International Journal of Foundations of Computer Science
© World Scientific Publishing Company

The Power of Weighted Regularity-Preserving Multi Bottom-up Tree Transducers*

ANDREAS MALETTI†

Universität Leipzig, Institute of Computer Science
Augustusplatz 10–11, 04109 Leipzig, Germany
maletti@informatik.uni-leipzig.de

Received (20 December 2014)
Communicated by (xxxxxxxxxx)

The expressive power of regularity-preserving ε-free weighted linear multi bottom-up tree transducers is investigated. These models have very attractive theoretical and algorithmic properties, but their expressive power is not well understood especially in the weighted setting. It is proved that despite the restriction to preserve regularity their power still exceeds that of composition chains of ε-free weighted linear extended top-down tree transducers with regular look-ahead, which are a natural super-class of weighted synchronous tree substitution grammars that are commonly used in statistical machine translation. In particular, the linguistically motivated discontinuous transformation of topicalization can be modeled by such multi bottom-up tree transducers, whereas composition chains of such extended top-down tree transducers cannot implement it. On the negative side, the inverse of topicalization cannot even be implemented by any such multi bottom-up tree transducers, which confirms their bottom-up nature. An interesting, promising, and widely applicable proof technique is used to prove those statements.

1. Introduction

The area of statistical machine translation [24] deals with the automatic translation of natural language texts from one language into another. The central component of each such system is the *translation model*, which determines which transformations are possible in the system. The translation model is supported by additional models (such as language models), which we will not discuss here. Currently, several translation models are popular: (i) phrase-based systems [33], which use a finite-state transducer [10], (ii) hierarchical phrase-based systems [7], which use a synchronous context-free grammar, and (iii) syntax-based systems, which use a form of synchronous tree grammar such as synchronous tree substitution grammars [11], syn-

*This article is an extended and revised version of [Maletti: *The power of regularity-preserving multi bottom-up tree transducers*. In Proc. Conference on Implementation and Application of Automata, LNCS 8587, pages 278–289, 2014].

†The author was financially supported of the German Research Foundation (DFG) grant MA / 4959 / 1-1.

chronous tree-adjoining grammars [34], or synchronous tree-sequence substitution grammars [35]. All these models are weighted (typically over the field of rational numbers) in order to help resolve the ambiguity that is inherent in translation. Since those systems are trained on huge data sets, the used translation model must meet two contradictory goals. On the one hand, it should have nice algorithmic properties and its key operations should have (very) low computational complexity. On the other hand, its expressive power should be high in order to be able to model all typical phenomena (complex reorderings, etc.) that occur in translation. The mentioned models cover a wide spectrum along these axes. Thus an essential part of model evaluation for machine translation is the accurate determination of the expressive power and the complexity of the key operations [23].

A relatively recent proposal for another translation model suggests the linear multi bottom-up tree transducer (MBOT) [27, 29], which can be understood as an extension of STSG that allows discontinuity on the output side. It was already demonstrated [27, 29] that MBOTs have very good theoretical and algorithmic properties in comparison to STSG, so they were implemented [5] in the machine translation framework Moses [25]. In experiments the new model significantly outperforms a STSG baseline when evaluated in a standard English-to-German translation task. In general, MBOTs have the power of *finite copying* [13], which yields that their output tree languages are not necessarily regular [18, 19, 17] (or: the output string languages are not necessarily context-free), which was proved in [12, 20]. As expected, this increase in power comes at the price of worse algorithmic properties. However, it is currently unclear whether this added complexity is necessary to model common discontinuities like topicalization [8].

Consequently, we restrict our MBOTs such that they only produce regular (weighted) output tree languages and demonstrate that this model retains the power to compute common discontinuities. In addition, these models remain more powerful than arbitrary composition chains [32] of STSG, which is demonstrated on the example of topicalization. Whereas STSG can trivially be inverted, neither MBOTs nor regularity-preserving MBOTs can be inverted in general. In fact, we show that the inverse of topicalization cannot be implemented by any MBOT. Overall, these results help us to relate the expressive power of regularity-preserving MBOT to the other classes (see Figure 7). In the unweighted setting, the relation is quite clear, but in the weighted setting the relation remains unclear, although we manage to prove the mentioned separation. Additionally, we want to promote the use of explicit links [30, 16], which naturally record which parts of the input and output tree developped synchronously in a derivation step [3]. We investigate the properties of these links and then use those properties to prove our main results. In fact, the proofs of our main statements split into a standard technical part that establishes certain mandatory links with the help of the linking theorems of [16] and a rather straightforward high-level argumentation that refutes that the obtained link ensemble is well-formed. We believe that this proof method holds much potential and can successfully be applied to many similar scenarios.

2. Preliminaries

We use $\mathbb{N}$ for the set of all nonnegative integers. Let S and T be two sets and $\rho \subseteq S \times T$ be a relation. For every $S' \subseteq S$ we let $\rho(S') = \{t \in T \mid \exists s \in S' \colon (s,t) \in \rho\}$ be the set of those elements of T that are related to an element of S'. Moreover, the inverse of ρ is the relation $\rho^{-1} \subseteq T \times S$ given by $\rho^{-1} = \{(t,s) \mid (s,t) \in \rho\}$. For every $s \in S$ we write $\rho(s)$ instead of $\rho(\{s\})$. Given another relation $\tau \subseteq T \times U$, the composition of ρ followed by τ is the relation $\rho \,;\tau \subseteq S \times U$ defined by

$$\rho \,; \tau = \{(s,u) \mid \rho(s) \cap \tau^{-1}(u) \neq \emptyset\} \enspace .$$

We continue to use the set S. The k-fold CARTESIAN product of S with itself is denoted by S^k. Note that $S^0 = \{\varepsilon\}$. We let $S^* = \bigcup_{k \in \mathbb{N}} S^k$, which is the set of all (finite) words with letters from S. Thus, for every $w \in S^*$ there exists (a unique) $k \in \mathbb{N}$ such that $w \in S^k$; this integer k is called the length of w and denoted by $|w|$. Let $v = (v_1, \dots, v_k) \in S^*$ and $w = (w_1, \dots, w_n) \in S^*$ be two words of length k and n, respectively. The concatenation of v with w is $(v_1, \dots, v_k, w_1, \dots, w_n) \in S^*$. Whenever confusion is impossible, we simply write $w_1 \cdots w_n$ instead of $(w_1, \dots, w_n)$ and thus concatenation is simply vw or $v.w$, when we want to stress the separation into letters. Given two languages $L, L' \subseteq S^*$, we let $LL' = \{vw \mid v \in L,\, w \in L'\}$ be the set of all concatenations of words from L by words from L'. An alphabet Σ is a nonempty and finite set of symbols. Given an alphabet Σ, the set $T_\Sigma(S)$ of Σ-trees indexed by S is the smallest set T such that the following two conditions hold: (i) $S \subseteq T$ and (ii) $\sigma(t_1, \dots, t_k) \in T$ for all $k \in \mathbb{N}$, $\sigma \in \Sigma$, and $t_1, \dots, t_k \in T$. For simplicity, we commonly write T_Σ instead of $T_\Sigma(\emptyset)$.

Next we recall some common notions for trees. In the following, let Σ be an alphabet and S be a set. To avoid confusion, we assume that $\Sigma \cap S = \emptyset$, which allows us to write just σ instead of the tree $\sigma()$ for every $\sigma \in \Sigma$. Let $t \in T_\Sigma(S)$ be a tree. The set $\mathrm{pos}(t) \subseteq \mathbb{N}^*$ of all positions in t is recursively defined by (i) $\mathrm{pos}(s) = \{\varepsilon\}$ for every $s \in S$ and (ii) $\mathrm{pos}(\sigma(t_1, \dots, t_k)) = \{\varepsilon\} \cup \{i.w \mid 1 \leq i \leq k,\, w \in \mathrm{pos}(t_i)\}$ for every $k \in \mathbb{N}$, $\sigma \in \Sigma$, and $t_1, \dots, t_k \in T_\Sigma(S)$. The size $|t|$ of t is the number of its positions [i.e., $|t| = |\mathrm{pos}(t)|$], and the height $\mathrm{ht}(t)$ of t is the length of a maximal (in terms of length) position in $\mathrm{pos}(t)$ [i.e., $\mathrm{ht}(t) = \max\,\{|w| \mid w \in \mathrm{pos}(t)\}$]. We assume the standard ordering on $\mathbb{N}$, which extends to the lexicographic order $\sqsubseteq$ on $\mathbb{N}^*$, which is a linear order. This linear order $\sqsubseteq$ allows us to turn every finite set $W \subseteq \mathbb{N}^*$ into a (unique) word $\vec{W} \in (\mathbb{N}^*)^*$, which is given by $\vec{W} = (w_1, \dots, w_n)$ if $W = \{w_1, \dots, w_n\}$ with $w_1 \sqsubset \cdots \sqsubset w_n$, where $\sqsubset$ is the strict version of $\sqsubseteq$ as usual. In addition, we use the prefix order $\leq$ on $\mathbb{N}^*$, which for all $v, w \in \mathbb{N}^*$ is defined by $v \leq w$ if and only if there exists $u \in \mathbb{N}^*$ such that $vu = w$.

Now let us recall some essential operations on trees. As before, let Σ be an alphabet and S be a set. In addition, let $t, u \in T_\Sigma(S)$ be trees and $w \in \mathrm{pos}(t)$ be a position in t. The label of t at w is denoted by $t(w)$, the subtree of t rooted in w

is denoted by $t|_w$, and the tree obtained from t by replacing the subtree at w by u is denoted by $t[u]_w$. Formally, these are defined recursively for every $s \in S$, $k \in \mathbb{N}$, $\sigma \in \Sigma$, and $t_1, \dots, t_k \in T_\Sigma(S)$ as follows:

$$s(\varepsilon) = s \qquad \big(\sigma(t_1, \dots, t_k)\big)(w) = \begin{cases} \sigma & \text{if } w = \varepsilon \\ t_i(w') & \text{if } w = i.w' \end{cases}$$

$$s|_\varepsilon = s \qquad \big(\sigma(t_1, \dots, t_k)\big)|_w = \begin{cases} \sigma(t_1, \dots, t_k) & \text{if } w = \varepsilon \\ t_i|_{w'} & \text{if } w = i.w' \end{cases}$$

$$s[u]_\varepsilon = u \quad \big(\sigma(t_1, \dots, t_k)\big)[u]_w = \begin{cases} u & \text{if } w = \varepsilon \\ \sigma(t_1, \dots, t_{i-1}, t_i[u]_{w'}, t_{i+1}, \dots, t_k) & \text{if } w = i.w' \end{cases}$$

where $i \in \mathbb{N}$ in those definitions. For every $s \in S$, the set $\mathrm{pos}_s(t)$ of s-labeled positions in t is given by $\mathrm{pos}_s(t) = \{w \in \mathrm{pos}(t) \mid t(w) = s\}$. The tree t is linear if $|\mathrm{pos}_s(t)| \leq 1$ for every $s \in S$, and $\mathrm{idx}(t) = \{s \in S \mid \mathrm{pos}_s(t) \neq \emptyset\}$ is the set of indexes that occur in t. Next, we extend the substitution operation $t[u]_w$ to words of trees and positions. Let $\vec{u} = (u_1, \dots, u_n)$ be a word of length n such that $u_i \in T_\Sigma(S)$ is a tree for all $1 \leq i \leq n$. Correspondingly, let $\vec{w} = (w_1, \dots, w_n)$ be a word of length n such that $w_i \in \mathrm{pos}(t)$ is a position in t for all $1 \leq i \leq n$. Additionally, we demand that the elements of $\{w_1, \dots, w_n\}$ are pairwise incomparable with respect to the prefix order $\leq$. Then $t[\vec{u}]_{\vec{w}}$ denotes the tree obtained from t by replacing the subtree at w_i by u_i for all $1 \leq i \leq n$. Formally, we have $t[\vec{u}]_{\vec{w}} = (\cdots (t[u_1]_{w_1}) \cdots)[u_n]_{w_n}$. A solid introduction into trees, their main notions, and their operations can be found in [18, 19, 9].

Finally, we recall the algebraic structures, from which we draw our weights, and weighted tree automata, which recognize the regular weighted tree languages. A commutative semiring $(A, +, \cdot, 0, 1)$ is an algebraic structure consisting of two commutative monoids $(A, +, 0)$ and $(A, \cdot, 1)$ such that (i) $0 \cdot a = 0$ for all $a \in A$, and (ii) $a \cdot (a_1 + a_2) = (a \cdot a_1) + (a \cdot a_2)$ for all $a, a_1, a_2 \in A$. Typical commutative semirings include that natural numbers $(\mathbb{N}, +, \cdot, 0, 1)$ and $([0, 1], \max, \cdot, 0, 1)$, where $[0, 1] = \{x \in \mathbb{R} \mid 0 \leq x \leq 1\}$. A detailed introduction into semirings is provided by [22, 21]. A weighted tree automaton is a tuple $\mathcal{A} = (Q, \Sigma, q_0, R, \mathrm{wt})$, where Q is a finite set of states, Σ an alphabet, $q_0 \in Q$ is an initial state, R is a finite set of rules of the form $q \to \sigma(q_1, \dots, q_k)$ for some $k \in \mathbb{N}$, $q, q_1, \dots, q_k \in Q$, and $\sigma \in \Sigma$, and $\mathrm{wt}\colon R \to A$ is a weight function. For such a weighted tree automaton $\mathcal{A}$, we define a function $h_\mathcal{A}\colon T_\Sigma \times Q \to A$ recursively as follows:

$$h_\mathcal{A}\big(\sigma(t_1, \dots, t_k), q\big) = \sum_{q_1, \dots, q_k \in Q} \mathrm{wt}\big(q \to \sigma(q_1, \dots, q_k)\big) \cdot \prod_{i=1}^{k} h_\mathcal{A}(t_i, q_i)$$

for all $k \in \mathbb{N}$, $\sigma \in \Sigma$, and $t_1, \dots, t_k \in T_\Sigma$. A weighted tree language is simply a mapping $\varphi\colon T_\Delta \to A$ for some alphabet Δ. The weighted tree automaton $\mathcal{A}$ recognizes the weighted tree language $\varphi_\mathcal{A}\colon T_\Sigma \to A$ such that $\varphi_\mathcal{A}(t) = h_\mathcal{A}(t, q_0)$ for all $t \in T_\Sigma$. The weighted tree language $\varphi\colon T_\Sigma \to A$ is regular [2] if there exists a weighted tree automaton $\mathcal{A}$ such that $\varphi = \varphi_{\mathcal{A}}$. A detailed introduction into regular weighted tree languages can be found in [17].

3. Multi bottom-up tree transducers

Our main model is the weighted linear multi bottom-up tree transducer (MBOT), whose unweighted variant was introduced in [26, 1]. An English presentation of the unweighted model and some recent results for it can be found in [12]. The weighted model is discussed in [28]. Contrary to [12, 28] we present MBOTs as synchronous grammars [6], which was elaborated for a stateless variant already in [29]. The distinguishing feature of MBOTs compared to traditional linear bottom-up tree transducers [17] is the potential to have several output tree fragments per rule. Consequently, each MBOT rule specifies an input tree fragment together with potentially several output tree fragments. We essentially follow the definition of [30], but add a weight function. In order to avoid repetition, we henceforth assume that $(A, +, \cdot, 0, 1)$ is a commutative semiring. In illustrations we will commonly use the semiring $(\mathbb{R}, +, \cdot, 0, 1)$ of real numbers.

Definition 1. A *weighted linear multi bottom-up tree transducer* (for short: MBOT) is a tuple $M = (Q, \Sigma, q_0, R, \mathrm{wt})$, where

- Q is the finite set of *states*,
- Σ is the alphabet of *input* and *output symbols*,
- $q_0 \in Q$ is the *initial* state,
- $R \subseteq T_\Sigma(Q) \times Q \times T_\Sigma(Q)^*$ is the finite set of *rules* such that for every rule $\langle \ell, q, (r_1, \dots, r_n) \rangle \in R$ we have that
  - ℓ is linear and
  - $\mathrm{idx}(r_1) \cup \cdots \cup \mathrm{idx}(r_n) \subseteq \mathrm{idx}(\ell)$
- $\mathrm{wt} \colon R \to A$ is the *weight function*.

Such an MBOT M is *ε-free* if $\ell \notin Q$ for all rules $\langle \ell, q, v \rangle \in R$. Moreover, if $n \leq 1$ (respectively, $n = 1$) for all rules $\langle \ell, q, (r_1, \dots, r_n) \rangle \in R$, then M is a linear *extended top-down tree transducer with regular look-ahead* (l-XTOP$^{\mathrm{R}}$) [31] (respectively, a linear *nondeleting extended top-down tree transducer* [ln-XTOP]). ◇

Let $M = (Q, \Sigma, q_0, R, \mathrm{wt})$ be an arbitrary MBOT. Each rule $\langle \ell, q, v \rangle \in R$ such that $\ell \in Q$ is called ε-rule. Clearly, an ε-free MBOT has no ε-rules. Since ε-rules can be used to create arbitrarily long derivations, which are problematic in the weighted setting (since this would require infinite summation), we exclude them and only consider ε-free MBOTs. Henceforth, let $M = (Q, \Sigma, q_0, R, \mathrm{wt})$ be an ε-free MBOT.

To allow a simpler discussion, we call ℓ and v of a rule $\langle \ell, q, v \rangle \in R$ the *left-* and *right-hand side*, respectively, and correspondingly write the rule as $\ell \stackrel{q}{\text{---}} v$. The state q is called the *governing state* of the rule. The rules of our running example

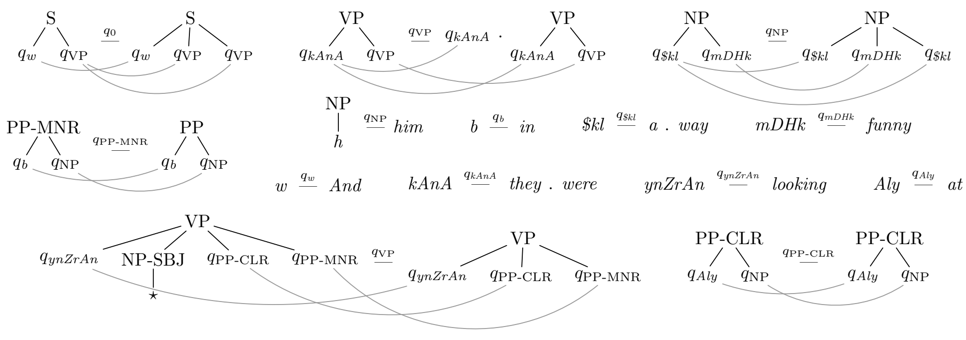

Figure 1. Example rules.

MBOT are displayed in Figure 1 (where we ignore the splines connecting the states for the moment).

Example 2. Our running example MBOT $M_{\mathrm{ex}} = (Q, \Sigma, q_0, R, \mathrm{wt})$ is given by

- $Q = \{q_0\} \cup \{q_\sigma \mid \sigma \in \Sigma\}$,
- $\Sigma = \{\mathrm{S}, \mathrm{VP}, \dots, \mathrm{funny}, \dots, \mathrm{PP\text{-}CLR}, \dots, \mathrm{mDHk}, \mathrm{kAnA}, \dots\}$,
- R contains exactly the rules displayed in Figure 1, and
- $\mathrm{wt}(\rho) = 0.5$ for all $\rho \in R$.

The first (top, left) rule in Figure 1 is textually represented by

$$\Big\langle \mathrm{S}(q_w, q_{\mathrm{VP}}),\ q_0,\ \big(\mathrm{S}(q_w, q_{\mathrm{VP}}, q_{\mathrm{VP}})\big) \Big\rangle \ . \qquad \diamond$$

Next, we present a top-down semantics for our MBOT in the style of [30]. The splines, representing *links*, connecting the state occurrences in Figure 1 already indicate that certain states are supposed to develop synchronously. In the rules these links exist implicitly (in the sense that they are not syntactically represented; see the rule in Example 2), but all occurrences of a single state should develop synchronously. For this reason, we connect all occurrences of a state in the output tree fragments with the unique occurrence of the same state in the input tree fragment. However, in our sentential forms we need to make these links explicit. Formally, such links are simply pairs of positions consisting of a position of the current input tree fragment and a position of the current output tree fragment. Our *link structure* is simply a set of such links. Although the link structure is an overhead (since it is not required for other equivalent semantics [12, 29]), all our later formal arguments are based on this link structure. In fact, we believe that its presence simplifies many traditional arguments, and thus want to promote a detailed investigation into its properties. Next, we define the link structure induced by a given rule.

Definition 3. *For all rules $\ell \xrightarrow{p} (r_1, \dots, r_n) \in R$ and $w, w_1, \dots, w_n \in \mathbb{N}^*$, we define* *the* link structure $\mathrm{links}_{w,(w_1,\dots,w_n)}(\ell \stackrel{p}{—} (r_1,\dots,r_n)) \subseteq \mathbb{N}^* \times \mathbb{N}^*$ *by*

$$\mathrm{links}_{w,(w_1,\dots,w_n)}(\ell \stackrel{p}{—} (r_1,\dots,r_n)) = \bigcup_{\substack{q\in Q\\ 1\leq i\leq n}} \Big(\{w\}\,\mathrm{pos}_q(\ell)\Big) \times \Big(\{w_i\}\,\mathrm{pos}_q(r_i)\Big) \ . \quad \diamond$$

Roughly speaking, the set $\mathrm{links}_{w,(w_1,\dots,w_n)}(\ell \stackrel{p}{—} (r_1,\dots,r_n))$ relates an occurrence (i.e., position) of a state q with each occurrence of it in the output tree fragments. However, it additionally prefixes w to each position in the input tree fragment ℓ and w_i to each position from the output tree fragment r_i. In our derivation semantics, these arguments $w, w_1, \dots, w_n$ hold the positions at which the rule fragments are applied.

Another peculiarity of our approach is the distinction of two types of links. Namely, we consider both *active* and *inactive* links. Active links are links that have not been used in a derivation step and thus record state occurrences that should develop synchronously in the following derivation steps. On the other hand, inactive links are those links that have already been used in a derivation step. We record those for reference, since we want to reason about them later on. Consequently, our sentential forms consist of four components: (i) an input tree fragment, (ii) active links, (iii) inactive links, and (iv) an output tree fragment. Formally, a sentential form is a tuple $\langle t, L, I, u\rangle$ with $t, u \in T_\Sigma(Q)$ and $L, I \subseteq \mathrm{pos}(t)\times\mathrm{pos}(u)$. The elements of L are called *active* links and those of I are called *inactive* links.

Now we can explain a derivation step. Let $\xi = \langle t, L, I, u\rangle$ and $\zeta = \langle t', L', I', u'\rangle$ be sentential forms. Roughly speaking, to derive ζ from ξ (in a single step) we select the lexicographically smallest position w in t that is labeled by a state together with the positions $L(w)$ [i.e., the output tree positions that are actively linked to w]. Finally, we select a rule $\ell \stackrel{q}{—} (r_1,\dots,r_n) \in R$ that (i) has the right governing state $q = t(w)$ and (ii) has the right number $n = |L(w)|$ of output tree fragments. We obtain $t' = t[\ell]_w$ by substituting the input tree fragment of the rule into the selected position, and similarly we obtain $u' = u[(r_1,\dots,r_n)]_{L(\vec{w})}$ by substituting the (potentially several) output tree fragments into the linked positions in u. The used links $U = \{(w, w') \mid w' \in L(w)\}$ are added to the set I of inactive links (i.e., $I' = I \cup U$) and removed from the set L of active links. However, the links derived from the applied rule still need to be made new active links [i.e., $L' = (L \setminus U) \cup \mathrm{links}_{w,L(\vec{w})}(\ell \stackrel{p}{—} (r_1,\dots,r_n))$].

Definition 4. Let $\mathcal{SF}(M)$ be the set of all sentential forms. Given two sentential forms $\langle t, L, I, u\rangle, \langle t', L', I', u'\rangle \in \mathcal{SF}(M)$ we write $\langle t, L, I, u\rangle \Rightarrow_M \langle t', L', I', u'\rangle$, if

- $\rho = t'|_w \stackrel{t(w)}{—} (u'|_{w_1},\dots,u'|_{w_n}) \in R$
- $L' = (L\setminus U)\cup\mathrm{links}_{w,L(\vec{w})}(\rho)$ and $I' = I\cup U$ with $U = \{(w, w') \mid w' \in L(w)\}$,

where $w = \min_\sqsubseteq\big(\bigcup_{p\in Q}\mathrm{pos}_p(t)\big)$ is the (lexicographically) smallest state-labeled position in t and $L(\vec{w}) = (w_1,\dots,w_n)$. Whenever we want to stress the used rule ρ,

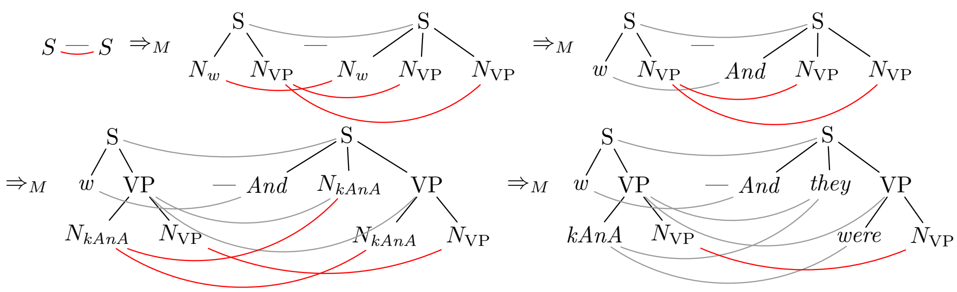

Figure 2. A few derivation steps using the rules of Figure 1. The weight of this derivation is 0.5^4. The active links are clearly marked, whereas inactive links are light.

then we write $\langle t, L, I, u\rangle \Rightarrow^{\rho}_{M} \langle t', L', I', u'\rangle$. $\diamond$

Note that the rule used to derive one sentential form from another is unique. In graphical representations we only present the input and output tree fragments and illustrate the active and inactive links as clear and light splines, respectively. A few derivation steps using the rules of Figure 1 are presented in Figure 2. Each MBOT computes a weighted tree transformation, which is simply a mapping $\tau \colon T_\Delta \times T_\Delta \to A$ for some alphabet Δ.

Definition 5. A *derivation* is a word $(\xi_0, \dots, \xi_k) \in \mathcal{SF}(M)^*$ such that $\xi_0 \Rightarrow_M \xi_1 \Rightarrow_M \cdots \Rightarrow_M \xi_k$. For each derivation $(\xi_0, \dots, \xi_k)$ the unique rule word $(\rho_1, \dots, \rho_k) \in R^*$ such that $\xi_{i-1} \Rightarrow^{\rho_i}_{M} \xi_i$ for all $1 \leq i \leq k$ is called its rule word. The weight $\mathrm{wt}(d) \in A$ of the derivation d is $\mathrm{wt}(d) = \prod_{i=1}^{k} \mathrm{wt}(\rho_i)$, where $(\rho_1, \dots, \rho_k)$ is the rule word of d. For every $t, u \in T_\Sigma$, let $\mathcal{D}_M(t, u)$ be the set of all derivations $(\xi_0, \dots, \xi_k)$ such that

- $\xi_0 = \langle q_0, \{(\varepsilon, \varepsilon)\}, \emptyset, q_0\rangle$, which is the initial sentential form, and
- $\xi_k = \langle t, \emptyset, I, u\rangle$ for some $I \subseteq \mathrm{pos}(t) \times \mathrm{pos}(u)$.

The MBOT M computes the weighted tree transformation $\tau_M \colon T_\Sigma \times T_\Sigma \to A$ such that $\tau_M(t, u) = \sum_{d \in \mathcal{D}_M(t,u)} \mathrm{wt}(d)$ for every $t, u \in T_\Sigma$. $\diamond$

Note that due to the ε-freeness, the set $\mathcal{D}_M(t, u)$ is finite for all $t, u \in T_\Sigma$ (because each applied rule replaces a state in the input tree fragment by the input tree fragment of the rule, which contains at least one non-state symbol, and the overall size of the input tree fragment is bound by $|t|$). In fact, for every $t \in T_\Sigma$ there exist only finitely many $u \in T_\Sigma$ such that $\tau_M(t, u) \neq 0$. Correspondingly, a weighted tree transformation $\tau \colon T_\Sigma \times T_\Sigma \to A$ is *finitary* if for every $t \in T_\Sigma$ there exist finitely many $u \in T_\Sigma$ such that $\tau(t, u) \neq 0$. Clearly, τ_M is finitary. Note that there is an asymmetry here; for a given $u \in T_\Sigma$ there may be infinitely many $t \in T_\Sigma$ such that $\tau_M(t, u) \neq 0$.

We close this section by recalling some properties for the sentential forms occurring in a derivation [30, 16]. We only define those properties for the input side,

Table 1. Summary of the known properties of M.

	hierarchical		link-distance bounded	
Model \ Property	input	output	input	output
ε-free XTOP	strictly	strictly	strictly	strictly
ε-free XTOP$^{\mathrm{R}}$	strictly	strictly	✓	strictly
ε-free MBOT	✓	strictly	✓	strictly
reg.-pres. ε-free MBOT	✓	strictly	✓	strictly

but assume that they are also defined (in the same manner) for the output side.

Definition 6. A sentential form $\langle t, L, I, u\rangle \in \mathcal{SF}(M)$ is

- *input hierarchical* if $z' \not< w'$ and there exists $(y, y') \in L \cup I$ with $y' \leq z'$ for all $(w, w'), (z, z') \in L \cup I$ with $w < z$,
- *strictly input hierarchical* if (i) $w < z$ implies $w' \leq z'$ and (ii) $w = z$ implies $w' \leq z'$ or $z' \leq w'$ for all $(w, w'), (z, z') \in L \cup I$,
- *input link-distance bounded by* $b \in \mathbb{N}$ if for all links $(w, w'), (wz, z') \in L \cup I$ with $|z| > b$ there exists a link $(wy, y') \in L \cup I$ such that $y < z$ and $1 \leq |y| \leq b$,
- *strict input link-distance bounded by* b if for all positions $w, wz \in \mathrm{pos}(t)$ with $|z| > b$ there exists $(wy, y') \in L \cup I$ such that $y < z$ and $1 \leq |y| \leq b$.

The MBOT M has those properties if, for all $t, u \in T_\Sigma$, each sentential form occurring in a derivation of $\mathcal{D}_M(t, u)$ has them. ◇

We also say that the MBOT M is *input link-distance bounded* if there exists an integer $b \in \mathbb{N}$ such that it is input link-distance bounded by b. We summarize the known properties [30, 16] in Table 1.

4. Expressive power

A weighted tree transformation $\tau\colon T_\Sigma \times T_\Sigma \to A$ *preserves regularity* if for every regular weighted tree language $\varphi\colon T_\Sigma \to A$ and $u \in T_\Sigma$ we have that $\big(\tau(\varphi)\big)(u) = \sum_{t \in T_\Sigma} \varphi(t) \cdot \tau(t, u)$ is (i) well-defined in the sense that only finitely many summands are different from 0 and (ii) the weighted tree language $\tau(\varphi)$ defined in this manner is regular. Note that this property is stricter than in the unweighted setting [18, 19] because we additionally require (i). The MBOT M is regularity preserving if its computed weighted tree transformation τ_M preserves regularity. A chain of MBOTs is simply a word $C = (M_1, \dots, M_k)$ for MBOT $M_1, \dots, M_k$ over the alphabet Σ. It computes the weighted tree transformation $\tau_C = \tau_{M_1} ; \cdots ; \tau_{M_k}$, where for any two finitary weighted tree transformations $\tau, \tau'\colon T_\Sigma \times T_\Sigma \to A$ and $s, u \in T_\Sigma$ we have

$$(\tau \,;\, \tau')(s, u) = \sum_{t \in T_\Sigma} \tau(s, t) \cdot \tau'(t, u) \enspace ,$$

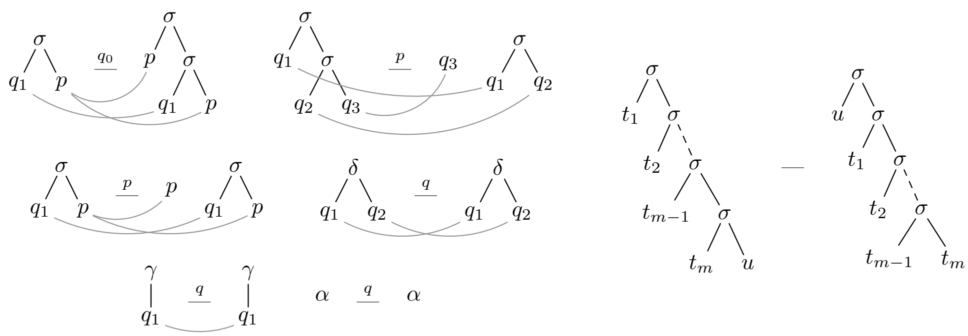

Figure 3. Rules of the MBOT Tpc with $q \in \{q_1, q_2, q_3\}$ and illustration of the weighted tree transformation (topicalization) computed by it for all $m \geq 3$ and arbitrary trees $u, t_1, \ldots, t_m$, which can contain binary δ-symbols, unary γ-symbols, and nullary α-symbols.

which is well-defined because τ is finitary. The weighted tree transformation computed by a chain of ε-free MBOTs is well-defined because all weighted tree transformations computed by ε-free MBOTs are finitary. Here we want to answer whether regularity preserving ε-free MBOTs can compute weighted tree transformations that cannot be computed by any chain of ε-free XTOP$^{\mathrm{R}}$. In the unweighted setting, this is easily confirmed by observing that all linear ε-free extended top-down tree transducers with regular look-ahead are regularity preserving [31], whereas some linear ε-free multi bottom-up tree transducers are not [12].

However, it remained open already in the unweighted case whether regularity preserving ε-free MBOTs are still more powerful than chains of ε-free XTOP$^{\mathrm{R}}$. Moreover, in the weighted setting discussed here, the simple unweighted approach does not work at all because already some ε-free XTOP$^{\mathrm{R}}$s are not regularity preserving (due to our stricter definition of regularity preservation). We will show that our approach based on the linking structures can successfully be applied to answer such questions. It thus provides are very powerful proof method, which works in any commutative semiring. This contrast another common lifting technique, which is also used to import unweighted results into the weighted setting. However, for the lifting technique the commutative semiring often has to fulfill additional properties (like zero-sum freeness, which effectively excludes all rings [22, 21]).

Example 7. Let $\mathrm{Tpc} = (Q, \Sigma, q_0, R, \mathrm{wt})$ be the ε-free MBOT with the states $Q = \{q_0, q_1, q_2, q_3, p\}$, the symbols $\Sigma = \{\sigma, \delta, \gamma, \alpha\}$, the rules R illustrated in Figure 3, and the weight function wt such that $\mathrm{wt}(\rho) = 1$ for all $\rho \in R$. The computed weighted tree transformation τ_M assigns weight 1 to pairs of trees of the shape depicted in Figure 3 and 0 otherwise. $\diamond$

The MBOT Tpc even implements a linguistically relevant transformation called *topicalization* [8]. We illustrate topicalization on two English examples in Figure 4. Topicalization is known as a discontiguous (and thus difficult) transformation on phrase-structure parse trees. First we show that it preserves regularity to establish that Tpc is regularity preserving.

Lemma 8. The MBOT Tpc is regularity preserving. ◇

Proof. For illustration we again refer to Figure 3. It is straightforward to confirm that for each input tree $t \in T_\Sigma$ there exists at most one output tree $u \in T_\Sigma$ such that $\tau_{\mathrm{Tpc}}(t, u) \neq 0$. Moreover, given $t_1, t_2, u_1, u_2 \in T_\Sigma$ with $\tau_{\mathrm{Tpc}}(t_1, u_1) \neq 0 \neq \tau_{\mathrm{Tpc}}(t_2, u_2)$ and $t_1 \neq t_2$, we can easily observe that $u_1 \neq u_2$. Consequently, for every output tree $u \in T_\Sigma$ there exists at most one input tree $t \in T_\Sigma$ such that $\tau_{\mathrm{Tpc}}(t, u) \neq 0$. In the following, let $\varphi\colon T_\Sigma \to A$ be a regular weighted tree language. For every $u \in T_\Sigma$ we have that $\sum_{t \in T_\Sigma} \varphi(t) \cdot \tau_{\mathrm{Tpc}}(t, u)$ is well-defined because the summand is different from 0 for at most one $t \in T_\Sigma$.

It remains to show that $\tau_M(\varphi)$ is regular. Let $\mathcal{A}' = (Q', \Sigma, q'_0, R', \mathrm{wt}')$ be a weighted tree automaton such that $\varphi_{\mathcal{A}'} = \varphi$. Since τ_{Tpc} is finitary, the weighted tree language $\mathrm{dom}(\tau_{\mathrm{Tpc}})\colon T_\Sigma \to A$ given by $\big(\mathrm{dom}(\tau_{\mathrm{Tpc}})\big)(t) = \sum_{u \in T_\Sigma} \tau_{\mathrm{Tpc}}(t, u)$ for every $t \in T_\Sigma$ is well-defined and thus regular by [28]. Moreover, $\mathrm{dom}(\tau_{\mathrm{Tpc}})\colon T_\Sigma \to \{0, 1\}$. Regular weighted tree languages are closed under HADAMARD product [4] using a standard product construction. The obtained weighted tree automaton now recognizes trees in the domain of the weighted tree transformation with the weight assigned to them by φ. In the final step we need to modify this weighted tree automaton. Let us discuss this change shortly. Let $t, u \in T_\Sigma$ be such that $\tau_{\mathrm{Tpc}}(t, u) \neq 0$. The modified weighted tree automaton should recognize u, but assign the weight $\varphi(t)$ to it. This can be achieved using a guess-and-check strategy, which processes the σ-spine of u as if it were that of t. To this end, we need to remember one guess (that for u) for a long time, whereas all other guesses can be checked and cancelled already in the next step. We leave the details of the construction as an exercise. □

Thus we have identified a weighted tree transformation that preserves regularity and can be computed by an ε-free MBOT. To distinguish the expressive power of regularity preserving ε-free MBOT from that of chains of ε-free XTOP$^{\mathrm{R}}$, it remains to demonstrate that this weighted tree transformation cannot be computed by any chain of ε-free XTOP$^{\mathrm{R}}$. Note that we strongly suspect that this result is also true without the restriction to ε-free XTOP$^{\mathrm{R}}$ since ε-rules would not essentially help in the computation of the weighted tree transformation τ_{Tpc}, but our proof method using the links relies on ε-free XTOP$^{\mathrm{R}}$.

Theorem 9. The weighted tree transformation τ_{Tpc} cannot be computed by any chain of ε-free XTOP$^{\mathrm{R}}$. ◇

Proof. First we need to reduce the problem from the weighted setting to the unweighted setting. By way of a contradiction, suppose that the transformation can be computed by a chain $C = (M_1, \dots, M_k)$ of ε-free XTOP$^{\mathrm{R}}$. Next, for each $1 \leq i \leq k$ we replace each $M_i = (Q_i, \Sigma, q_i, R_i, \mathrm{wt}_i)$ by the corresponding unweighted ε-free

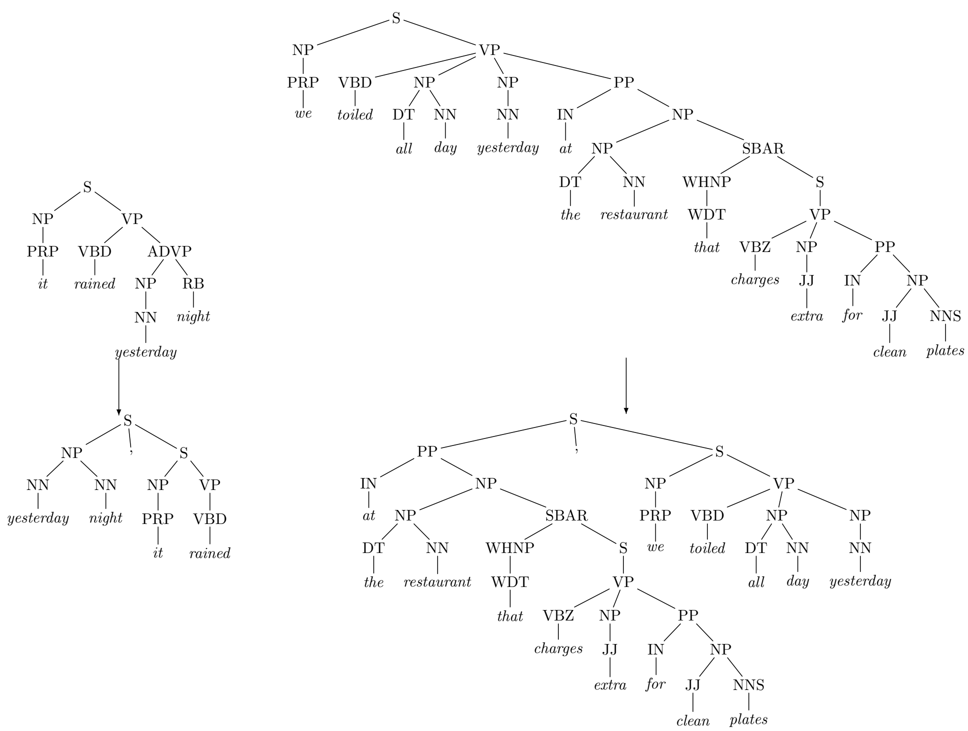

Figure 4. Topicalization on two English examples.

XTOP$^{\rm R}$ $N'_i = (Q_i, \Sigma, q_i, R_i)$. Moreover, $\tau_C(t, u) = 0$ for all $t, u \in T_\Sigma$ such that $(t, u) \notin T(N'_1) \mathbin{;} \cdots \mathbin{;} T(N'_k)$, where

$$T(N'_i) = \{(t, u) \in T_\Sigma \times T_\Sigma \mid \mathcal{D}_{N'_i}(t, u) \neq \emptyset\}$$

for all $1 \leq i \leq k$ and $\mathcal{D}_{N'_i}(t, u)$ is defined for unweighted linear multi bottom-up tree transducers in the same manner as for MBOT (note that it does not depend on the weight function wt). Thus, the composition chain $C' = (N'_1, \ldots, N'_k)$ computes at least all pairs of trees like those demonstrated in Figure 3, but potentially more.

Recent progress [15] showed that a chain of 3 ε-free XTOP$^{\rm R}$ can simulate any chain of ε-free XTOP$^{\rm R}$. Consequently, there exist 3 unweighted ε-free XTOP$^{\rm R}$ N_1, N_2, N_3 such that $T(N_1) \mathbin{;} T(N_2) \mathbin{;} T(N_3) = T(N'_1) \mathbin{;} \cdots \mathbin{;} T(N'_k)$. By the mentioned property, the (unweighted) tree transformation $T = T(N_1) \mathbin{;} T(N_2) \mathbin{;} T(N_3)$ contains all pairs indicated in Figure 3 (but potentially many more pairs). Additionally, we know that N_1, N_2, N_3 are input and output link-distance bounded (see Table 1), so let $b \in \mathbb{N}$ be such that all link-distances (for all 3 XTOP$^{\rm R}$) are bounded by b. Using the link theorem of [16] for chains of unweighted ε-free XTOP$^{\rm R}$, we can deduce the existence of an input tree t (leftmost in Figure 5), an output tree t''' (rightmost), two intermediate trees t' and t'' such that for all derivations $D \in \mathcal{D}_{N_1}(t, t')$, $D' \in \mathcal{D}_{N_2}(t', t'')$, and $D'' \in \mathcal{D}_{N_3}(t'', t''')$ terminating in the senten-

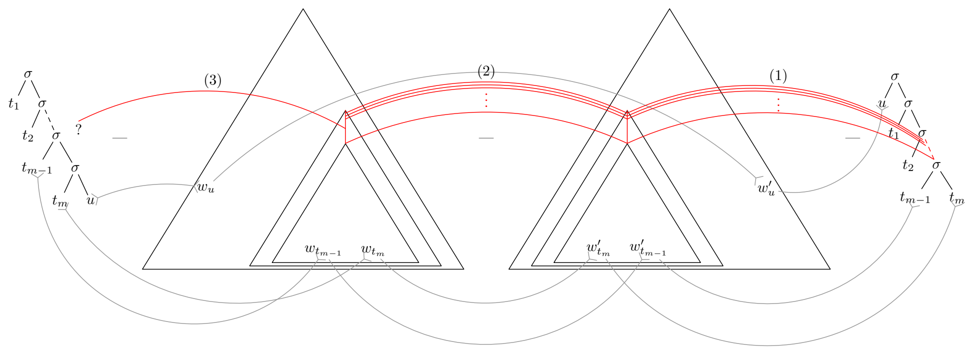

Figure 5. Illustration of the links discussed in the proof of Theorem 9. Inverted arrow heads indicate that the link points to a position below the one indicated by the spline. The links relating the roots of the trees are omitted.

tial forms $\langle t, \emptyset, I, t'\rangle$, $\langle t', \emptyset, I', t''\rangle$, and $\langle t'', \emptyset, I'', t'''\rangle$, respectively, we have that the light links depicted in Figure 5 (for the input and output tree and two intermediate trees without the clearly marked links) belong to I, I', or I'', in which $m \gg b^3$.

Now we will reason with those links and demonstrate that we can derive a contradiction. The ellipsis (clearly marked dots) in the output tree (rightmost in Figure 5) hides at least b^2 links that point to this part of the output because there must be a link every b positions by the link-distance bound. Let $(v''_1, w''_1), \dots, (v''_{m''}, w''_{m''}) \in I''$ with $m'' \gg b^2$ be those links such that $w''_1 < \cdots < w''_{m''}$. These links are marked with (1) in Figure 5. Clearly, $w''_{m''}$ dominates (via $\leq$) the root positions of the subtrees t_{m-1} and t_m, but it does not dominate that of the subtree u. The input positions of those links, which point to positions inside the third tree in Figure 5, automatically fulfill $v''_1 \leq \cdots \leq v''_{m''}$ since N_3 is strictly output hierarchical. A straightforward induction can be used to show that (for any ε-free XTOP$^{\mathrm{R}}$) all links sharing the same input positions must be incomparable with respect to the prefix order $\leq$ (also proved in [16]), which uses the ε-freeness of N_3. Consequently, $v''_1 < \cdots < v''_{m''}$. Similarly, we can conclude $v''_{m''} < w'_{t_{m-1}}$, $v''_{m''} < w'_{t_m}$, and $v''_1 \not\leq w'_u$, where the last statement uses that N_3 is strictly input hierarchical. Thus we have identified a chain of $m'' \gg b^2$ positions $v''_1, \dots, v''_{m''}$ in t'', which all dominate w'_{t_m} and $w'_{t_{m-1}}$ but are incomparable to w'_u. Repeating essentially the same arguments for N_2, we obtain links $(v'_1, w'_1), \dots, (v'_{m'}, w'_{m'}) \in I'$ with $m' \gg b$ such that $v''_1 \leq w'_1 < \cdots < w'_{m'} \leq v''_{m''}$ and $v'_1 < \cdots < v'_{m'}$. These links are labeled by (2) in Figure 5. Moreover, $v'_{m'} < w_{t_{m-1}}$, $v'_{m'} < w_{t_m}$, and $v'_1 \not\leq w_u$. Thus we have a chain of $m' \gg b$ positions $v'_1, \dots, v'_{m'}$ in t', which all dominate $w_{t_{m-1}}$ and w_{t_m}, but are incomparable to w_u. Using the same arguments once more for N_1, we obtain a single link $(v, w) \in I$ such that $v'_1 \leq w \leq v'_{m'}$. This final link is marked (3) in Figure 5. Moreover, we have $v < v_{t_{m-1}}$ and $v < v_{t_m}$, but $v \not\leq v_u$, where $v_{t_{m-1}}$, v_{t_m}, and v_u are the root positions of the subtrees t_{m-1}, t_m, and u, respectively. In other words, v must be a position in the σ-spine of t (since it dominates both the root position of t_{m-1} and that of t_m), but all positions along the σ-spine also dominate the root position of u. Consequently, such a position does not exist in the input tree t, which completes the desired contradiction. Consequently, such unweighted ε-free XTOP$^{\mathrm{R}}$ cannot exist, which yields that no chain of unweighted ε-free XTOP$^{\mathrm{R}}$ can compute an (unweighted) overapproximation of τ_{Tpc}. However, this contradicts the known statement that such an overapproximation must exist if τ_{Tpc} is computed by a chain of ε-free XTOP$^{\mathrm{R}}$. Consequently, such a chain cannot exist, which proves the statement. □

It is noteworthy that the proof can be achieved using high-level arguments based on the links and their properties. In fact, the whole proof is rather straightforward once the elementary links (light in Figure 5) are established using the linking theorem of [16]. Arguably, the proof of that linking theorem is quite technical and involved (using size arguments and thus the particular shape of the trees $u, t_1, \ldots, t_m$), but it can be checked once and reused in similar setups as it generally establishes links in the presented way between identical subtrees (for which infinitely many trees are possible). The proof nicely demonstrates that the difficult argumentation via two unknown intermediate trees t' and t'' and an unweighted tree transformation that overapproximates the support of τ_{Tpc}, where the support contains those pairs mapped to a weight different from 0 by τ_{Tpc}, reduces to (relatively) simple arguments with the help of the links. The author believes that the links will provide a powerful and versatile tool in the future and have been neglected for too long. This is especially true in the weighted setting since the absence of negative information from the linking theorems of [16] easily permits such overapproximations, which allows us to perform the reduction from the weighted to the unweighted case without additional restrictions.

From Theorem 9 it follows that (some) topicalizations cannot be computed by any chain of ε-free XTOP$^{\mathrm{R}}$ (or any chain of ε-free XTOP), and since τ_{Tpc} is computed by the regularity-preserving ε-free MBOT Tpc, we can conclude that regularity-preserving ε-free MBOT are strictly more powerful than chains of ε-free XTOP$^{\mathrm{R}}$.

Corollary 10. Regularity-preserving ε-free MBOT are strictly more powerful than composition chains of ε-free XTOP$^{\mathrm{R}}$ (and composition chains of ε-free XTOP). ◇

Our next result will limit the expressive power of ε-free MBOT. Using the corresponding linking theorem for unweighted MBOT [16] and our approach based on links once more, we will prove that the inverse weighted tree transformation $\tau_{\mathrm{Tpc}}^{-1} \colon T_\Sigma \times T_\Sigma \to A$, which is given by $\tau_{\mathrm{Tpc}}^{-1}(u, t) = \tau_{\mathrm{Tpc}}(t, u)$ for all $t, u \in T_\Sigma$, cannot be computed by any ε-free MBOT. This confirms the bottom-up nature of MBOT as a bottom-up device. It can "grab" deeply nested subtrees and transport them towards the root, but it cannot achieve the converse.

Theorem 11. The relation τ_{Tpc}^{-1} cannot be computed by any (composition chain of) ε-free MBOT. ◇

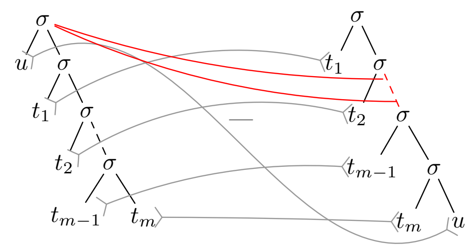

Figure 6. Illustration of the links discussed in the proof of Theorem 11. Inverted arrow heads indicate that the link points to a position below the one indicated by the spline. The links relating the roots of the trees are omitted.

Proof. Since ε-free MBOT are closed under composition [12], which extends to the weighted setting [27], we need to consider only a single ε-free MBOT. In order to derive a contradiction, let $M = (Q, \Sigma, q_0, R, \mathrm{wt})$ be an ε-free MBOT such that $\tau_M = \tau_{\mathrm{Tpc}}^{-1}$. As in the proof of Theorem 9 we first move from the weighted to the unweighted setting by considering the unweighted ε-free MBOT $N = (Q, \Sigma, q_0, R)$. Clearly, it computes a tree transformation $T(N) = \{(t, u) \in T_\Sigma \times T_\Sigma \mid \mathcal{D}_N(t, u) \neq \emptyset\}$ such that $\mathrm{supp}(\tau_M) \subseteq T(N)$, where $\mathrm{supp}(\tau_N) = \{(t, u) \in T_\Sigma \times T_\Sigma \mid \tau_M(t, u) \neq 0\}$.

We know that N is input and output link-distance bounded (see Table 1), so let $b \in \mathbb{N}$ be a suitable bound. Moreover, let $a > |v|$ for all rules $\langle \ell, q, v\rangle \in R$. Hence a is an upper bound for the length of the right-hand sides. Finally, let $k > \max(a, b)$ be our main constant. Applying the linking theorem for ε-free MBOT [16] there exist an input tree t (leftmost in Figure 6 and an output tree t' (rightmost) such that for all derivations $D \in \mathcal{D}_N(t, t')$ terminating in the sentential form $\langle t, \emptyset, I, t'\rangle$ we have the links lightly shown in Figure 6 in I, in which $m \gg 2k$. Consequently, the ellipsis (clearly marked dots) in the output tree t' hides at least 2 links that point to this part of the output tree t' because there must be a link every b positions by the link-distance bound. Let $(v, w), (v', w') \in I$ be those links such that $w < w'$. These links are clearly indicated in Figure 6.

Clearly, w' dominates the root positions of the subtrees t_m and u, and thus the output positions of the links pointing into t_m and u in t'. Since N is strictly output hierarchical (see Table 1), we obtain that (i) $v \leq v'$ and (ii) v' dominates the input positions of the (light) links pointing into the subtrees t_m and u. However, there is only one position in t that dominates the input position of the link into t_m and the input position of the link into u, which is the root of t. Consequently, $v = v' = \varepsilon$ as already indicated in Figure 6. Another straightforward induction can be used to show that (for any ε-free MBOT) all links sharing the same input positions must be incomparable with respect to the prefix order $\leq$, which is also proved [16]. This part again uses the ε-freeness of N. However, (ε, w) and (ε, w') are two links with the same source and comparable target positions because $w < w'$, so we derived the desired contradiction. Hence no unweighted ε-free MBOT can compute an overapproximation of $\mathrm{supp}(\tau_{\mathrm{Tpc}}^{-1})$. However, since this is always possible provided that τ_{Tpc}^{-1} can be computed by some ε-free MBOT, the latter provision must be false, which proves the statement. □

Again we note that the proof could be straightforwardly achieved using high-level arguments on the links and their interrelation after establishing the elementary links (light in Figure 6) relating equal subtrees in the input and output tree. Then the link-distance can be used to conclude the existence of links and their input and output target can be related to existing links using the hierarchical properties. In this way, we could in both cases derive a contradiction in rather straightforward ways, which would not have been possible without the links. Typically, such (negative) statements are proved using the fooling technique (see [1] or [31] for examples), which requires a rather detailed case analysis of all possible intermediate trees and applied rules, which then individually have to be contradicted. In a scenario with 2 unknown intermediate trees such an approach becomes (nearly) impossible to handle. In addition, the fooling technique relies on the knowledge that certain pairs are not in the unweighted tree transformation. Since we only have an overapproximation (due to the reduction from the weighted to the unweighted setting), we typically cannot exclude pairs from the tree transformation computed by the overapproximation. Thus, the fooling technique fails completely in the weighted setting [unless additional restrictions are placed on the commutative semiring, which yield that for each MBOT M the corresponding unweighted MBOT computes exactly the set $\mathrm{supp}(\tau_M)$]. This latter approach is, for example, discussed in [14]. Thus, we strongly want to promote the use of links and their interrelations in the analysis of tree transducers.

Theorem 11 yields that regularity-preserving ε-free MBOT are not closed under inversion. In other words, there are regularity-preserving ε-free MBOT M (such as Tpc), whose inverted computed weighted tree transformation τ_M^{-1} cannot be computed by any ε-free MBOT. This is not very surprising since ε-freeness is already an asymmetric restriction.

Corollary 12. Regularity-preserving ε-free MBOT (and general ε-free MBOT) are not closed under inversion. ◇

We collect the obtained results together with some minor consequences in a HASSE diagram in Figure 7). Note that all weighted versions of the classes mentioned in it refer to the regularity preserving and ε-free variants. Additionally, note that very little in known in general (i.e., for all commutative semirings) in the weighted setting.

Bibliography

[1] A. Arnold and M. Dauchet, Morphismes et bimorphismes d'arbres, *Theoret. Comput. Sci.* **20**(1) (1982) 33–93.

[2] J. Berstel and C. Reutenauer, Recognizable formal power series on trees, *Theoret. Comput. Sci.* **18**(2) (1982) 115–148.

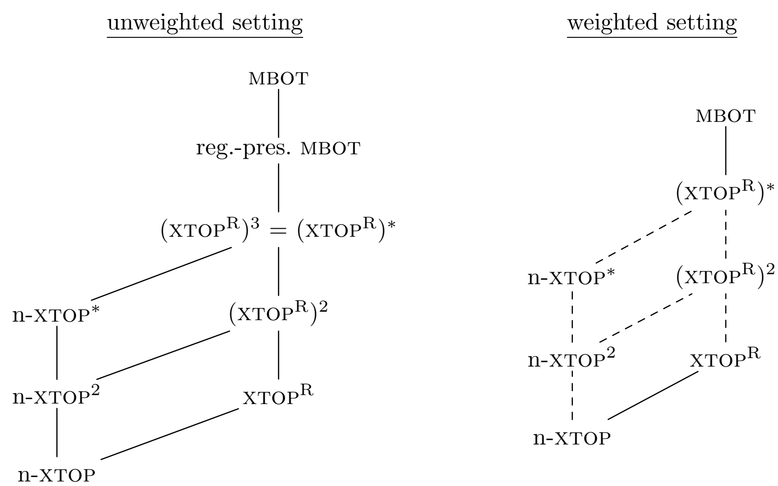

Figure 7. Hasse diagram for the classes of (weighted) tree transformations computed by various classes of MBOTs, where $\mathcal{C}^*$ is the composition closure of class $\mathcal{C}$ and dashed lines indicates the inclusion might not be strict. Note that in order not to clutter the diagram all classes in the weighted setting are restricted to be the ε-free and regularity preserving variants of the displayed models.

[3] M. Bojanczyk, Transducers with origin information, *Proc. 41st ICALP*, *LNCS* **8573**, (Springer, 2014), pp. 26–37.
[4] B. Borchardt, A pumping lemma and decidability problems for recognizable tree series, *Acta Cybernet.* **16**(4) (2004) 509–544.
[5] F. Braune, N. Seemann, D. Quernheim and A. Maletti, Shallow local multi bottom-up tree transducers in statistical machine translation, *Proc. 51st ACL*, (Association for Computational Linguistics, 2013), pp. 811–821.
[6] D. Chiang, An introduction to synchronous grammars, *Proc. 44th ACL*, (Association for Computational Linguistics, 2006). Part of a tutorial given with Kevin Knight.
[7] D. Chiang, Hierarchical phrase-based translation, *Comput. Linguist.* **33**(2) (2007) 201–228.
[8] N. Chomsky, *The Minimalist Program* (MIT Press, 1995).
[9] H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, C. Löding, S. Tison and M. Tommasi, Tree automata — techniques and applications available online at `http://tata.gforge.inria.fr/`, (2007).
[10] A. de Gispert, G. Iglesias, G. W. Blackwood, E. R. Banga and W. Byrne, Hierarchical phrase-based translation with weighted finite-state transducers and shallow-n grammars, *Comput. Linguist.* **36**(3) (2010) 505–533.
[11] J. Eisner, Learning non-isomorphic tree mappings for machine translation, *Proc. 41st ACL*, (Association for Computational Linguistics, 2003), pp. 205–208.
[12] J. Engelfriet, E. Lilin and A. Maletti, Composition and decomposition of extended multi bottom-up tree transducers, *Acta Inform.* **46**(8) (2009) 561–590.
[13] J. Engelfriet, G. Rozenberg and G. Slutzki, Tree transducers, L systems, and two-way machines, *J. Comput. System Sci.* **20**(2) (1980) 150–202.
[14] Z. Fülöp, Z. Gazdag and H. Vogler, Hierarchies of tree series transformations, *Theoret. Comput. Sci.* **314**(3) (2004) 387–429.

[15] Z. Fülöp and A. Maletti, Composition closure of ε-free linear extended top-down tree transducers, *Proc. 17th DLT*, LNCS **7907**, (Springer, 2013), pp. 239–251.
[16] Z. Fülöp and A. Maletti, Linking theorems for tree transducers, Manuscript (2014).
[17] Z. Fülöp and H. Vogler, Weighted tree automata and tree transducers, *Handbook of Weighted Automata*, eds. M. Droste, W. Kuich and H. Vogler, *EATCS Monographs on Theoret. Comput. Sci.* (Springer, 2009), pp. 313–403.
[18] F. Gécseg and M. Steinby, *Tree Automata* (Akadémiai Kiadó, Budapest, 1984).
[19] F. Gécseg and M. Steinby, Tree languages, *Handbook of Formal Languages*, eds. G. Rozenberg and A. Salomaa, **3** (Springer, 1997), pp. 1–68.
[20] D. Gildea, On the string translations produced by multi bottom-up tree transducers, *Comput. Linguist.* **38**(3) (2012) 673–693.
[21] J. S. Golan, *Semirings and their Applications* (Kluwer Academic, Dordrecht, 1999).
[22] U. Hebisch and H. J. Weinert, *Semirings—Algebraic Theory and Applications in Computer Science* (World Scientific, 1998).
[23] K. Knight and J. Graehl, An overview of probabilistic tree transducers for natural language processing, *Proc. 6th CICLing*, LNCS **3406**, (Springer, 2005), pp. 1–24.
[24] P. Koehn, *Statistical Machine Translation* (Cambridge University Press, 2010).
[25] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin and E. Herbst, Moses: Open source toolkit for statistical machine translation, *Proc. 45th ACL*, (Association for Computational Linguistics, 2007), pp. 177–180.
[26] E. Lilin, Propriétés de clôture d'une extension de transducteurs d'arbres déterministes, *Proc. 6th CAAP*, LNCS **112**, (Springer, 1981), pp. 280–289.
[27] A. Maletti, Why synchronous tree substitution grammars?, *Proc. HLT-NAACL 2010*, (Association for Computational Linguistics, 2010), pp. 876–884.
[28] A. Maletti, An alternative to synchronous tree substitution grammars, *J. Natur. Lang. Engrg.* **17**(2) (2011) 221–242.
[29] A. Maletti, How to train your multi bottom-up tree transducer, *Proc. 49th ACL*, (Association for Computational Linguistics, 2011), pp. 825–834.
[30] A. Maletti, Tree transformations and dependencies, *Proc. 12th MOL*, LNAI **6878**, (Springer, 2011), pp. 1–20.
[31] A. Maletti, J. Graehl, M. Hopkins and K. Knight, The power of extended top-down tree transducers, *SIAM J. Comput.* **39**(2) (2009) 410–430.
[32] J. May, K. Knight and H. Vogler, Efficient inference through cascades of weighted tree transducers, *Proc. 48th ACL*, (Association for Computational Linguistics, 2010), pp. 1058–1066.
[33] F. J. Och and H. Ney, The alignment template approach to statistical machine translation, *Comput. Linguist.* **30**(4) (2004) 417–449.
[34] S. M. Shieber and Y. Schabes, Synchronous tree-adjoining grammars, *Proc. 13th CoLing*, **3** (1990), pp. 253–258.
[35] M. Zhang, H. Jiang, A. Aw, H. Li, C. L. Tan and S. Li, A tree sequence alignment-based tree-to-tree translation model, *Proc. 46th ACL*, (Association for Computational Linguistics, 2008), pp. 559–567.