i+1$, $v^{i+1}_j = z_j$ for $j : i+2 \leq j \leq n$, and $w^{i+1} = z$.

The induction proof above implies that $c[u^n_1,\ldots,u^n_n] \Longrightarrow w^n$. From the definition of the language inclusion preorder and the premises $\xrightarrow{f_i} q'_i$ and $q'_iDu^n_i$, it follows that $\xrightarrow{f_i} u^n_i$ for each $i : 1 \leq i \leq n$. It follows that $t = c[f_1,\ldots,f_n] \Longrightarrow w^n$. By the definition of U and the fact that $r \in F$ if $\mathsf{B} \cap F \neq \emptyset$, it follows that for all $i : 1 \leq i \leq n$, $w^i \in F$ provided that $\mathsf{B} \in F_{\equiv_W}$. Thus, in the claim of the lemma, it suffices to take $w = w^n$. □

Proof (Theorem 1). The inclusion $L(A_{\equiv_W}) \supseteq L(A)$ is trivial. Let $t \in L(A_{\equiv_W})$, i.e., $t \Longrightarrow \mathsf{B}$ for some block B where $\mathsf{B} \cap F \neq \emptyset$. Lemma 8 implies that $t \Longrightarrow w$ such that $w \in F$. □