Information and Computation 206 (2008) 1187–1196

Contents lists available at ScienceDirect

Information and Computation

journal homepage: www.elsevier.com/locate/ic

Compositions of extended top-down tree transducers☆

Andreas Maletti

International Computer Science Institute, 1947 Center Street, Suite 600, Berkeley, CA 94704, USA

ARTICLE INFO

Article history:
Received 6 October 2007
Revised 3 March 2008
Available online 7 June 2008

Keywords:
Tree transducer
Composition
Bimorphism
Natural language processing
Tree transformation

ABSTRACT

Unfortunately, the class of transformations computed by linear extended top-down tree transducers with regular look-ahead is not closed under composition. It is shown that the class of transformations computed by certain linear bimorphisms coincides with the previously mentioned class. Moreover, it is demonstrated that every linear epsilon-free extended top-down tree transducer with regular look-ahead can be implemented by a linear multi bottom-up tree transducer. The class of transformations computed by the latter device is shown to be closed under composition, and to be included in the composition of the class of transformations computed by top-down tree transducers with itself. More precisely, it constitutes the composition closure of the class of transformations computed by finite-copying top-down tree transducers.

1. Introduction

The top-down tree transducer (tdtt) was introduced in [1,2] and intensively studied thereafter (see [3,4] for a survey). It was originally motivated from natural language processing [5] and syntax-directed semantics [6], but was later successfully applied to problems as diverse as: functional programming [7], analysis of cryptographic protocols [8], and decidability of the first-order theory of ground rewriting [9].

In particular, compositions of tdtt are considered in [10,11]. In this paper, we study compositions of extended tdtt, which were introduced in [12–14] and subsequently led to several improvements [15] in machine translation (see [16] for a survey). In fact, [16] mentions that closure under composition is a desirable property of any class of transformations with applications in natural language processing. However, nondeleting and linear extended tdtt as well as linear extended tdtt with regular look-ahead [17] compute classes of transformations that are not closed under composition [13,18,19]. In essence, this requires us to consider either slightly more restricted classes or slightly larger classes. In this paper, we will follow a combination of both approaches; we first restrict ourselves to extended tdtt without epsilon rules and then slightly generalize.

An extended tdtt essentially is a tdtt whose left-hand sides of rules offer not only shallow patterns of the form $\sigma(x_1,\ldots,x_k)$ for some k-ary symbol σ, but allow arbitrary patterns (without repeated variables) as left-hand sides. In this paper, we will mostly consider linear extended tdtt, in which the right-hand side of a rule may not contain several occurrences of the same variable. Two example rules are shown in Fig. 1. The semantics of extended tdtt is given by term rewriting. An instance of the left-hand side of a rule is replaced by the appropriately instantiated right-hand side of that rule. We start this rewriting process with $q(t)$ where q is an initial state and t is the input tree. An extended tdtt may thus transform an input tree t into an output tree u if there exists an initial state q such that $q(t)$ can be rewritten to u.

☆ Financially supported by a German Academic Exchange Service (DAAD) grant.
E-mail address: maletti@icsi.berkeley.edu
URL: http://www.icsi.berkeley.edu/~maletti

0890-5401/$ - see front matter
doi:10.1016/j.ic.2008.03.019

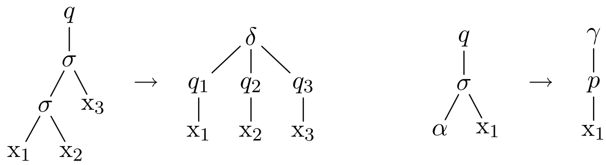

Fig. 1. Illustration of extended top-down tree transducer rules.

It is shown in [20] that synchronized tree substitution grammars [21] are as powerful (up to relabeling) as bimorphisms (see survey [22]) of type (LC,LC). As a variation of this, we show that nondeleting and linear extended tdtt are exactly as powerful as bimorphisms of type (LC,LC). These two results are in fact straightforward generalizations of a similar result in [13] for a subclass of such extended tdtt and bimorphisms of type (LCE,LCE). We also show that linear extended tdtt with regular look-ahead are as powerful as bimorphisms of type (LC,L). It was proved in [18, Section 3.4] that no class of bimorphisms that contains all bimorphisms of type (LCE,LCE) computes a class of transformations that is closed under composition. Consequently, nondeleting and linear extended tdtt, synchronized tree substitution grammars, and linear extended tdtt with regular look-ahead compute nonclosed classes.

In this paper, we approach the issue by first restricting ourselves to extended tdtt without epsilon rules [equivalently, bimorphisms of types (LCE,LC) and (LCE,L)]. Second, we recall a bottom-up device: the multi bottom-up tree transducer [23–25] (mbutt). We show that linear epsilon-free extended tdtt can be simulated by linear mbutt. We also show that the class of transformations computed by linear mbutt is closed under composition. This is rather unexpected because linear mbutt can reproduce certain forms of top-down copying [10]. Finally, we discuss how to implement mbutt in a top-down fashion, alas not as linear extended tdtt as this would be impossible in general because every linear extended tdtt preserves recognizability [3,4], whereas some linear mbutt do not. Specifically, the class of transformations computed by linear mbutt coincides with the composition closure of finite-copying tdtt [26] (which in turn equals the class of compositions of a finite-copying tdtt and a single-use tdtt [27–29,7,30]). Thus, we do not solve the problem as originally posed but, for the epsilon-free case, present a suitable superclass of transformations that enjoys the much required closure under composition. As a side result we obtain that every linear mbutt is equivalent to a nondeleting and linear one.

2. Preliminaries

We use $\mathbb{N}$ to denote the set of natural numbers including 0. Let $X = \{x_1,x_2,\dots\}$ be a fixed set of variables, and for every $k \in \mathbb{N}$ let $X_k = \{x_i \mid 1 \leqslant i \leqslant k\}$. Since we need the restriction $1 \leqslant i \leqslant k$ often, we abbreviate $\{i \mid 1 \leqslant i \leqslant k\}$ by $[k]$. Alphabets and ranked alphabets are defined as usual. We use $\Sigma^{(k)}$ to denote the set of k-ary symbols of a ranked alphabet Σ and write rk_Σ for the rank function associated to Σ. The set of Σ-trees indexed by a set V is denoted by $T_\Sigma(V)$. We generally assume that all used ranked alphabets draw their symbols from a common ranked alphabet (i.e., a symbol is assigned only one rank). Thus, if $V \subseteq T_\Delta(V')$ for some set V', then we can view elements of $T_\Sigma(V)$ also as elements of $T_{\Sigma\cup\Delta}(V')$.

Let $V \subseteq X$. The set of variables occurring in a tree $t \in T_\Sigma(V)$ is denoted by $\mathrm{var}(t)$. We call t nondeleting (respectively, linear) in V if every $v \in V$ occurs at least (respectively, at most) once in t. Let $\Delta \subseteq \Sigma \cup X$. The mapping $\mathrm{preorder}_\Delta \colon T_\Sigma(X) \to \Delta^*$ is defined as follows: $\mathrm{preorder}_\Delta(x)$ is x if $x \in \Delta$ and ε otherwise for every $x \in X$ (where ε is the empty string), and

$$\mathrm{preorder}_\Delta(\sigma(t_1,\dots,t_k)) = \begin{cases} \sigma\,\mathrm{preorder}_\Delta(t_1)\cdots\mathrm{preorder}_\Delta(t_k) & \text{if } \sigma \in \Delta \\ \mathrm{preorder}_\Delta(t_1)\cdots\mathrm{preorder}_\Delta(t_k) & \text{otherwise} \end{cases}$$

for every $\sigma \in \Sigma^{(k)}$ and $t_1,\dots,t_k \in T_\Sigma(X)$. The set $\mathrm{pos}(t)$ denotes the set of positions (or nodes) of t and is defined as usual. For every $w \in \mathrm{pos}(t)$ we write $t(w)$ for the symbol that occurs at position w in t. By $t|_w$ we denote the subtree of t that is rooted at w, and by $t[u]_w$ we denote the tree obtained from t by replacing the subtree rooted at w by u. Moreover, $\mathrm{pos}_\Delta(t) = \{w \in \mathrm{pos}(t) \mid t(w) \in \Delta\}$ and $\mathrm{pos}_\delta(t) = \mathrm{pos}_{\{\delta\}}(t)$ for every $\delta \in \Sigma \cup X$. Any $\theta \colon V \to T_\Sigma(X)$ is a substitution. It extends to a mapping $\theta \colon T_\Sigma(V) \to T_\Sigma(X)$ by $\sigma(t_1,\dots,t_k)\theta = \sigma(t_1\theta,\dots,t_k\theta)$ for every $\sigma \in \Sigma^{(k)}$ and $t_1,\dots,t_k \in T_\Sigma(V)$ [note that we prefer the post-fix notation with substitutions]. Given $\sigma \in \Sigma^{(k)}$ and $L_1,\dots,L_k \subseteq T_\Sigma(X)$ we write $\sigma(L_1,\dots,L_k)$ for the set $\{\sigma(t_1,\dots,t_k) \mid t_1 \in L_1,\dots,t_k \in L_k\}$ and $\Sigma(L_1)$ for the set

$$\bigcup_{\sigma\in\Sigma} \sigma(L_1,\dots,L_1).$$

For a mapping $f \colon A \to B$ and a set $C \subseteq A$, we write $f(C)$ to denote $\{f(c) \mid c \in C\}$. The powerset of A, i.e., the set of all subsets of A, is denoted by $\mathcal{P}(A)$. Finally, we write ; for function composition provided that the types are compatible, i.e., given $f \colon A \to B$ and $g \colon B \to C$ the expression $f ; g$ denotes the function from A to C such that $(f ; g)(a) = g(f(a))$ for every $a \in A$.

Any subset of T_Σ is a tree language [4]. A (top-down) tree automaton [4] is a tuple $N = (Q,\Sigma,I,\delta)$ where Q is a finite set, Σ is a ranked alphabet, $I \subseteq Q$, and $\delta = (\delta_k)_{k\in\mathbb{N}}$ where $\delta_k \subseteq Q \times \Sigma^{(k)} \times Q^k$. A run of N on an input tree $t \in T_\Sigma$ is a mapping $d \colon \mathrm{pos}(t) \to Q$ such that $(d(w),t(w),d(w1),\dots,d(wk)) \in \delta_k$ for every $w \in \mathrm{pos}(t)$ with $t(w) \in \Sigma^{(k)}$. For every $q \in Q$ we denote by $L(N)_q$ the set of trees t in T_Σ for which there exists a run d of N on t with $d(\varepsilon) = q$. The tree language recognized by N is

$L(N) = \bigcup_{q \in I} L(N)_q$. Any tree language $L \subseteq T_\Sigma$ that is recognized by some tree automaton is called recognizable and we denote the set of all such tree languages by $\mathrm{Rec}(\Sigma)$.

Finally, let us recall the bimorphism approach to tree transformations [3,22]. Suppose that $\varphi\colon \Sigma \to T_\Delta(X)$ is such that $\varphi(\sigma) \in T_\Delta(X_k)$ for every $\sigma \in \Sigma^{(k)}$. Such a mapping extends uniquely to a (tree) homomorphism $\varphi\colon T_\Sigma \to T_\Delta$ by $\varphi(\sigma(t_1,\ldots,t_k)) = \varphi(\sigma)\theta$ where $\theta(x_i) = \varphi(t_i)$ for every $i \in [k]$. A homomorphism φ is called nondeleting (respectively, linear), if $\varphi(\sigma)$ is nondeleting (respectively, linear) in X_k for every $\sigma \in \Sigma^{(k)}$. It is nonerasing if $\varphi(\sigma) \notin X$ for every $\sigma \in \Sigma$. A bimorphism just consists of a recognizable tree language and two homomorphisms. Let Σ, Γ, and Δ be ranked alphabets. A bimorphism is a triple $B = (\varphi,L,\psi)$ where (i) $\varphi\colon T_\Gamma \to T_\Sigma$ is the input homomorphism, (ii) $L \subseteq T_\Gamma$ is the recognizable tree language (control language), and (iii) $\psi\colon T_\Gamma \to T_\Delta$ is the output homomorphism. The tree transformation computed by B is $\|B\| = \{(\varphi(s),\psi(s)) \in T_\Sigma \times T_\Delta \mid s \in L\}$. We call the bimorphism B linear if φ and ψ are linear. The class of tree transformations computable by bimorphisms is denoted by $\mathrm{B}(w_1,w_2)$ where w_1 and w_2 list the restrictions on the input- and output-homomorphism, respectively. The restrictions are abbreviated ‘L’ for “linear”, ‘C’ for “nondeleting” (complete), and ‘E’ for “nonerasing”. Thus, e.g., B(LCE,L) denotes the class of transformations computable by linear bimorphisms with a nondeleting and nonerasing input homomorphism.

3. Extended top-down tree transducer

In this section, we quickly recall the notion of an extended top-down tree transducer (*transducteur généralisé descendant*) from [12–14]. Essentially, an extended top-down tree transducer has rules in which the left-hand side may contain arbitrary, not just shallow, patterns. Since we will also need regular look-ahead [17], we immediately introduce the extended top-down tree transducer with regular look-ahead of [19].

Definition 1. An extended (top-down) tree transducer with regular look-ahead ($\mathrm{xtt^R}$) is a tuple (Q,Σ,Δ,I,R,c) such that

- Q is a ranked alphabet (the *states*) such that $Q = Q^{(1)}$ and $Q \cap (\Sigma \cup \Delta) = \emptyset$;
- Σ and Δ are ranked alphabets (the *input* and *output symbols*);
- $I \subseteq Q$ (the *initial states*);
- $R \subseteq Q(T_\Sigma(X)) \times T_\Delta(Q(X))$ is a finite set (the *rules*) such that l is linear in X and $\mathrm{var}(r) \subseteq \mathrm{var}(l)$ for every $(l,r) \in R$; and
- $c\colon R \to \mathrm{Rec}(\Sigma)$ (the *look-ahead*).

Without loss of generality we commonly assume that for every rule $(l,r) \in R$ there exists $k \in \mathbb{N}$ such that $\mathrm{preorder}_X(l) = x_1 \cdots x_k$. Moreover, we commonly write $l \to r$ instead of (l,r) when handling rules in order to save parentheses. Let us define some properties of $\mathrm{xtt^R}$ next. Note that we define “deterministic” only for top-down tree transducers [1,2] with regular look-ahead [17].

Definition 2. The $\mathrm{xtt^R}$ (Q,Σ,Δ,I,R,c) is

- an extended (top-down) tree transducer (xtt) if $c(l \to r) = T_\Sigma$ for every $l \to r \in R$;
- a top-down tree transducer with regular look-ahead($\mathrm{tdtt^R}$) if $R \subseteq Q(\Sigma(X)) \times T_\Delta(Q(X))$;
- a top-down tree transducer (tdtt) if it is an xtt and a $\mathrm{tdtt^R}$;
- nondeleting (respectively, *linear*) if r is nondeleting (respectively, linear) in $\mathrm{var}(l)$ for every $l \to r \in R$;
- epsilon-free [respectively, *nonerasing*] if $l \notin Q(X)$ [respectively, $r \notin Q(X)$] for every $l \to r \in R$.

Finally, a $\mathrm{tdtt^R}$ (Q,Σ,Δ,I,R,c) is *deterministic* if $\mathrm{card}(I) = 1$ and for every $l \in Q(\Sigma(X))$ and $t \in T_\Sigma$ there exist at most one $\theta\colon \mathrm{var}(l) \to X$ and r such that $l\theta \to r \in R$ and $t \in c(l\theta \to r)$.

We drop the look-ahead component from the tuple for all xtt. The semantics of $\mathrm{xtt^R}$ is given by a straightforward term rewriting. We identify an instance of the left-hand side in a sentential form, verify that the look-ahead is satisfied, and replace this instance by a correspondingly (according to the rules) instantiated right-hand side.

Definition 3. Let $M = (Q,\Sigma,\Delta,I,R,c)$ be an $\mathrm{xtt^R}$. For every $\xi,\zeta \in T_\Delta(Q(T_\Sigma))$ let $\xi \Rightarrow_M \zeta$ if there exist

- a position $w \in \mathrm{pos}(\xi)$,
- a rule $l \to r \in R$, and
- a substitution $\theta\colon X \to T_\Sigma$

such that (i) $\xi|_w = l\theta$, (ii) $\xi|_{w1} \in c(l \to r)$, and (iii) $\zeta = \xi[r\theta]_w$. The *tree transformation computed by M* is

$$\|M\| = \{(t,u) \in T_\Sigma \times T_\Delta \mid \exists q \in I\colon q(t) \Rightarrow_M^* u\}.$$

We denote the classes of transformations computed by $\mathrm{xtt^R}$, xtt, $\mathrm{tdtt^R}$, and tdtt by $\mathrm{XTOP^R}$, XTOP, $\mathrm{TOP^R}$, and TOP. Moreover, we use the prefixes ‘l’ and ‘n’ (and ‘d’ for $\mathrm{tdtt^R}$) to restrict to linear and nondeleting (and deterministic) devices, respectively.

Thus, the class of tree transformations computed by linear xtt^{R} is denoted by $\text{l}-\text{XTOP}^{\text{R}}$. We first relate linear xtt^{R} and particular linear bimorphisms. The following theorem shows that the power of linear xtt^{R} and linear bimorphisms with nondeleting input homomorphism coincides [13]. Note that $\text{nl}-\text{XTOP}^{\text{R}} = \text{nl}-\text{XTOP}$ can easily be shown.

Theorem 4. $\text{B(LC,LC)} = \text{nl–XTOP}$ *and* $\text{B(LC,L)} = \text{l–XTOP}^{\text{R}}$.

Proof. In [13], it is proved that B(LCE,LCE) coincides with the class of transformations computed by linear, nondeleting, epsilon-free, and nonerasing xtt. We slightly extend their approach to obtain the stated results.

Let $B = (\varphi,L,\psi)$ be a linear bimorphism such that $\varphi: T_\Gamma \to T_\Sigma$ is nondeleting and $\psi: T_\Gamma \to T_\Delta$. Moreover, let $N = (Q,\Gamma,I,\delta)$ be a tree automaton such that $L(N) = L$. Roughly speaking, we use the control structure of N as control structure of the xtt^{R} and use φ and ψ to determine the left- and right-hand sides of the rules, respectively. Additionally, we use the look-ahead to verify that the input tree is suitable (which is essential only in the case of deletion). Formally, we construct the linear xtt^{R} $M = (Q,\Sigma,\Delta,I,R,c)$ as follows. For every $\gamma \in \Gamma^{(k)}$ and $q,q_1,\ldots,q_k \in Q$, if $(q,\gamma,q_1,\ldots,q_k) \in \delta_k$, then $\rho = q(\varphi(\gamma)) \to \psi(\gamma)\theta \in R$ where θ is the substitution such that $x_i\theta = q_i(x_i)$ for every $i \in [k]$, and $c(\rho) = \varphi(\gamma(L(N)_{q_1},\ldots,L(N)_{q_k}))$. Note that M is nondeleting whenever ψ is so. It remains to prove that $\|M\| = \|B\|$. To this end, it can be shown for every $q \in Q$, $t \in T_\Sigma$, and $u \in T_\Delta$ that $q(t) \Rightarrow_M^* u$ if and only if there exists $s \in L(N)_q$ such that $(t,u) = (\varphi(s),\psi(s))$. The "if"-direction of this statement can be proved by induction on the structure of s, and the "only-if"-direction can be proved by induction on the length of the derivation $q(t) \Rightarrow_M^* u$.

For the converse, we only consider the linear case. The nondeleting and linear case can be handled as in [13]. We first extract the control structure from the extended tdtt^{R} M. We combine the obtained tree automaton, which works on trees of rules of M, with the one needed to check the look-ahead, which shall also work on trees of rules of M. Since the look-ahead is performed on the input tree, we need to make sure that for each input symbol at least one processing rule exists. The left- and right-hand sides of the rules then determine the homomorphisms φ and ψ, respectively. Formally, let a linear xtt^{R} $M = (Q,\Sigma,\Delta,I,R,c)$ be given. Without loss of generality, suppose that there exist $\perp \in Q$ and $\alpha \in \Delta^{(0)}$ such that $\rho_\sigma = \perp(\sigma(x_1,\ldots,x_k)) \to \alpha \in R$ and $c(\rho_\sigma) = T_\Sigma$ for every $\sigma \in \Sigma^{(k)}$. We first construct φ, ψ, and a tree automaton $N = (Q,R,I,\delta)$ as follows. Let $\rho \in R$ be such that $\rho = q(l) \to r\theta$ for some $l \in T_\Sigma(X)$ with $\text{preorder}_X(l) = x_1 \cdots x_k$, $r \in T_\Delta(X)$, and $q,q_1,\ldots,q_k \in Q$ where θ is the substitution such that $x_i\theta = q_i(x_i)$ for every $i \in [k]$. We already noted that, without loss of generality, any rule can be written in this way. Then, let $\text{rk}_R(\rho) = k$, $\varphi(\rho) = l$, $\psi(\rho) = r$, and $(q,\rho,q'_1,\ldots,q'_k) \in \delta_k$ where for every $i \in [k]$

$$q'_i = \begin{cases} q_i & \text{if } x_i \in \text{var}(r) \\ \perp & \text{otherwise.} \end{cases}$$

Note that this in particular yields that $\text{rk}_R(\rho_\sigma) = \text{rk}_\Sigma(\sigma)$ and $(\perp,\rho_\sigma,\perp,\ldots,\perp) \in \delta_k$ for every $\sigma \in \Sigma^{(k)}$. In essence, this means that R contains a copy of Σ. Now let us consider the look-ahead. Let

$$L = \{s \in T_R \mid \forall w \in \text{pos}(s): \varphi(s|_w) \in c(s(w))\}.$$

It can easily be shown that L is recognizable. Consequently, we obtain the linear bimorphism $B = (\varphi,L(N) \cap L,\psi)$. To prove that $\|B\| = \|M\|$, we show for every $t \in T_\Sigma$, $u \in T_\Delta$, and $q \in Q$ we have $q(t) \Rightarrow_M^* u$ if and only if there exists $s \in L(N)_q \cap L$ such that $(t,u) = (\varphi(s),\psi(s))$. This can be achieved as in the converse direction. $\square$

By [18] there exist $\tau_1,\tau_2 \in \text{B(LCE,LCE)}$ such that $\tau_1 ; \tau_2 \notin \text{B(L,L)}$. Hence, there exist $\tau_1,\tau_2 \in \text{nl}-\text{XTOP}$ such that $\tau_1 ; \tau_2 \notin \text{l}-\text{XTOP}^{\text{R}}$.

Corollary 5. nl–XTOP, l–XTOP, *and* l–XTOP^{R} *are not closed under composition.*

4. Multi bottom-up tree transducer

Next, let us recall the multi bottom-up tree transducer (mbutt; also called STA or *S-transducteur ascendant*) of [23–25,31]. We slightly adapt the model by omitting the special root symbol, which is required in [25,31] to deterministically identify the root of the input tree. In [25] the root symbol is needed to show that deterministic mbutt are as powerful as deterministic tdtt^{R}. Compared to [23,24], we disallow rules that do not consume any input symbol (epsilon rules). Essentially, an mbutt is a bottom-up tree transducer [32,10], in which states may have arbitrary rank. Let Σ and Q be disjoint ranked alphabets. We define

$$\text{Lhs}(\Sigma,Q) = \{l \in \Sigma(Q(X)) \mid \text{preorder}_X(l) = x_1 \cdots x_m \text{ for some } m \in \mathbb{N}\}.$$

Definition 6. A *multi bottom-up tree transducer* (mbutt) is a tuple (Q,Σ,Δ,F,R) such that

- Q is a ranked alphabet (the *states*) disjoint with $\Sigma \cup \Delta$,
- Σ and Δ are ranked alphabets (the *input* and *output* symbols),

- $F \subseteq Q^{(1)}$ (the *final states*), and
- $R \subseteq \mathrm{Lhs}(\Sigma,Q) \times Q(T_\Delta(X))$ is a finite set (the *rules*) such that $\mathrm{var}(r) \subseteq \mathrm{var}(l)$ for every $(l,r) \in R$.

It is *nondeleting* (respectively, *linear*), if r is nondeleting (respectively, linear) in $\mathrm{var}(l)$ for every $(l,r) \in R$. Finally, it is *deterministic* (respectively, *total*) if for every l there exists at most (respectively, at least) one r such that $(l,r) \in R$.

Again we write $l \to r$ for rules (l,r). The semantics of mbutt is also given by term rewriting. Note that the set X of variables is not needed to define the tree transformation computed by an mbutt, but we will need it for the composition construction.

Definition 7. Let $M = (Q,\Sigma,\Delta,F,R)$ be an mbutt. For every $\xi,\zeta \in T_\Sigma(Q(T_\Delta(X)))$ let $\xi \Rightarrow_M \zeta$ if there exist

- a position $w \in \mathrm{pos}(\xi)$,
- a rule $l \to r \in R$, and
- a substitution $\theta\colon X \to T_\Delta(X)$

such that $\xi|_w = l\theta$ and $\zeta = \xi[r\theta]_w$. The tree transformation *computed by M* is

$$\|M\| = \{(t,u) \in T_\Sigma \times T_\Delta \mid \exists q \in F\colon t \Rightarrow_M^* q(u)\}.$$

Two mbutt are equivalent if their computed tree transformations coincide. By MBOT we denote the class of tree transformations computable by mbutt. We use the prefixes 'n', 'l', 'd', and 't' for nondeletion, linearity, determinism, and totality, respectively; e.g., the class nl − MBOT comprises all tree transformations computable by nondeleting and linear mbutt.

Lemma 8. *For every mbutt there exists an equivalent total mbutt. The involved construction preserves linearity and determinism.*

Proof. The construction is entirely similar to the classical construction for bottom-up tree transducers [10]. The newly added state can be nullary in our case. □

Next, we present a composition result, which is similar to the composition results of [23,33] for linear STA and deterministic mbutt, respectively. First let us prepare the definition of the composition of two mbutt. The general idea is the classic one: take the cross-product of the sets of states and simulate the second transducer on the right-hand sides of the first transducer. However, a k-ary state of the first transducer has k prepared (partial) output trees. Thus we also need to process those k trees with the second transducer, which gives states of the form $q\langle p_1,\dots,p_k\rangle$. This idea was already used in the composition constructions of [23,33]. For all disjoint ranked alphabets Q and P, we define the ranked alphabet

$$Q\langle P\rangle = \{q\langle p_1,\dots,p_n\rangle \mid q \in Q^{(n)}, p_1,\dots,p_n \in P\}$$

such that $\mathrm{rk}(q\langle p_1,\dots,p_n\rangle) = \sum_{i=1}^n \mathrm{rk}(p_i)$ for every $q \in Q^{(n)}$ and $p_1,\dots,p_n \in P$. Moreover, let $U = T_\Delta(X)$ and $\varphi\colon Q\langle P\rangle(U) \to Q(P(U))$ be such that

$$\varphi(q\langle p_1,\dots,p_n\rangle(u_1,\dots,u_k)) = q(p_1(u_1,\dots,u_{\mathrm{rk}(p_1)}),\dots,p_n(u_{k-\mathrm{rk}(p_n)+1},\dots,u_k))$$

for every symbol $q\langle p_1,\dots,p_n\rangle \in Q\langle P\rangle^{(k)}$ and $u_1,\dots,u_k \in U$. We extend this map to $\varphi\colon T_\Sigma(Q\langle P\rangle(U)) \to T_\Sigma(Q(P(U)))$ by $\varphi(\sigma(t_1,\dots,t_k)) = \sigma(\varphi(t_1),\dots,\varphi(t_k))$ for every $\sigma \in \Sigma^{(k)}$ and $t_1,\dots,t_k \in T_\Sigma(Q\langle P\rangle(U))$.

Definition 9. Let $M_1 = (Q,\Sigma,\Gamma,F_1,R_1)$ and $M_2 = (P,\Gamma,\Delta,F_2,R_2)$ be mbutt such that Q, P, and $\Sigma \cup \Gamma \cup \Delta$ are mutually disjoint. Moreover, let

$$M_1' = (Q,\Sigma,\Gamma \cup P \cup \Delta,F_1,R_1) \quad \text{and} \quad M_2' = (P,\Sigma \cup Q \cup \Gamma,\Delta,F_2,R_2).$$

The *composition of* M_1 *and* M_2 is the mbutt $M_1 \,;\, M_2 = (Q\langle P\rangle,\Sigma,\Delta,F_1\langle F_2\rangle,R)$ where

$$R = \{(l,r) \in \mathrm{Lhs}(\Sigma,Q\langle P\rangle) \times Q\langle P\rangle(T_\Delta(X)) \mid \varphi(l)\,(\Rightarrow_{M_1'} \,;\, \Rightarrow_{M_2'}^*)\,\varphi(r)\}.$$

Note that the construction preserves nondeletion, linearity, and determinism. Moreover, our construction generalizes the composition construction of [11] for bottom-up tree transducers (i.e., mbutt with unary states only). Let us recall the main correctness theorem from that paper: Let M_1 and M_2 be bottom-up tree transducers and M be the composition of M_1 and M_2 according to [11]. Then M computes the composition of the transformations computed by M_1 and M_2 if

- M_1 is linear or M_2 is deterministic; and
- M_1 is nondeleting or M_2 is total.

Since the construction of [11] also preserves nondeletion, linearity, and determinism, we obtain that the classes of transformations computed by linear, nondeleting and linear, and deterministic bottom-up tree transducers are all closed under composition [10,11]. This follows from the previous conditions because every bottom-up tree transducer can be turned into an equivalent total one (preserving linearity and determinism; cf. Lemma 8). The following lemma states the central property that is required to show the correctness of the construction of Definition 9. In fact, our restrictions are exactly the mentioned restrictions for bottom-up tree transducers. To avoid repetition, we assume the symbols of Definition 9.

Lemma 10. *Let (i) M_1 be linear or M_2 be deterministic; and (ii) M_1 be nondeleting or M_2 be total. In addition, let $t \in T_\Sigma$ and $\xi \in Q\langle P\rangle(T_\Delta)$. Then $t \Rightarrow^*_{M_1;M_2} \xi$ if and only if $t\,(\Rightarrow^*_{M_1} \,;\, \Rightarrow^*_{M_2'})\,\varphi(\xi)$. In particular, $\|M_1 \,;\, M_2\| = \|M_1\| \,;\, \|M_2\|$.*

Proof. Let $t = \sigma(t_1,\ldots,t_k)$ for some symbol $\sigma \in \Sigma^{(k)}$ and $t_1,\ldots,t_k \in T_\Sigma$. We first prove the "if"-direction by induction on the length of the derivation $\Rightarrow^*_{M_1}$. Let $l \to r \in R_1$ and $\theta: X \to T_\Gamma$ be such that $t \Rightarrow^*_{M_1} l\theta \Rightarrow_{M_1} r\theta \Rightarrow^*_{M_2'} \varphi(\xi)$. Since $r\theta \Rightarrow^*_{M_2'} \varphi(\xi)$, we have that for every $w \in \mathrm{pos}_X(r)$ there exists $\xi_w \in P(T_\Delta)$ such that $r\theta \Rightarrow^*_{M_2'} r\theta[\xi_w]_w \Rightarrow^*_{M_2'} \varphi(\xi)$. Since either M_1 is linear [and thus $\mathrm{card}(\mathrm{pos}_x(r)) \le 1$ for every $x \in \mathrm{var}(r)$] or M_2 is deterministic [and thus $r\theta|_w$ completely determines ξ_w for every $w \in \mathrm{pos}_X(r)$], we obtain that $\xi_v = \xi_w$ for every $v,w \in \mathrm{pos}_X(r)$ such that $r(v) = r(w)$. Consequently, let $\theta': \mathrm{var}(r) \to P(T_\Delta)$ be such that $x\theta' = \xi_w$ for some $w \in \mathrm{pos}_x(r)$. We observe that $r\theta \Rightarrow^*_{M_2'} r\theta' \Rightarrow^*_{M_2'} \varphi(\xi)$. Now, we extend θ' to a substitution $\theta': \mathrm{var}(l) \to P(T_\Delta)$ such that additionally $x\theta \Rightarrow^*_{M_2} x\theta'$ for every $x \in \mathrm{var}(l)$. This can be achieved because either M_1 is nondeleting [and thus $\mathrm{var}(l) = \mathrm{var}(r)$] or M_2 is total [and thus such $x\theta'$ exists for every $x \in \mathrm{var}(l)$]. Consequently, $t_i \Rightarrow^*_{M_1} l|_i\theta \Rightarrow^*_{M_2'} l|_i\theta'$ for every $i \in [k]$. Invoking the induction hypothesis k times, we obtain

$$\sigma(t_1,\ldots,t_k) \Rightarrow^*_{M_1;M_2} \sigma(\varphi^{-1}(l|_1\theta'),\ldots,\varphi^{-1}(l|_k\theta')) = \varphi^{-1}(l\theta').$$

Since $l\theta' \Rightarrow_{M_1'} r\theta' \Rightarrow^*_{M_2'} \varphi(\xi)$, we also obtain $\varphi^{-1}(l\theta') \Rightarrow_{M_1;M_2} \xi$ by the definition of R.

For the converse, which is proved by induction on the length of the derivation $\Rightarrow^*_{M_1;M_2}$, let $l \to r \in R$ and $\theta: X \to T_\Delta$ be such that $t \Rightarrow^*_{M_1;M_2} l\theta \Rightarrow_{M_1;M_2} r\theta = \xi$. Since $t_i \Rightarrow^*_{M_1;M_2} l|_i\theta$ for every $i \in [k]$, the induction hypothesis implies that there exist $\zeta_i \in Q(T_\Gamma)$ such that $t_i \Rightarrow^*_{M_1} \zeta_i \Rightarrow^*_{M_2'} \varphi(l|_i)\theta$. Taking $\zeta = \sigma(\zeta_1,\ldots,\zeta_k)$, we obtain that $t \Rightarrow^*_{M_1} \zeta \Rightarrow^*_{M_2'} \varphi(l)\theta$. By the definition of R, there exist $l' \to r' \in R_1$ and $\theta': X \to P(X)$ such that $l'\theta' = \varphi(l)$ and $\varphi(l) \Rightarrow_{M_1'} r'\theta' \Rightarrow^*_{M_2'} \varphi(r)$. Clearly, $\zeta = l'\theta''$ for some $\theta'': X \to T_\Gamma$, and consequently,

$$t \Rightarrow^*_{M_1} \zeta \Rightarrow_{M_1} r'\theta'' \Rightarrow^*_{M_2'} r'\theta'\theta \Rightarrow^*_{M_2'} \varphi(r)\theta = \varphi(\xi)\ ,$$

where we used that $x\theta'' \Rightarrow^*_{M_2} x\theta'\theta$ for every $x \in \mathrm{var}(l')$ [because $\zeta \Rightarrow^*_{M_2'} l'\theta'\theta$]. □

We thus obtain the main composition theorem. Note that it is known that d−MBOT is closed under composition [24]. In [25] it is shown that their deterministic mbutt, which are more powerful than our deterministic mbutt, compute exactly the class of transformations computed by deterministic tdtt$^{\mathrm{R}}$, which is closed under composition [17]. In addition, [23] proves that the classes of transformations computed by linear STA and nondeleting and linear STA are closed.

Theorem 11

$$\text{l–MBOT} \,;\, \text{MBOT} \subseteq \text{MBOT} \quad \text{and} \quad \text{MBOT} \,;\, \text{d–MBOT} \subseteq \text{MBOT}.$$

In particular, l–MBOT, nl–MBOT, *and* d–MBOT *are closed under composition.*

Proof. The inequalities follow immediately from Lemma 10 with the help of Lemma 8. The closure results are essentially due to [23,24,33], but can also be obtained by the observation that the construction of Definition 9 preserves linearity, nondeletion, and determinism. □

5. Relation to top-down devices

Let us consider how mbutt relate to tdtt$^{\mathrm{R}}$ and xtt$^{\mathrm{R}}$. An important result in this respect can be found in [25]. It is shown there that every deterministic mbutt (note that their deterministic mbutt are slightly more powerful than ours) can be simulated by a deterministic tdtt$^{\mathrm{R}}$. Here, we present a slightly different construction. Our construction is a faithful generalization of the decomposition [10] of bottom-up tree transducers. We first need to recall two more properties of tdtt$^{\mathrm{R}}$. Let $M = (Q,\Sigma,\Delta,I,R,c)$ be a tdtt$^{\mathrm{R}}$. Then M is *single-use* [30,27–29,7] if for every $q(x) \in Q(X)$ and $t \in T_\Sigma$ there exist at most one $l \to r \in R$ and $w \in \mathrm{pos}(r)$ such that $l(1) = t(\varepsilon)$, $t \in c(l \to r)$, and $r|_w = q(x)$. In the notations for classes of transformations, we use the subscript 'su' to restrict to single-use tdtt$^{\mathrm{R}}$; e.g., $\mathrm{d}-\mathrm{TOP}_{\mathrm{su}}$ denotes the class of transformations computed by deterministic single-use tdtt. Finally, a *finite-state relabeling* [10] is a tdtt (Q,Σ,Δ,I,R) such that $r \in \Delta(Q(X))$ and $\mathrm{preorder}_X(l) = \mathrm{preorder}_X(r)$ for every $l \to r \in R$, and we use QREL for the class of transformations computed by such relabelings.

Lemma 12

$$\text{l–MBOT} \subseteq \text{QREL} \,;\, \text{d–TOP}_{\text{su}} \quad \text{and} \quad \text{MBOT} \subseteq \text{QREL} \,;\, \text{d–TOP}.$$

Proof. The finite-state relabeling annotates the input tree with the transitions applied by a run of the mbutt. It thus takes care of the nondeterminism. The deterministic tdtt then executes the annotated transitions using a state for each parameter position. Note that we could obtain the second result by proving that MBOT $\subseteq$ QREL ; d $-$ MBOT and then applying the result of [25].

Let $M = (Q,\Sigma,\Delta,F,R)$ be an mbutt. We define the rank of a rule $l \to r \in R$ by $\text{rk}_R(l \to r) = \text{card}(\text{pos}_Q(l))$. Thus, R is a ranked alphabet. We construct the finite-state relabeling $M_1 = (Q,\Sigma,R,F,R_1)$ where all states in Q have rank 1 and

$$R_1 = \{r(\varepsilon)(l(\varepsilon)(x_1,\ldots,x_k)) \to (l \to r)(l(1)(x_1),\ldots,l(k)(x_k)) \mid l \to r \in R^{(k)}\}.$$

Clearly, M_1 relabels the input tree by applicable rules. The deterministic tdtt can now simply execute the annotated rules. Let $M_2 = ([\text{mx}],R,\Delta,\{1\},R_2)$ be the deterministic tdtt with $\text{mx} = \max \text{rk}(Q)$ and

$$R_2 = \{n((l \to r)(x_1,\ldots,x_k)) \to r|_n\theta_l \mid n \in [\text{mx}] \quad \text{and} \quad l \to r \in R^{(k)}\},$$

where for every $l \in \text{Lhs}(\Sigma,Q)$ the substitution $\theta_l : X \to [\text{mx}](X)$ is such that for every $x \in \text{var}(l)$ we have $\theta_l(x) = j(x_i)$ with $ij \in \text{pos}_x(l)$. Note that M_2 is single-use if M is linear.

We only sketch the correctness proof. Let $t \in T_\Sigma$, $q \in Q^{(m)}$, and $u_1,\ldots,u_m \in T_\Delta$. Suppose that $t \Rightarrow_M^* q(u_1,\ldots,u_m)$ and consider one fixed derivation d. Since one rule of M is applied at each position of the input tree, we can consider the tree s that has the same shape as t and each position is labeled with the rule that is applied at this position of t in the derivation d. It is straightforward to show that $q(t) \Rightarrow_{M_1}^* s$, i.e., the finite-state relabeling can transform t into s (in state q). Finally, we have to take care of the output. This is achieved by M_2 and it is easily seen that for every $n \in [m]$ we have $n(s) \Rightarrow_{M_2}^* u_n$. Thus, the proof obligation is

$$t \Rightarrow_M^* q(u_1,\ldots,u_m) \quad \Longleftrightarrow \quad \exists s \in T_R : q(t) \Rightarrow_{M_1}^* s \quad \text{and} \quad \forall n \in [m] : n(s) \Rightarrow_{M_2}^* u_n.$$

This can be proved by induction in a straightforward fashion. □

Now let us investigate whether the inclusions of Lemma 12 are strict. It will turn out that the inequalities are actually equalities. For this, we show how to implement a deterministic tdtt with the help of a nondeleting mbutt by a variation of [25, Lemma 4.2].

Lemma 13

$$\text{d–TOP}_{\text{su}} \subseteq \text{nl–MBOT} \quad \text{and} \quad \text{d–TOP} \subseteq \text{n–MBOT}.$$

Proof. The mbutt guesses at each position of the input tree, which states of the top-down tree transducer would process this subtree. Since the tdtt is deterministic, processing the same subtree in the same state yields the same output tree, so that the mbutt can simply copy the generated output tree. Formally, let $M = (Q,\Sigma,\Delta,I,R)$ be a deterministic tdtt such that $\text{preorder}_X(l) = x_1 \cdots x_k$ with $k = \text{rk}(l(1))$ for every $l \to r \in R$. We construct the mbutt $M' = (\mathcal{P}(Q),\Sigma,\Delta,\{I\},R')$ where $\text{rk}(P) = \text{card}(P)$ for every $P \subseteq Q$. In addition, for every $P \subseteq Q$ fix a bijection $f_P : P \to [\text{card}(P)]$. For better readability, we occasionally write $f(P,p)$ instead of $f_P(p)$. We then construct the rules of R' as follows. Let $\sigma \in \Sigma^{(k)}$, $P \in \mathcal{P}(Q)^{(n)}$, and $f_P^{-1}(j)(\sigma(x_1,\ldots,x_k)) \to r_j \in R$ for every $j \in [n]$. Moreover, for every $i \in [k]$ let

$$P_i = \bigcup_{j=1}^{n} \{q \in Q \mid \exists w \in \text{pos}(r_j) : r_j|_w = q(x_i)\}.$$

We then construct the rule $l \to P(r'_1,\ldots,r'_n)$ where $l \in \text{Lhs}(\Sigma,\mathcal{P}(Q))$ is such that $l(\varepsilon) = \sigma$ and $l(i) = P_i$ for every $i \in [k]$. Moreover, for every $j \in [n]$ the tree r'_j is obtained from r_j by replacing all occurrences of $q(x_i)$ by $l(im)$ where $m = f(P_i,q)$. Note that M' is nondeleting. Moreover, if M is single-use, then $P(r'_1,\ldots,r'_n)$ is linear in X, and hence, M' is linear. It remains to prove that for every $P \in \mathcal{P}(Q)^{(n)}$, $t \in T_\Sigma$, and $u_1,\ldots,u_n \in T_\Delta$ we have

$$t \Rightarrow_{M'}^* P(u_1,\ldots,u_n) \quad \Longleftrightarrow \quad \forall p \in P : p(t) \Rightarrow_M^* u_{f(P,p)}.$$

Induction on the length of the derivation can be used to show both directions of this statement. We obtain $\tau_{M'} = \tau_M$ for $P = I$. □

Thus, we obtain the following characterization of the power of mbutt. It also shows that every mbutt (respectively, linear mbutt) is equivalent to a nondeleting (respectively, nondeleting and linear) one.

Theorem 14

$$\begin{aligned} \text{l–MBOT} &= \text{QREL} \,;\, \text{d–TOP}_{\text{su}} = \text{nl–MBOT} \\ \text{MBOT} &= \text{QREL} \,;\, \text{d–TOP} \quad = \text{n–MBOT}. \end{aligned}$$

Proof. Since obviously, QREL $\subseteq$ nl – MBOT, the equalities follow directly from (the proof of) Theorem 11 and Lemmata 12 and 13. $\square$

The following development of the relation of mbutt to finite-copying tdtts [26] is essentially due to an anonymous referee [34]. Roughly speaking, a tdtt is finite-copying if it processes each input subtree at most a bounded number of times. Formally, a tdtt $M = (Q,\Sigma,\Delta,I,R)$ is *m-copying* for some $m \in \mathbb{N}$ if $\text{card}(\text{pos}_{\bigstar}(u)) \leq m$ for every $t \in T_\Sigma(\{\bigstar\})$ and $u \in T_\Delta(\{\bigstar\})$ such that $\text{card}(\text{pos}_{\bigstar}(t)) = 1$ and $(t,u) \in \|M'\|$ where $M' = (Q,\Sigma \cup \{\bigstar^{(0)}\},\Delta \cup \{\bigstar^{(0)}\},I,R \cup \{q(\bigstar) \to \bigstar \mid q \in Q\})$. The tdtt M is *finite-copying* if there exists an $m \in \mathbb{N}$ such that it is m-copying. We use the subscript 'fc' for classes of transformations computed by finite-copying tdtt, e.g., d$-$TOP$_{\text{fc}}$ denotes the class of all transformations computed by deterministic finite-copying tdtt. The equality QREL ; d–TOP$_{\text{su}}$ = QREL ; d–TOP$_{\text{fc}}$ is due to [30, Theorems 5.10 and 7.4], and could be added to the characterization of Theorem 14. Let us now show that every finite-copying tdtt can be simulated by a linear mbutt.

Lemma 15

$$\text{TOP}_{\text{fc}} \subseteq \text{l–MBOT}.$$

Proof. It is already hinted in [26, Lemma 3.2.3] (in the context of tree-to-string-transducers) that TOP$_{\text{fc}}$ $\subseteq$ QREL ; d-TOP$_{\text{fc}}$, which would prove the statement by Theorem 14. Again, the relabeling annotates the input tree with rules. However, since the tdtt might make a bounded number of copies of input subtrees, we annotate several rules to each position. The deterministic tdtt should execute the first rule when running on the first copy, the second rule when running on the second copy, etc. Note that this approach is closely related to the construction of Lemma 12.

Let $M = (Q,\Sigma,\Delta,I,R)$ be an m-copying tdtt such that $\text{preorder}_X(l) = x_1 \cdots x_k$ with $k = \text{rk}(l(1))$ for every $l \to r \in R$. Let $P = Q \times [m]$ be a ranked alphabet of unary symbols and $f\colon T_\Delta(P(X)) \to T_\Delta(Q(X))$ be such that $f((q,j)(x)) = q(x)$ for every $q \in Q$, $j \in [m]$, and $x \in X$ and $f(\delta(u_1,\ldots,u_k)) = \delta(f(u_1),\ldots f(u_k))$ for every $\delta \in \Delta^{(k)}$ and $u_1,\ldots,u_k \in T_\Delta(P(X))$. For every $\sigma \in \Sigma^{(k)}$, let

$$R_\sigma = \{(q,\sigma) \to r \mid r \in T_\Delta(P(X)) \text{ and } q(\sigma(x_1,\ldots,x_k)) \to f(r) \in R\}.$$

We turn $R' = \bigcup_{\sigma\in\Sigma} R_\sigma^m$ into a ranked alphabet by $\text{rk}_{R'}(\rho) = \text{rk}(\sigma)$ for every $\sigma \in \Sigma$ and $\rho \in R_\sigma^m$. The finite-state relabeling $M_1 = (\{\dagger\},\Sigma,R',\{\dagger\},R_1)$ is such that $\dagger \notin \Sigma \cup R'$ and

$$R_1 = \{\dagger(\sigma(x_1,\ldots,x_k)) \to \rho(\dagger(x_1),\ldots,\dagger(x_k)) \mid \sigma \in \Sigma^{(k)} \text{ and } \rho \in R_\sigma^m\}.$$

Finally, let $\top \notin P$. We construct the tdtt $M_2 = (P \cup \{\top\},R',\Delta,\{\top\},R_2)$ such that for every $q,q_1,\ldots,q_m \in Q$, $j \in [m]$, $r_1,\ldots,r_m \in T_\Delta(P(X))$, and $\sigma \in \Sigma^{(k)}$ with $\rho_i = (q_i,\sigma) \to r_i \in R_\sigma$ for every $i \in [m]$ we have

- $(q,j)((\rho_1,\ldots,\rho_m)(x_1,\ldots,x_k)) \to r_j \in R_2$ if $q = q_j$; and
- $\top((\rho_1,\ldots,\rho_m)(x_1,\ldots,x_k)) \to r_1 \in R_2$ if $q_1 \in I$.

It is obvious that M_2 is deterministic. In addition, we can easily prove that $q(t) \Rightarrow_M^* u$ if $(t,t') \in \|M_1\|$ and $(q,1)(t') \Rightarrow_{M_2}^* u$, for every $q \in Q$, $t \in T_\Sigma$, $u \in T_\Delta$, and $t' \in T_{R'}$. In fact, we can obtain a derivation $q(t) \Rightarrow_M^* u$ from $(q,1)(t') \Rightarrow_{M_2}^* u$ by simply changing states from (q',j) to just q' and replacing symbols of R_σ^m by σ. Hence, $\|M_1\| \,;\, \|M_2\| \subseteq \|M\|$. The same implication can also be extended to $T_\Sigma(\{\bigstar\})$ and thus be used to show that M_2 is m-copying. It remains to prove $\|M\| \subseteq \|M_1\| \,;\, \|M_2\|$. To this aim, let $w \in Q^*$ be a *state sequence of* M if there exist $q \in I$, $t \in T_\Sigma(\{\bigstar\})$, and $\xi \in T_\Delta(Q(\{\bigstar\}))$ such that $\text{card}(\text{pos}_{\bigstar}(t)) = 1$, $q(t) \Rightarrow_{M'}^* \xi$, and $w = \text{preorder}_Q(\xi)$ where M' is the extension of M given in the definition of m-copying. Clearly, every state sequence of M is at most of length m because M is m-copying. We can prove by a straightforward induction that for every state sequence $q_1 \cdots q_n$ of M, $t \in T_\Sigma$, and $u_1,\ldots,u_n \in T_\Delta$: if $q_j \Rightarrow_M^* u_j$ for every $j \in [n]$, then there exists $t' \in T_{R'}$ such that $(t,t') \in \|M_1\|$ and $(q_j,j)(t') \Rightarrow_{M_2}^* u_j$ for every $j \in [n]$. Since every initial state $q \in I$ is a state sequence of M, this proves that $\|M\| \subseteq \|M_1\| \,;\, \|M_2\|$. $\square$

Hence, we identified the composition closure of TOP$_{\text{fc}}$. It is l–MBOT, and in addition, it coincides with the second level of the composition hierarchy.

Theorem 16

$$\text{l–MBOT} = \text{TOP}_{\text{fc}} \,;\, \text{TOP}_{\text{fc}}$$

and this class is closed under composition.

Proof. The statements follow trivially from Theorems 11 and 14 and Lemma 15 because every linear tdtt is 1-copying and every single-use tdtt is n-copying where n is the number of its states. □

Let us finally investigate the relation of mbutt to xtt. We immediately observe that l–XTOP is too rich because there exist $\tau \in$ l–XTOP and $t \in T_\Sigma$ such that $\tau \cap (\{t\} \times T_\Delta)$ is infinite. However, for an mbutt M the set $\|M\| \cap (\{t\} \times T_\Delta)$ is always finite. Consequently, we restrict ourselves to epsilon-free xtt. We use the stems $\mathrm{XTOP}_{\mathrm{ef}}$ and $\mathrm{XTOP}^{\mathrm{R}}_{\mathrm{ef}}$ (with the usual prefixes) for the classes of transformations computable by epsilon-free xtt and $\mathrm{xtt}^{\mathrm{R}}$, respectively. Note that every $\mathrm{tdtt}^{\mathrm{R}}$ is epsilon-free. The following theorem follows from the proof of Theorem 4.

Theorem 17. $\mathrm{B(LCE,LC)} = \mathrm{nl–XTOP}_{\mathrm{ef}}$ *and* $\mathrm{B(LCE,L)} = \mathrm{l–XTOP}^{\mathrm{R}}_{\mathrm{ef}}$.

Corollary 18. $\mathrm{nl–XTOP}_{\mathrm{ef}}$, $\mathrm{l–XTOP}_{\mathrm{ef}}$, *and* $\mathrm{l–XTOP}^{\mathrm{R}}_{\mathrm{ef}}$ *are not closed under composition.*

By [19] we have $\mathrm{XTOP}^{\mathrm{R}}_{\mathrm{ef}} = \mathrm{TOP}^{\mathrm{R}}$, and if we reconsider the proof, then we see that if the $\mathrm{xtt}^{\mathrm{R}}$ is linear, then the constructed $\mathrm{tdtt}^{\mathrm{R}}$ also has a "finite-copying" property (note that we did not define "finite-copying" for $\mathrm{tdtt}^{\mathrm{R}}$). In fact, the resulting $\mathrm{tdtt}^{\mathrm{R}}$ will be m-copying where $m = \max\{\mathrm{card}(\mathrm{var}(r)) \mid l \to r \in R\}$ with R being the set of rules of the given $\mathrm{xtt}^{\mathrm{R}}$. We can state this as $\mathrm{l–XTOP}^{\mathrm{R}}_{\mathrm{ef}} \subseteq \mathrm{QREL} ; \mathrm{TOP}_{\mathrm{fc}}$. It can thus be shown that compositions of epsilon-free and linear xtt can be simulated by a composition of a finite-state relabeling and a deterministic tdtt, and hence by a linear mbutt. This is our main theorem for compositions of extended tdtts.

Theorem 19

$$\bigcup_{n \in \mathbb{N}} \mathrm{l–XTOP}^{n}_{\mathrm{ef}} \subset \mathrm{l–MBOT} = \mathrm{QREL} ; \mathrm{d–TOP}_{\mathrm{su}}.$$

Proof. We have the inclusions

$$\mathrm{l–XTOP}^{n}_{\mathrm{ef}} \subseteq (\mathrm{QREL} ; \mathrm{TOP}_{\mathrm{fc}})^n \subseteq \mathrm{l–MBOT}^n \subseteq \mathrm{l–MBOT}$$

by [19, Lemma 7] and Theorems 16 and 11. The equality is due to Theorem 14. Strictness follows because (by Theorem 4) every transformation of $\mathrm{l-XTOP}_{\mathrm{ef}}$ preserves recognizability [3,4] whereas some transformations of l−MBOT do not. □

6. Conclusions and open problems

We have identified a class, namely nl − MBOT, that is closed under composition and contains all transformations that can be computed by epsilon-free and linear extended tdtt. We further showed that compositions of epsilon-free and linear extended tdtt can be implemented by a single composition of a finite-state relabeling and a deterministic (single-use) tdtt.

It remains an open problem to decide whether the composition of the transformations computed by two extended tdtts can be computed by just a single extended tdtt. In the relevant subcase where the two extended tdtts are epsilon-free one can investigate how to implement (restricted) mbutts using just one extended tdtt.

Acknowledgments

The author is grateful to the anonymous referees for their insightful remarks on the draft version of the paper. Especially one referee improved the presentation dramatically. Finally, the author would like to thank Joost Engelfriet for reading an early draft of the paper.

References

[1] W.C. Rounds, Mappings and grammars on trees, Math. Syst. Theory, 4 (3) (1970) 257–287.
[2] J.W. Thatcher, Generalized^2 sequential machine maps, J. Comput. Syst. Sci. 4 (4) (1970) 339–367.
[3] F. Gécseg, M. Steinby, Tree Automata, Akadémiai Kiadó, Budapest, 1984.
[4] F. Gécseg, M. Steinby, Tree languages, in: G. Rozenberg, A. Salomaa (Eds.), Handbook of Formal Languages, vol. 3, Springer, 1997, pp. 1–68. (Chapter 1).
[5] J. Doner, Tree acceptors and some of their applications, J. Comput. Syst. Sci. 4 (4) (1970) 406–451.
[6] Z. Fülöp, H. Vogler, Syntax-Directed Semantics—Formal Models Based on Tree Transducers, Monographs in Theoret. Comput. Sci., Springer, 1998.
[7] A. Kühnemann, Benefits of tree transducers for optimizing functional programs, in: V. Arvind, R. Ramanujam (Eds.), Proc. 18th Int. Conf. Found. Software Tech. Theoret. Comput. Sci, LNCS, vol. 1530, Springer, 1998, pp. 146–157.
[8] R. Kusters, T. Wilke, Automata-based analysis of recursive cryptographic protocols, in: V. Diekert, M. Habib (Eds.), Proc. 21st Annual Symp. Theoret. Aspects Comput. Sci., LNCS, vol. 2996, Springer, 2004, pp. 382–393.
[9] M. Dauchet, S. Tison, The theory of ground rewrite systems is decidable, in: Proc. 5th Annual IEEE Symp. Logic in Comput. Sci., IEEE Computer Society, 1990, pp. 242–248.
[10] J. Engelfriet, Bottom-up and top-down tree transformations—a comparison, Math. Syst. Theory 9 (3) (1975) 198–231.
[11] B.S. Baker, Composition of top-down and bottom-up tree transductions, Inform. Control 41 (2) (1979) 186–213.

[12] A. Arnold, M. Dauchet, Transductions inversibles de forêts, Thèse 3ème cycle M. Dauchet, Université de Lille, 1975.
[13] A. Arnold, M. Dauchet, Bi-transductions de forêts, in: S. Michaelson, R. Milner (Eds.), Proc. 3rd Int. Coll. Automata, Languages and Programming, Edinburgh University Press, 1976, pp. 74–86.
[14] J. Graehl, K. Knight, Training tree transducers, in: Proc. HLT/NAACL, Association for Computational Linguists, 2004, pp. 105–112.
[15] B. Huang, K. Knight, Relabeling syntax trees to improve syntax-based machine translation quality, in: Proc. HLT/NAACL, Association for Computational Linguistics, 2006, pp. 240–247.
[16] K. Knight, J. Graehl, An overview of probabilistic tree transducers for natural language processing, in: A. F. Gelbukh (Ed.), Proc. 6th Int. Conf. Comput. Linguistics Intel. Text Proc., LNCS, vol. 3406, Springer, 2005, pp. 1–24.
[17] J. Engelfriet, Top-down tree transducers with regular look-ahead, Math. Syst. Theory 10 (1977) 289–303.
[18] A. Arnold, M. Dauchet, Morphismes et bimorphismes d'arbres, Theoret. Comput. Sci. 20 (1) (1982) 33–93.
[19] A. Maletti, J. Graehl, M. Hopkins, K. Knight, The power of extended top-down tree transducers, SIAM J. Comput. (2008), submitted for publication.
[20] S.M. Shieber, Synchronous grammars as tree transducers, in: Proc. 7th Int. Workshop Tree Adjoining Grammars and Related Formalisms, 2004, pp. 88–95.
[21] Y. Schabes, Mathematical and computational aspects of lexicalized grammars, Ph.D. thesis, University of Pennsylvania, 1990.
[22] M. Dauchet, S. Tison, Structural complexity of classes of tree languages, in: M. Nivat, A. Podelski (Eds.), Tree Automata and Languages, Elsevier Science, 1992, pp. 327–353.
[23] E. Lilin, Une généralisation des transducteurs d'états finis d'arbres:les S-transducteurs, Thèse 3ème cycle, Université de Lille, 1978.
[24] E. Lilin, Propriétés de clôture d'une extension de transducteurs d'arbres déterministes, in: E. Astesiano, C. Böhm (Eds.), Proc. 6th Coll. Trees in Algebra and Programming, LNCS, vol. 112, Springer, 1981, pp. 280–289.
[25] Z. Fülöp, A. Kühnemann, H. Vogler, A bottom-up characterization of deterministic top-down tree transducers with regular look-ahead, Inform. Proc. Lett. 91 (2004) 57–67.
[26] J. Engelfriet, G. Rozenberg, G. Slutzki, Tree transducers, L systems, and two-way machines, J. Comput. Syst. Sci. 20 (2) (1980) 150–202.
[27] R. Giegerich, Composition and evaluation of attribute coupled grammars, Acta Inform. 25 (4) (1988) 355–423.
[28] H. Ganzinger, Increasing modularity and language-independency in automatically generated compilers, Sci. Comput. Program. 3 (3) (1983) 223–278.
[29] A. Kühnemann, Berechnungsstärken von Teilklassen primitiv-rekursiver Programmschemata, Ph.D. thesis, Technische Universität Dresden, 1997.
[30] J. Engelfriet, S. Maneth, Macro tree transducers, attribute grammars, and MSO definable tree translations, Inform. Comput. 154 (1) (1999) 34–91.
[31] Z. Fülöp, A. Kühnemann, H. Vogler, Linear deterministic multi bottom-up tree transducers, Theoret. Comput. Sci. 347 (1–2) (2005) 276–287.
[32] J.W. Thatcher, Tree automata: an informal survey, in: Currents in the Theory of Computing, Prentice Hall, 1973, pp. 143–172.
[33] A. Kühnemann, Composition of deterministic multi bottom-up tree transducers, manuscript, 2006.
[34] Anonymous, Referee report, 2008.