Minimizing Weighted Tree Grammars using Simulation[⋆]

Andreas Maletti

Universitat Rovira i Virgili, Departament de Filologies Romàniques
Avinguda de Catalunya 35, 43002 Tarragona, Spain
andreas.maletti@urv.cat

Abstract. Weighted tree grammars (for short: WTG) are an extension of weighted context-free grammars that generate trees instead of strings. They can be used in natural language parsing to directly generate the parse tree of a sentence or to encode the set of all parse trees of a sentence. Two types of simulations for WTG over idempotent, commutative semirings are introduced. They generalize the existing notions of simulation and bisimulation for WTG. Both simulations can be used to reduce the size of WTG while preserving the semantics, and are thus an important tool in toolkits. Since the new notions are more general than the existing ones, they yield the best reduction rates achievable by all minimization procedures that rely on simulation or bisimulation. However, the existing notions might allow faster minimization.

1 Introduction

Grammar reduction and minimization are well-studied subjects [1]. Here we consider weighted tree grammars (Wtg), which are widely used in applications such as model checking [2] and in several areas of natural language processing [3]. Such Wtg are an extension of weighted (probabilistic) context-free grammars that generate trees instead of strings. They can, for example, be used to generate parse trees (with a weight) directly. Several toolkits implement Wtg [4,5,6].

Let us review the existing results on minimization of Wtg. There exists a direct correspondence between Wtg and weighted tree automata [7,8,9]. Deterministic bottom-up (unweighted) tree automata can be minimized efficiently using the algorithms inspired by Hopcroft [10,11]. This work has been extended to deterministic bottom-up weighted tree automata over semifields (i.e., commutative semirings with multiplicative inverses) in [12]. However, minimizing general (unweighted) tree grammars is Pspace-complete [13] and cannot be approximated well [14,15] unless P = Pspace. These negative results extend to Wtg over idempotent, commutative semirings. Consequently, alternative (efficient) methods to reduce the size of tree grammars are explored in [11,16,17,18]. In contrast, general Wtg over fields (i.e., commutative semirings with multiplicative and additive inverses) can efficiently be minimized to a unique (up to

⋆ This work was financially supported by the *Ministerio de Educación y Ciencia* (MEC) grants JDCI-2007-760 and MTM-2007-63422.

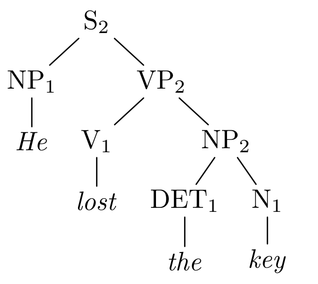
a change of basis) minimal Wtg [19,20]. Finally, efficient reductions of Wtg with the help of bisimulation relations are considered in [21].

Here we extend the simulation approach for tree grammars of [17] to Wtg over idempotent, commutative semirings, and in the process, overcome the major problems of [18]. Let us explain Wtg in more detail. A Wtg is a tree grammar, in which each production is assigned a weight of a semiring. Instead of just generating a certain set of trees, a Wtg assigns a weight to each tree. In [17] two types of simulation relations, called downward and upward simulations, are investigated for tree grammars. These notions were generalized to Wtg in [18]. This generalization required semirings with a partial order, so idempotent (i.e., $a + a = a$ for all a) semirings equipped with their natural order were considered. Idempotent semirings are used, for example, when extracting the n-best derivations [22] from a parser (even if the parser was not trained over an idempotent semiring) and Wtg over certain idempotent semirings (like the tropical semiring) directly represent all "best" derivations.

We generalize backward and forward simulation, which are defined in [18]. Intuitively, these notions correspond to backward and forward bisimulation of [21], but the notions of [18] did not properly generalize them. In addition, backward simulation generalizes downward simulation of [17] and our forward simulation generalizes upward simulation with respect to the identity as downward simulation [17]. We choose not to generalize upward simulations with respect to arbitrary downward simulations since we believe that two completely separate notions are easier to handle and understand. Our new notions now generalize all existing simulation and bisimulation notions for Wtg over commutative, idempotent semirings, and in addition, enjoy better properties than the corresponding notions of [18]. While a reduction with respect to a simulation of [18] might yield a Wtg that can be reduced further with the same type of simulation, this cannot occur with our simulations.

A backward simulation is a quasi-order (i.e., a reflexive, transitive relation) on the nonterminals of the Wtg such that larger nonterminals dominate the smaller ones; i.e., if a nonterminal allows us to generate a tree with weight a, then any larger nonterminal must be able to generate the same tree with a weight that is larger than a. This relation even holds at the transition level for the notions of [18], but it is lost at the transition level in our generalization. Two nonterminals that can simulate each other are considered equivalent, and we can reduce the Wtg with this equivalence relation. The obtained Wtg is equivalent to the original Wtg, so our reduction procedure can be applied before and after lossy reduction techniques such as pruning.

We also generalize forward simulation and show similar properties. Our minimization algorithms compute the greatest backward and forward simulations, which yield the best reduction. In addition, once a Wtg has been minimized with one type of simulation, it cannot be reduced any further using the same type. However, alternating backward and forward simulation can yield even smaller Wtg. In addition, pruning and other methods might also enable further nonterminals to become equivalent, so we might obtain even further reductions.



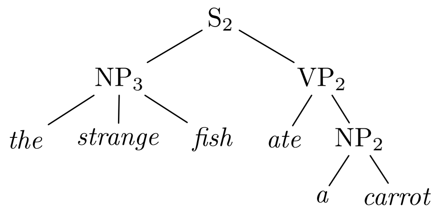


Fig. 1. Example tree (left) and tree used in Example 1 (right).

2 Trees and Weighted Tree Grammars

A quasi-order $\preceq$ on S is a reflexive, transitive relation on S. An up-set $A \subseteq S$ (with respect to $\preceq$) is such that for every $s \preceq s'$ with $s \in A$ also $s' \in A$. The smallest up-set containing $A \subseteq S$ is denoted by $\uparrow A$. If the quasi-order is not obvious from the context, then we write $\uparrow_{\preceq}(A)$. Moreover, if $A = \{a\}$, then we write $\uparrow a$ [and $\uparrow_{\preceq}(a)$].

For simplicity, we consider trees over ranked alphabets; i.e., each symbol we use has a fixed rank. Given an alphabet Σ, we write Σ_k for the set of symbols of Σ that have rank k. The rank of a symbol determines how many children a node marked with that symbol has in a tree. Consequently, our trees are formed by putting a symbol of rank k above k subtrees. Formally, the set T_Σ of trees over Σ is the smallest set such that $\sigma(t_1, \dots, t_k) \in T_\Sigma$ for every $\sigma \in \Sigma_k$ and $t_1, \dots, t_k \in T_\Sigma$.

A commutative semiring is an algebraic structure $(A, +, \cdot)$ such that $(A, +)$ and $(A, \cdot)$ are commutative monoids and $\cdot$ distributes over finite sums. It is idempotent if $a + a = a$ for every $a \in A$. In an idempotent, commutative semiring $(A, +, \cdot)$ the natural order $\sqsubseteq$ on A is defined by $a \sqsubseteq b$ if $a + b = b$. In the following, let $(A, +, \cdot)$ be an idempotent, commutative semiring. For example, the tropical semiring $(\mathbb{R} \cup \{\infty\}, \min, +)$ is such a semiring, in which $\sqsubseteq \, = \, \geq$.

To present our approach in a general setting, we recall weighted tree grammars (WTG) [7,23,24,25][1]. Weighted context-free grammars can be modelled as such grammars. Essentially, a WTG defines a weighted hypergraph [22], and formally, a WTG (in normal form) is a structure (N, Σ, P, I) such that

- N is a finite set of *nonterminals*,
- Σ is a ranked alphabet of *terminals*,
- P is a finite set of *productions* of the form $S \xrightarrow{a} \sigma(S_1, \dots, S_k)$ where $\sigma \in \Sigma_k$, $a \in A$ is a *weight*, and $S, S_1, \dots, S_k \in N$ are nonterminals, and
- $I \colon N \to A$ is an initial weight assignment.

Intuitively, a production $S \xrightarrow{a} \sigma(S_1, \dots, S_k)$ yields that a tree $\sigma(t_1, \dots, t_k)$ can be generated by S provided that the subtrees $t_1, \dots, t_k$ can be generated

[1] Note that most of the cited references investigate weighted tree automata, which are an equivalent formalism.

by $S_1, \dots, S_k$, respectively. The production incurs the weight a. In general, we assume that no two productions differ only in the weight. Left-most derivations are defined as usual. The weight of a derivation is the product (using $\cdot$) of the weights of the productions involved (counting multiple occurrences of the same production). The weight $\mathrm{wt}(t, S)$ of a terminal tree $t \in T_\Sigma$ and a start nonterminal S is obtained by adding the weights of all derivations of t from S.[2] Finally, the weight $\mathrm{wt}(t)$ is $\mathrm{wt}(t) = \sum_{S \in N} I(S) \cdot \mathrm{wt}(t, S)$. Let us illustrate this on a small example of [5]. Note that the weights are just made up and do not reflect usage probabilities.

Example 1. Let the semiring be the arctic semiring $(\mathbb{R} \cup \{-\infty\}, \max, +)$. Consider the WTG (N, Σ, P, I) such that

- $N = \{s, np, dt, jj, nn, vp, v\}$,
- Σ contains $\{\mathrm{S}_2, \mathrm{NP}_2, \mathrm{NP}_3, \mathrm{VP}_2\}$ and the English words "the", "a", "funny", "blue", "strange", "fish", "carrot", "ate", and "created",
- P contains the productions:

$$\begin{array}{llll}
s \xrightarrow{1} \mathrm{S}_2(np, vp) & dt \xrightarrow{.5} \text{the} & dt \xrightarrow{.5} \text{a} & \\
np \xrightarrow{.4} \mathrm{NP}_2(dt, nn) & jj \xrightarrow{.2} \text{funny} & jj \xrightarrow{.3} \text{blue} & jj \xrightarrow{.5} \text{strange} \\
np \xrightarrow{.6} \mathrm{NP}_3(dt, jj, nn) & nn \xrightarrow{.8} \text{fish} & nn \xrightarrow{.2} \text{carrot} & \\
vp \xrightarrow{1} \mathrm{VP}_2(v, np) & v \xrightarrow{.7} \text{ate} & v \xrightarrow{.3} \text{created}\ , &
\end{array}$$

- $I(s) = 1$ and $I(S) = -\infty$ for all remaining $S \in N$.

Now let us show a derivation. Consider tree t of Fig. 1 (right), which represents a parse tree of the English sentence "The strange fish ate a carrot." We can derive it as follows (at the end of the line we display the accumulated weight):

$$\begin{aligned}
s &\Rightarrow \mathrm{S}_2(np, vp) \Rightarrow \mathrm{S}_2(\mathrm{NP}_3(dt, jj, nn), vp) && (1.6) \\
&\Rightarrow \mathrm{S}_2(\mathrm{NP}_3(\text{the}, jj, nn), vp) \Rightarrow \mathrm{S}_2(\mathrm{NP}_3(\text{the}, \text{strange}, nn), vp) && (2.6) \\
&\Rightarrow^* \mathrm{S}_2(\mathrm{NP}_3(\text{the}, \text{strange}, \text{fish}), \mathrm{VP}_2(\text{ate}, \mathrm{NP}_2(\text{a}, \text{carrot}))) && (6.2)
\end{aligned}$$

Since this is the only derivation that yields t (see Fig. 1), we have $\mathrm{wt}(t, s) = 6.2$. If there would be several derivations (starting with s), then we would take the maximum weight among all such derivations. Since $I(S) = -\infty$ for all S besides s, we can conclude $\mathrm{wt}(t) = 6.2$.

3 Backward Simulation

Backward simulation was investigated for unweighted tree automata in [17] and generalized to our setting in [18]. Here we develop a general notion that generalizes the mentioned notions and the backward bisimulations of [21], which were

[2] In proofs we sometimes use the equivalent initial-algebra semantics [9], which is given by: $\mathrm{wt}(\sigma(t_1, \dots, t_k), S) = \sum_{S \xrightarrow{a} \sigma(S_1, \dots, S_k) \in P} a \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, S_i)$ for every $S \in N$, $\sigma \in \Sigma_k$, and $t_1, \dots, t_k \in T_\Sigma$.

not generalized by the notions of [18]. This latter fact led to the strange situation that a backward bisimulation was not at the same time also a backward simulation. Our notions repair this, and in addition, our minimization procedure is guaranteed to reduce more than all the minimization procedures developed for the mentioned simulations and bisimulations. From now on, let $G = (N, \Sigma, P, I)$ be a WTG, and for every $\sigma \in \Sigma_k$ and $\mathcal{T}_0, \dots, \mathcal{T}_k \subseteq N$, let

$$\mathrm{pwt}_\sigma(\mathcal{T}_0, \dots, \mathcal{T}_k) = \sum_{\substack{S_0 \xrightarrow{a} \sigma(S_1, \dots, S_k) \in P \\ S_0 \in \mathcal{T}_0, \dots, S_k \in \mathcal{T}_k}} a \enspace .$$

Intuitively, $\mathrm{pwt}_\sigma(\mathcal{T}_0, \dots, \mathcal{T}_k)$ is the weight of all productions generating the symbol σ and using the states of $\mathcal{T}_0, \dots, \mathcal{T}_k$.

Definition 2 (cf. [18, Definition 1]). *A quasi-order $\preceq$ on N is a* backward simulation *if for every $S \preceq T$, symbol $\sigma \in \Sigma_k$, and $T_1, \dots, T_k \in N$*

$$\mathrm{pwt}_\sigma(\{S\}, \uparrow T_1, \dots, \uparrow T_k) \sqsubseteq \mathrm{pwt}_\sigma(\{T\}, \uparrow T_1, \dots, \uparrow T_k) \enspace .$$

Note that the notion of a backward bisimulation is obtained by requiring that $\preceq$ is an equivalence relation in the previous definition (in that case $\uparrow T$ is the equivalence class of T). For the following discussions and results, let $\preceq$ be a backward simulation[3]. Unfortunately, our definition does not easily offer an intuitive explanation, thus let us continue to explore some central properties of such simulations. First, there exists a greatest (with respect to $\subseteq$) backward simulation. This can be proved by showing that for any two backward simulations also the reflexive, transitive closure of their union is a backward simulation.

The main property of nonterminals $S \preceq T$ is that S generates trees t with a weight that is smaller than the weight with which the same tree t is generated by T. This immediately yields that nonterminals S and T that simulate each other (i.e., $S \preceq T \preceq S$) generate t with the same weight.

Lemma 3 (see [18, Lemma 3]). *We have* $\mathrm{wt}(t, S) \sqsubseteq \mathrm{wt}(t, T)$ *for every* $t \in T_\Sigma$ *and* $S \preceq T$.

Proof. Since this property is essential for our approach, let us present full proof details. First, we remark that the semiring operations $+$ and $\cdot$ are monotone with respect to the natural order $\sqsubseteq$. Moreover, $a \sqsubseteq a + b$ for every $a, b \in A$. Second, suppose that $t = \sigma(t_1, \dots, t_k)$ for some $\sigma \in \Sigma_k$ and $t_1, \dots, t_k \in T_\Sigma$. We prove the statement by induction as follows (recall that the first line is the definition of the initial-algebra semantics):

$$\mathrm{wt}(t, S) = \sum_{S \xrightarrow{a} \sigma(S_1, \dots, S_k) \in P} a \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, S_i)$$

[3] Note that the name "backward simulation" makes more sense if we consider weighted bottom-up tree automata.

$$\begin{aligned}
&\sqsubseteq \sum_{S_1,\dots,S_k \in N} \mathrm{pwt}_\sigma(\{S\}, \uparrow S_1, \dots, \uparrow S_k) \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, S_i) \\
&\overset{\dagger}{\sqsubseteq} \sum_{S_1,\dots,S_k \in N} \mathrm{pwt}_\sigma(\{T\}, \uparrow S_1, \dots, \uparrow S_k) \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, S_i) \\
&= \sum_{\substack{S_1,\dots,S_k \in N \\ T_1 \in \uparrow S_1,\dots,T_k \in \uparrow S_k \\ T \xrightarrow{a} \sigma(T_1,\dots,T_k) \in P}} a \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, S_i) \overset{\ddagger}{\sqsubseteq} \sum_{\substack{S_1,\dots,S_k \in N \\ T_1 \in \uparrow S_1,\dots,T_k \in \uparrow S_k \\ T \xrightarrow{a} \sigma(T_1,\dots,T_k) \in P}} a \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, T_i) \\
&= \sum_{T \xrightarrow{a} \sigma(T_1,\dots,T_k) \in P} a \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, T_i) = \mathrm{wt}(t, T) \enspace ,
\end{aligned}$$

where we use $S \preceq T$ at † and k times the induction hypothesis at ‡, which is applicable because $S_i \preceq T_i$ for every $1 \leq i \leq k$. ☐

Clearly, nonterminals that simulate each other are superfluous. We can use this to reduce the number of nonterminals and the number of productions of the Wtg. Let us remark that $\{(S,T) \mid S \preceq T \preceq S\}$ is an equivalence relation on N, which we denote by $\simeq$. The equivalence class of $S \in N$ is denoted by $[S]$.

Definition 4 (see [18, Definition 6]). *The collapsed* Wtg*, denoted by $G/{\simeq}$, is (N', Σ, P', I') where*

- *$N' = \{[S] \mid S \in N\}$,*
- *P' contains, for every $\sigma \in \Sigma_k$ and $S, S_1, \dots, S_k \in N$, the production*

$$[S] \xrightarrow{\mathrm{pwt}_\sigma(\{S\}, \uparrow S_1, \dots, \uparrow S_k)} \sigma([S_1], \dots, [S_k]) \enspace ,$$

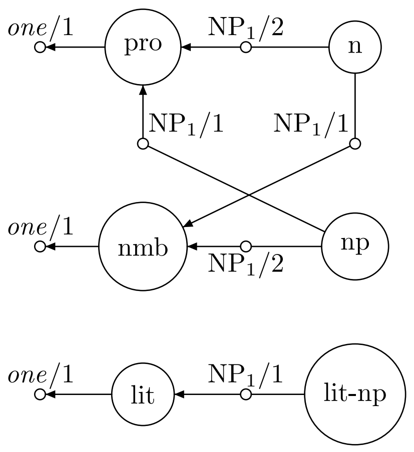
- *$I'([S]) = \sum_{T \in [S]} I(T)$ for every $S \in N$.*

An easy check shows that the weight of the production in the second item in Definition 4 is independent of the chosen representatives $S, S_1, \dots, S_k$. In addition, the collapsed Wtg $G/{\simeq}$ never has more nonterminals than G.

Theorem 5 (see [18, Theorem 7]). *The* Wtg *G and $G/{\simeq}$ are equivalent.*

Proof. This is the most important theorem of this section, so let us present some detail. Let $(G/{\simeq}) = (N', \Sigma, P', I')$, and we write wt′ for the weight computed with respect to $G/{\simeq}$. In a manner similar to the proof of Lemma 3 we first prove that $\mathrm{wt}'(t, [S]) = \mathrm{wt}(t, S)$ for every $t \in T_\Sigma$ and $S \in N$. Let $t = \sigma(t_1, \dots, t_k)$ for some $\sigma \in \Sigma_k$ and $t_1, \dots, t_k \in T_\Sigma$. Then

$$\mathrm{wt}'(t, [S]) = \sum_{[S] \xrightarrow{a'} \sigma([S_1],\dots,[S_k]) \in P'} a' \cdot \prod_{i=1}^{k} \mathrm{wt}'(t_i, [S_i])$$



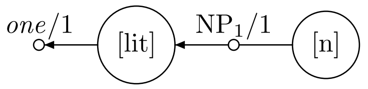


Fig. 2. The WTG (left) and the collapsed WTG (right) of Example 6.

$$
\begin{aligned}
&\overset{\dagger}{=} \sum_{S_1,\dots,S_k \in N} \mathrm{pwt}_\sigma(\{S\}, {\uparrow}S_1, \dots, {\uparrow}S_k) \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, S_i) \\
&= \sum_{\substack{S_1,\dots,S_k \in N \\ T_1 \in {\uparrow}S_1,\dots,T_k \in {\uparrow}S_k \\ S \xrightarrow{a} \sigma(T_1,\dots,T_k) \in P}} a \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, S_i) = \sum_{S \xrightarrow{a} \sigma(S_1,\dots,S_k) \in P} a \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, S_i) \\
&= \mathrm{wt}(t, S) \enspace ,
\end{aligned}
$$

where we used the induction hypothesis at †. With this auxiliary result, we immediately obtain

$$
\mathrm{wt}'(t) = \sum_{T \in N'} I'(T) \cdot \mathrm{wt}'(t, T) = \sum_{S \in N} I(S) \cdot \mathrm{wt}(t, S) = \mathrm{wt}(t) \enspace . \qquad \square
$$

In contrast to the backward simulation of [18], $G/{\simeq}$ cannot be reduced any further with the help of backward simulation, if $\preceq$ is our greatest backward simulation. Let us illustrate the definitions on a very simplistic example.

Example 6. Consider the tropical semiring $(\mathbb{R} \cup \{\infty\}, \min, +)$, and let G be such that $N = \{\mathit{pro}, \mathit{nmb}, n, \mathit{np}, \mathit{lit}, \mathit{lit\text{-}np}\}$ and $\Sigma = \{\mathrm{one}, \mathrm{NP}_1\}$, and P contains the following productions:

$$
\begin{array}{llll}
\mathit{np} \xrightarrow{1} \mathrm{NP}_1(\mathit{pro}) & n \xrightarrow{2} \mathrm{NP}_1(\mathit{pro}) & \mathit{lit\text{-}np} \xrightarrow{1} \mathrm{NP}_1(\mathit{lit}) & \mathit{lit} \xrightarrow{1} \mathrm{one} \\
\mathit{np} \xrightarrow{2} \mathrm{NP}_1(\mathit{nmb}) & n \xrightarrow{1} \mathrm{NP}_1(\mathit{nmb}) & \mathit{pro} \xrightarrow{1} \mathrm{one} & \mathit{nmb} \xrightarrow{1} \mathrm{one} \enspace .
\end{array}
$$

Finally, $I(S) = 1$ for every $S \in N$.

Let $\preceq$ be the greatest backward simulation. Then $\mathit{pro} \preceq \mathit{nmb} \preceq \mathit{lit} \preceq \mathit{pro}$ and $n \preceq \mathit{np} \preceq \mathit{lit\text{-}np} \preceq n$. The collapsed WTG is (N', Σ, P', I') with $N' = \{[\mathit{lit}], [n]\}$,

Algorithm 1 Minimization algorithm using backward simulation.

```
$R_0 \leftarrow N \times N$
$i \leftarrow 0$
**repeat**
    $j \leftarrow i$
    **for all** $\sigma \in \Sigma_k$ and $T_1, \dots, T_k \in N$, let $\mathcal{T}_1 = \uparrow_{R_i}(T_1), \dots, \mathcal{T}_k = \uparrow_{R_i}(T_k)$ and **do**
        $R_{i+1} \leftarrow \{(S,T) \in R_i \mid \mathrm{pwt}_\sigma(\{S\}, \mathcal{T}_1, \dots, \mathcal{T}_k) \sqsubseteq \mathrm{pwt}_\sigma(\{T\}, \mathcal{T}_1, \dots, \mathcal{T}_k)\}$
        $i \leftarrow i + 1$
**until** $R_i = R_j$
```

$I'(T) = 1$ for every $T \in N'$, and P' contains the two productions: $[lit] \to \mathrm{one}$ and $[n] \to \mathrm{NP}_1([lit])$ both with weight 1. The two WTG are displayed in Fig. 2.

Finally, let us develop an algorithm that computes the greatest backward simulation, which we denote by $\preceq$ for the rest of this section. Our algorithm, which is displayed in Algorithm 1, proceeds in a similar fashion as the algorithm for the greatest simulation of a labeled transition system [17] and the algorithm for the coarsest backward bisimulation of [21]. Let r be the maximal rank of symbol in Σ. Our algorithm is conceptually simple, but its run-time complexity is $\mathcal{O}(|N|^{2+r}|P|)$, which is very high compared to the algorithms of [21,17], which run in time $\mathcal{O}(|N|r|P|)$. A more efficient implementation, which utilizes the ideas of [21,17], remains a topic for further research. We compute $\preceq$ directly by refining the trivial quasi-order $Q \times Q$ iteratively. The main property that allows the refinement step is outlined in the next lemma.

Lemma 7. *Let $\sigma \in \Sigma_k$ and $\mathcal{T}_1, \dots, \mathcal{T}_k \subseteq N$ be up-sets. Then for every $S \preceq T$*

$$\mathrm{pwt}_\sigma(\{S\}, \mathcal{T}_1, \dots, \mathcal{T}_k) \sqsubseteq \mathrm{pwt}_\sigma(\{T\}, \mathcal{T}_1, \dots, \mathcal{T}_k) \enspace .$$

Proof. The proof can easily be obtained from the definitions and is omitted. □

Thus, in our algorithm we need to select up-sets (with respect to $\preceq$) and can then discard some pairs (S,T) of nonterminals such that T cannot simulate S. A simple implementation of Algorithm 1 runs in time $\mathcal{O}(|N|^{2+r}|P|)$, if we select the up-sets in order of their cardinality and reuse the already computed sums.

Theorem 8 (cf. [18, Theorem 9]). *Algorithm 1 can be implemented to run in time $\mathcal{O}(|N|^{2+r}|P|)$ and returns $R_i = \preceq$.*

Proof. The time bound is easy to obtain. For the correctness, we prove the following two statements for every i (encountered during execution): (i) $\preceq \subseteq R_i$ and (ii) $\uparrow_{R_i}(S)$ is an up-set (with respect to $\preceq$) for every $S \in N$. Let us proceed by induction on i. Both statements are true for $i = 0$ because $R_0 = N \times N$ and $\uparrow_{R_0}(S) = N$ for every $S \in N$. Now, suppose that $S \preceq T$. Then $(S,T) \in R_i$ by the induction hypothesis. Moreover, by Lemma 7, $(S,T) \in R_{i+1}$. Thus $\preceq \subseteq R_{i+1}$. Now, let $S, T, U \in N$ be such that $S \in \uparrow_{R_{i+1}}(T)$ and $S \preceq U$. Since $\preceq \subseteq R_{i+1}$, we also have $(S,U) \in R_{i+1}$ and thus $U \in \uparrow_{R_{i+1}}(T)$, which proves that $\uparrow_{R_{i+1}}(T)$ is an up-set (with respect to $\preceq$). Clearly, at termination, R_i is a backward simulation (see Definition 2). Since $\preceq \subseteq R_i$ and $\preceq$ is the greatest backward simulation for M, we can conclude that $R_i = \preceq$. $\square$

4 Forward Simulation

The previous section established a new method to reduce Wtg. However, once we reduce with the help of the greatest backward simulation, we cannot reduce the Wtg any further with the help of backward simulation. Next, we introduce an alternative procedure, which can, in principle, reduce such Wtg further. In fact, the two minimization procedures can be alternated for maximal reduction. The new procedure uses a forward version of the simulation of Sect. 3. Our forward simulation is a generalization of the forward bisimulation of [21] and the forward simulation of [18], which in turn is the weighted analogue to the composed simulations of [17]. For a definition of 'pwt' see the paragraph before Definition 2. To simplify the following discussion, we will generally omit the set braces for singleton sets.

Definition 9 (cf. [18, Definition 10]). *A quasi-order $\preceq$ on N is a* forward simulation *if for every $S \preceq T$:*

- *$I(S) \sqsubseteq I(T)$ and*
- *for every symbol $\sigma \in \Sigma_k$, all nonterminals $S', S_1, \dots, S_k \in N$, and $1 \leq i \leq k$*

$$\mathrm{pwt}_\sigma(\uparrow S', S_1, \dots, S, \dots, S_k) \sqsubseteq \mathrm{pwt}_\sigma(\uparrow S', S_1, \dots, T, \dots, S_k)$$

where S and T occur at the $(i+1)^{th}$ position.

Our definition of forward simulation is more general than the existing notions of [21,17,18]. However, we do not consider general upward simulations [17] with respect to arbitrary downward simulations here since we believe that two independent simulations (our forward simulation does not depend on a backward simulation) are easier to understand and analyze. Moreover, we can always first use backward-simulation minimization and then forward-simulation minimization to achieve roughly the same as with an upward-simulation minimization of [17]. As already remarked in the previous section, the more general the notion of simulation, the better the reduction rate (potentially at the expense of the run-time of the reduction algorithm).

Another similarity to the backward case is that our definition of forward simulation is hard to illustrate. Thus, let us proceed with the principal properties of forward simulations. As in the backward case, there exists a greatest forward simulation. This follows from the fact that the reflexive, transitive closure of the union of any two forward simulations is again a forward simulation. From now on, let $\preceq$ be a forward simulation.

The main property of similar states is slightly more complicated this time. We need an additional notion. A *context* is a tree of $T_{\Sigma \cup \{\square\}}$, where $\square$ is a new nullary symbol, such that the symbol $\Box$ occurs exactly once. The set of all contexts is denoted by C_Σ. The tree $c[t]$ is obtained by replacing the symbol $\Box$ in the context $c \in C_\Sigma$ by the tree $t \in T_\Sigma$.

Lemma 10 (see [18, Lemma 12]). *Let $c \in C_\Sigma$, $S' \in N$, and $S \preceq T$. Then*

$$\sum_{T' \in \uparrow S'} \mathrm{wt}(c[S], T') \sqsubseteq \sum_{T' \in \uparrow S'} \mathrm{wt}(c[T], T') \enspace .$$

Proof. This property is not essential for our goals, so we omit the proof. $\Box$

Again, we want to use the simulation to reduce the size of the Wtg. The definition of the collapsed WTG is slightly different this time. As before, let $\simeq$ be the equivalence relation $\{(S, T) \mid S \preceq T \preceq S\}$.

Definition 11 (see [18, Definition 13]). *The collapsed* Wtg*, which is denoted by $G/{\simeq}$, is (N', Σ, P', I') where*

- *$N' = \{[S] \mid S \in N\}$,*
- *P' contains, for every $\sigma \in \Sigma_k$ and $S, S_1, \ldots, S_k \in N$, the production*

$$[S] \xrightarrow{\mathrm{pwt}_\sigma(\uparrow S, S_1, \ldots, S_k)} \sigma([S_1], \ldots, [S_k]) \enspace ,$$

- *$I'([S]) = I(S)$ for every $S \in N$.*

It is again simple to show that the second and third item in Definition 11 are well-defined (i.e., independent of the chosen representative). Next, we prove the correctness of our construction.

Theorem 12 (see [18, Theorem 15]). *The* Wtg *G and $G/{\simeq}$ are equivalent.*

Proof. Let $(G/{\simeq}) = (N', \Sigma, P', I')$. As before, we use wt′ for weights computed using $G/{\simeq}$. As a first step, we prove that $\mathrm{wt}'(t, [S]) = \sum_{T \in \uparrow S} \mathrm{wt}(t, T)$ for every $t \in T_\Sigma$ and $S \in N$. Suppose that $t = \sigma(t_1, \ldots, t_k)$ for some $\sigma \in \Sigma_k$ and $t_1, \ldots, t_k \in T_\Sigma$. Then we compute as follows where the equality marked † is explained below.

$$\begin{aligned}
\mathrm{wt}'(t, [S]) &= \sum_{S_1, \ldots, S_k \in N} \mathrm{pwt}_\sigma(\uparrow S, S_1, \ldots, S_k) \cdot \prod_{i=1}^{k} \mathrm{wt}'(t_i, [S_i]) \\
&= \sum_{\substack{S_1, \ldots, S_k \in N \\ T_1 \in \uparrow S_1, \ldots, T_k \in \uparrow S_k}} \mathrm{pwt}_\sigma(\uparrow S, S_1, \ldots, S_k) \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, T_i) \\
&\overset{\dagger}{=} \sum_{T_1, \ldots, T_k \in N} \mathrm{pwt}_\sigma(\uparrow S, T_1, \ldots, T_k) \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, T_i) = \sum_{T \in \uparrow S} \mathrm{wt}(t, T) \enspace .
\end{aligned}$$

Let us take a closer look at the equation marked †. We can show this equality by showing both directions. Let us consider $\sqsubseteq$ first. Clearly, it is sufficient to show that for each summand of the left-hand side there exists a larger summand in the right-hand side. For this we consider a summand

$$\mathrm{pwt}_\sigma(\uparrow S, S_1, \dots, S_k) \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, T_i)$$

of the left-hand side of † for some nonterminals $S_1, \dots, S_k, T_1, \dots, T_k \in N$ such that $T_i \in \uparrow S_i$ for every $1 \leq i \leq k$. By Definition 9 and $S_i \preceq T_i$ we have $\mathrm{pwt}_\sigma(\uparrow S, S_1, \dots, S_k) \sqsubseteq \mathrm{pwt}_\sigma(\uparrow S, T_1, \dots, T_k)$. Consequently,

$$\mathrm{pwt}_\sigma(\uparrow S, S_1, \dots, S_k) \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, T_i) \sqsubseteq \mathrm{pwt}_\sigma(\uparrow S, T_1, \dots, T_k) \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, T_i)$$

and the latter is a summand on the right-hand side of †. For the converse, let us consider a summand $\mathrm{pwt}_\sigma(\uparrow S, T_1, \dots, T_k) \cdot \prod_{i=1}^{k} \mathrm{wt}(t_i, T_i)$ in the right-hand side where $T_1, \dots, T_k \in N$. Then this is clearly also a summand of the left-hand side of † (by setting $S_i = T_i$). This completes the proof of our auxiliary statement. For the statement of the theorem, we compute as follows:

$$\begin{aligned} \mathrm{wt}'(t) &= \sum_{S \in N} I'([S]) \cdot \mathrm{wt}'(t, [S]) = \sum_{S \in N} I(S) \cdot \Big(\sum_{T \in \uparrow S} \mathrm{wt}(t, T) \Big) \\ &= \sum_{S \in N} I(S) \cdot \mathrm{wt}(t, S) = \mathrm{wt}(t) \end{aligned}$$

because $I(S) \sqsubseteq I(T)$ if $S \preceq T$. This proves our theorem. □

As in the backward case, the WTG obtained by reducing with respect to the greatest forward simulation cannot be reduced any further with the help of forward simulation. Let us look at an example for illustration.

Example 13. Consider the original WTG of Example 6. Let $\preceq$ be the greatest forward simulation. Then *pro* $\preceq$ *nmb* $\preceq$ *lit* $\preceq$ *pro* and *n* $\preceq$ *np* $\preceq$ *lit-np* $\preceq$ *n*. The reduced WTG coincides with the one of Example 6. Note that this is not a general property, but we rather chose a WTG with this property to save space.

Now let us develop an algorithm (see Algorithm 2) for the greatest forward simulation, which we denote by $\preceq$ for the rest of the section. It will run in time $\mathcal{O}(|N|^3 r|P|)$, which is high compared to the run-time $\mathcal{O}(|N| r|P|)$ of the preceding algorithms [21,17]. The initial weight of a nontermial T that simulates S must be larger than that of S. Thus, we start with this restricted quasi-order in Algorithm 2. Again, we use a simple property that allows us to refine iteratively.

Lemma 14. *Let $\sigma \in \Sigma_k$, $S_1, \dots, S_k \in N$, $1 \leq i \leq k$, and $\mathcal{T} \subseteq N$ be an up-set. Then* $\mathrm{pwt}_\sigma(\mathcal{T}, S_1, \dots, S, \dots, S_k) \sqsubseteq \mathrm{pwt}_\sigma(\mathcal{T}, S_1, \dots, T, \dots, S_k)$ *for every $S \preceq T$ where S and T occur at the $(i+1)^{th}$ position.*

Proof. The proof can be obtained easily from Definition 9. □

Algorithm 2 Minimization algorithm using forward simulation.

```
$R_0 \leftarrow \{(S, T) \in N \times N \mid I(S) \sqsubseteq I(T)\}$
$i \leftarrow 0$
**repeat**
    $j \leftarrow i$
    **for all** $\sigma \in \Sigma_k$, $n \in \{1, \dots, k\}$, and $S', S_1, \dots, S_k \in N$, let $T' = \uparrow_{R_i}(S')$ and
    **do**
        $R_{i+1} \leftarrow \{(S, T) \in R_i \mid \mathrm{pwt}_\sigma(T', \dots, S, \dots) \sqsubseteq \mathrm{pwt}_\sigma(T', \dots, T, \dots)\}$
        $i \leftarrow i + 1$
**until** $R_i = R_j$
```

Theorem 15 (see [18, Theorem 19]). *Algorithm 2 can be implemented to run in time $\mathcal{O}(|N|^3 r|P|)$ and returns $R_i = \preceq$.*

Proof. Again, the given time bound is easy to obtain. We prove the following two statements for every relevant i: (i) $\preceq \subseteq R_i$ and (ii) $\uparrow_{R_i}(S')$ is an up-set (with respect to $\preceq$) for every $S' \in N$. By the first condition of Definition 9 we have $\preceq \subseteq R_0$. Moreover, if $\preceq \subseteq R_i$, then $\uparrow_{R_i}(S')$ is an up-set with respect to $\preceq$ for every $S' \in N$ (see the proof of Theorem 8). Now, suppose that $S \preceq T$. Then $(S, T) \in R_i$ by the induction hypothesis. Moreover, by Lemma 14 we also have $(S, T) \in R_{i+1}$ because $\uparrow_{R_i}(S')$ is an up-set (with respect to $\preceq$). Thus $\preceq \subseteq R_{i+1}$. Clearly, R_i is a forward simulation (see Definition 9) at termination. Since $\preceq \subseteq R_i$, we can conclude that $R_i = \preceq$. □

Conclusion

We introduced the most general simulation relations for weighted tree automata, which generalize all the existing notions of [21,17,26]. Such simulations enjoy the theoretical properties we expect, but the computation of the greatest backward and forward simulation is significantly more expensive than the corresponding computations for the notions of [21,17,26]. Earlier work in [21,17,26] reports reductions of 7–76.1% (with an average around 50%) with less general forms of (bi)simulation, so our work will result in at least as much reduction. Future implementation work will be undertaken to verify how much our approach improves upon those results.

References

1. Dale, R., Moisl, H., Somers, H.L., eds.: Handbook of Natural Language Processing. CRC Press (2000)
2. Abdulla, P.A., Jonsson, B., Mahata, P., d'Orso, J.: Regular tree model checking. In: Proc. CAV. Volume 2404 of LNCS., Springer (2002) 555–568
3. Knight, K., Graehl, J.: An overview of probabilistic tree transducers for natural language processing. In: Proc. CICLing. Volume 3406 of LNCS., Springer (2005) 1–24

4. Klarlund, N., Møller, A.: MONA Version 1.4 User Manual. BRICS, Department of Computer Science, University of Aarhus. (2001)
5. May, J., Knight, K.: Tiburon: A weighted tree automata toolkit. In: Proc. CIAA. Volume 4094 of LNCS., Springer (2006) 102–113
6. Cleophas, L.: Forest FIRE and FIRE wood: Tools for tree automata and tree algorithms. In: Proc. FSMNLP. (2008) 191–198
7. Berstel, J., Reutenauer, C.: Recognizable formal power series on trees. Theoret. Comput. Sci. **18**(2) (1982) 115–148
8. Ésik, Z., Kuich, W.: Formal tree series. J. Autom. Lang. Combin. **8**(2) (2003) 219–285
9. Borchardt, B.: The Theory of Recognizable Tree Series. PhD thesis, Technische Universität Dresden (2005)
10. Hopcroft, J.E.: An $n \log n$ algorithm for minimizing states in a finite automaton. In: Theory of Machines and Computations. Academic Press (1971) 189–196
11. Högberg, J., Maletti, A., May, J.: Backward and forward bisimulation minimisation of tree automata. In: Proc. CIAA. Volume 4783 of LNCS., Springer (2007) 109–121
12. Maletti, A.: Minimizing deterministic weighted tree automata. Inform. and Comput. **207**(11) (2009) 1284–1299
13. Meyer, A.R., Stockmeyer, L.J.: The equivalence problem for regular expressions with squaring requires exponential space. In: Proc. FOCS, IEEE Computer Society (1972) 125–129
14. Gramlich, G., Schnitger, G.: Minimizing nfa's and regular expressions. J. Comput. System Sci. **73**(6) (2007) 908–923
15. Gruber, H., Holzer, M.: Inapproximability of nondeterministic state and transition complexity assuming P $\neq$ NP. In: Proc. DLT. Volume 4588 of LNCS., Springer (2007) 205–216
16. Abdulla, P.A., Högberg, J., Kaati, L.: Bisimulation minimization of tree automata. Int. J. Found. Comput. Sci. **18**(4) (2007) 699–713
17. Abdulla, P.A., Bouajjani, A., Holík, L., Kaati, L., Vojnar, T.: Computing simulations over tree automata. In: Proc. TACAS. Volume 4963 of LNCS., Springer (2008) 93–108
18. Maletti, A.: A backward and a forward simulation for weighted tree automata. In: Proc. CAI. Volume 5725 of LNCS., Springer (2009) 288–304
19. Bozapalidis, S., Louscou-Bozapalidou, O.: The rank of a formal tree power series. Theoret. Comput. Sci. **27**(1–2) (1983) 211–215
20. Bozapalidis, S.: Effective construction of the syntactic algebra of a recognizable series on trees. Acta Inform. **28**(4) (1991) 351–363
21. Högberg, J., Maletti, A., May, J.: Bisimulation minimisation for weighted tree automata. In: Proc. DLT. Volume 4588 of LNCS., Springer (2007) 229–241
22. Huang, L., Chiang, D.: Better k-best parsing. In: Proc. IWPT. (2005) 53–64
23. Alexandrakis, A., Bozapalidis, S.: Weighted grammars and Kleene's theorem. Information Processing Letters **24**(1) (1987) 1–4
24. Bozapalidis, S.: Equational elements in additive algebras. Theory Comput. Systems **32**(1) (1999) 1–33
25. Borchardt, B., Vogler, H.: Determinization of finite state weighted tree automata. J. Autom. Lang. Combin. **8**(3) (2003) 417–463
26. Abdulla, P.A., Holík, L., Kaati, L., Vojnar, T.: A uniform (bi-)simulation-based framework for reducing tree automata. In: Proc. MEMICS. (2008) 3–11