Acta Informatica manuscript No.
(will be inserted by the editor)

Extended Multi Bottom-Up Tree Transducers
Composition and Decomposition

Joost Engelfriet · Eric Lilin · Andreas Maletti

September 4, 2009

Abstract Extended multi bottom-up tree transducers are defined and investigated. They are an extension of multi bottom-up tree transducers by arbitrary, not just shallow, left-hand sides of rules; this includes rules that do not consume input. It is shown that such transducers, even linear ones, can compute all transformations that are computed by linear extended top-down tree transducers, which are a theoretical model for syntax-based machine translation. Moreover, the classical composition results for bottom-up tree transducers are generalized to extended multi bottom-up tree transducers. Finally, characterizations in terms of extended top-down tree transducers and tree bimorphisms are presented.

Keywords tree transducer · tree transformation · tree language · composition · tree homomorphism · natural language processing

Mathematics Subject Classification (2000) 68Q45 · 68T50

CR Subject Classification F.4.3 · F.1.1

This is an extended and revised version of: J. Engelfriet, E. Lilin, A. Maletti. *Extended Multi Bottom-up Tree Transducers.* Proc. 12th Int. Conf. Developments in Language Theory. LNCS 5257, 289–300, Springer-Verlag 2008.

A. Maletti was supported by a fellowship within the Postdoc-Programme of the German Academic Exchange Service (DAAD) and by the *Ministerio de Educación y Ciencia* (MEC) grant JDCI-2007-760.

Corresponding author: A. Maletti, Universitat Rovira i Virgili, Departament de Filologies Romàniques, Av. Catalunya 35, 43002 Tarragona, Spain. E-mail: `andreas.maletti@urv.cat`, Phone: +34-977-558-382, Fax: +34-977-558-386

J. Engelfriet
Leiden Institute of Advanced Computer Science, Leiden University, P.O. Box 9512
2300 RA Leiden, The Netherlands. E-mail: `engelfri@liacs.nl`

E. Lilin
Université des Sciences et Technologies de Lille, UFR IEEA 59655, Villeneuve d'Ascq, France.
E-mail: `eric.lilin@lifl.fr`

A. Maletti
International Computer Science Institute, 1947 Center Street, Suite 600, Berkeley, CA 94704, USA. E-mail: `maletti@icsi.berkeley.edu`

1 Introduction

The area of machine translation recently embraced syntactical translation models [29], which rely on the syntactical structure of a sentence to perform the translation. The new subfield *syntax-based machine translation* [10] was successfully established, and the first results look very promising. However, there are two major problems: (i) The syntax-based approach is computationally much more expensive (up to the point of computational infeasibility) than the classical phrase-based approach and (ii) there currently is a wealth of formal models that compete to become the implementation standard that finite-state transducers [28] became for phrase-based machine translation.

In syntax-based machine translation, the translation of an input sentence uses not only the words of the sentence and the context in which they appear, but rather also uses the syntactic structure of the sentence. Thus, the input sentence is first parsed and the transformation is then started on the syntax trees so obtained. We can distinguish two modes of translation: tree-to-string and tree-to-tree. In the former mode, the translation device immediately generates sentences of the output language whereas, in the latter mode, the translation device translates syntax trees of an input sentence into syntax trees of output sentences. In this contribution we will focus on the tree-to-tree mode; in fact, most tree-to-tree devices can be turned into tree-to-string devices by only considering the yield (the sequence of nullary output symbols) of the output they generate.

In [29] and personal communication, Kevin Knight proposed the following criteria that any reasonable formal tree-to-tree model of syntax-based machine translation should fulfil:

(a) It should be a genuine generalization of finite-state transducers [7, 28].
(b) It should be efficiently trainable.
(c) It should be able to handle rotations (on the tree level).
(d) Its induced class of transformations should be closed under composition.

Let us explain why (a) and (d) are important. The feature of finite-state transducers of particular relevance is that they have 'extended' rules, i.e., rules that consume any number of input symbols and produce any number of output symbols. This includes the use of epsilon rules, i.e., rules that do not consume any input symbol. There exists a wealth of phrase- and word-based models that are built as finite-state transducers. These models represent vast knowledge about translations based on the words and the context in which they appear. This information should be reused in syntax-based models, but extended rules occur quite frequently in those finite-state transducers. It might be argued that extended rules are, strictly speaking, not necessary in machine translation, but designers use them quite freely. In particular, especially in probabilistic models, epsilon rules can be a compact representation because excessive use of epsilon rules deteriorates the likelihood of the derivation. We note that the classical bottom-up and top-down tree transducers [22, 23, 45] have rules that consume exactly one input symbol.

The closure under composition is important because it is infeasible to train one transducer that handles the complete machine translation task. Instead, "small" task-specific transducers (like transducers that reorder the syntax tree, insert output subtrees that have no counterpart in the input syntax tree, or do word-by-word translation [47]) are trained. Once those "small" transducers are trained, we would like

to compose them into a single transducer, so that further operations (like composing with a language model transducer, obtaining a regular tree grammar [22, 23] for the output syntax trees generated for a specific input syntax tree, etc.) can be applied to a single transducer and not a chain of transducers. Second, the success of finite-state transducer toolkits (like CARMEL [25] and OPENFSM [2]) is largely due to the fact that they rely on only one model. This simplifies the usage of the toolkits and allows rapid development of translation systems.

GRAEHL and KNIGHT [26] proposed the linear nondeleting extended (top-down) tree transducer (ln-xtt) [3, 4] as a suitable formal model of syntax-based machine translation. It fulfils (a)–(c) but fails to fulfil (d); see [34, Corollary 5] and [35, Theorem 5.2] for the failure of (d). Further models were proposed but, to the authors' knowledge, they all fail at least one criterion. Table 1 shows some important models and their properties.

We propose a formal model that satisfies criteria (a), (c), and (d), and has more expressive power than the ln-xtt. The device is called linear extended multi bottom-up tree transducer, where, as usual, 'linear' means that no variable occurs twice in the right-hand side of a rule. The extended multi bottom-up tree transducer (xmbutt) is obtained from the bottom-up tree transducer [11, 45] by the addition of two features: 'multi' means that each input tree is translated into a sequence of output trees rather than just one output tree, formalized by allowing states of arbitrary rank, and 'extended' means that the left-hand side of a rule can contain any number of input symbols rather than just one input symbol, as explained above. In this paper we formally define and investigate the xmbutt and various restrictions (e.g., *linear*, *nondeleting*, and *deterministic*). Note that we consider the xmbutt in general, not just its linear restriction. The (non-extended) multi bottom-up tree transducer was recently studied in [19, 20, 34], and we generalize several results of those papers to the extended case.

In Section 3 we present a number of basic properties of xmbutts. For example, we construct for every xmbutt an equivalent nondeleting xmbutt (see Proposition 9). This can be achieved by guessing the required translations. Though the construction preserves linearity, it obviously destroys determinism. In Section 4 we prove a *one-symbol normal form* for xmbutts (see Theorem 15): each rule of the xmbutt either consumes one input symbol (without producing output), or produces one output symbol (without consuming input). The transformation into this normal form preserves all three restrictions above. This shows in particular that the (linear) xmbutt has the same expressive power as the (linear) multi bottom-up tree transducer of [19, 20, 34], enhanced with epsilon rules. In the deterministic case the epsilon rules can even be removed (Corollary 17), showing that the extension does not add expressive power. Thus, by the result of [19], the deterministic xmbutt is as powerful as the deterministic top-down tree transducer with regular look-ahead [12], and we prove (Theorem 18) that in the linear case the latter transducer has the so-called single-use property (known from attribute grammars [16, 21, 24, 30, 31]). The one-symbol normal form allows us to give an easy, intuitively clear way of composing xmbutts as opposed to the usual constructions in the literature, where possibly large right-hand sides of rules have to be processed.

Our main result (Theorem 23 in Section 5) states that the class of transformations computed by xmbutts is closed under pre-composition with transformations computed by linear xmbutts and under post-composition with those computed by deterministic xmbutts. In particular, we also obtain that the classes of transformations computed by linear and/or deterministic xmbutts are closed under composition. These results are analogous to classical results (see [11, Theorems 4.5 and 4.6] and [6, Corollary 7]) for

Model \ Criterion	(a)	(b)	(c)	(d)
Linear nondeleting top-down tree transducer [38, 44]	–	x	–	x
Quasi-alphabetic tree bimorphism [43]	–	x	–	x
Synchronous context-free grammar [1]	x	x	–	x
Synchronous tree substitution grammar [40]	x	x	x	–
Synchronous tree adjoining grammar [5, 41, 42]	x	x	x	–
Linear complete tree bimorphism [5]	x	x	x	–
Linear nondeleting extended top-down tree transducer [3, 4, 26, 27, 35]	x	x	x	–
Linear multi bottom-up tree transducer [19, 20, 34]	–	?	x	x
Linear extended multi bottom-up tree transducer [*this paper*]	x	?	x	x

Table 1 Overview of formal models with respect to desired criteria. "x" marks fulfilment; "–" marks failure to fulfil. A question mark shows that this remains open though we conjecture fulfilment.

bottom-up tree transducers and thus show the "bottom-up" nature of xmbutts. Also, they are analogous to the composition results of [34, Theorem 11] for (non-extended) multi bottom-up tree transducers. As in [34], our proof essentially uses the principle set forth in [6, Theorem 6], but, as observed above, the one-symbol normal form allows us to present a very simple composition construction for xmbutts and verify that it is correct, provided that the first input transducer is linear or the second is deterministic. We observe here that the "extension" of a tree transducer model (or even just the addition of epsilon rules) can, in general, destroy closure under composition, as can be seen from the linear nondeleting top-down tree transducer. This seems to be due to the non-existence of a one-symbol normal form in the top-down case.

We verify in Section 3 that linear xmbutts have sufficient power for syntax-based machine translation. This is because, as mentioned before (and shown in Proposition 5), they can simulate all ln-xtts. Thus, we have a lower bound to the power of linear xmbutts. In fact, even the composition closure of the class of transformations computed by ln-xtts is strictly contained in the class of transformations computed by linear xmbutts (see Corollary 7). In Section 6, we present an exact characterization in terms of extended top-down tree transducers (Theorem 25): xmbutts are as powerful as compositions of an ln-xtt with a deterministic top-down tree transducer, and in the linear case the latter transducer has the single-use property. Thus, the composition of two extended top-down tree transducers forms an upper bound to the power of the linear xmbutt. The obtained results are summarized in an inclusion diagram (see Figure 5). Additionally, we characterize the power of xmbutts in terms of bimorphisms (Theorem 24 in Section 5). In particular, this yields that every xmbutt has a recognizable set of derivations, which suggests that xmbutts can be efficiently trained.

Overall, this paper is an extended and revised version of the conference paper [15]. Several results were first presented in 1978 in the thesis [32] (in French) of Lilin.

2 Preliminaries

Let A, B, C be sets. The powerset of A is denoted by $\mathcal{P}(A)$. A relation from A to B is a subset of $A \times B$. Let $\tau_1 \subseteq A \times B$ and $\tau_2 \subseteq B \times C$. The composition of τ_1 and τ_2 is

$$\tau_1 \,;\, \tau_2 = \{(a, c) \mid \exists b \in B\colon (a, b) \in \tau_1, (b, c) \in \tau_2\} \ .$$

This composition is lifted to classes of relations in the usual manner.

The nonnegative integers are denoted by $\mathbb{N}$ and $\{i \mid 1 \leq i \leq k\}$ is denoted by $[k]$. A ranked set is a set Σ of symbols with a relation $\mathrm{rk} \subseteq \Sigma \times \mathbb{N}$ such that $\{k \mid (\sigma, k) \in \mathrm{rk}\}$ is finite for every $\sigma \in \Sigma$. Commonly, we denote the ranked set only by Σ and the set of k-ary symbols of Σ by $\Sigma^{(k)} = \{\sigma \in \Sigma \mid (\sigma, k) \in \mathrm{rk}\}$. We also denote that $\sigma \in \Sigma^{(k)}$ by writing $\sigma^{(k)}$. Given two ranked sets Σ and Δ with associated rank relations rk_Σ and rk_Δ, respectively, the set $\Sigma \cup \Delta$ is associated the rank relation $\mathrm{rk}_\Sigma \cup \mathrm{rk}_\Delta$. A ranked set Σ is uniquely-ranked if for every $\sigma \in \Sigma$ there exists exactly one k such that $(\sigma, k) \in \mathrm{rk}$. For uniquely-ranked sets, we denote this k simply by $\mathrm{rk}(\sigma)$. An alphabet is a finite set, and a ranked alphabet is a ranked set Σ such that Σ is an alphabet.

Let Σ be a ranked set. The set of Σ-trees, denoted by T_Σ, is the smallest set T such that $\sigma(t_1, \ldots, t_k) \in T$ for every $k \in \mathbb{N}$, $\sigma \in \Sigma^{(k)}$, and $t_1, \ldots, t_k \in T$. We write α instead of $\alpha()$ if $\alpha \in \Sigma^{(0)}$. Let $\Gamma \subseteq \Sigma$ and $H \subseteq T_\Sigma$. By $\Gamma(H)$ we denote $\{\gamma(t_1, \ldots, t_k) \mid \gamma \in \Gamma^{(k)}, t_1, \ldots, t_k \in H\}$. Now, let Δ be a ranked set. We denote by $T_\Delta(H)$ the smallest set $T \subseteq T_{\Sigma \cup \Delta}$ such that $H \subseteq T$ and $\Delta(T) \subseteq T$.

Let $t \in T_\Sigma$. The set of positions of t, denoted by $\mathrm{pos}(t)$, is defined by

$$\mathrm{pos}(\sigma(t_1, \ldots, t_k)) = \{\varepsilon\} \cup \{iw \mid i \in [k], w \in \mathrm{pos}(t_i)\}$$

for every $\sigma \in \Sigma^{(k)}$ and $t_1, \ldots, t_k \in T_\Sigma$. Note that we denote the empty string by ε and that $\mathrm{pos}(t) \subseteq \mathbb{N}^*$. The height of t is $\mathrm{height}(t) = \max\{|w| + 1 \mid w \in \mathrm{pos}(t)\}$, and the size of t is $\mathrm{size}(t) = \mathrm{card}(\mathrm{pos}(t))$. Let $w \in \mathrm{pos}(t)$ and $u \in T_\Sigma$. The subtree of t that is rooted in w is denoted by $t|_w$, the symbol of t at w is denoted by $t(w)$, and the tree obtained from t by replacing the subtree rooted at w by u is denoted by $t[u]_w$. For every $\Gamma \subseteq \Sigma$ and $\sigma \in \Sigma$, let $\mathrm{pos}_\Gamma(t) = \{w \in \mathrm{pos}(t) \mid t(w) \in \Gamma\}$ and $\mathrm{pos}_\sigma(t) = \mathrm{pos}_{\{\sigma\}}(t)$.

Let $X = \{x_i \mid i \geq 1\}$ be a set of formal variables, each variable is considered to have the unique rank 0. For every $k \geq 0$, let $X_k = \{x_i \mid i \in [k]\}$. In what follows, we assume that the input, output, and state alphabets of tree transducers do not contain variables. A tree $t \in T_\Sigma(X)$ is linear (respectively, nondeleting) in $V \subseteq X$ if $\mathrm{card}(\mathrm{pos}_v(t)) \leq 1$ (respectively, $\mathrm{card}(\mathrm{pos}_v(t)) \geq 1$) for every $v \in V$. The set of variables of t is $\mathrm{var}(t) = \{v \in X \mid \mathrm{pos}_v(t) \neq \emptyset\}$ and the sequence of variables is given by $\mathrm{yield}_X \colon T_\Sigma(X) \to X^*$ with $\mathrm{yield}_X(v) = v$ for every $v \in X$ and

$$\mathrm{yield}_X(\sigma(t_1, \ldots, t_k)) = \mathrm{yield}_X(t_1) \cdots \mathrm{yield}_X(t_k)$$

for every $\sigma \in \Sigma^{(k)}$ and $t_1, \ldots, t_k \in T_\Sigma(X)$. A tree $t \in T_\Sigma(X)$ is *normalized* if $\mathrm{yield}_X(t) = x_1 \cdots x_m$ for some $m \in \mathbb{N}$. Every mapping $\theta \colon V \to T_\Sigma(X)$ with $V \subseteq X$ is a substitution. We define the application of θ to a tree in $T_\Sigma(V)$ inductively by $v\theta = \theta(v)$ for every $v \in V$ and $\sigma(t_1, \ldots, t_k)\theta = \sigma(t_1\theta, \ldots, t_k\theta)$ for every $\sigma \in \Sigma^{(k)}$ and $t_1, \ldots, t_k \in T_\Sigma(V)$.

An *extended* (top-down) *tree transducer* (xtt, or *transducteur généralisé descendant*) [3, 4] is a tuple $M = (Q, \Sigma, \Delta, I, R)$ where Q is a uniquely-ranked alphabet such that $Q = Q^{(1)}$, Σ and Δ are ranked alphabets, $I \subseteq Q$, and R is a finite set of rules of the form $l \to r$ with $l \in Q(T_\Sigma(X))$ linear in X, and $r \in T_\Delta(Q(\mathrm{var}(l)))$. The xtt M is linear (respectively, nondeleting) if r is linear (respectively, nondeleting) in $\mathrm{var}(l)$ for every $l \to r \in R$. It is epsilon-free, if $l \notin Q(X)$ for every $l \to r \in R$. The semantics of the xtt is given by term rewriting. Let $\xi, \zeta \in T_\Delta(Q(T_\Sigma))$. We write $\xi \Rightarrow_M \zeta$ if there exist a rule $l \to r \in R$, a position $w \in \mathrm{pos}(\xi)$, and a substitution $\theta \colon X \to T_\Sigma$ such that $\xi|_w = l\theta$ and $\zeta = \xi[r\theta]_w$. The tree transformation computed by M is the relation $\tau_M = \{(t, u) \in T_\Sigma \times T_\Delta \mid \exists q \in I \colon q(t) \Rightarrow_M^* u\}$. In Figure 1 we display two example

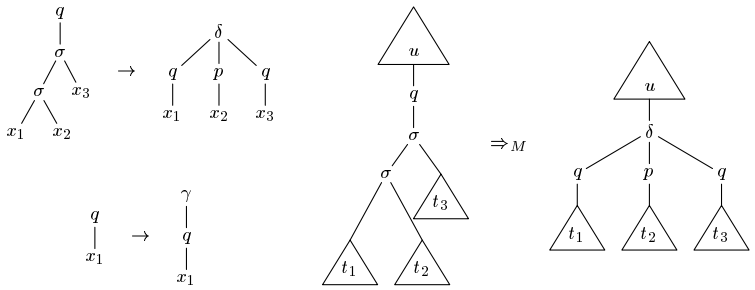

Fig. 1 Two example rules of an xtt M and a derivation step using the first rule.

rules and a derivation step to illustrate the above definitions. The class of all tree transformations computed by xtts is denoted by XTOP. The prefixes 'l', 'n', and 'e' are used to restrict to linear, nondeleting, and epsilon-free devices, respectively. Thus, ln-XTOP denotes the class of all tree transformations computable by linear and nondeleting xtts.

An xtt $M = (Q, \Sigma, \Delta, I, R)$ is a *top-down tree transducer* [38, 44] if for every rule $l \to r \in R$ there exist $q \in Q$ and $\sigma \in \Sigma^{(k)}$ such that $l = q(\sigma(x_1, \ldots, x_k))$. Note that every left-hand side of a rule of a top-down tree transducer is normalized. The top-down tree transducer M is *deterministic* (respectively, *total*) if (i) $\mathrm{card}(I) = 1$ (respectively, $\mathrm{card}(I) \geq 1$) and (ii) for every $l \in Q(\Sigma(X))$ there exists at most (respectively, at least) one r such that $l \to r \in R$. Finally, M is *single-use* [21, 24, 30, 31] if for every $q(v) \in Q(X)$, $k \in \mathbb{N}$, and $\sigma \in \Sigma^{(k)}$ there exist at most one $l \to r \in R$ and at most one $w \in \mathrm{pos}(r)$ such that $l(1) = \sigma$ and $r|_w = q(v)$. We use TOP and $\mathrm{TOP}_{\mathrm{su}}$ to denote the classes of transformations computed by top-down tree transducers and single-use top-down tree transducers, respectively. We also use the prefixes 'l', 'n', 'd' and 't' to restrict to linear, nondeleting, deterministic, and total devices, respectively.

We use the standard notion of a *recognizable* (or *regular*) tree language [22, 23], and we denote the class of all recognizable tree languages by REC. Moreover, we let $\mathrm{REC}(\Sigma) = \{L \subseteq T_\Sigma \mid L \in \mathrm{REC}\}$ and $\mathcal{T}(\mathrm{REC}) = \{\tau(L) \mid \tau \in \mathcal{T} \text{ and } L \in \mathrm{REC}\}$ for a class $\mathcal{T}$ of tree transformations.

Next, we recall top-down tree transducers with regular look-ahead [12]. A *top-down tree transducer with regular look-ahead* is a pair $\langle M, c\rangle$ such that $M = (Q, \Sigma, \Delta, I, R)$ is a top-down tree transducer and $c\colon R \to \mathrm{REC}(\Sigma)$. We say that such a transducer $\langle M, c\rangle$ is *deterministic* if (i) $\mathrm{card}(I) = 1$ and (ii) for every $l \in Q(\Sigma(X))$ and $t \in T_\Sigma$ there exists at most one r such that $l \to r \in R$, $l(1) = t(\varepsilon)$, and $t \in c(l \to r)$. Similarly, $\langle M, c\rangle$ is *single-use* (called *strongly single use restricted* in [16, Definition 5.5]) if for every $q(v) \in Q(X)$ and $t \in T_\Sigma$ there exist at most one $l \to r \in R$ and at most one $w \in \mathrm{pos}(r)$ such that $l(1) = t(\varepsilon)$, $t \in c(l \to r)$, and $r|_w = q(v)$. The semantics $\Rightarrow_{\langle M,c\rangle}$ of $\langle M, c\rangle$ is defined in the same manner as for an xtt with the additional restriction that $l\theta|_1 \in c(l \to r)$. The computed tree transformation $\tau_{\langle M,c\rangle}$ is defined as for an xtt. We use $\mathrm{TOP}^{\mathrm{R}}$ and $\mathrm{TOP}^{\mathrm{R}}_{\mathrm{su}}$ to denote the classes of transformations computed by top-down tree transducers with regular look-ahead and single-use top-down tree transducers with regular look-ahead, respectively. We use the prefix 'd' in the usual manner.

For further information on recognizable tree languages and tree transducers, we refer the reader to [22, 23].

3 Extended Multi Bottom-up Tree Transducers

In this section, we recall *S-transducteurs ascendants généralisés* [32], which are a generalization of *S-transducteurs ascendants* (STA) [32, 33]. We choose to call them *extended multi bottom-up tree transducers* here in line with [19, 20, 34], where 'extended' refers to the fact that trees of arbitrary size are allowed on both sides of the rules and 'multi' refers to the fact that states may have ranks different from one.

Definition 1 An *extended multi bottom-up tree transducer* (for short: xmbutt) is a tuple $(Q, \Sigma, \Delta, F, R)$ where

- Q is a uniquely-ranked alphabet of *states*, disjoint with $\Sigma \cup \Delta$;
- Σ and Δ are ranked alphabets of *input* and *output symbols*, respectively;
- $F \subseteq Q \setminus Q^{(0)}$ is a set of *final states*; and
- R is a finite set of *rules* of the form $l \to r$ where $l \in T_\Sigma(Q(X))$ is linear in X and $r \in Q(T_\Delta(\mathrm{var}(l)))$.

A rule $l \to r \in R$ is an *epsilon rule* if $l \in Q(X)$; otherwise it is *input-consuming*. Note that the right-hand side of an epsilon rule need not be in $Q(X)$, i.e., can contain output symbols. The sets of epsilon and input-consuming rules are denoted by R^ε and R^Σ, respectively.

An xmbutt $M = (Q, \Sigma, \Delta, F, R)$ is a *multi bottom-up tree transducer* (mbutt) [respectively, an STA] if $l \in \Sigma(Q(X))$ [respectively, $l \in \Sigma(Q(X)) \cup Q(X)$] for every $l \to r \in R$. It is an *extended bottom-up tree transducer* (xbutt) if all states of Q are of rank 1, and it is a bottom-up tree transducer [11, 45] if it is both an mbutt and an xbutt. Linearity and nondeletion of xmbutts are defined in the natural manner. The xmbutt M is *linear* if r is linear in $\mathrm{var}(l)$ for every rule $l \to r \in R$. It is *nondeleting* if (i) $F \subseteq Q^{(1)}$ and (ii) r is nondeleting in $\mathrm{var}(l)$ for every $l \to r \in R$. Moreover, M is *deterministic* if (i) there do not exist two distinct rules $l_1 \to r_1 \in R$ and $l_2 \to r_2 \in R$, a substitution $\theta\colon X \to X$, and $w \in \mathrm{pos}(l_2)$ such that $l_1\theta = l_2|_w$, and (ii) there does not exist an epsilon rule $l \to r \in R$ such that $l(\varepsilon) \in F$. Note that, intuitively speaking, in a deterministic xmbutt there exist no useful states that contribute to a cycle of epsilon rules and thus epsilon rules can be removed in the standard manner from a deterministic xmbutt (see Proposition 10 for a detailed account). Finally, an mbutt M is *total* if for every $k \in \mathbb{N}$, every $\sigma \in \Sigma^{(k)}$, and every sequence $q_1, \ldots, q_k \in Q$, there exists a rule $l \to r \in R$ such that $l(\varepsilon) = \sigma$ and $l(i) = q_i$ for all $i \in [k]$. It is not easy to define totality for an arbitrary xmbutt in a syntactic way; fortunately we will not need such a definition.

Let us now present a rewrite semantics. For later use (in Proposition 10 and Section 5) we define the rewriting for ranked sets larger than Σ and Δ (and possibly containing variables).

In the rest of this and the next section, let $M = (Q, \Sigma, \Delta, F, R)$ be an xmbutt (unless otherwise specified).

Definition 2 Let Σ' and Δ' be ranked sets disjoint with Q. Moreover, let $l \to r \in R$, $\xi, \zeta \in T_{\Sigma\cup\Sigma'}(Q(T_{\Delta\cup\Delta'}))$, and $w \in \mathrm{pos}(\xi)$. We write $\xi \Rightarrow_M^{l\to r,w} \zeta$ if there exists a substitution $\theta\colon X \to T_{\Delta\cup\Delta'}$ such that $\xi|_w = l\theta$ and $\zeta = \xi[r\theta]_w$. We write $\xi \Rightarrow_M^{l\to r} \zeta$ if there exists $w \in \mathrm{pos}(\xi)$ such that $\xi \Rightarrow_M^{l\to r,w} \zeta$, and we write $\xi \Rightarrow_M \zeta$ if there exists $\rho \in R$ such that $\xi \Rightarrow_M^{\rho} \zeta$. The *tree transformation computed by* M is

$$\tau_M = \{(t, \xi|_1) \mid t \in T_\Sigma, \xi \in F(T_\Delta), t \Rightarrow_M^* \xi\} \ .$$

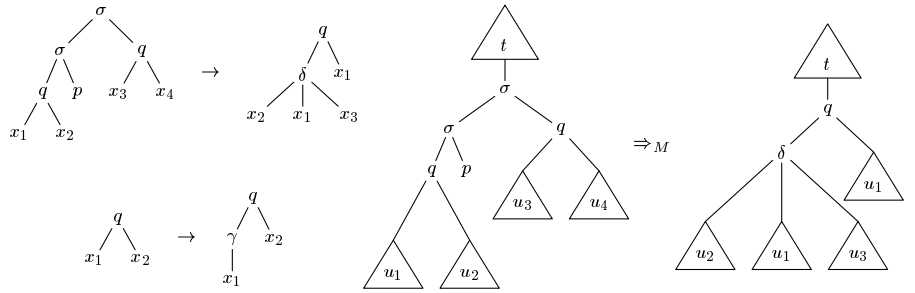

Fig. 2 Two example rules (the upper one is input-consuming and the lower one is an epsilon rule) of an xmbutt M and a derivation step using the first rule.

Clearly, in a derivation $t \Rightarrow_M^* \xi$ with $t \in T_\Sigma$ and $\xi \in Q(T_\Delta)$, the first applied rule $l \to r \in R$ cannot contain states in the left-hand side (i.e., $l \in T_\Sigma$). Moreover, if an xmbutt M is deterministic, then for every $t \in T_\Sigma$ there exists at most one $\xi \in F(T_\Delta)$ such that $t \Rightarrow_M^* \xi$. Hence, τ_M is a partial function for every deterministic xmbutt M. Note that if M is a deterministic (respectively, total) mbutt and $t \in T_\Sigma$, then there exists at most one (respectively, at least one) $\xi \in Q(T_\Delta)$ such that $t \Rightarrow_M^* \xi$.

Figure 2 displays example rules and a derivation step. The xmbutts M and N are *equivalent* if $\tau_M = \tau_N$. We denote by XMBOT, MBOT, XBOT, and BOT the classes of tree transformations computed by xmbutts, mbutts, xbutts, and bottom-up tree transducers, respectively. We use the prefixes 'l', 'n', and 'd' to restrict to linear, nondeleting, and deterministic devices, respectively. For example, l-XMBOT denotes the class of all tree transformations computed by linear xmbutts, which is the formal model for syntax-based machine translation that we proposed in the Introduction. Similarly, ld-XMBOT is the subclass of transformations computed by linear and deterministic xmbutts.

Example 3 Let $\Sigma = \{\delta^{(2)}, \sigma^{(1)}, \gamma^{(1)}, \alpha^{(0)}\}$ and $M = (\{p^{(1)}, q^{(2)}\}, \Sigma, \Sigma, \{p\}, R)$ be the xmbutt with R consisting of the following rules

$$\alpha \to q(\alpha, \alpha) \qquad \sigma(q(x_1, x_2)) \to p(\delta(x_1, x_2))$$
$$\gamma(q(x_1, x_2)) \to q(\gamma(x_1), \gamma(x_2)) \qquad p(x_1) \to p(\sigma(x_1)) \ .$$

This xmbutt M is linear and nondeleting and computes the transformation

$$\{(\sigma(t), \sigma^n(\delta(t,t))) \mid t \in T_\Gamma, n \in \mathbb{N}\}$$

where $\Gamma = \{\gamma^{(1)}, \alpha^{(0)}\}$. If we remove the epsilon rule $p(x_1) \to p(\sigma(x_1))$, then the xmbutt M becomes a linear, nondeleting, and deterministic mbutt that computes the transformation $\{(\sigma(t), \delta(t,t)) \mid t \in T_\Gamma\}$. □

Example 3 shows that linear xmbutts have some copying power, which implies that not every transformation of l-XMBOT (not even of lnd-MBOT) preserves recognizability. Obviously, this is due to the fact that states may have ranks larger than one. In fact, it will be shown in Corollary 7 that the transformations of l-XBOT preserve recognizability.

The following terminology and lemma will be useful in formal constructions and formal correctness proofs. Since the lemma is intuitively obvious, its proof is left to

the reader. Let $\xi \in T_\Sigma(Q(T_\Delta(X)))$ with $k = \mathrm{card}(\mathrm{pos}_Q(\xi))$. We can write ξ (uniquely) as $\xi = \bar{\xi}\theta_\xi$ such that $\bar{\xi} \in T_\Sigma(X_k)$ is normalized with $\mathrm{yield}_X(\bar{\xi}) = x_1 \cdots x_k$ and $\theta_\xi \colon X_k \to Q(T_\Delta(X))$ is a substitution.

Lemma 4 *For all $t \in T_\Sigma$, $n \in \mathbb{N}$, and $\xi \in T_\Sigma(Q(T_\Delta))$ with $k = \mathrm{card}(\mathrm{pos}_Q(\xi))$, we have $t \Rightarrow_M^n \xi$ if and only if there is a substitution $\theta_{\mathrm{in}} \colon X_k \to T_\Sigma$ and there are $n_1, \ldots, n_k \in \mathbb{N}$ with $\sum_{i \in [k]} n_i = n$, such that $t = \bar{\xi}\theta_{\mathrm{in}}$ and $\theta_{\mathrm{in}}(x_i) \Rightarrow_M^{n_i} \theta_\xi(x_i)$ for all $i \in [k]$.*

This lemma is similar to the usual one for context-free grammars: viewing the elements of $Q(T_\Delta)$ as nonterminals, the application of a rule $l \to r$ of M can be viewed as backwards application of a rule $r\theta \to l\theta$ of a context-free grammar (with $\theta \colon X \to T_\Delta$ as in Definition 2).

In the remainder of this section we establish a number of basic properties of xmbutts. First, we verify that l-XMBOT is suitably powerful for applications in machine translation. We do this by showing that ln-XTOP coincides with ln-XBOT. In particular, this shows that even linear and nondeleting xbutts can handle rotations [29].

Proposition 5

$$\text{ln-XBOT} = \text{ln-XTOP} \subsetneq \text{l-XTOP} \subsetneq \text{l-XBOT} \ .$$

Proof The inclusion l-XTOP $\subseteq$ l-XBOT can be proved in a similar manner as the inclusion l-TOP $\subseteq$ l-BOT [11, Theorem 2.8]. We now recall the idea. Let $q(l) \to r$ be a rule of the linear xtt M. We construct a rule $l\theta \to q(r')$ of the xbutt M' where θ is the substitution that maps every variable v to $\bot(v)$ for a new state $\bot$, if v does not occur in r, or to $p(v)$ with p being the state such that $p(v)$ occurs in r. Finally, r' is obtained from r by replacing all occurrences of $p(v)$ simply by v. In addition, we add the rule $\sigma(\bot(x_1), \ldots, \bot(x_k)) \to \bot(\sigma(x_1, \ldots, x_k))$ for every $k \in \mathbb{N}$ and $\sigma \in \Sigma^{(k)}$. Note that if M is nondeleting, then the constructed xbutt M' will also be nondeleting. Moreover, in that case $\bot$ is not needed, and thus our construction actually establishes a bijection between linear nondeleting xtt and linear nondeleting xbutt. We can easily prove that $t \Rightarrow_{M'}^* q(u)$ iff $q(t) \Rightarrow_M^* u$ with the help of Lemma 4 for M' and the dual lemma for the xtt M. Thus we proved the inclusion and the equality ln-XBOT = ln-XTOP, which generalizes the equality ln-BOT = ln-TOP [11, Theorem 2.9].

We leave the strictness proof of the inclusion ln-XTOP $\subsetneq$ l-XTOP to the reader (see also (8) in the proof of Theorem 26). The strictness of the inclusion l-XTOP $\subsetneq$ l-XBOT, which generalizes the strict inclusion l-TOP $\subsetneq$ l-BOT [11, Theorem 2.8], follows from l-TOP$^{\mathrm{R}} \not\subseteq$ XTOP [35, Lemma 4.3] and l-BOT = l-TOP$^{\mathrm{R}}$ [12, Theorem 2.8]. □

The equality l-BOT = l-TOP$^{\mathrm{R}}$ [12, Theorem 2.8] mentioned in the previous proof can easily be generalized to l-XBOT = l-XTOP$^{\mathrm{R}}$, where l-XTOP$^{\mathrm{R}}$ is the class of transformations computed by linear xtts with regular look-ahead as defined (in the obvious way) in [34, 35].

Next, let us discuss the relation of xmbutts to bimorphisms [4, 5]. Nivat's Theorem [7, Theorem III.3.2] shows that finite-state transducers compute exactly the transformations of the form $\{(f(s), g(s)) \mid s \in L\}$ where f and g are string homomorphisms and L is a regular string language. In what follows we start to develop a similar characterization for classes of transformations computed by xmbutts.

Let X and Y be classes of total functions on trees. We indicate by $\mathrm{B}(X, Y)$ the class of all *bimorphisms* of the form $\{(f(s), g(s)) \mid s \in L\}$ where $f \in X$ with $f \colon T_\Gamma \to T_\Sigma$,

$g \in Y$ with $g\colon T_\Gamma \to T_\Delta$, and $L \in \mathrm{REC}(\Gamma)$. An mbutt $M = (Q, \Sigma, \Delta, F, R)$ is *homomorphic* if (i) M is total deterministic, (ii) Q is a singleton, say $Q = \{\star\}$, and (iii) $F = Q$. Note that, for every such homomorphic mbutt M and every $t \in T_\Sigma$, there are unique $u_1, \ldots, u_m \in T_\Delta$ such that $t \Rightarrow_M^* \star(u_1, \ldots, u_m)$, where $m = \mathrm{rk}(\star)$. Consequently, τ_M is a total function. For classes of transformations computed by mbutts, we use the prefix 'h' to restrict to homomorphic devices. The elements of h-MBOT are called m-morphisms (if $\mathrm{rk}(\star) = m$) in [5, 32]. Note that 1-morphisms, i.e., the elements of h-BOT, are the usual tree homomorphisms [22, 23]. The m-morphisms could be called multi-tree homomorphisms since they associate a sequence of m trees (with variables) to each input symbol.

We will use the relation to bimorphisms to prove that certain transformations preserve recognizability. Thus, we recall that every transformation τ in B(h-BOT, lh-BOT) preserves recognizability. In fact, let $\tau = \{(f(s), g(s)) \mid s \in L\}$ where $f \in$ h-BOT, $g \in$ lh-BOT, and $L \in$ REC. If L' is now also a recognizable tree language, then $\tau(L') = g(f^{-1}(L') \cap L)$ is recognizable too, because REC is closed under inverse tree homomorphisms, intersection, and linear tree homomorphisms [22].

It is shown in [34, Theorem 4] (as a variation of a result of [4]) that

$$\text{ln-XTOP} = \mathrm{B}(\text{lnh-BOT}, \text{lnh-BOT}) \ .$$

Thus, by Proposition 5, the same bimorphism characterization holds for ln-XBOT: it equals B(lnh-BOT, lnh-BOT). The inclusion of the class ln-XBOT in the class B(lnh-BOT, lnh-BOT) essentially shows that the derivations of a linear and nondeleting xbutt M form a recognizable tree language from which the transformation τ_M can be recovered by two linear and nondeleting tree homomorphisms, one for the input trees and one for the output trees. Let us investigate this in more detail because a recognizable set of derivations is an important property for trainability. Recall that $M = (Q, \Sigma, \Delta, F, R)$ is an xmbutt. We turn R into a uniquely-ranked alphabet such that $\mathrm{rk}(\rho) = \mathrm{card}(\mathrm{pos}_Q(l))$ for every rule $\rho = (l \to r) \in R$. In addition, let $\{w_1, \ldots, w_k\} = \mathrm{pos}_Q(l)$ such that $w_1 < \cdots < w_k$ with respect to the lexicographic ordering on $\mathbb{N}^*$. We define $\mathrm{in}(\rho) = (l(w_1), \ldots, l(w_k))$ and $\mathrm{out}(\rho) = r(\varepsilon)$. Note that, using the notation introduced before Lemma 4, $\mathrm{in}(\rho) = (\theta_l(x_1)(\varepsilon), \ldots, \theta_l(x_{\mathrm{rk}(\rho)})(\varepsilon))$. For every $q \in Q$, let $D_q(M)$ be the set of trees $s \in T_R$ such that

(i) $\mathrm{out}(s(\varepsilon)) = q$ and
(ii) $\mathrm{in}(s(w)) = (\mathrm{out}(s(w1)), \ldots, \mathrm{out}(s(wk)))$ where $k = \mathrm{rk}(s(w))$ for every position $w \in \mathrm{pos}(s)$.

Finally, let $D(M) = \bigcup_{q \in F} D_q(M)$. Obviously, $D(M) \subseteq T_R$ is a recognizable tree language.

Next, we generalize the inclusion ln-XBOT $\subseteq$ B(lnh-BOT, lnh-BOT) to arbitrary xmbutts, using multi-tree homomorphisms instead of ordinary ones for the output trees. We also show that the first two components of the bimorphism can be replaced by a linear and nondeleting xbutt.

Lemma 6

$$\begin{aligned}
\text{XMBOT} &\subseteq \mathrm{B}(\text{lnh-BOT}, \text{h-MBOT}) &&\subseteq \text{ln-XBOT ; h-MBOT} \\
\text{l-XMBOT} &\subseteq \mathrm{B}(\text{lnh-BOT}, \text{lh-MBOT}) &&\subseteq \text{ln-XBOT ; lh-MBOT} \\
\text{XBOT} &\subseteq \mathrm{B}(\text{lnh-BOT}, \text{h-BOT}) &&\subseteq \text{ln-XBOT ; h-BOT} \\
\text{l-XBOT} &\subseteq \mathrm{B}(\text{lnh-BOT}, \text{lh-BOT}) &&\subseteq \text{ln-XBOT ; lh-BOT}
\end{aligned}$$

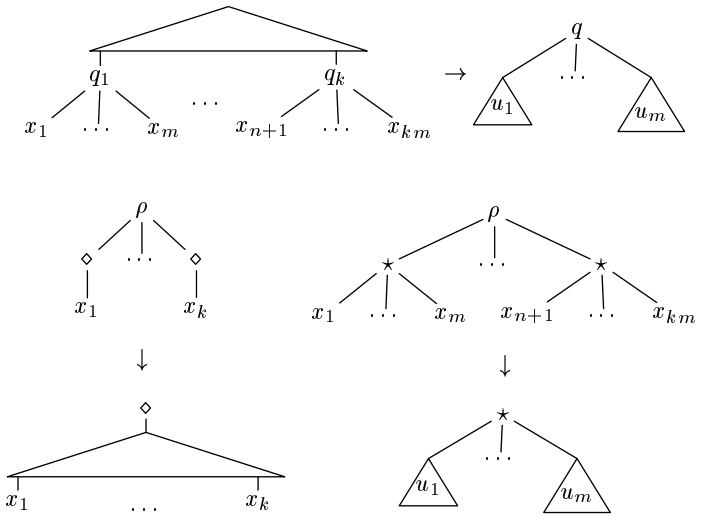

Fig. 3 Illustration of the construction of the rules in Lemma 6 where $n = (k-1)m$. Top: rule ρ in M. Bottom: corresponding rules in M_1 (left) and M_2 (right).

Proof Let $M = (Q, \Sigma, \Delta, F, R)$ be an xmbutt. To avoid uninteresting technical details, we assume, without loss of generality, that all left-hand sides of rules are normalized (see Section 2), and that all states have the same rank, say rank m. Note that the latter property can easily be satisfied by adding dummy variables to the states in the left-hand sides of rules, and adding dummy output trees to the states in the right-hand sides.

Let $M_1 = (\{\diamond\}, R, \Sigma, \{\diamond\}, R_1)$ with $\mathrm{rk}(\diamond) = 1$ and $M_2 = (\{\star\}, R, \Delta, \{\star\}, R_2)$ with $\mathrm{rk}(\star) = m$. It remains to define R_1 and R_2. Every rule $\rho \in R$ gives rise to a rule in R_1 and a rule in R_2 as follows. Let $\rho = l \to r$ with $\mathrm{rk}(\rho) = k$. Then R_1 and R_2 contain the normalized rules

$$\rho(\diamond(x_1), \ldots, \diamond(x_k)) \to \diamond(\bar{l})$$
$$\rho(\star(x_1, \ldots, x_m), \ldots, \star(x_{(k-1)m+1}, \ldots, x_{km})) \to \star(r|_1, \ldots, r|_m) \ ,$$

respectively, where $\bar{l}$ is defined before Lemma 4. The constructed rules are illustrated in Figure 3.

We claim that $\tau_M = \{(\tau_{M_1}(s), \tau_{M_2}(s)) \mid s \in D(M)\}$. Clearly, M_1 and M_2 satisfy the syntactic requirements. The semantic correctness of the construction follows from the following statement. For every $t \in T_\Sigma$, $q \in Q$, and $u_1, \ldots, u_m \in T_\Delta$, we have $t \Rightarrow_M^* q(u_1, \ldots, u_m)$ if and only if there exists $s \in D_q(M)$ such that

(i) $s \Rightarrow_{M_1}^* \diamond(t)$ and
(ii) $s \Rightarrow_{M_2}^* \star(u_1, \ldots, u_m)$.

This statement can easily be shown by induction on the length of the first derivation and by structural induction on s. As an example of a correctness proof using Lemma 4, we will give the details of the proof for one direction. Suppose that $t \Rightarrow_M^* \xi \Rightarrow_M^\rho q(u_1, \ldots, u_m)$ with $\rho = l \to r$. Due to the form of $q(u_1, \ldots, u_m)$, the rule ρ is applied at the root of ξ, i.e., $\xi \Rightarrow_M^{\rho,\varepsilon} q(u_1, \ldots, u_m)$. Thus, there exists a substitution $\theta\colon X \to T_\Delta$ such that $\xi = l\theta$ and $q(u_1, \ldots, u_m) = r\theta$. This implies that

$\bar{\xi} = \bar{l}$ and $\theta_\xi = \theta_l \,;\theta$ (the composition of the substitutions θ_l and θ). By Lemma 4 there is a substitution $\theta_{\mathrm{in}}\colon X \to T_\Sigma$ such that $t = \bar{\xi}\theta_{\mathrm{in}}$ and $\theta_{\mathrm{in}}(x_i) \Rightarrow_M^* \theta_\xi(x_i)$ for all $i \in [k]$, where $k = \mathrm{card}(\mathrm{var}(\bar{\xi})) = \mathrm{rk}(\rho)$. Note that, by Lemma 4, the derivations $\theta_{\mathrm{in}}(x_i) \Rightarrow_M^* \theta_\xi(x_i)$ can be chosen to be shorter than $t \Rightarrow_M^* q(u_1, \ldots, u_m)$. Hence, by the induction hypothesis, there are trees $s_1, \ldots, s_k \in T_R$ such that $s_i \in D_{q_i}(M)$, where $q_i = \theta_\xi(x_i)(\varepsilon) = \theta_l(x_i)(\varepsilon)$ and so $\mathrm{in}(\rho) = (q_1, \ldots, q_k)$; moreover,

$$s_i \Rightarrow_{M_1}^* \diamond(\theta_{\mathrm{in}}(x_i)) \qquad \text{and} \qquad s_i \Rightarrow_{M_2}^* \star(\theta_\xi(x_i)|_1, \ldots, \theta_\xi(x_i)|_m) \ .$$

Now take $s = \rho(s_1, \ldots, s_k)$, which is in $D_q(M)$ because $\mathrm{out}(\rho) = r(\varepsilon) = q$. By Lemma 4 applied to M_1 and using the rule of R_1 displayed above, we obtain that

$$s \Rightarrow_{M_1}^* \rho(\diamond(\theta_{\mathrm{in}}(x_1)), \ldots, \diamond(\theta_{\mathrm{in}}(x_k))) \Rightarrow_{M_1} \diamond(\bar{l}\theta_{\mathrm{in}}) = \diamond(\bar{\xi}\theta_{\mathrm{in}}) = \diamond(t) \ .$$

Similarly, and using the fact that $\theta_\xi = \theta_l \,;\theta$, we obtain for M_2 that

$$\begin{aligned} s &\Rightarrow_{M_2}^* \rho(\star(\theta_\xi(x_1)|_1, \ldots, \theta_\xi(x_1)|_m), \ldots, \star(\theta_\xi(x_k)|_1, \ldots, \theta_\xi(x_k)|_m)) \\ &= \rho(\star(\theta(x_1), \ldots, \theta(x_m)), \ldots, \star(\theta(x_{(k-1)m+1}), \ldots, \theta(x_{km}))) \\ &\Rightarrow_{M_2} \star(r\theta|_1, \ldots, r\theta|_m) = \star(u_1, \ldots, u_m) \ . \end{aligned}$$

The other direction can be shown in a similar way: If $s = \rho(s_1, \ldots, s_k) \in D_q(M)$ and s satisfies items (i) and (ii), then $t \ (\Rightarrow_M^* \,; \Rightarrow_M^\rho) \ q(u_1, \ldots, u_m)$. The details are left to the reader.

For the final inclusions, we note that the identity transformation is contained in lnh-BOT. Thus, with the help of ln-XBOT = B(lnh-BOT, lnh-BOT) (see the discussion before this Lemma), we can write

$$\mathrm{B}(\text{lnh-BOT}, Y) \subseteq \mathrm{B}(\text{lnh-BOT}, \text{lnh-BOT}) \,; Y = \text{ln-XBOT} \,; Y \ ,$$

which yields the given inclusions. □

Let us quickly discuss the implications of Lemma 6. First, it shows that we can separate input and output behavior on the one hand (performed by the homomorphic mbutts) from nondeterminism and state checking on the other hand (encoded in the tree language $D(M)$). In addition, Lemma 6 shows that the set of derivations of an arbitrary xmbutt is recognizable; it is modelled by the set of "derivation trees" $D(M)$. This is a strong indication toward the existence of efficient training algorithms. Finally, Lemma 6 also limits the power of xmbutts, but we will provide more detail and tighter bounds later on. In fact, it is straightforward to prove directly that the left-most inclusions of Lemma 6 are actually equalities. However, we will prove this indirectly for XMBOT and l-XMBOT in Theorem 24 after establishing our composition results. We note that the equality l-XBOT = B(lnh-BOT, lh-BOT) is shown in [34, Theorem 4] for l-$\mathrm{XTOP}^{\mathrm{R}}$ instead of l-XBOT (see the remark after the proof of Proposition 5).

By Lemma 6 and one of the observations made before that lemma, every transformation of l-XBOT preserves recognizability. Thus, by Example 3, we obtain the following result, which shows that the copying power of linear xmbutts is due to the states with rank larger than one.

Corollary 7 *Every transformation of* l-XBOT *preserves recognizability. Consequently,* l-XBOT $\subsetneq$ l-XMBOT.

Let us now investigate the effect of extended left-hand sides. We will prove in Lemma 14 that for every xmbutt there is an equivalent STA, which shows that rules with several input symbols can be avoided. However, epsilon rules are essential, as illustrated in the next lemma, which is clearly false in the presence of epsilon rules.

Lemma 8 *Let M be an xmbutt without epsilon rules. Then there exists $c \in \mathbb{N}$ such that* $\mathrm{height}(u) \leq c \cdot \mathrm{height}(t)$ *for every $(t,u) \in \tau_M$. Moreover, if M is linear, then* $\mathrm{size}(u) \leq c \cdot \mathrm{size}(t)$ *for every $(t,u) \in \tau_M$.*

Proof In the absence of epsilon rules we can clearly apply at most $\mathrm{size}(t)$ derivation steps in any derivation starting with the input tree t. If M is linear, then each derivation step increases the total number of output symbols by at most the number of output symbols in the applied rule. In the general case, a similar observation applies to the height of the output tree if we consider the paths in the trees. Formally, we can prove the statements as follows.

Let $M = (Q, \Sigma, \Delta, F, R)$ be an xmbutt such that $R^{\varepsilon} = \emptyset$. For the first statement, let $f = \mathrm{height}$ and $\bigoplus = \max$, and for the second statement, let $f = \mathrm{size}$ and $\bigoplus = \sum$. Moreover, let $c = f(r)$ where r is a maximal (with respect to f) right-hand side of all rules of R. We now inductively prove that $f(\zeta) \leq c \cdot f(t)$ for every $t \in T_\Sigma$ and $\zeta \in Q(T_\Delta)$ such that $t \Rightarrow_M^* \zeta$. Thus, there exists a rule $l \to r \in R$ such that $t \Rightarrow_M^* \xi \Rightarrow_M^{l \to r} \zeta$, and due to the form of ζ, there exists a substitution $\theta \colon X \to T_\Delta$ such that $\xi = l\theta$ and $\zeta = r\theta$. As in the proof of Lemma 6, this implies that $\bar{\xi} = \bar{l}$ and $\theta_\xi = \theta_l \,;\, \theta$. Note that $\bar{l} \notin X$ (because $R^{\varepsilon} = \emptyset$). By Lemma 4 there is a substitution $\theta_{\mathrm{in}} \colon X \to T_\Sigma$ such that $t = \bar{\xi}\theta_{\mathrm{in}}$ and $\theta_{\mathrm{in}}(x) \Rightarrow_M^* \theta_\xi(x)$ for every $x \in \mathrm{var}(\bar{\xi})$. By the induction hypothesis we obtain that $f(\theta_\xi(x)) \leq c \cdot f(\theta_{\mathrm{in}}(x))$ for every $x \in \mathrm{var}(\bar{\xi})$. Since $\xi = l\theta$ and $\zeta = r\theta$, the linearity of r (for the second statement) implies that $f(\zeta) \leq f(r) + \bigoplus_{v \in \mathrm{var}(l)} f(\theta(v))$. From the fact that $\theta_\xi = \theta_l \,;\, \theta$ we obtain that $f(\theta_\xi(x)) = 1 + \bigoplus_{v \in \mathrm{var}(\theta_l(x))} f(\theta(v))$ for every $x \in \mathrm{var}(\bar{\xi})$. Hence,

$$f(\zeta) < f(r) + \bigoplus_{x \in \mathrm{var}(\bar{\xi})} f(\theta_\xi(x)) \leq c + \bigoplus_{x \in \mathrm{var}(\bar{\xi})} c \cdot f(\theta_{\mathrm{in}}(x)) \leq c \cdot f(t)$$

because $\bigoplus_{x \in \mathrm{var}(\bar{\xi})} f(\theta_{\mathrm{in}}(x)) < f(t)$, due to the fact that $t = \bar{\xi}\theta_{\mathrm{in}}$ and $\bar{\xi} \notin X$. This completes the induction proof of the auxiliary statement and we can easily conclude both statements. □

The statements of Lemma 8 generalize the corresponding statements for bottom-up tree transducers. A special case of the second statement is shown in [20, Lemma 3.7].

Next, we discuss in which cases we can actually remove the epsilon rules. We need a preliminary result first. It shows that every xmbutt can be turned into an equivalent nondeleting one. Unfortunately, the construction does not preserve determinism. A similar result is proved for mbutts in [34, Theorem 14] in an indirect way; for linear xmbutts it was first presented in [32, Theorem VI.1], also in an indirect way. Here we give a direct construction.

Roughly speaking, all subtrees of a final state at the root of the input tree except the first subtree are deleted. The remaining deleted subtrees can then be computed in a top-down fashion down to the leaves of the input tree. More precisely, they can be computed by a deterministic top-down tree automaton that works on a tree of $D(M)$ that models the derivation of the given xmbutt M. We simulate the computation of the top-down tree automaton nondeterministically bottom-up in the usual way, without building up the deleted trees.

Proposition 9 XMBOT = n-XMBOT *and* l-XMBOT = ln-XMBOT.

Proof For the xmbutt $M = (Q, \Sigma, \Delta, F, R)$ we construct an equivalent nondeleting xmbutt $N = (P, \Sigma, \Delta, F', R')$. The set P of states of N consists of all pairs $\langle q, J\rangle$ with $q \in Q$ and $J \subseteq [\mathrm{rk}(q)]$. The rank of $\langle q, J\rangle$ is $\mathrm{card}(J)$; moreover, $F' = \{\langle q, \{1\}\rangle \mid q \in F\}$.

For a tree $t = q(t_1, \ldots, t_k)$ with $q \in Q^{(k)}$ and $t_1, \ldots, t_k \in T_\Delta(X)$, we define $\varphi(t) = \{\langle q, \{i_1, \ldots, i_m\}\rangle(t_{i_1}, \ldots, t_{i_m}) \mid 1 \leq i_1 < \cdots < i_m \leq k\}$. The mapping φ is extended to trees $t \in T_\Sigma(Q(T_\Delta(X)))$ by

$$\varphi(\sigma(t_1, \ldots, t_k)) = \{\sigma(u_1, \ldots, u_k) \mid u_i \in \varphi(t_i) \text{ for all } i \in [k]\}$$

for every $\sigma \in \Sigma^{(k)}$ and $t_1, \ldots, t_k \in T_\Sigma(Q(T_\Delta(X)))$. The set R' of rules of N is now defined to be

$$\{l' \to r' \mid \exists\, l \to r \in R\colon l' \in \varphi(l), r' \in \varphi(r), \mathrm{var}(l') = \mathrm{var}(r')\}\ .$$

By definition, N is nondeleting. Since the rules of N are obtained from those of M by removing subtrees (and renaming states), N is linear whenever M is.

Let $t \in T_\Sigma$ be an input tree, $q \in Q^{(k)}$, $J = \{i_1, \ldots, i_m\} \subseteq [k]$ with $i_1 < \cdots < i_m$, and $u_{i_1}, \ldots, u_{i_m} \in T_\Delta$. It can easily be shown that $t \Rightarrow_N^* \langle q, J\rangle(u_{i_1}, \ldots, u_{i_m})$ if and only if there exist $u_i \in T_\Delta$ for every $i \in [k] \setminus J$ such that $t \Rightarrow_M^* q(u_1, \ldots, u_k)$. The proof is by induction on the length of the derivations. For the if-direction, note that for every $l \to r \in R$ and $r' \in \varphi(r)$ there is a (unique) $l' \in \varphi(l)$ such that $l' \to r' \in R'$; this formalizes the (deterministic) top-down intuition discussed before this theorem. Taking $q \in F$ and $J = \{1\}$ this shows that $\tau_N = \tau_M$. □

Now we return to investigate when epsilon rules can be eliminated. A tree transformation $\tau \subseteq T_\Sigma \times T_\Delta$ is *finitary* if $\{u \mid (t, u) \in \tau\}$ is finite for every $t \in T_\Sigma$. Lemma 8 shows that τ_M is finitary for every xmbutt M without epsilon rules (see [35, Lemma 4.9]). For every deterministic xmbutt M, the transformation τ_M is a partial function and thus finitary. The next proposition shows that every finitary transformation of XMBOT can actually be computed by an xmbutt without epsilon rules. Using Lemma 14 we can even obtain an equivalent mbutt.

Proposition 10 *For every xmbutt M such that τ_M is finitary, there exists an equivalent xmbutt N without epsilon rules. Moreover, if M is linear (respectively, nondeleting, deterministic), then so is N.*

Proof Intuitively, epsilon rules can be removed in the standard way because if τ_M is finitary, then only a bounded number of output symbols can be produced (provided M is nondeleting or deterministic) in a series of consecutive applications of epsilon rules. We now turn to the formal proof. It is rather long because we wish to show how the notion of a useful rule can be formalized and applied, on the basis of Lemma 6.

Let $M = (Q, \Sigma, \Delta, F, R)$. Without loss of generality, we can assume that M is nondeleting or deterministic by Proposition 9. Moreover, we may assume, again without loss of generality, that for every rule there exists a successful derivation in which it participates, i.e., formally, that for every rule $\rho \in R$ there exists a tree $s \in D(M)$ such that ρ occurs in s (where $D(M)$ is defined before Lemma 6). This condition can easily be tested because $D(M)$ is a recognizable tree language, and all rules that fail the test can safely be removed because $\tau_M = \{(\tau_{M_1}(s), \tau_{M_2}(s)) \mid s \in D(M)\}$, as shown in the proof of Lemma 6. This assumption implies that for every $\rho \in R$ with $\mathrm{out}(\rho) = q \in Q^{(m)}$,

(i) there exist $t \in T_\Sigma$ and $\zeta \in Q(T_\Delta)$ such that $t\ (\Rightarrow_M^* ; \Rightarrow_M^\rho)\ \zeta$, and
(ii) there exist a linear tree $t \in T_\Sigma(Q(X_m))$ and a tree $\xi \in F(T_\Delta(X_m))$ such that $t|_w = q(x_1, \ldots, x_m)$ for some $w \in \mathrm{pos}(t)$ and $t \Rightarrow_M^* \xi$.

In fact, by the above assumption, there exist $s_0 \in T_R(\{x_1\})$ and $s_1, \ldots, s_k \in T_R$, where $k = \mathrm{rk}(\rho)$, such that x_1 occurs exactly once in s_0 and $s_0\theta' \in D(M)$ with $\theta'(x_1) = \rho(s_1, \ldots, s_k)$. Clearly, $\rho(s_1, \ldots, s_k) \in D_q(M)$. This implies item (i) by the proof of Lemma 6 (note that M_1 and M_2 in that proof are total). Similarly, item (ii) can be obtained from the correctness statement in the proof of Lemma 6, applied to the xmbutt M' that is obtained from M by adding a new m-ary symbol $\tilde{q}$ to Σ, new nullary symbols $\tilde{x}_1, \ldots, \tilde{x}_m$ to both Σ and Δ, and the new rule $\tilde{\rho}$, which is $\tilde{q}(\tilde{x}_1, \ldots, \tilde{x}_m) \to q(\tilde{x}_1, \ldots, \tilde{x}_m)$ to R. The statement should be applied to $s_0\theta''$ with $\theta''(x_1) = \tilde{\rho}$. Since $\mathrm{in}(\tilde{\rho})$ is the empty sequence and $\mathrm{out}(\tilde{\rho}) = \mathrm{out}(\rho)$, we have that $s_0\theta'' \in D(M')$. Further details are left to the reader. Note that we may also assume that item (ii) holds for every state $q \in Q^{(m)}$, because those states that do not appear in the right-hand sides of rules can obviously be removed.

Let us move on to the main statement. We call a state $q \in Q$ an *end state* if there exists an input-consuming rule $l \to r \in R^\Sigma$ such that $\mathrm{pos}_q(l) \neq \emptyset$. We denote the set of all end states of Q by E. For every input-consuming rule $\rho = (l \to r) \in R^\Sigma$, we construct the set $\mathrm{rhs}(\rho) = \{r' \in Q(T_\Delta(X)) \mid r \Rightarrow_M^* r' \text{ and } r'(\varepsilon) \in E \cup F\}$. Note that the shape of r restricts any derivation $r \Rightarrow_M^* r'$ to use only epsilon rules. Moreover, $\mathrm{rhs}(\rho)$ is finite for every $\rho \in R^\Sigma$ because M is deterministic or nondeleting. In the former case, $\mathrm{rhs}(\rho)$ contains at most one element (in fact, exactly one element by item (ii)), which is easily seen from the definition of determinism. Now consider the latter case, in which M is nondeleting. Suppose that $\mathrm{rhs}(\rho)$ is infinite. Then there exists a state $q \in E \cup F$ such that $T = \{r' \in \mathrm{rhs}(\rho) \mid r'(\varepsilon) = q\}$ is infinite by the pigeon-hole principle. Now let $t' \in T_\Sigma$ and $\zeta \in Q(T_\Delta)$ be such that $t'\ (\Rightarrow_M^* ; \Rightarrow_M^\rho)\ \zeta$, and let $t \in T_\Sigma(Q(X_m))$ be a linear tree, $p \in F$ a final state (which is of rank 1), and $u \in T_\Delta(X_m)$ such that $t|_w = q(x_1, \ldots, x_m)$ for some $w \in \mathrm{pos}(t)$ and $t \Rightarrow_M^* p(u)$. Note that u is nondeleting in X_m because M is nondeleting. Moreover, let $\theta\colon X \to T_\Delta$ be such that $\zeta = r\theta$ (as in Definition 2). Finally, let $\theta'\colon X_m \to T_\Delta$ be such that $\theta'(x_i) = r'\theta|_i$ for every $i \in [m]$. Then

$$t[t']_w \Rightarrow_M^* t[\zeta]_w = t[r\theta]_w \Rightarrow_M^* t[r'\theta]_w \Rightarrow_M^* p(u\theta')$$

and thus $(t[t']_w, u\theta') \in \tau_M$ for every $r' \in T$. Note that $u\theta'$ contains all output symbols from r' since u is nondeleting in X_m. Consequently, τ_M is not finitary because T is infinite. This contradicts the assumption and allows us to conclude that $\mathrm{rhs}(\rho)$ is finite.

Next, we construct the xmbutt $M' = (Q, \Sigma, \Delta, F, R')$ where

$$R' = \bigcup_{l \to r \in R^\Sigma} \{l \to r' \mid r' \in \mathrm{rhs}(l \to r)\} \ .$$

Clearly, the xmbutt M' is linear (respectively, nondeleting, deterministic) if M is so. By the definition of $\mathrm{rhs}(l \to r)$ it should also be clear that M' contains no epsilon rules and that M' and M are equivalent. □

Since every deterministic xmbutt computes a finitary transformation, we immediately obtain that the same transformation can also be computed by a deterministic xmbutt without epsilon rules, or even a deterministic mbutt by Lemma 14. More generally, using Proposition 10, Lemma 14, and a straightforward subset construction, it

can be shown that for every xmbutt M such that τ_M is a partial function, there exists an equivalent deterministic mbutt.

We have seen in Example 3 that linear xmbutts possess some copying power, and we can now use Proposition 10 to show that this copying power of linear xmbutts is restricted.

Proposition 11 d-XMBOT $\not\subseteq$ l-XMBOT.

Proof Let us prove that h-BOT $\not\subseteq$ l-XMBOT. Let $M = (\{\star\}, \Sigma, \Delta, \{\star\}, R')$ with $\Sigma = \{a^{(1)}, e^{(0)}\}$, $\Delta = \{a^{(2)}, e^{(0)}\}$, and R' contains the two rules

$$e \to \star(e) \quad \text{and} \quad a(\star(x_1)) \to \star(a(x_1, x_1)) \ .$$

The transformation $\tau_M \in$ h-BOT transforms a unary tree $s \in T_\Sigma$ into a full binary tree such that along all paths from the root to a leaf we read s. Now suppose that $\tau_M \in$ l-XMBOT. Since τ_M is finitary, we can conclude by Proposition 10 that τ_M can be computed by a linear xmbutt without epsilon rules. Thus, by Lemma 8 there exists an integer $c \in \mathbb{N}$ such that $\mathrm{size}(u) \leq c \cdot \mathrm{size}(t)$ for every $(t, u) \in \tau_M$. Clearly, this is a contradiction, which proves $\tau_M \notin$ l-XMBOT. □

We finally discuss totality. It is easy to see that every xmbutt can be turned into an equivalent one that is "dynamically total", which means that for every $t \in T_\Sigma$ there exists at least one $\xi \in Q(T_\Delta)$ such that $t \Rightarrow_M^* \xi$. This extends a classical result for bottom-up tree transducers [22, Lemma IV.3.5]. It will follow from Corollary 17 and the next lemma that this also holds for deterministic xmbutts. The lemma treats the easy case of deterministic mbutts. A similar result is also stated in [34, Lemma 8].

Lemma 12 *For every deterministic mbutt M there exists an equivalent total deterministic mbutt N. Moreover, if M is linear, then so is N.*

Proof We use a construction that is similar to the one for bottom-up tree transducers [22, Lemma IV.3.5]. Specifically, let $M = (Q, \Sigma, \Delta, F, R)$ and let $\bot \notin Q$ be a new state of rank 0. We construct the mbutt $N = (Q', \Sigma, \Delta, F, R \cup R')$ where $Q' = Q \cup \{\bot^{(0)}\}$ and R' contains the normalized rule

$$\sigma(q_1(x_1, \ldots, x_{\mathrm{rk}(q_1)}), \ldots, q_k(x_{m-\mathrm{rk}(q_k)+1}, \ldots, x_m)) \to \bot$$

with $m = \sum_{i=1}^{k} \mathrm{rk}(q_i)$ for every $\sigma \in \Sigma^{(k)}$ and $q_1, \ldots, q_k \in Q'$ such that there exists no $l \to r \in R$ with $l(\varepsilon) = \sigma$ and $l(j) = q_j$ for every $j \in [k]$.

It should be clear that N and M are equivalent since new transitions only yield the new state $\bot$, which is not final. Moreover, N is total deterministic, and N is linear whenever M is so. □

4 One-symbol Normal form

Recall that $M = (Q, \Sigma, \Delta, F, R)$ is an xmbutt. The following normal form will be at the heart of our composition construction in the next section. It says that exactly one symbol, irrespective whether input or output symbol, occurs in every rule. It is a generalization of the similar well-known normal form for finite-state transducers (see [7, Corollary III.6.2]).

Definition 13 The rule $l \to r \in R$ is a *one-symbol rule* if

$$\mathrm{card}(\mathrm{pos}_\Sigma(l)) + \mathrm{card}(\mathrm{pos}_\Delta(r)) = 1 \ .$$

The xmbutt M is in *one-symbol normal form* if it has only one-symbol rules.

For every one-symbol rule $l \to r \in R$ we either have

- $l \in \Sigma(Q(X))$ and $r \in Q(X)$, or
- $l \in Q(X)$ and $r = q(u_1, \ldots, u_n)$ for some $q \in Q^{(n)}$ and $u_1, \ldots, u_n \in X \cup \Delta(X)$ such that $u_i \in \Delta(X)$ for exactly one $i \in [n]$.

In particular, every xmbutt in one-symbol normal form is an STA. Next, we show that every xmbutt can be transformed into an equivalent xmbutt in one-symbol normal form. This is achieved in two steps: first we construct an equivalent STA and then an equivalent STA in one-symbol normal form.

Lemma 14 *For every xmbutt M there exists an equivalent STA N. Moreover, if M is linear (respectively, nondeleting, deterministic) then so is N, and if M has no epsilon rules then N is an mbutt.*

Proof Let us assume, without loss of generality, that the left-hand side l of every rule of M is normalized (see Section 2; i.e., $\mathrm{yield}_X(l) = x_1 \cdots x_m$ for some $m \in \mathbb{N}$). We decompose rules with more than one input symbol in the left-hand side into several rules, using (a variation of) the construction of [32, Proposition II.B.5]. Choose a uniquely-ranked set P and a bijection $f : T_\Sigma(Q(X)) \to P$ such that

(i) $Q \subseteq P$,
(ii) $f(q(x_1, \ldots, x_n)) = q$ for every $q \in Q^{(n)}$, and
(iii) $\mathrm{rk}(f(l)) = \mathrm{card}(\mathrm{var}(l))$ for every $l \in T_\Sigma(Q(X))$.

In fact, we will only use $f(l)$ for normalized $l \in T_\Sigma(Q(X))$.

A finite number of elements of P will be used as states of the STA to be constructed. The idea is that if $f(l) = p$ and $u_1, \ldots, u_m \in T_\Delta(X)$, where l is normalized and $m = \mathrm{rk}(p)$, then the tree $p(u_1, \ldots, u_m) \in P(T_\Delta(X))$ encodes the tree $l\theta \in T_\Sigma(Q(T_\Delta(X)))$ with $\theta(x_i) = u_i$ for $i \in [m]$. Note that it is essential here that states can have an arbitrary number of arguments.

Let $l \to r \in R$ be an input-consuming rule such that $l \notin \Sigma(P(X))$. Suppose that $l = \sigma(l_1, \ldots, l_k)$ for some $\sigma \in \Sigma^{(k)}$ and $l_1, \ldots, l_k \in T_\Sigma(Q(X))$. Moreover, let $\theta_1, \ldots, \theta_k : X \to X$ be bijections such that $l_i\theta_i$ is normalized. Finally, for every $i \in [k]$ let $p_i = f(l_i\theta_i)$ and $r_i = p_i(x_1, \ldots, x_m)$ where $m = \mathrm{rk}(p_i)$. We construct the xmbutt

$$M_1 = (Q \cup \{p_1, \ldots, p_k\}, \Sigma, \Delta, F, (R \setminus \{l \to r\}) \cup R_{1,1} \cup R_{1,2})$$

where $R_{1,1} = \{l_i\theta_i \to r_i \mid i \in [k],\ l_i \notin Q(X)\}$ and $R_{1,2} = \{l' \to r\}$, in which l' is the unique normalized tree of $\{\sigma\}(P(X))$ such that $l'(i) = p_i$ for every $i \in [k]$ (note that if $l_i \in Q(X)$, then $p_i = l_i(\varepsilon)$ and so $l'|_i = l|_i$). It is easy to see that the tree r_i encodes $l_i\theta_i$, in the sense discussed above. Also, since $\mathrm{yield}_X(l'|_i) = \mathrm{yield}_X(l_i)$, the tree $l'|_i$ encodes $l_i\theta_i\theta_i^{-1} = l_i$. Furthermore, note that $R_{1,1} \cup R_{1,2}$ does not contain epsilon rules.

It is straightforward to prove that $\tau_{M_1} = \tau_M$, and moreover, M_1 is linear (respectively, nondeleting) whenever M is so. Note, however, that M_1 need not be deterministic. Clearly, all rules of $R_{1,1}$ have fewer input symbols in the left-hand side than $l \to r$,

and the left-hand side of the rule in $R_{1,2}$ is in $\Sigma(P(X))$. Hence, repeated application of the above construction (keeping P and the mapping f fixed) eventually yields an equivalent STA M'. Moreover, due to the special choice of P and f, and the fact that all new rules are input-consuming, the STA M' is deterministic if M is. □

Theorem 15 *For every xmbutt M there exists an equivalent xmbutt N in one-symbol normal form. Moreover, if M is linear (respectively, nondeleting, deterministic), then so is N.*

Proof By Lemma 14 we can assume that M is an STA. Next, we remove all epsilon rules $l \to r \in R$ such that $r \in Q(X)$. This can be achieved with the standard construction that preserves determinism, nondeletion, and linearity (see the proof of Proposition 10).

Finally, we decompose the right-hand sides. Let $M' = (Q, \Sigma, \Delta, F, R')$ be the xmbutt obtained so far and $l \to r \in R'$ a rule that is not a one-symbol rule. Let $r = q(u_1, \ldots, u_{i-1}, \delta(u'_1, \ldots, u'_k), u_{i+1}, \ldots, u_n)$ for some $q \in Q^{(n)}$, $i \in [n]$, $\delta \in \Delta^{(k)}$, and $u_1, \ldots, u_{i-1}, u_{i+1}, \ldots, u_n, u'_1, \ldots, u'_k \in T_\Delta(X)$. Also, let $p \notin Q$ be a new state of rank $k + n - 1$. We construct the xmbutt

$$M'_1 = (Q \cup \{p\}, \Sigma, \Delta, F, R'_1)$$

with $R'_1 = (R' \setminus \{l \to r\}) \cup \overline{R}_1$ where $\overline{R}_1$ contains the two rules:

- $l \to p(u_1, \ldots, u_{i-1}, u'_1, \ldots, u'_k, u_{i+1}, \ldots, u_n)$ and
- $p(x_1, \ldots, x_{k+n-1}) \to q(x_1, \ldots, x_{i-1}, \delta(x_i, \ldots, x_{i+k-1}), x_{i+k}, \ldots, x_{k+n-1})$.

Note that M'_1 is linear (respectively, nondeleting, deterministic) whenever M' is so. The proof of $\tau_{M'_1} = \tau_{M'}$ is straightforward and omitted. The second rule of $\overline{R}_1$ is a one-symbol rule and the first rule has one output symbol less in its right-hand side than $l \to r$. Thus, repeated application of the above procedure eventually yields an equivalent xmbutt N in one-symbol normal form. □

Let us illustrate Lemma 14 and Theorem 15 on a small example.

Example 16 Consider the xmbutt $M = (Q, \Sigma, \Gamma, F, R)$ such that $Q = F = \{q^{(1)}\}$, $\Sigma = \{\sigma^{(1)}, \alpha^{(0)}\}$, $\Gamma = \{\gamma^{(2)}, \alpha^{(0)}\}$, and

$$R = \{\sigma(\alpha) \to q(\alpha),\ \sigma(q(x_1)) \to q(\gamma(x_1, \alpha))\}\ .$$

Clearly, M is linear, nondeleting, and deterministic. Applying first the procedure of Lemma 14 and then the procedures of Theorem 15 stepwise we obtain the states and rules (we only show the newly constructed states and rules, and the rules that are not one-symbol rules):

- $\{q^{(1)}, q_1^{(0)}\}$ and $\{\alpha \to q_1,\ \sigma(q_1) \to q(\alpha),\ \sigma(q(x_1)) \to q(\gamma(x_1, \alpha))\}$
- $\{\ldots, q_2^{(0)}\}$ and $\{\ldots, \sigma(q_1) \to q_2,\ q_2 \to q(\alpha),\ \sigma(q(x_1)) \to q(\gamma(x_1, \alpha))\}$
- $\{\ldots, q_3^{(2)}\}$ and $\{\ldots, \sigma(q(x_1)) \to q_3(x_1, \alpha),\ q_3(x_1, x_2) \to q(\gamma(x_1, x_2))\}$
- $\{\ldots, q_4^{(1)}\}$ and $\{\ldots, \sigma(q(x_1)) \to q_4(x_1),\ q_4(x_1) \to q_3(x_1, \alpha), \ldots\}$

Note that only the first line corresponds to the construction in the proof of Lemma 14; also note that, for example, $f(\alpha) = q_1$ and $f(q(x_1)) = q$ (see the proof of Lemma 14 for the use of f). Overall, we obtain the xmbutt $N = (P, \Sigma, \Gamma, F, R')$ with

$$P = \{q^{(1)}, q_1^{(0)}, q_2^{(0)}, q_3^{(2)}, q_4^{(1)}\}$$

and R' contains the rules

$$\begin{array}{ccc} \alpha \to q_1 & q_2 \to q(\alpha) & q_4(x_1) \to q_3(x_1, \alpha) \\ \sigma(q_1) \to q_2 & \sigma(q(x_1)) \to q_4(x_1) & q_3(x_1, x_2) \to q(\gamma(x_1, x_2)) \ . \end{array}$$ □

The next corollary is a simple consequence of Proposition 10 and Lemma 14.

Corollary 17 *Let M be a deterministic xmbutt. Then there exists an equivalent deterministic mbutt N. Moreover, if M is linear (respectively, nondeleting), then so is N. In particular,*

$$\begin{aligned} \text{d-XMBOT} &= \text{d-MBOT} \\ \text{ld-XMBOT} &= \text{ld-MBOT} \\ \text{lnd-XMBOT} &= \text{lnd-MBOT} \ . \end{aligned}$$

This allows us to characterize the classes d-XMBOT and ld-XMBOT in terms of top-down tree transducers using the result of [19]. The equality d-MBOT = d-TOP$^{\mathrm{R}}$ is shown in [19]. However, there is a subtle difference between the mbutt of [19] and ours: the mbutt of [19] uses a special root symbol, which allows it to recognize the root of the input tree. It is an easy exercise to show that the two models are equivalent, and so we can conclude that d-XMBOT = d-TOP$^{\mathrm{R}}$ with the help of Corollary 17. However, this equivalence does not preserve linearity; in that case the presence of the special root symbol adds expressive power (as we will discuss further after the next theorem). Also, there is no characterization of ld-MBOT in [19], but one is suggested in the Conclusion of [20]. Thus, to characterize ld-XMBOT (as d-TOP$^{\mathrm{R}}_{\mathrm{su}}$) we have to reconsider the proofs of [19, Lemmata 4.1 and 4.2]. It turns out that the direction d-TOP$^{\mathrm{R}}_{\mathrm{su}} \subseteq$ ld-MBOT is a straightforward consequence of the proof of [19, Lemma 4.2], whereas for the other direction a slight variation of the proof of [19, Lemma 4.1] is needed. Hence, for completeness sake, we repeat the proofs in terms of the present formalism (which also differs from the one of [19] in other, minor details).

Theorem 18

$$\begin{array}{cc} \text{d-XMBOT} = \text{d-TOP}^{\mathrm{R}} & \text{h-MBOT} = \text{td-TOP} \\ \text{ld-XMBOT} = \text{d-TOP}^{\mathrm{R}}_{\mathrm{su}} & \text{lh-MBOT} = \text{td-TOP}_{\mathrm{su}} \ . \end{array}$$

Proof Let us start with the inclusion d-XMBOT $\subseteq$ d-TOP$^{\mathrm{R}}$. Let $M = (Q, \Sigma, \Delta, F, R)$ be a total deterministic mbutt (see Corollary 17 and Lemma 12), and let $f : Q \to \{0, 1\}$ be the characteristic function of F. We define an equivalent top-down tree transducer $\langle M', c\rangle$ with regular look-ahead, with $M' = (Q', \Sigma, \Delta, I, R')$. Intuitively, for every input subtree t, $\langle M', c\rangle$ uses its look-ahead to determine the last rule that M applied in its (unique) derivation $t \Rightarrow^*_M q(u_1, \ldots, u_m)$ with $q \in Q$. Then, for each $n \in [m]$, it uses a state n to compute u_n. To simulate the final states of M, state 1 (which computes u_1) is split into two states: one to be used as initial state (when q is in F) and one to be

used otherwise. In fact, for technical convenience, *every* state n is split into two states, by storing $f(q)$.

Thus we define $Q' = \{0,1\} \times [\mathrm{mx}]$ where $\mathrm{mx} = \max\{\mathrm{rk}(q) \mid q \in Q\}$. Moreover, $I = \{\langle 1,1\rangle\}$ and R' is defined as follows. Let $\rho\colon l \to r$ be a rule in R, with $l(\varepsilon) = \sigma \in \Sigma^{(k)}$, $l(i) = q_i$ for every $i \in [k]$, and $r(\varepsilon) = q \in Q^{(m)}$. Then, for every $n \in [m]$, the rule $\rho(n)\colon \langle f(q), n\rangle(\sigma(x_1, \ldots, x_k)) \to r|_n\theta$ is in R' where the substitution $\theta\colon \mathrm{var}(l) \to Q'(X)$ is defined for every $i \in [k]$ and $j \in [\mathrm{rk}(q_i)]$ by $\theta(l(ij)) = \langle f(q_i), j\rangle(x_i)$. The regular look-ahead $c(\rho(n))$ consists of all trees $t \in T_\Sigma$ such that (i) $t(\varepsilon) = \sigma$ and (ii) for every $i \in [k]$ there exists $\xi_i \in Q(T_\Delta)$ with $t|_i \Rightarrow_M^* \xi_i$ and $\xi_i(\varepsilon) = q_i$. Note that $c(\rho(n))$ is a recognizable tree language because the state behavior of an mbutt is the same as that of a finite-state tree automaton. Note also that, as observed after Definition 2, there is exactly one such ξ_i for every $t|_i$.

Clearly $\langle M', c\rangle$ is deterministic: for every $l' = \langle b, n\rangle(\sigma(x_1, \ldots, x_k))$ and $t \in T_\Sigma$, if $\rho(n)\colon l' \to r'$ with $\sigma = t(\varepsilon)$ and $t \in c(l' \to r')$, then the look-ahead $c(\rho(n))$ determines the states q_i in the left-hand side of ρ, and hence ρ itself (by the determinism of M) and thus $\rho(n)$. Moreover, if M is linear, then for every j and x_i there is at most one occurrence of $\langle f(q_i), j\rangle(x_i)$ in $r\theta$; hence, $\langle M', c\rangle$ is single-use. If M is homomorphic, then the rule that is applied at the root of an input tree is uniquely determined by the symbol at the root, so the look-ahead is not required in this case.

It is straightforward to show by structural induction on t that for every $\langle b, n\rangle \in Q'$ and $(t, u) \in T_\Sigma \times T_\Delta$ the following statement holds: $\langle b, n\rangle(t) \Rightarrow^*_{\langle M', c\rangle} u$ if and only if there exists $\xi \in Q(T_\Delta)$ such that (i) $t \Rightarrow_M^* \xi$, (ii) $b = f(\xi(\varepsilon))$, (iii) $n \in [\mathrm{rk}(\xi(\varepsilon))]$, and (iv) $u = \xi|_n$. Taking $b = n = 1$ this shows that $\tau_{\langle M', c\rangle} = \tau_M$.

Now, let us move on to the inclusion d-TOP$^{\mathrm{R}} \subseteq$ d-XMBOT. Let $\langle M, c\rangle$ be a deterministic top-down tree transducer with regular look-ahead, with $M = (Q, \Sigma, \Delta, I, R)$. We assume that $Q = [s]$ for some $s \geq 1$, and that $I = \{1\}$. Note that for every $i \in Q$ and $t \in T_\Sigma$ there is at most one $u \in T_\Delta$ such that $i(t) \Rightarrow_M^* u$.

According to a standard way of handling look-ahead, the finitely many recognizable tree languages $c(\rho)$, where $\rho \in R$, can be recognized simultaneously by one (total deterministic bottom-up) finite-state tree automaton. Here, we conveniently model this automaton as a pair (N, γ) where (i) $N = (P, \Sigma, \emptyset, \emptyset, R_N)$ is a total deterministic mbutt such that all states in P have rank 0 and (ii) γ is a mapping $\gamma\colon R \to \mathcal{P}(P)$ such that, for every $t \in T_\Sigma$, $t \in c(\rho)$ if and only if $t \Rightarrow_N^* p$ for some $p \in \gamma(\rho)$. Note that for every $t \in T_\Sigma$ there is a unique $p \in P$ such that $t \Rightarrow_N^* p$. Moreover, we can assume that for every $p \in P$ there exists $t \in T_\Sigma$ such that $t \Rightarrow_N^* p$.

Now we construct a total deterministic mbutt $M' = (Q', \Sigma, \Delta, F, R')$ that is equivalent with M. The set of states of M' is $Q' = \{0,1\}^s \times P$, and each state $\langle b, p\rangle$ has rank s. For a bit sequence $b \in \{0,1\}^s$ and $i \in [s]$, we denote the ith bit in b by $b[i]$. Let *nil* be an arbitrary, fixed element of $\Delta^{(0)}$. The rules in R' will be constructed in such a way that for every $t \in T_\Sigma$ there exist (unique) $\langle b, p\rangle \in Q'$ and $u_1, \ldots, u_s \in T_\Delta$ such that (1) $t \Rightarrow_{M'}^* \langle b, p\rangle(u_1, \ldots, u_s)$, (2) $t \Rightarrow_N^* p$, and (3) for every $i \in [s]$, either $b[i] = 1$ and $i(t) \Rightarrow_M^* u_i$, or $b[i] = 0$, $u_i = nil$, and there is no $u \in T_\Delta$ such that $i(t) \Rightarrow_M^* u$. Thus, taking the set F of final states to consist of all $\langle b, p\rangle$ such that $b[1] = 1$, we obtain that $\tau_{M'} = \tau_M$.

For every $\sigma \in \Sigma^{(k)}$ and $\langle b_1, p_1\rangle, \ldots, \langle b_k, p_k\rangle \in Q'$, R' contains a unique rule $l \to r$ with $l(\varepsilon) = \sigma$ and $l(j) = \langle b_j, p_j\rangle$ for every $j \in [k]$. The right-hand side r is defined to be $\langle b, p\rangle(u_1, \ldots, u_s)$ where (1) the rule $\sigma(p_1, \ldots, p_k) \to p$ is in R_N, and (2) for every $i \in [s]$,

- *if* there is a rule $\rho\colon i(\sigma(x_1,\ldots,x_k)) \to r_i$ in R such that (a) $p \in \gamma(\rho)$ and (b) for every $j \in [s]$ and $m \in [k]$, if $j(x_m)$ occurs in r_i then $b_m[j] = 1$,
- *then* $b[i] = 1$ and u_i is obtained from r_i by replacing all occurrences of $j(x_m)$ by the variable $l(mj)$,
- *and otherwise* $b[i] = 0$ and $u_i = nil$.

Note that there is at most one rule ρ that satifies the requirements, due to the determinism of M (consider $t = \sigma(t_1,\ldots,t_k)$ with $t_m \Rightarrow_N^* p_m$ for every $m \in [k]$).

If $\langle M, c\rangle$ is single-use, then for every j and x_m there is at most one occurrence of $j(x_m)$ in $r_1,\ldots,r_s$, and hence the variable $l(mj)$ occurs at most once in $u_1,\ldots,u_s$. Thus, M' is linear. Moreover, if M is a total deterministic top-down tree transducer, then we can safely omit the look-ahead component P of the states (i.e., it is sufficient to use $\{0,1\}^s$ as set of states of M'). Moreover, for every $i \in [s]$ and $t \in T_\Sigma$ there exists an output tree $u \in T_\Delta$ such that $i(t) \Rightarrow_M^* u$ by the totality of M. A simple analysis then shows that any state $b \in Q'$ with a '0' in its bit sequence cannot be reached [i.e., there does not exist $t \in T_\Sigma$ and $u_1,\ldots,u_s \in T_\Delta$ such that $t \Rightarrow_{M'}^* b(u_1,\ldots,u_s)$]. Consequently, we can drop all states with a '0' bit in them and obtain an equivalent homomorphic mbutt. □

We note that ld-XMBOT is strictly included in the class ld-MBOT$_{\mathrm{fkv}}$ of the linear deterministic multi bottom-up tree transformations discussed in [20]. The inclusion is an easy exercise: we add the special root symbol of [19, 20] and add rules that, while consuming that symbol, project on the first argument of a final state. The inclusion is strict, due to the use of the special root symbol. For instance, it is easy to see that the transformation $\{(t, \delta(t,t)) \mid t \in T_\Gamma\}$, with $\delta \in \Gamma^{(2)}$, is in ld-MBOT$_{\mathrm{fkv}}$ (cf. Example 3) but not in ld-MBOT (though it *is* in both d-MBOT and l-MBOT). The results shown in [20] for ld-MBOT$_{\mathrm{fkv}}$ also hold for ld-XMBOT; in particular, ld-XMBOT is incomparable with ld-TOP$^{\mathrm{R}}$, which is the class of transformations computed by linear deterministic top-down tree transducers with regular look-ahead (where 'linear' is defined as usual).

For readers familiar with attribute grammars we observe that Theorem 18 is not surprising. Obviously, in a deterministic mbutt, the arguments of each state can be viewed as synthesized attributes (of type 'output tree'), and the state itself can also be viewed as a synthesized attribute (of finite type). One rule of the mbutt combines all the semantic rules for the attributes, for a given input symbol. It is well known that attribute grammars with synthesized attributes only (of type 'output tree') correspond to deterministic top-down tree transducers, with the states of the top-down tree transducer corresponding to the attributes (see, e.g., [9, 18]). Moreover, a synthesized attribute of finite type obviously corresponds to regular look-ahead (cf. the 'flags' in [13]). The single-use property of top-down tree transducers is defined in such a way that it corresponds to the well-known single-use property of attribute grammars: each attribute is used at most once. For an mbutt this means that each argument of a state is used at most once, i.e., the mbutt is linear. Thus, the only difficulties in the proof of Theorem 18 (and the proof in [19]) are caused by the details of the formal models, which do not precisely correspond to attribute grammars. We elected to avoid the special root symbol of [19] in order to obtain the strong characterizations of Theorem 18. We note that the analogy between mbutts and attribute grammars breaks down in the nondeterministic case (at least for the nondeterministic attributed tree transducers of [18]), and certainly in the extended case. Let us finally note that the characterization of ld-XMBOT as d-TOP$^{\mathrm{R}}_{\mathrm{su}}$ shows that the transformations in ld-XMBOT are MSO trans-

ductions, i.e., can be defined in terms of monadic second-order logic (as shown in [8]). The same can be shown for l-MBOT, because every linear mbutt can be simulated by the composition of an MSO relabelling and a deterministic linear mbutt. However, it does neither hold for d-XMBOT nor for l-XMBOT, because if τ is an MSO transduction then there exists $c \in \mathbb{N}$ such that $\mathrm{size}(u) \leq c \cdot \mathrm{size}(t)$ for every $(t, u) \in \tau$ (and hence τ is finitary); cf. Lemma 8 and the proof of Proposition 11.

5 Composition

Let us now investigate compositions of tree transformations computed by xmbutts. We first recall the classical composition results for bottom-up tree transducers [6, 11]. Let M and N be bottom-up tree transducers. If M is linear or N deterministic, then the composition of the transformations computed by M and N can be computed by a bottom-up tree transducer. As a special case, the classes of transformations computed by linear, linear and nondeleting, and deterministic bottom-up tree transducers are closed under composition.

In our setting, let M and N be xmbutts. We will prove that if M is linear or N is deterministic, then there is an xmbutt $M\,;N$ that computes $\tau_M\,;\tau_N$ (the composition of τ_M and τ_N). In particular, we prove that l-XMBOT, d-XMBOT, and ld-XMBOT are closed under composition. The closure of l-XMBOT was first presented in [32, Propositions II.B.5 and II.B.7]. The closure of d-XMBOT is already known: Theorem 18 shows that d-XMBOT coincides with d-$\mathrm{TOP}^{\mathrm{R}}$, which is closed under composition [12, Theorem 2.11]. In [33, Proposition 2.5] closure under composition of STA transformations was shown for a different type of determinism, in which final states are allowed to have epsilon rules, but must have rank 1; such transformations need not be finitary. Finally, the closure of ld-XMBOT is to be expected from its characterization in Theorem 18 and the fact that the single-use restriction was introduced in [21, 24] to guarantee the closure under composition of attribute grammar transformations (see [31, Theorem 3]).

The standard way to construct the rules of a transducer $M\,;N$, which computes the composition of the transformations computed by two given transducers M and N, is to "run" N on the right-hand sides of the rules of M (see, e.g., [6, 34]). The disadvantage is that, in general, possibly large right-hand sides have to be processed, which complicates the intuition and the correctness proof. In our case this construction is even impossible because it leads to infinitely many rules if τ_N is not finitary. Fortunately, the one-symbol normal form allows a very easy alternative: the xmbutt $M\,;N$ consumes input by simulating M, it produces output by simulating N, and when M produces an output symbol, then it feeds that symbol immediately as input into N.

Let us now prepare the composition construction. In the following, let

$$M = (Q, \Sigma, \Gamma, F_M, R_M) \quad \text{and} \quad N = (P, \Gamma, \Delta, F_N, R_N)$$

be xmbutts such that Q, P, and $\Sigma \cup \Gamma \cup \Delta$ are pairwise disjoint (which can always be assumed without loss of generality). We define the uniquely-ranked alphabet

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

such that $\mathrm{rk}(q\langle p_1, \ldots, p_n\rangle) = \sum_{i=1}^{n} \mathrm{rk}(p_i)$ for every $q \in Q^{(n)}$ and $p_1, \ldots, p_n \in P$. Let $\Pi = \Sigma \cup \Gamma \cup \Delta \cup X$. We define the mapping $\varphi\colon T_{\Pi \cup Q\langle P\rangle} \to T_{\Pi \cup Q \cup P}$ such that for

Fig. 4 Tree homomorphism φ where $q \in Q^{(2)}$, $p_1 \in P^{(1)}$, and $p_2 \in P^{(2)}$.

every $q\langle p_1, \ldots, p_n\rangle \in Q\langle P\rangle^{(k)}$, $\pi \in \Pi^{(k)}$, and $t_1, \ldots, t_k \in T_{\Pi \cup Q\langle P\rangle}$

$$\varphi(q\langle p_1, \ldots, p_n\rangle(t_1, \ldots, t_k)) = q(p_1(\varphi(t_1), \ldots, \varphi(t_l)), \ldots, p_n(\varphi(t_m), \ldots, \varphi(t_k)))$$
$$\varphi(\pi(t_1, \ldots, t_k)) = \pi(\varphi(t_1), \ldots, \varphi(t_k))$$

where, for $j \in [n]$, the jth argument of the right-hand side of the first equation is $p_j(\varphi(t_r), \ldots, \varphi(t_s))$ with $r = 1 + \sum_{i=1}^{j-1} \mathrm{rk}(p_i)$ and $s = \sum_{i=1}^{j} \mathrm{rk}(p_i)$. In particular, $l = \mathrm{rk}(p_1)$ and $m = 1+k-\mathrm{rk}(p_n)$. Thus, we group the subtrees below the corresponding state p_j. Note that φ is a linear and nondeleting tree homomorphism, which acts as a bijection from $T_\Sigma(Q\langle P\rangle(T_\Delta(X)))$ to $T_\Sigma(Q(P(T_\Delta(X))))$. In the sequel, we identify t with $\varphi(t)$ for all $t \in T_\Sigma(Q\langle P\rangle(T_\Delta(X)))$. We display φ in Figure 4. Now let us present the composition construction.

Definition 19 Let $M = (Q, \Sigma, \Gamma, F_M, R_M)$ be an xmbutt in one-symbol normal form and $N = (P, \Gamma, \Delta, F_N, R_N)$ an STA. The *composition* of M and N is the STA

$$M\,;N = (Q\langle P\rangle, \Sigma, \Delta, F, R_1 \cup R_2 \cup R_3)$$

with $F = \{q\langle p_1, \ldots, p_n\rangle \mid n \geq 1, q \in F_M^{(n)}, p_1 \in F_N, p_2, \ldots, p_n \in P\}$ and:

$$R_1 = \{l \to r \mid l \in \mathrm{LHS}(\Sigma) \text{ and } \exists \rho \in R_M^\Sigma : l \Rightarrow_M^\rho r\}$$
$$R_2 = \{l \to r \mid l \in \mathrm{LHS}(\varepsilon) \text{ and } \exists \rho \in R_N^\varepsilon : l \Rightarrow_N^\rho r\}$$
$$R_3 = \{l \to r \mid l \in \mathrm{LHS}(\varepsilon) \text{ and } \exists \rho_1 \in R_M^\varepsilon, \rho_2 \in R_N^\Gamma : l\,(\Rightarrow_M^{\rho_1} ; \Rightarrow_N^{\rho_2})\, r\}$$

where $\mathrm{LHS}(\Sigma)$ and $\mathrm{LHS}(\varepsilon)$ are the sets of normalized (defined in Section 2) trees of $\Sigma(Q\langle P\rangle(X))$ and $Q\langle P\rangle(X)$, respectively.

To illustrate the implicit use of φ, let us show the "official" definition of R_1:

$$R_1 = \{l \to r \mid l \in \mathrm{LHS}(\Sigma), r \in Q\langle P\rangle(T_\Delta(X)), \text{ and } \exists \rho \in R_M^\Sigma : \varphi(l) \Rightarrow_M^\rho \varphi(r)\}\ .$$

Note that the construction preserves linearity. Moreover, it preserves determinism if N is an mbutt (in which case R_2 is empty, of course).

Since we identify t with $\varphi(t)$, the trees that occur in the derivations of $M\,;N$ are in $T_\Sigma(Q(P(T_\Delta)))$. The rules in R_1 are defined in such a way that $M\,;N$ can simulate the input-consuming rules of M, which do not produce output. Those in R_2 allow $M\,;N$ to simulate the epsilon rules of N. Finally, the rules in R_3 allow $M\,;N$ to simulate consecutively an epsilon rule ρ_1 of M and an input-consuming rule ρ_2 of N. Note that ρ_1 produces exactly one output symbol $\gamma \in \Gamma$, and that ρ_2 immediately consumes that same γ as input. Thus, the application of the rule ρ_1 to a given tree in $T_\Sigma(Q(P(T_\Delta)))$ leads to a tree $\xi \in T_\Sigma(Q(T_\Gamma(P(T_\Delta))))$ with $\mathrm{card}(\mathrm{pos}_\Gamma(\xi)) = 1$ (assuming, for simplicity, that Γ is disjoint with $\Sigma \cup \Delta$). The application of ρ_2 to ξ leads back to $T_\Sigma(Q(P(T_\Delta)))$. This means that derivations of $M\,;N$ can be turned into derivations that use the rules of both M and N.

The remaining part of this section will investigate when $\tau_{M;N} = \tau_M\,;\tau_N$, but first let us illustrate the construction on our small running example.

Example 20 Let M be the xmbutt N of Example 16 in one-symbol normal form, and let $N = (\{g^{(1)}, h^{(1)}\}, \Gamma, \Delta, \{g\}, R_N)$ be the linear and nondeleting STA with $\Delta = \Gamma \cup \{\delta^{(1)}\}$ and

$$R_N = \{\alpha \to h(\alpha),\ h(x_1) \to h(\delta(x_1)),\ \gamma(h(x_1), h(x_2)) \to g(\gamma(x_1, x_2))\}\ .$$

Clearly, N computes $\{(\gamma(\alpha,\alpha), \gamma(\delta^i(\alpha), \delta^j(\alpha))) \mid i, j \in \mathbb{N}\}$, and hence $\tau_M\,;\tau_N$ is $\{(\sigma(\sigma(\alpha)), \gamma(\delta^i(\alpha), \delta^j(\alpha))) \mid i, j \in \mathbb{N}\}$. Note that τ_N is not finitary. Now let us construct $M\,;N$. As states we obtain

$$\begin{aligned}&\{q\langle g\rangle^{(1)}, q\langle h\rangle^{(1)}, q_1\langle\rangle^{(0)}, q_2\langle\rangle^{(0)},\\ &\ q_3\langle g,g\rangle^{(2)}, q_3\langle g,h\rangle^{(2)}, q_3\langle h,g\rangle^{(2)}, q_3\langle h,h\rangle^{(2)}, q_4\langle g\rangle^{(1)}, q_4\langle h\rangle^{(1)}\}\ ,\end{aligned}$$

of which only $q\langle g\rangle$ is final. We present some relevant rules only [left in official form $l \to r$; right in alternative notation $\varphi(l) \to \varphi(r)$]. The 1st, 2nd, and 5th rules are in R_1 (of Definition 19), the 4th rule is in R_2, and the remaining rules are in R_3.

$$\begin{aligned}
\alpha &\to q_1\langle\rangle & \alpha &\to q_1\\
\sigma(q_1\langle\rangle) &\to q_2\langle\rangle & \sigma(q_1) &\to q_2\\
q_2\langle\rangle &\to q\langle h\rangle(\alpha) & q_2 &\to q(h(\alpha))\\
q\langle h\rangle(x_1) &\to q\langle h\rangle(\delta(x_1)) & q(h(x_1)) &\to q(h(\delta(x_1)))\\
\sigma(q\langle h\rangle(x_1)) &\to q_4\langle h\rangle(x_1) & \sigma(q(h(x_1))) &\to q_4(h(x_1))\\
q_4\langle h\rangle(x_1) &\to q_3\langle h,h\rangle(x_1,\alpha) & q_4(h(x_1)) &\to q_3(h(x_1),h(\alpha))\\
q_3\langle h,h\rangle(x_1,x_2) &\to q\langle g\rangle(\gamma(x_1,x_2)) & q_3(h(x_1),h(x_2)) &\to q(g(\gamma(x_1,x_2)))
\end{aligned}$$ □

Next, we will prove that $\tau_M\,;\tau_N$ is in XMBOT provided that (i) M is linear or (ii) N is deterministic. Without loss of generality, we can assume that M is in one-symbol normal form, by Theorem 15, and that it is nondeleting in case (i), by Proposition 9. We can also assume that N is an STA in case (i), by Lemma 14, and a total deterministic mbutt in case (ii) by Corollary 17 and Lemma 12. Thus, we can meet the requirements of Definition 19, and we will prove that with the above assumptions the composition construction is correct, i.e., that $M;N$ computes $\tau_M;\tau_N$ (cf. [6, Theorem 6]). Henceforth we assume the notation of Definition 19.

We start with a simple lemma. It shows that in a derivation that uses steps of M and N (like the derivations of $M\,;N$) we can always perform all steps of M first and only then perform the derivation steps of N. This already proves one direction needed for the correctness of the composition construction.

Lemma 21 *Let $t \in T_\Sigma$ and $\xi \in Q(P(T_\Delta))$. If $t \Rightarrow^* \xi$ where $\Rightarrow$ is $\Rightarrow_M \cup \Rightarrow_N$, then $t\ (\Rightarrow_M^*\,;\Rightarrow_N^*)\ \xi$. In particular, $\tau_{M;N} \subseteq \tau_M\,;\tau_N$.*

Proof Let $\mathrm{SF} = T_\Sigma(Q(T_\Gamma(P(T_\Delta))))$. It obviously suffices to prove the following statement: For $\xi, \xi' \in \mathrm{SF}$, if $\xi\ (\Rightarrow_N\,;\Rightarrow_M)\ \xi'$, then $\xi\ (\Rightarrow_M\,;\Rightarrow_N^*)\ \xi'$. To this end, let $\rho_2 \in R_N$, $w_2 \in \mathrm{pos}(\xi)$, $\zeta \in \mathrm{SF}$, $\rho_1 \in R_M$, and $w_1 \in \mathrm{pos}(\zeta)$ be such that

$$\xi \Rightarrow_N^{\rho_2, w_2} \zeta \Rightarrow_M^{\rho_1, w_1} \xi'\ .$$

Clearly, w_2 is not a prefix of w_1, and thus, ρ_1 is applicable to ξ at w_1. Hence there exists $\zeta' \in \mathrm{SF}$ such that $\xi \Rightarrow_M^{\rho_1, w_1} \zeta'$. If w_1 is not a prefix of w_2, then $\zeta' \Rightarrow_N^{\rho_2, w_2} \xi'$ because rewriting occurs in incomparable positions. Now suppose that w_1 is a prefix

of w_2. Let ρ_1 be $l \to r$, and let $v \in \mathrm{pos}_X(l)$ and $y \in \mathbb{N}^*$ be such that $w_1 v y = w_2$. Then $\zeta' \, (\Rightarrow_N^{\rho_2, w_1 v_1 y} ; \cdots ; \Rightarrow_N^{\rho_2, w_1 v_m y}) \, \xi'$ where $\mathrm{pos}_{l(v)}(r) = \{v_1, \ldots, v_m\}$.

Now suppose that $(t, u_1) \in \tau_{M;N}$. Then $t \Rightarrow_{M;N}^* q\langle p_1, \ldots, p_n\rangle(u_1, \ldots, u_k)$ for some $k \geq 1$, $q\langle p_1, \ldots, p_n\rangle \in Q\langle P\rangle^{(k)}$, and $u_2, \ldots, u_k \in T_\Delta$ such that $q \in F_M$ and $p_1 \in F_N$. By the first part of the statement, there exists $\zeta \in \mathrm{SF}$ such that $t \Rightarrow_M^* \zeta \Rightarrow_N^* q\langle p_1, \ldots, p_n\rangle(u_1, \ldots, u_k)$. It is easy to see that $\zeta = q(t_1, \ldots, t_n)$ for some $t_1, \ldots, t_n \in T_\Gamma$, which proves $(t, t_1) \in \tau_M$. Moreover, $\zeta \Rightarrow_N^* q\langle p_1, \ldots, p_n\rangle(u_1, \ldots, u_k)$ yields $t_1 \Rightarrow_N^* p_1(u_1, \ldots, u_{\mathrm{rk}(p_1)})$ by Lemma 4. Consequently, $(t_1, u_1) \in \tau_N$, and thus $(t, u_1) \in \tau_M \, ; \tau_N$. □

Next, we prove that $\tau_M \, ; \tau_N \subseteq \tau_{M;N}$ under the above assumptions on M and N. We achieve this by a standard induction over the length of the derivation.

Lemma 22 *Let $t \in T_\Sigma$ and $\xi \in Q(P(T_\Delta))$ be such that $t \, (\Rightarrow_M^* ; \Rightarrow_N^*) \, \xi$. If (i) M is linear and nondeleting, or (ii) N is a total deterministic mbutt, then $t \Rightarrow_{M;N}^* \xi$. In particular, $\tau_M \, ; \tau_N \subseteq \tau_{M;N}$.*

Proof Let $t = \sigma(t_1, \ldots, t_k)$ for some $\sigma \in \Sigma^{(k)}$ and $t_1, \ldots, t_k \in T_\Sigma$. The proof is by induction on the length of the total given derivation in case (i) and by induction on the length of the derivation using M in case (ii). Let $\eta \in Q(T_\Gamma)$ such that $t \Rightarrow_M^* \eta \Rightarrow_N^* \xi$, and let $\eta(\varepsilon) = \xi(\varepsilon) = q \in Q^{(m)}$. It follows from Lemma 4 that $\eta|_i \Rightarrow_N^* \xi|_i \in P(T_\Delta)$ for every $i \in [m]$. First, suppose that there exists $i \in [m]$ such that $\eta|_i \Rightarrow_N^* \chi \Rightarrow_N^\rho \xi|_i$ with $\chi \in P(T_\Delta)$ and $\rho \in R_N^\varepsilon$. In other words, suppose that one of the derivations ends with the application of an epsilon rule ρ and we are thus in case (i). Then, by the induction hypothesis, we obtain that $t \Rightarrow_{M;N}^* q(\xi|_1, \ldots, \xi|_{i-1}, \chi, \xi|_{i+1}, \ldots, \xi|_m)$. By the definition of R_2, we furthermore have $q(\xi|_1, \ldots, \xi|_{i-1}, \chi, \xi|_{i+1}, \ldots, \xi|_m) \Rightarrow_{M;N} \xi$.

Now, suppose that, for every $i \in [m]$, the derivation $\eta|_i \Rightarrow_N^* \xi|_i$ ends with the application of an input-consuming rule. Consider the last step of the derivation $t \Rightarrow_M^* \eta$, and let $l \to r \in R_M$ and $\theta \colon X \to T_\Gamma$ be such that $t \Rightarrow_M^* l\theta \Rightarrow_M r\theta = \eta$. Since M is in one-symbol normal form, we have $\mathrm{card}(\mathrm{pos}_\Gamma(r)) \leq 1$. Next we distinguish two cases for $l \to r$ to define a tree ζ. If $l \to r$ is input-consuming, then let $\zeta = \xi$. Otherwise $l \to r$ is an epsilon rule, and let $\{i\} = \mathrm{pos}_\Gamma(r)$. Then the derivation $\eta|_i \Rightarrow_N^* \xi|_i$ first applies rules at non-root positions and then applies an input-consuming rule at the root that consumes $\eta(i) = r(i)$. Hence, by Lemma 4, there exists ζ such that $\eta \Rightarrow_N^* \zeta \Rightarrow_N \xi$ and $\zeta(i) = \eta(i) \in \Gamma$; i.e., the last derivation step applies an input-consuming rule at position i. Moreover, for every $n \in [m]$ with $n \neq i$ we have $\zeta|_n \in P(T_\Delta)$ and $\eta|_n \Rightarrow_N^* \zeta|_n$. Also, $\zeta|_i = \zeta(i)(\zeta|_{i1}, \ldots, \zeta|_{is})$ for some $s \in \mathbb{N}$ with $\zeta|_{ij} \in P(T_\Delta)$ and $\eta|_{ij} \Rightarrow_N^* \zeta|_{ij}$ for every $j \in [s]$.

With this definition of ζ, it should now be clear that, since either M is linear or N is a deterministic mbutt, there exists a substitution $\theta' \colon X \to P(T_\Delta)$ such that $\zeta = r\theta'$. In the deterministic case this uses the property of deterministic mbutts mentioned between Definition 2 and Example 3. It should also be clear that $\theta(x) \Rightarrow_N^* \theta'(x)$ for every $x \in \mathrm{var}(r)$. Moreover, since M is nondeleting or N is total, we may assume that this even holds for every $x \in \mathrm{var}(l)$ with the help of the property mentioned for total mbutts between Definition 2 and Example 3. Thus, $t \Rightarrow_M^* l\theta \Rightarrow_N^* l\theta'$. By the induction hypothesis (applied to $t_1, \ldots, t_k$ if $l \to r$ is input-consuming, and to t if it is an epsilon rule), we obtain $t \Rightarrow_{M;N}^* l\theta' \Rightarrow_M r\theta' \Rightarrow_N^* \xi$. Note that the induction hypothesis is applicable in case (i) because M is nondeleting and is also applicable in case (ii) because fewer derivation steps of M are applied. If $l \to r$ is an input-consuming rule,

then $r\theta' = \xi$, and thus by the definition of R_1 we conclude $t \Rightarrow^*_{M;N} l\theta' \Rightarrow_{M;N} r\theta' = \xi$. On the other hand, if $l \to r$ is an epsilon rule, then $r\theta' \Rightarrow^\rho_N \xi$ for an input-consuming rule ρ of N, and thus by the definition of R_3 we conclude $t \Rightarrow^*_{M;N} l\theta' \Rightarrow_{M;N} \xi$.

For the second part of the statement, let $(t, u_1) \in \tau_M ; \tau_N$. Then there exists $t_1 \in T_\Gamma$ such that $(t, t_1) \in \tau_M$ and $(t_1, u_1) \in \tau_N$. Consequently,

- there exist $q \in F^{(n)}_M$ and $t_2, \ldots, t_n \in T_\Gamma$ such that $t \Rightarrow^*_M q(t_1, \ldots, t_n)$, and
- there exist $p \in F^{(k)}_N$ and $u_2, \ldots, u_k \in T_\Delta$ such that $t_1 \Rightarrow^*_N p(u_1, \ldots, u_k)$.

Since M is nondeleting (in which case we have $n = 1$) or N is total, there exist $\zeta_2, \ldots, \zeta_n \in P(T_\Delta)$ such that $t_i \Rightarrow^*_N \zeta_i$ for every $2 \leq i \leq n$ by the remarks after Definition 2. With the help of Lemma 4 we have

$$t \Rightarrow^*_M q(t_1, \ldots, t_n) \Rightarrow^*_N q(p(u_1, \ldots, u_k), \zeta_2, \ldots, \zeta_n) \ .$$

Thus, by the first part of the statement we obtain $t \Rightarrow^*_{M;N} q(p(u_1, \ldots, u_k), \zeta_2, \ldots, \zeta_n)$, and thus $(t, u_1) \in \tau_{M;N}$. □

Now we are ready to prove the main theorem of this section.

Theorem 23

$$\text{l-XMBOT} ; \text{XMBOT} \subseteq \text{XMBOT} \quad \textit{and} \quad \text{XMBOT} ; \text{d-XMBOT} \subseteq \text{XMBOT} \ .$$

Moreover, l-XMBOT, d-XMBOT, *and* ld-XMBOT *are closed under composition.*

Proof The inclusions follow directly from Lemmata 21 and 22 using Theorem 15, Proposition 9, Corollary 17, and Lemmata 12 and 14 to establish the preconditions of Definition 19 and Lemma 22. The closure results follow from the fact that the composition construction preserves linearity and determinism (provided that N is a total deterministic mbutt in the latter case). □

Note that with the help of Proposition 10 and Lemma 14, our approach can be used to reprove [34, Theorem 11] (which is obtained from Theorem 23 by removing every X). In addition, we can now use the composition theorem to obtain bimorphism characterizations for XMBOT and l-XMBOT, which sharpens the inclusions of Lemma 6. We also show that the first two components of the bimorphism can be replaced by a linear and nondeleting xbutt. Finally, we note that the statement l-XMBOT = B(lnh-BOT, lh-MBOT) was first presented in [32, Theorem II.B.4].

Theorem 24

$$\begin{aligned} \text{XMBOT} &= \text{B}(\text{lnh-BOT}, \text{h-MBOT}) &&= \text{ln-XBOT} ; \text{h-MBOT} \\ \text{l-XMBOT} &= \text{B}(\text{lnh-BOT}, \text{lh-MBOT}) &&= \text{ln-XBOT} ; \text{lh-MBOT} \end{aligned}$$

Proof The left-to-right chain of inclusions is proved in Lemma 6. The missing inclusions ln-XBOT ; h-MBOT $\subseteq$ XMBOT and ln-XBOT ; lh-MBOT $\subseteq$ l-XMBOT follow from Theorem 23. □

6 Relation to Extended Top-down Tree Transducers

Now, let us focus on the limitation of the power of xmbutts, as compared to xtts. By [34, Theorem 14] every mbutt computes a transformation of ln-TOP ; d-TOP and can thus be simulated by two top-down tree transducers. It is obvious from the proof of [34, Theorem 14] that the second top-down tree transducer can be made total (by adding dummy rules). Moreover, if the mbutt is linear, then the second top-down tree transducer is single-use. Here we easily derive a similar result for xmbutts.

Theorem 25

$$\text{XMBOT} = \text{ln-XTOP} \,;\, \text{td-TOP} \quad \textit{and} \quad \text{l-XMBOT} = \text{ln-XTOP} \,;\, \text{td-TOP}_{\text{su}} \ .$$

Proof By Theorem 24 we have

$$\begin{aligned} \text{XMBOT} &= \text{ln-XBOT} \,;\, \text{h-MBOT} \\ \text{l-XMBOT} &= \text{ln-XBOT} \,;\, \text{lh-MBOT} \ . \end{aligned}$$

Applying the equation ln-XBOT = ln-XTOP of Proposition 5 and the equations h-MBOT = td-TOP and lh-MBOT = $\text{td-TOP}_{\text{su}}$ of Theorem 18, we obtain the statement. □

Theorem 25 limits the power of xmbutts. They have the same power as a composition of two (special) xtts. In particular, the second equation of Theorem 25 shows in a precise way how much stronger l-XMBOT is with respect to ln-XTOP. Since ln-XTOP transformations preserve recognizability (by Proposition 5 and Corollary 7), this characterization implies that

$$\text{l-XMBOT}(\text{REC}) = \text{lh-MBOT}(\text{REC}) = \text{td-TOP}_{\text{su}}(\text{REC}) \ .$$

Obviously, by Theorem 23, this class of tree languages is closed under all tree transformations of l-XMBOT. This is, in some sense, similar to the fact that the class of regular string languages is closed under all finite-state transductions.

The single-use top-down tree transducer has a bounded copying facility: it makes at most $\text{card}(Q)$ copies of each input subtree; thus it is a so-called "finite-copying" top-down tree transducer, see [17, Definition 3.1.9]. Hence, denoting the class of total deterministic finite-copying top-down tree transformations by $\text{td-TOP}_{\text{fc}}$, we have $\text{td-TOP}_{\text{su}}(\text{REC}) \subseteq \text{td-TOP}_{\text{fc}}(\text{REC})$. Actually, by [16, Corollary 7.5], these classes are equal, and so

$$\text{l-XMBOT}(\text{REC}) = \text{l-MBOT}(\text{REC}) = \text{td-TOP}_{\text{fc}}(\text{REC}) \ .$$

By [17, Corollary 3.2.8], $\text{td-TOP}_{\text{fc}}(\text{REC})$ is a proper subclass of td-TOP(REC), which equals XMBOT(REC) by Theorem 25; intuitively, this is because top-down tree transducers can do unbounded copying. Consequently, l-XMBOT and td-TOP are incomparable subclasses of XMBOT; this was already shown in the proof of Proposition 11: the tree homomorphism τ_M in that proof is in td-TOP.

The class of string languages $\text{yield}(\text{td-TOP}_{\text{fc}}(\text{REC}))$, consisting of the yields of tree languages in $\text{td-TOP}_{\text{fc}}(\text{REC})$, has been investigated in [36, 39, 46], motivated by the grammatical generation of discontinuous constituents in natural languages. Several formal models that generate this class are discussed in [36, Section 5] (cf. also the

discussion in [14, Section 6]). One of these models is the multiple context-free grammar of [39]. In fact, the so-called parallel multiple context-free grammar ([39, Section 2.2]) can be viewed as a variation of the mbutt, generating the yields of its output trees; similarly, the multiple context-free grammar is a variation of the linear mbutt. Thus, it is immediate that the class PMCFL of parallel multiple context-free languages equals yield(XMBOT(REC)) and that the class MCFL of multiple context-free languages equals yield(l-XMBOT(REC)). Let us also mention that the relational tree grammars of [37] (which are closely related to the multiple context-free grammar) generate the class td-$\mathrm{TOP}_{\mathrm{fc}}$(REC), as proved in [37, Proposition 4.8].

Finally, let us summarize the relations between classes of tree transformations computed by several tree transducer models in an inclusion diagram (see Figure 5). Note that all lines in the diagram are oriented upwards, and we use solid lines to denote strict inclusion and dashed lines to denote inclusion. Moreover, for any class $\mathcal{T}$ of transformations, $\mathcal{T}^2$ and $\mathcal{T}^*$ denote $\mathcal{T}\,;\mathcal{T}$ and the composition closure of $\mathcal{T}$, respectively.

Theorem 26 *Figure 5 is an inclusion diagram.*

Proof The nontrivial inclusions are by Propositions 5 and 9, Lemma 6, Corollary 17, Theorems 23 and 25, and [34, Theorems 14 and 19]. Note that by the last line of Lemma 6, we have l-XBOT $\subseteq$ ln-XBOT ; lh-BOT, which is included in l-XTOP^2 by Proposition 5 and the well-known (and obvious) fact that lh-BOT $\subseteq$ ltd-TOP. To prove strictness and incomparability, we need to prove the following statements:

$$\text{ld-XMBOT} \not\subseteq \text{lnd-XMBOT} \tag{1}$$
$$\text{lnd-XMBOT} \not\subseteq \text{l-XTOP}^* \tag{2}$$
$$\text{d-XMBOT} \not\subseteq \text{l-XMBOT} \tag{3}$$
$$\text{TOP}^2 \not\subseteq \text{XMBOT} \tag{4}$$
$$\text{lne-XTOP} \not\subseteq \text{d-XMBOT} \tag{5}$$
$$\text{ln-XTOP} \not\subseteq \text{TOP}^2 \tag{6}$$
$$\text{lne-XTOP}^2 \not\subseteq \text{l-XBOT} \tag{7}$$
$$\text{le-XTOP} \not\subseteq \text{ln-XTOP}^2 \tag{8}$$
$$\text{l-XBOT} \not\subseteq \text{l-XTOP} \ . \tag{9}$$

We leave the proof of (1) and (8) as an exercise. For statement (2) recall the deterministic mbutt of Example 3, which does not preserve recognizability. However, all transformations of l-XTOP^* preserve recognizability (by Proposition 5 and Corollary 7), which proves the statement. Statement (3) is proved in Proposition 11. By [17, Corollary 3.2.16] we have

$$\text{XMBOT(REC)} = \text{td-TOP(REC)} \subsetneq \text{TOP(REC)} \ ,$$

which yields (4). The statements (5) and (6) are trivial, because the tree transformations in d-XMBOT are functions and those in TOP^2 are finitary. It is shown in [5, Section 3.4] and [35, Theorem 5.2] that lne-$\mathrm{XTOP}^2 \not\subseteq$ B(lnh-BOT, lh-BOT), which proves statement (7) by Lemma 6. Finally, statement (9) is proved in Proposition 5. □

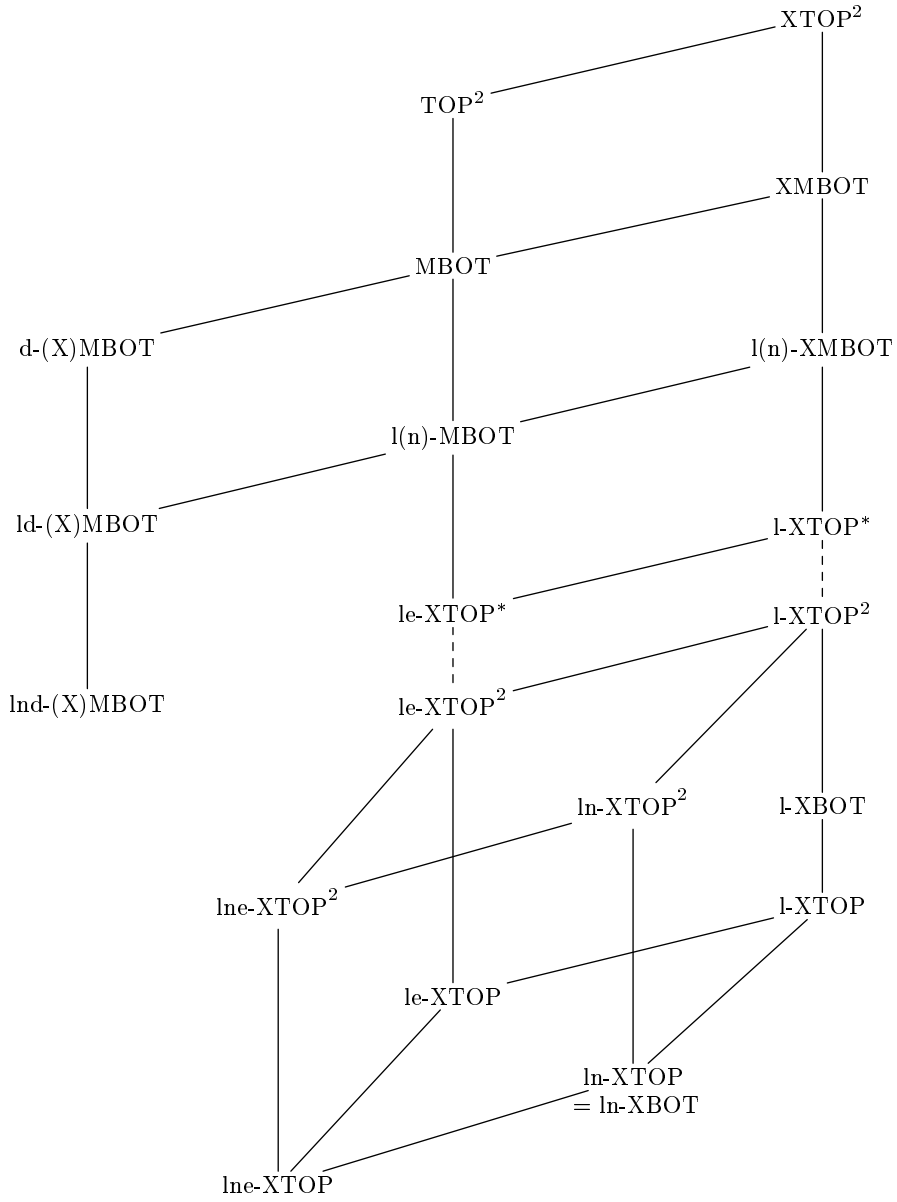

Fig. 5 Inclusion diagram of classes of tree transformations (solid lines denote strict inclusion and dashed lines denote inclusion).

Conclusion and Open Problems

We have shown that linear xmbutts are suitably powerful to compute any transformation that can be computed by linear extended top-down tree transducers (see Proposition 5). Moreover, we generalized the main composition results of [6, 11] for bottom-up tree transducers to xmbutts (see Theorem 23). In particular, we showed that l-XMBOT

is closed under composition. Finally, we characterized l-XMBOT as the composition of ln-XTOP and td-TOP_{su} (see Theorem 25), which shows that, analogously to bottom-up tree transducers, nondeterminism and evaluation can be separated.

As shown in Theorem 26, the composition closure of l-XTOP is strictly contained in l-XMBOT. This raises two interesting questions:

(a) Which linear xmbutts can be transformed into a linear xtt?
(b) Can we characterize the composition closure of l-XTOP?

Acknowledgements The authors are grateful to the reviewer for his insightful remarks, which improved the paper.

References

1. Alfred V. Aho and Jeffrey D. Ullman. Syntax directed translations and the pushdown assembler. *J. Comput. System Sci.*, 3(1):37–56, 1969.
2. Cyril Allauzen, Michael Riley, Johan Schalkwyk, Wojciech Skut, and Mehryar Mohri. OpenFst: A general and efficient weighted finite-state transducer library. In *CIAA*, volume 4783 of LNCS, pages 11–23. Springer, 2007.
3. André Arnold and Max Dauchet. *Transductions inversibles de forêts.* Thèse 3ème cycle M. Dauchet, Université de Lille, 1975.
4. André Arnold and Max Dauchet. Bi-transductions de forêts. In *ICALP*, pages 74–86. Edinburgh University Press, 1976.
5. André Arnold and Max Dauchet. Morphismes et bimorphismes d'arbres. *Theoret. Comput. Sci.*, 20(1):33–93, 1982.
6. Brenda S. Baker. Composition of top-down and bottom-up tree transductions. *Inform. and Control*, 41(2):186–213, 1979.
7. Jean Berstel. *Transductions and context-free languages.* Teubner, Stuttgart, 1979.
8. Roderick Bloem and Joost Engelfriet. A comparison of tree transductions defined by monadic second-order logic and by attribute grammars. *J. Comput. System Sci.*, 61(1):1–50, 2000.
9. Bruno Courcelle and Paul Franchi-Zannettacci. Attribute grammars and recursive program schemes. *Theoret. Comput. Sci.*, 17:163–191, 235–257, 1982.
10. Steve DeNeefe, Kevin Knight, Wei Wang, and Daniel Marcu. What can syntax-based MT learn from phrase-based MT? In *EMNLP & CoNLL*, pages 755–763. The Association for Computational Linguistics, 2007.
11. Joost Engelfriet. Bottom-up and top-down tree transformations: A comparison. *Math. Systems Theory*, 9(3):198–231, 1975.
12. Joost Engelfriet. Top-down tree transducers with regular look-ahead. *Math. Systems Theory*, 10(1):289–303, 1977.
13. Joost Engelfriet. Tree transducers and syntax-directed semantics. Technical Report Memorandum 363, Technische Hogeschool Twente, 1981. Also in: Proc. 7th CAAP, pages 82–107, 1982.
14. Joost Engelfriet. Context-free graph grammars. In *Handbook of Formal Languages*, volume 3, chapter 3, pages 125–213. Springer, 1997.
15. Joost Engelfriet, Eric Lilin, and Andreas Maletti. Extended multi bottom-up tree transducers. In *DLT*, volume 5257 of LNCS, pages 289–300. Springer, 2008.

16. Joost Engelfriet and Sebastian Maneth. Macro tree transducers, attribute grammars, and MSO definable tree translations. *Inform. and Comput.*, 154(1):34–91, 1999.
17. Joost Engelfriet, Grzegorz Rozenberg, and Giora Slutzki. Tree transducers, L systems, and two-way machines. *J. Comput. System Sci.*, 20(2):150–202, 1980.
18. Zoltán Fülöp. On attributed tree transducers. *Acta Cybernet.*, 5:261–279, 1981.
19. Zoltán Fülöp, Armin Kühnemann, and Heiko Vogler. A bottom-up characterization of deterministic top-down tree transducers with regular look-ahead. *Inf. Process. Lett.*, 91(2):57–67, 2004.
20. Zoltán Fülöp, Armin Kühnemann, and Heiko Vogler. Linear deterministic multi bottom-up tree transducers. *Theoret. Comput. Sci.*, 347(1–2):276–287, 2005.
21. Harald Ganzinger. Increasing modularity and language-independency in automatically generated compilers. *Sci. Comput. Prog.*, 3(3):223–278, 1983.
22. Ferenc Gécseg and Magnus Steinby. *Tree Automata*. Akadémiai Kiadó, Budapest, 1984.
23. Ferenc Gécseg and Magnus Steinby. Tree languages. In *Handbook of Formal Languages*, volume 3, chapter 1, pages 1–68. Springer, 1997.
24. Robert Giegerich. Composition and evaluation of attribute coupled grammars. *Acta Inform.*, 25(4):355–423, 1988.
25. Jonathan Graehl. CARMEL. ISI/USC, `http://www.isi.edu/licensed-sw/carmel`, 1997.
26. Jonathan Graehl and Kevin Knight. Training tree transducers. In *HLT-NAACL*, pages 105–112. The Association for Computational Linguistics, 2004. See also [27].
27. Jonathan Graehl, Kevin Knight, and Jonathan May. Training tree transducers. *Computational Linguistics*, 34(3):391–427, 2008.
28. John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley, 1979.
29. Kevin Knight and Jonathan Graehl. An overview of probabilistic tree transducers for natural language processing. In *CICLing*, volume 3406 of LNCS, pages 1–24. Springer, 2005.
30. Armin Kühnemann. *Berechnungsstärken von Teilklassen primitiv-rekursiver Programmschemata*. PhD thesis, Technische Universität Dresden, 1997.
31. Armin Kühnemann. Benefits of tree transducers for optimizing functional programs. In *FSTTCS*, volume 1530 of LNCS, pages 146–157. Springer, 1998.
32. Eric Lilin. *Une généralisation des transducteurs d'états finis d'arbres: les S-transducteurs*. Thèse 3ème cycle, Université de Lille, 1978.
33. Eric Lilin. Propriétés de clôture d'une extension de transducteurs d'arbres déterministes. In *CAAP*, volume 112 of LNCS, pages 280–289. Springer, 1981.
34. Andreas Maletti. Compositions of extended top-down tree transducers. *Inform. and Comput.*, 206(9–10):1187–1196, 2008.
35. Andreas Maletti, Jonathan Graehl, Mark Hopkins, and Kevin Knight. The power of extended top-down tree transducers. *SIAM J. Comput.*, 39(2):410–430, 2009.
36. Owen Rambow and Giorgio Satta. Independent parallelism in finite copying parallel rewriting systems. *Theoret. Comput. Sci.*, 223(1–2):87–120, 1999.
37. Jean-Claude Raoult. Rational tree relations. *Bull. Belg. Math. Soc.*, 4:149–176, 1997.
38. William C. Rounds. Mappings and grammars on trees. *Math. Systems Theory*, 4(3):257–287, 1970.

39. Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. On multiple context-free grammars. *Theoret. Comput. Sci.*, 88(2):191–229, 1991.
40. Yves Shabes. *Mathematical and Computational Aspects of Lexicalized Grammars.* PhD thesis, University of Pennsylvania, 1990.
41. Stuart M. Shieber. Unifying synchronous tree adjoining grammars and tree transducers via bimorphisms. In *EACL*, pages 377–384. The Association for Computer Linguistics, 2006.
42. Stuart M. Shieber and Yves Shabes. Synchronous tree-adjoining grammars. In *COLING*, pages 1–6. The Association for Computer Linguistics, 1990.
43. Magnus Steinby and Cătălin Ionuţ Tîrnăucă. Syntax-directed translations and quasi-alphabetic tree bimorphisms. In *CIAA*, volume 4783 of LNCS, pages 265–276. Springer, 2007.
44. James W. Thatcher. Generalized^2 sequential machine maps. *J. Comput. System Sci.*, 4(4):339–367, 1970.
45. James W. Thatcher. Tree automata: An informal survey. In *Currents in the Theory of Computing*, pages 143–172. Prentice Hall, 1973.
46. David J. Weir. Linear context-free rewriting systems and deterministic tree-walking transducers. In *ACL*, pages 136–143. The Association for Computational Linguistics, 1992.
47. Kenji Yamada and Kevin Knight. A decoder for syntax-based statistical MT. In *ACL*, pages 303–310. The Association for Computational Linguistics, 2002.