Composition Closure of Linear Extended Top-down Tree Transducers

Zoltán Fülöp[a,1], Andreas Maletti[b,2,*]

[a] *Department of Foundations of Computer Science, University of Szeged*
Árpád tér 2, H-6720 Szeged, Hungary
[b] *Universität Stuttgart, Institut für Maschinelle Sprachverarbeitung*
Pfaffenwaldring 5b, 70569 Stuttgart, Germany

Abstract

Linear extended top-down tree transducers (or synchronous tree-substitution grammars) are popular formal models of tree transformations. The expressive power of compositions of such transducers with and without regular look-ahead is investigated. In particular, the restrictions of nondeletion, ε-freeness, and strictness are considered. The composition hierarchy turns out to be finite for all ε-free (all rules consume input) variants of these transducers except for nondeleting ε-free linear extended top-down tree transducers. The least number of transducers needed for the full expressive power of arbitrary compositions is presented. In all remaining cases (incl. nondeleting ε-free linear extended top-down tree transducers) the composition hierarchy does not collapse.

Keywords: extended top-down tree transducer; composition hierarchy; bimorphism

1. Introduction

The top-down tree transducer is a simple formal model that encodes a tree transformation (i.e., a relation on trees). It was introduced in [22, 23] and intensively studied thereafter (see [13–15] for an overview). Roughly speaking, a top-down tree transducer processes the input tree symbol-by-symbol and specifies in its rules, how to translate an input symbol into an output tree fragment together with instructions on how to process the subtrees of the input symbol. This asymmetry between input (single symbol) and output (tree fragment) was removed in extended top-down tree transducers (xt), which were introduced and studied in [1, 2]. In an xt the left-hand side of a rule now contains an input tree fragment, in which each variable can occur at most once as a placeholder for a subtree. In particular, the input tree fragment can even be just a variable, which matches every tree, and such rules are called ε-rules. In this contribution we consider linear xt (l-xt), in which the right-hand side of each rule contains each variable at most once as well. Restricted variants of l-xt are used in most approaches to syntax-based machine translation [17, 18].

We also add regular look-ahead [8] (i.e., the ability to check a regular property for the subtrees in an input tree fragment) to l-xt, so our most expressive model is the linear extended top-down tree transducer with regular look-ahead (l-xt$^{\mathrm{R}}$). Contrary to most of the literature [8, 16] we present our model as a synchronized grammar [5] because we sometimes use the auxiliary link structure in our proofs. Instead of variables in the left-hand side and a state-variable combination in the right-hand side of a rule, we immediately only use states with the restriction that each state can occur at most once in the left-hand side and at most once in the right-hand side. Moreover, all states that occur in the right-hand side must also occur in the left-hand side. In this way, for each rule the states establish implicit links (a state links its occurrence in the left-hand side with its occurrence in the right-hand side), which form a bijection between a subset of the

*Corresponding author
[1]Supported by the program TÁMOP-4.2.1/B-09/1/KONV-2010-0005 of the Hungarian National Development Agency.
[2]Supported by the German Research Foundation (DFG) grant MA / 4959 / 1-1.

state occurrences in the left-hand side and all state occurrences in the right-hand side. The state occurrences (in the left-hand side) that do not participate in the bijection (i.e., those states that exclusively occur in the left-hand side) can restrict the acceptable subtrees at their position with the help of regular look-ahead [8]. The implicit links in a rule are made explicit in a derivation, and a rule application expands (explicitly) linked state occurrences at the same time. Example 2 shows an l-xt$^{\mathrm{R}}$, for which we illustrate a few derivation steps in Figure 2. The tree transformation computed by the example l-xt$^{\mathrm{R}}$ is shown in Example 8. In the following, we use l-XT$^{\mathrm{R}}$ and l-XT to denote the class of all tree transformations computed by l-xt$^{\mathrm{R}}$ and l-xt, respectively.

The expressive power of the various subclasses of l-XT$^{\mathrm{R}}$ is already well understood [12, 16]. However, in practice complex systems are often specified with the help of compositions of tree transformations [21] because it is much easier to develop (or train) small components that manage a part of the overall transformation. Consequently, [18] and others declare that closure under composition is a very desirable property for classes of tree transformations (especially in the area of natural language processing). If a class $\mathcal{C}$ of tree transformations is closed under composition, then any composition chain $\tau_1 ; \cdots ; \tau_n$ of tree transformations $\tau_1, \ldots, \tau_n$ of $\mathcal{C}$ can be replaced by a single tree transformation $\tau \in \mathcal{C}$. If $\mathcal{C}$ represents the class of all tree transformations computable by a device, then closure under composition means that we can replace any composition chain specified by several devices by just a single device, which enables an efficient modular development. Unfortunately, neither l-XT$^{\mathrm{R}}$ nor l-XT are closed under composition [2, 3, 16].

In general, for a class $\mathcal{C}$ of tree transformations (that contains the identity transformation) we obtain a composition hierarchy $\mathcal{C} \subseteq \mathcal{C}^2 \subseteq \mathcal{C}^3 \subseteq \cdots$, where $\mathcal{C}^n$ denotes the n-fold composition of $\mathcal{C}$. The class $\mathcal{C}$ might be closed under composition at power n (i.e., $\mathcal{C}^n = \mathcal{C}^{n+1}$) or its composition hierarchy might be infinite (i.e., $\mathcal{C}^n \subsetneq \mathcal{C}^{n+1}$ for all n). In the former case, we say that the composition hierarchy of $\mathcal{C}$ collapses at power n, which also yields that $\mathcal{C}^n = \mathcal{C}^m$ for all $m \geq n$. In particular, $\mathcal{C}$ is closed under composition if its composition hierarchy collapses at power 1. We note that in practice (e.g., in machine translation) the classes that are closed under composition at a small finite power are also important because for such classes we can limit the length of composition chains [21]. In this contribution, we investigate the composition hierarchy of the classes l-XT$^{\mathrm{R}}$ and l-XT together with their subclasses determined by the properties: ε-freeness, strictness, and nondeletion, which are abbreviated by '$\not\varepsilon$', 's', and 'n', respectively. Roughly speaking, ε-freeness yields that all rules are ε-free, strictness guarantees that the right-hand side of each rule contains an output symbol, and nondeletion requires that for each rule exactly the same states occur in the left- and right-hand side. We use the property abbreviations in front of l-XT$^{\mathrm{R}}$ and l-XT to obtain the class of all tree transformations computable by such restricted l-xt$^{\mathrm{R}}$ and l-xt, respectively. For instance, $\not\varepsilon$sl-XT$^{\mathrm{R}}$ denotes the class of all tree transformations computed by ε-free and strict l-xt$^{\mathrm{R}}$.

It is known that none of our considered classes is closed under composition [3, Section 3.4]. In addition, it is known that $\not\varepsilon$snl-XT = $\not\varepsilon$snl-XT$^{\mathrm{R}}$ is closed at power 2 [6, Section II-2-2-3-3]. We complete the picture as follows. For each of the remaining classes, we either provide the least power at which the class is closed under composition or show that the composition hierarchy of the class is infinite (denoted by ∞). Our results (together with the mentioned existing result) are presented in Table 1.

Our contribution is organized as follows. Section 2 recalls the necessary concepts and introduces our notation. We continue in Section 3 with the formal introduction of our main model (l-xt$^{\mathrm{R}}$) including its syntax and semantics and the restrictions that we consider later. In addition, we recall some known equalities between certain fundamental classes of tree transformations in preparation for our first main results. In Section 4 we give a power at which the classes $\not\varepsilon$sl-XT, $\not\varepsilon$sl-XT$^{\mathrm{R}}$, $\not\varepsilon$l-XT, and $\not\varepsilon$l-XT$^{\mathrm{R}}$ of tree transformations are closed under composition (cf. Table 1). This is completed in Section 5, where we conclude that the presented powers (Table 1) are minimal. Finally, in Section 6 we prove that the composition hierarchy of the remaining classes is infinite.

2. Notation

We denote the set of all nonnegative integers by $\mathbb{N}$. The set of all *finite words* (finite sequences) over a set S is $S^* = \bigcup_{n \in \mathbb{N}} S^n$, where $S^0 = \{\varepsilon\}$ contains only the *empty word* ε. The *length* of a word $w \in S^*$ is the

Class	Least power of closedness	Proved in
$\not\in$snl-XT $=$ $\not\in$snl-XT$^{\mathrm{R}}$	2	[6, Section II-2-2-3-3]
$\not\in$sl-XT$^{\mathrm{R}}$, $\not\in$sl-XT	2	Theorem 21
$\not\in$l-XT$^{\mathrm{R}}$	3	Theorem 25
$\not\in$l-XT	4	Corollary 26
otherwise	∞	Theorem 34

Table 1: Characterization of the composition hierarchies.

unique $n \in \mathbb{N}$ such that $w \in S^n$. We write $|w|$ for the length of w. The *concatenation* of two words $v, w \in S^*$ is denoted by $v.w$ or simply vw.

Every subset of $S \times T$ is a *relation* from S to T. Given relations $R_1 \subseteq S \times T$ and $R_2 \subseteq T \times U$, the *inverse* of R_1 is the relation $R_1^{-1} = \{(t,s) \mid (s,t) \in R_1\}$, and the *composition* of R_1 and R_2 is the relation

$$R_1 \,;\, R_2 = \{(s,u) \mid \exists t \in T\colon (s,t) \in R_1, (t,u) \in R_2\} \ .$$

These notions and notations are lifted to classes $\mathcal{C}_1$ and $\mathcal{C}_2$ of relations in the usual manner. Namely, we let $\mathcal{C}_1^{-1} = \{R_1^{-1} \mid R_1 \in \mathcal{C}_1\}$ and

$$\mathcal{C}_1 \,;\, \mathcal{C}_2 = \{R_1 \,;\, R_2 \mid R_1 \in \mathcal{C}_1, R_2 \in \mathcal{C}_2\} \ .$$

Moreover, the *powers* of a class $\mathcal{C}$ are defined by $\mathcal{C}^1 = \mathcal{C}$ and $\mathcal{C}^{n+1} = \mathcal{C}^n \,;\, \mathcal{C}$ for $n \geq 1$. The *composition hierarchy* (resp. *composition closure*) *of* $\mathcal{C}$ is the family $(\mathcal{C}^n \mid n \geq 1)$ (resp. the class $\bigcup_{n \geq 1} \mathcal{C}^n$). If $\mathcal{C}^{n+1} = \mathcal{C}^n$, then $\mathcal{C}$ *is closed under composition at power* n. For $n = 1$ we shorten this to just $\mathcal{C}$ *is closed under composition*. If $\mathcal{C}$ is closed under composition at power n, then $\bigcup_{1 \leq i \leq n} \mathcal{C}^i$ is the composition closure of $\mathcal{C}$. Moreover, we note that if $\mathcal{C}$ contains the identity relations, then $\mathcal{C}^n \subseteq \mathcal{C}^{n+1}$ for all $n \geq 1$, so that in this case $\mathcal{C}^n$ is the composition closure of $\mathcal{C}$ provided that $\mathcal{C}$ is closed under composition at power n. Our classes $\mathcal{C}$ of tree transformations will always contain the identity relations.

An *alphabet* Σ is a nonempty and finite set, of which the elements are called *symbols*. The alphabet Σ is *ranked* if there additionally is a mapping $\mathrm{rk}\colon \Sigma \to \mathbb{N}$ that assigns a rank to each symbol. We let $\Sigma_k = \{\sigma \in \Sigma \mid \mathrm{rk}(\sigma) = k\}$ for every $k \in \mathbb{N}$. Often the mapping 'rk' is obvious from the context, so we typically denote ranked alphabets by Σ alone. If it is not obvious, then we use the notation $\sigma^{(k)}$ to indicate that the symbol σ has rank k. For the rest of this paper, Σ, Δ, and Γ will denote arbitrary ranked alphabets if not specified otherwise.

For every set T, let

$$\Sigma(T) = \{\sigma(t_1, \ldots, t_k) \mid \sigma \in \Sigma_k, t_1, \ldots, t_k \in T\} \ .$$

Let S be a set with $S \cap \Sigma = \emptyset$. The set $T_\Sigma(S)$ of Σ-*trees with leaf labels* S is the smallest set U such that $S \subseteq U$ and $\Sigma(U) \subseteq U$. We write T_Σ for $T_\Sigma(\emptyset)$, and any subset of T_Σ is a *tree language*. The *height* $\mathrm{ht}(t)$ of a tree $t \in T_\Sigma(S)$ is defined such that $\mathrm{ht}(s) = 0$ for all $s \in S$ and

$$\mathrm{ht}(\sigma(t_1, \ldots, t_k)) = 1 + \max\{\mathrm{ht}(t_i) \mid 1 \leq i \leq k\}$$

for all $\sigma \in \Sigma_k$ and $t_1, \ldots, t_k \in T_\Sigma(S)$. Given a unary symbol $\gamma \in \Sigma_1$ and a tree $t \in T_\Sigma(S)$, we write $\gamma^k(t)$ for the tree $\underbrace{\gamma(\cdots\gamma}_{k \text{ times}}(t)\cdots)$.

The set $\mathrm{pos}(t) \subseteq \mathbb{N}^*$ of *positions* of $t \in T_\Sigma(S)$ is inductively defined by $\mathrm{pos}(s) = \{\varepsilon\}$ for every $s \in S$ and

$$\mathrm{pos}(\sigma(t_1, \ldots, t_k)) = \{\varepsilon\} \cup \bigcup_{i=1}^{k} \{iw \mid w \in \mathrm{pos}(t_i)\}$$

for every $\sigma \in \Sigma_k$ and $t_1, \ldots, t_k \in T_\Sigma(S)$. For words $v, w \in \mathbb{N}^*$, we denote the longest common prefix of v and w by $\mathrm{lcp}(v, w)$. Note that $\mathrm{lcp}(v, w) \in \mathrm{pos}(t)$ for all $v, w \in \mathrm{pos}(t)$ because $\mathrm{pos}(t)$ is prefix-closed. The positions $\mathrm{pos}(t)$ are partially ordered by the prefix order $\preceq$ on $\mathbb{N}^*$ [i.e., $v \preceq w$ if and only if $v = \mathrm{lcp}(v, w)$]. The size $|t|$ of the tree $t \in T_\Sigma(S)$ is $|\mathrm{pos}(t)|$; i.e., the number of its positions. Let $t \in T_\Sigma(S)$ and $w \in \mathrm{pos}(t)$. The *label* of t at w is $t(w)$, and the *w-rooted subtree* of t is $t|_w$. Formally, $s(\varepsilon) = s|_\varepsilon = s$ for every $s \in S$ and

$$t(w) = \begin{cases} \sigma & \text{if } w = \varepsilon \\ t_i(v) & \text{if } w = iv \text{ and } i \in \mathbb{N} \end{cases} \qquad \text{and} \qquad t|_w = \begin{cases} t & \text{if } w = \varepsilon \\ t_i|_v & \text{if } w = iv \text{ and } i \in \mathbb{N} \end{cases}$$

where $t = \sigma(t_1, \ldots, t_k)$ with $\sigma \in \Sigma_k$ and $t_1, \ldots, t_k \in T_\Sigma(S)$. For every selection $U \subseteq S$ of leaf symbols, we let $\mathrm{pos}_U(t) = \{w \in \mathrm{pos}(t) \mid t(w) \in U\}$ and $\mathrm{pos}_s(t) = \mathrm{pos}_{\{s\}}(t)$ for every $s \in S$. The tree t is *linear* (resp. *nondeleting*) in U if $|\mathrm{pos}_u(t)| \leq 1$ (resp. $|\mathrm{pos}_u(t)| \geq 1$) for every $u \in U$. Moreover, $\mathrm{var}(t) = \{s \in S \mid \mathrm{pos}_s(t) \neq \emptyset\}$. The expression $t[u]_w$ denotes the tree that is obtained from $t \in T_\Sigma(S)$ by replacing the subtree $t|_w$ at w by $u \in T_\Sigma(S)$.

Let $U \subseteq S$ be finite, $t \in T_\Sigma(S)$, and $\theta \colon U \to \{L \mid L \subseteq T_\Sigma(S)\}$. We define the tree language $t\theta$ by induction as follows.

- $s\theta = s$ for all $s \in S \setminus U$,
- $s\theta = \theta(s)$ for all $s \in U$, and
- for all $\sigma \in \Sigma_k$ and $t_1, \ldots, t_k \in T_\Sigma(S)$

$$\sigma(t_1, \ldots, t_k)\theta = \{\sigma(u_1, \ldots, u_k) \mid u_1 \in t_1\theta, \ldots, u_k \in t_k\theta\} \ .$$

For every $n \in \mathbb{N}$ we fix the set $X_n = \{x_1, \ldots, x_n\}$ of variables, which we assume to be disjoint from all ranked alphabets considered in the paper. Given $t \in T_\Sigma(S)$ and $\tau \colon X_n \to T_\Sigma(S)$, we write $t[\tau(x_1), \ldots, \tau(x_n)]$ for $t\theta$, where $\theta(x_i) = \{\tau(x_i)\}$ for all $1 \leq i \leq n$.

A *tree homomorphism from Σ to Δ* is a family of mappings $(h_k \mid k \in \mathbb{N})$ such that $h_k \colon \Sigma_k \to T_\Delta(X_k)$ for every $k \in \mathbb{N}$. Such a tree homomorphism is

- *linear* if for every $\sigma \in \Sigma_k$ the tree $h_k(\sigma)$ is linear in X_k,
- *complete* if for every $\sigma \in \Sigma_k$ the tree $h_k(\sigma)$ is nondeleting in X_k,
- *strict* if $h_k \colon \Sigma_k \to \Delta(T_\Delta(X_k))$ for every $k \in \mathbb{N}$, and
- *delabeling* if $h_k \colon \Sigma_k \to X_k \cup \Delta(X_k)$ for every $k \in \mathbb{N}$.

We abbreviate the above restrictions by 'l', 'c', 's', and 'd'. The tree homomorphism $(h_k \mid k \in \mathbb{N})$ induces a mapping $h \colon T_\Sigma(S) \to T_\Delta(S)$ defined inductively by $h(s) = s$ for all $s \in S$ and

$$h(\sigma(t_1, \ldots, t_k)) = h_k(\sigma)[h(t_1), \ldots, h(t_k)]$$

for all $\sigma \in \Sigma_k$ and $t_1, \ldots, t_k \in T_\Sigma(S)$. As usual, we also call the induced mapping h a tree homomorphism. We denote by H the class of all tree homomorphisms, and for any combination w of 'l', 'c', 's', and 'd' we denote by w-H the class of all w-tree homomorphisms. For instance, lcs-H is the class of all linear, complete and strict tree homomorphisms.

In the following, we need *regular tree languages* [14, 15] and basic results for tree automata, which recognize the regular tree languages. The set $\mathrm{Reg}(\Gamma)$ contains all regular tree languages $L \subseteq T_\Gamma$ over the ranked alphabet Γ. A particular well-known result, which we use, states that $t\theta \in \mathrm{Reg}(\Gamma)$ for all $t \in T_\Gamma(U)$ and $\theta \colon U \to \mathrm{Reg}(\Gamma)$. A *bimorphism* is a triple $B = (\psi, L, \varphi)$ consisting of a regular tree language $L \in \mathrm{Reg}(\Gamma)$, an input tree homomorphism $\psi \colon T_\Gamma \to T_\Sigma$, and an output tree homomorphism $\varphi \colon T_\Gamma \to T_\Delta$. The tree transformation $B \subseteq T_\Sigma \times T_\Delta$ computed by the bimorphism B is the relation $B = \{\langle \psi(t), \varphi(t) \rangle \mid t \in L\}$. Given two combinations v and w of restrictions for tree homomorphisms, we let $\mathrm{B}(v, w)$ denote the class of all tree transformations computed by bimorphisms $B = (\psi, L, \varphi)$ such that ψ and φ are tree homomorphisms of type v and w, respectively.

3. Linear extended top-down tree transducers

Our main model is the linear extended top-down tree transducer [1, 2, 17, 18] with regular look-ahead (l-xt$^{\mathrm{R}}$), which is based on the classical linear top-down tree transducer without [22, 23] and with regular

look-ahead [8]. We will present it in a form that is closer to synchronized grammars [5] because we will use an auxiliary structure, called the links, in a later proof. In synchronous grammars, equal states in the left and right-hand side of a rule are implicitly linked, and equal states are explicitly linked in a derivation, in which such linked states will be replaced at the same time by a rule for that state. In a rule of an l-xt$^{\mathrm{R}}$, these implicit links form a bijection between a subset of the states in the left-hand side and all states in the right-hand side. Thus, some states might exclusively occur in the left-hand side. For those states we can use the regular look-ahead to restrict the subtrees that are acceptable at these positions.

Definition 1 (see [16, Section 2.2]). A *linear extended top-down tree transducer* with regular look-ahead (l-xt$^{\mathrm{R}}$) is a tuple $M = (Q, \Sigma, \Delta, I, R, c)$, where

- Q is a finite set of *states*,
- Σ and Δ are ranked alphabets of *input* and *output symbols*,
- $I \subseteq Q$ is a set of *initial states*,
- $R \subseteq T_\Sigma(Q) \times Q \times T_\Delta(Q)$ is a finite set of *rules* such that ℓ and r are linear in Q and $\mathrm{var}(r) \subseteq \mathrm{var}(\ell)$ for every $(\ell, q, r) \in R$, and
- $c\colon Q \to \mathrm{Reg}(\Sigma)$ assigns regular look-ahead to each state.

It is worth noting that we assign look-ahead to each state, but in a derivation we will only use the look-ahead if the state is deleted (see Definition 6). Moreover, in contrast to other definitions [12, 16], we do not allow the same state to occur several times (neither in the left- nor in the right-hand side). However, with the help of a simple renaming, each traditional linear extended top-down tree transducer can be written in our slightly more restrictive format. Next, we recall some important syntactic properties of our model. To this end, let $M = (Q, \Sigma, \Delta, I, R, c)$ be an l-xt$^{\mathrm{R}}$ in the following. It is

- a *linear extended tree transducer* (without look-ahead) [l-xt], if $c(q) = T_\Sigma$ for every $q \in Q$,
- a *linear top-down tree transducer with regular look ahead* [l-t$^{\mathrm{R}}$] if $\ell \in \Sigma(Q)$ for every $(\ell, q, r) \in R$,
- a *linear top-down tree transducer* (without look ahead) [l-t] if it is both an l-xt and an l-t$^{\mathrm{R}}$,
- *ε-free* if $\ell \notin Q$ for every $(\ell, q, r) \in R$,
- *strict* if $r \notin Q$ for every $(\ell, q, r) \in R$,
- a *delabeling* if $\ell \in \Sigma(Q)$ and $r \in Q \cup \Delta(Q)$ for every $(\ell, q, r) \in R$,
- *nondeleting* if $\mathrm{var}(r) = \mathrm{var}(\ell)$ for every $(\ell, q, r) \in R$, and
- a *finite-state relabeling* [qr] if it is a nondeleting, strict delabeling l-t such that $\mathrm{pos}_p(\ell) = \mathrm{pos}_p(r)$ for every $(\ell, q, r) \in R$ and $p \in \mathrm{var}(r)$.

We note that each l-t$^{\mathrm{R}}$ and each l-t is automatically ε-free. To simplify the notation in examples and illustrations, we sometimes write rules as $\ell \xrightarrow{q} r$ instead of (ℓ, q, r). Similarly, we write $\ell \xrightarrow{q,p} r$ as a shorthand for the two rules $\ell \xrightarrow{q} r$ and $\ell \xrightarrow{p} r$. Moreover, for every $p \in Q$ and $(\ell, q, r) \in R$ we identify $\mathrm{pos}_p(\ell)$ and $\mathrm{pos}_p(r)$ with their unique element if the sets are non-empty. Since ℓ and r are linear in Q, there is indeed at most one element in $\mathrm{pos}_p(\ell)$ and $\mathrm{pos}_p(r)$. Finally, for every $q \in Q$, we let

$$R_q = \{\rho \in R \mid \rho = (\ell, q, r)\}$$

be the subset of R that contains all rules for state q.

Example 2. *Let us consider the l-xt^R $M_1 = (Q, \Sigma, \Delta, \{\star\}, R, c)$ given by*

- $Q = \{\star, p, q, q', \mathrm{id}, \mathrm{id}'\}$,
- $\Sigma = \{\sigma^{(2)}, \gamma_1^{(1)}, \gamma_2^{(1)}\} \cup \Delta$,
- $\Delta = \{\sigma_1^{(2)}, \sigma_2^{(2)}, \gamma^{(1)}, \alpha^{(0)}\}$,
- *R contains exactly the following rules*

$$\sigma_1(p, q) \xrightarrow{\star,p} \sigma_1(p, q) \qquad \sigma_2(\mathrm{id}, \mathrm{id}') \xrightarrow{p,q} \sigma_2(\mathrm{id}, \mathrm{id}') \qquad \gamma_1(p) \xrightarrow{p} p$$
$$\sigma(q, \mathrm{id}) \xrightarrow{q} q \qquad \sigma(q', q) \xrightarrow{q} q \qquad \gamma_2(q) \xrightarrow{q} q$$
$$\gamma(\mathrm{id}) \xrightarrow{\mathrm{id},\mathrm{id}'} \gamma(\mathrm{id}) \qquad \alpha \xrightarrow{\mathrm{id},\mathrm{id}'} \alpha$$

Figure 1: Illustration of two rules with their implicit links.

- $c(q') = \{t \in T_\Sigma \mid \mathrm{pos}_{\sigma_2}(t) \neq \emptyset\}$ *and* $c(f) = T_\Sigma$ *for all other states* f.

It can easily be checked that $c(q')$ *is a regular tree language. The l-xt*^{R} M_1 *is an* ε*-free, nondeleting, delabeling top-down tree transducer with regular look-ahead.*

Next, we recall the semantics of the l-xt$^\mathrm{R}$ $M = (Q, \Sigma, \Delta, I, R, c)$, which is (mostly) given by synchronous substitution. While the links in a rule are implicit and established due to occurrences of equal states, we need an explicit linking structure for our sentential forms. These links will form dependencies that are used in a proof later on. To encode the links, we store a relation between positions of the input and output tree. Let $\mathcal{L} = \{D \mid D \subseteq \mathbb{N}^* \times \mathbb{N}^*\}$ be the set of all link structures. First, we define general sentential forms. Roughly speaking, a sentential form consists of an input tree and an output tree, in which positions are linked.

Definition 3 (see [11, Section 3]). An element $\langle \xi, D, \zeta \rangle \in T_\Sigma(Q) \times \mathcal{L} \times T_\Delta(Q)$ is a *sentential form* (for M) if $v \in \mathrm{pos}(\xi)$ and $w \in \mathrm{pos}(\zeta)$ for every $(v, w) \in D$. For a set $\mathcal{S}$ of sentential forms we define

$$\mathrm{links}(\mathcal{S}) = \{D \mid \langle \xi, D, \zeta \rangle \in \mathcal{S} \text{ for some trees } \xi \text{ and } \zeta\} \ .$$

Now we formalize the implicit links in a rule ρ by defining the explicit link structure determined by ρ. This explicit link structure is added to the link structure of a sentential form whenever the rule ρ is applied in the derivation process.

Definition 4. Let $\rho \in R$ be the rule (ℓ, q, r), and let $v, w \in \mathbb{N}^*$. The explicit link structure of ρ for the positions v and w is defined as

$$\mathrm{links}_{v,w}(\rho) = \{(v.\,\mathrm{pos}_p(\ell), w.\,\mathrm{pos}_p(r)) \mid p \in \mathrm{var}(r)\} \ .$$

Example 5. *Let us compute two explicit link structures.*

$$\mathrm{links}_{1,21}(\sigma_1(p, q) \xrightarrow{\star} \sigma_1(p, q)) = \{(11, 211), (12, 212)\}$$
$$\mathrm{links}_{1,21}(\sigma(q', q) \xrightarrow{q} q) = \{(12, 21)\}$$

In illustrations we use grayed splines to indicate the links. The rules above and their implicit links [i.e., those of $\mathrm{links}_{\varepsilon,\varepsilon}(\rho)$*] are displayed in Figure 1.*

The derivation process is started with a simple sentential form $\langle q, \{(\varepsilon, \varepsilon)\}, q \rangle$ consisting of the input and output tree q for some initial state $q \in I$ and the trivial link relating both roots. The current sentential form can evolve in two ways. Either we (nondeterministically) apply a rule (ℓ, q, r) to a pair (v, w) of linked occurrences of the state q or we apply the look-ahead. In the former case, such a rule application replaces the linked occurrences of q by the left and right-hand side of the rule. The explicit link structure of the rule for v and w is added to the current (explicit) link structure to obtain a new sentential form. Since we are interested in the dependencies created during derivation, we preserve all links and never remove a link from the linking structure. This replacement process can be repeated until no linked occurrences of states remain. Thus, we obtain an output tree without states, but the input tree of the sentential form can still contain states, which do not take part in an active link (i.e., a link relating two states in the sentential form). Each occurrence of such a state q can simply be replaced by any tree of the regular look-ahead tree language $c(q)$, where different occurrences of the same state can be replaced by different trees of $c(q)$.

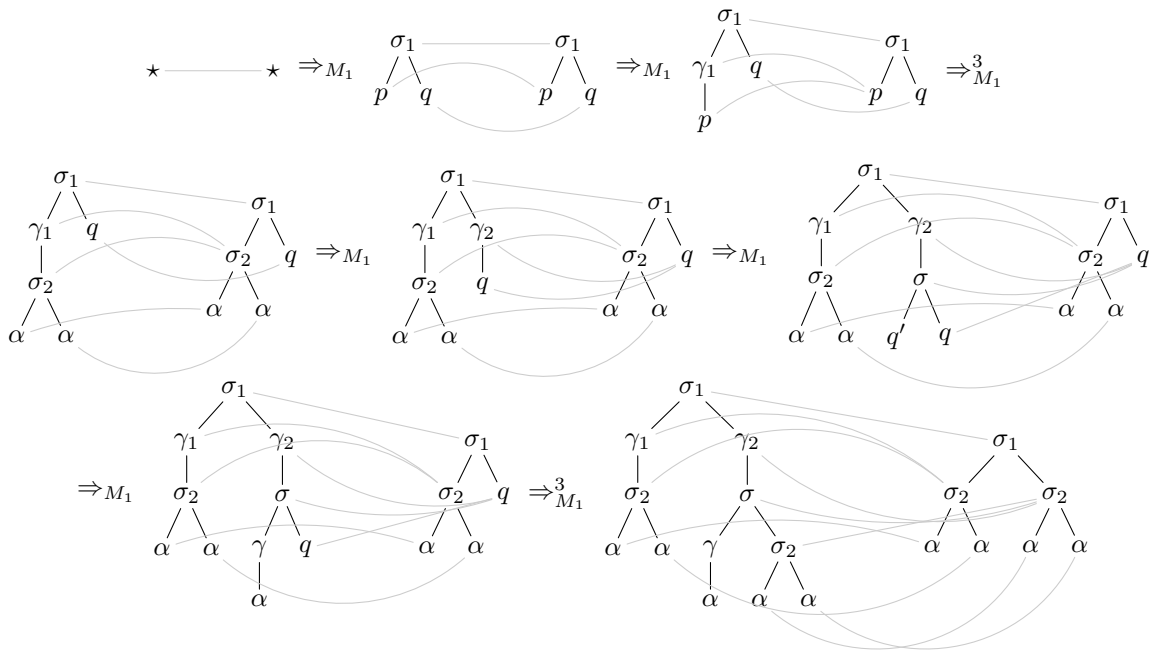

Figure 2: Derivation using the l-XT$^{\mathrm{R}}$ M_1 of Example 2.

Definition 6 (see [11, Section 3]). Given two sentential forms $\langle \xi, D, \zeta \rangle$ and $\langle \xi', D', \zeta' \rangle$, we write

$$\langle \xi, D, \zeta \rangle \Rightarrow_M \langle \xi', D', \zeta' \rangle$$

if

- there exists a rule $\rho = (\ell, q, r) \in R$ and input and output positions $(v, w) \in D \cap \big(\mathrm{pos}_q(\xi) \times \mathrm{pos}_q(\zeta)\big)$ such that
$$\xi' = \xi[\ell]_v \qquad \zeta' = \zeta[r]_w \qquad D' = D \cup \mathrm{links}_{v,w}(\rho) \ ,$$
- or there exists a position $v \in \mathrm{pos}_Q(\xi)$ with $w \notin \mathrm{pos}_Q(\zeta)$ for all $(v, w) \in D$ and there exists $t \in c(\xi(v))$ such that
$$\xi' = \xi[t]_v \qquad \zeta' = \zeta \qquad D' = D \ .$$

A few derivation steps using the l-xt$^{\mathrm{R}}$ M_1 of Example 2 are illustrated in Figure 2. Next, we define the tree transformation computed by an l-xt$^{\mathrm{R}}$.

Definition 7. The l-xt$^{\mathrm{R}}$ M computes the dependencies $\mathrm{dep}(M)$, which are given by

$$\mathrm{dep}(M) = \{(t, D, u) \in T_\Sigma \times \mathcal{L} \times T_\Delta \mid \exists q \in I \colon \langle q, \{(\varepsilon, \varepsilon)\}, q \rangle \Rightarrow_M^* (t, D, u)\} \ ,$$

where $\Rightarrow_M^*$ is the reflexive and transitive closure of $\Rightarrow_M$. Moreover, it computes the tree transformation M, which is given by $M = \{(t, u) \mid (t, D, u) \in \mathrm{dep}(M)\}$.

Example 8. *Let M_1 be the l-xt^R of Example 2. Then*

$$\big(\sigma_1(\gamma_1(\sigma_2(\alpha, \alpha)), \gamma_2(\sigma(\gamma(\alpha), \sigma_2(\alpha, \alpha)))), D, \sigma_1(\sigma_2(\alpha, \alpha), \sigma_2(\alpha, \alpha))\big) \in \mathrm{dep}(M_1)$$

where

$$D = \{(\varepsilon, \varepsilon), (1, 1), (11, 1), (111, 11), (112, 12), (2, 2), (21, 2), (212, 2), (2121, 21), (2122, 22)\} \ ,$$

which corresponds to the final element in the derivation displayed in Figure 2. Roughly speaking, the tree transformation computed by M_1 accepts only input trees of a certain shape. In such a tree it removes all γ_1-symbols on the left spine between the symbols σ_1 and σ_2. In addition, it also removes all symbols to the right of σ_1 except for those belonging to the left-most subtree rooted by σ_2.

Since every translation $(t, u) \in M$ is ultimately created by (at least) one successful derivation, we can inspect the links in the derivation process to exhibit the dependencies. Roughly speaking, the links establish which parts of the output tree were generated due to a particular part of the input tree. This correspondence is called *contribution* in [10].

Definition 9. To allow concise statements, we introduce the following short-hands:

$$\not\varepsilon = \varepsilon\text{-free} \qquad \text{s} = \text{strict} \qquad \text{d} = \text{delabeling} \qquad \text{n} = \text{nondeleting} \ .$$

We use these abbreviations in conjunction with l-xt$^{\text{R}}$ to restrict to devices with the indicated properties. For example, dnl-xt stands for "delabeling and nondeleting linear extended top-down tree transducer" (without look-ahead). We use the same abbreviations with the stem (i.e., the material behind the hyphen) in capital letters for the corresponding classes of computed tree transformations. For instance, dnl-XT stands for the class of all tree transformations computable by dnl-xt, and QR denotes the class of all tree transformations computable by qr.

Our device develops the input and output tree from top to bottom, which yields the name top-down tree transducer. Similarly, there exists a device called linear extended bottom-up tree transducer [12], which creates the trees from the leaves to the root (i.e., bottom-up) [7]. We do not recall the device formally here, but we use the abbreviations l-xb and l-XB together with the usual properties for it. We refer the reader to [12] for full details.

Next, we recall two important known statements for the class l-XT$^{\text{R}}$. The first statement relates the classes l-XT$^{\text{R}}$ and l-T$^{\text{R}}$ to l-XB and l-B, respectively. In addition, we demonstrate that look-ahead is superfluous in the nondeleting case. We use the brackets '[' and ']' for optional use of the restrictions $\not\varepsilon$, 'd', 's', and 'n' that have to be consistently applied.

Theorem 10 ([12, Proposition 3.3 and Corollary 4.1]).

$$[\not\varepsilon][\text{s}][\text{d}][\text{n}]\text{l-XT}^{\text{R}} = [\not\varepsilon][\text{s}][\text{d}][\text{n}]\text{l-XB} \qquad [\text{s}][\text{d}][\text{n}]\text{l-T}^{\text{R}} = [\text{s}][\text{d}][\text{n}]\text{l-B}$$
$$[\not\varepsilon][\text{s}][\text{d}]\text{nl-XT}^{\text{R}} = [\not\varepsilon][\text{s}][\text{d}]\text{nl-XT} \qquad [\text{s}][\text{d}]\text{nl-T}^{\text{R}} = [\text{s}][\text{d}]\text{nl-T}$$

To illustrate the consistent application of optional restrictions, the following statements are instances of the first result of the above theorem:

$$\text{l-XT}^{\text{R}} = \text{l-XB} \qquad \text{and} \qquad \not\varepsilon\text{sl-XT}^{\text{R}} = \not\varepsilon\text{sl-XB} \ .$$

Secondly, we relate the class l-XT$^{\text{R}}$ to l-T$^{\text{R}}$, which tells us how to emulate linear extended top-down tree transducers with regular look-ahead by linear top-down tree transducers with regular look-ahead [8].

Theorem 11.

$$\not\varepsilon[\text{s}][\text{d}][\text{n}]\text{l-XT}^{\text{R}} = \text{lcs-H}^{-1} \, ; [\text{s}][\text{d}][\text{n}]\text{l-T}^{\text{R}} \qquad [\text{s}][\text{d}][\text{n}]\text{l-XT}^{\text{R}} = \text{lc-H}^{-1} \, ; [\text{s}][\text{d}][\text{n}]\text{l-T}^{\text{R}}$$

Proof. The statements are easily obtained from Theorem 10 and [12, Lemma 4.1].

4. Four classes that are closed at a finite power

In this section, we show that the classes $\not\in$l-XT$^{\mathrm{R}}$, $\not\in$l-XT, $\not\in$sl-XT$^{\mathrm{R}}$, and $\not\in$sl-XT are closed under composition at a finite power. We first recall a central result [3] that proves that the (very restricted) class $\not\in$snl-XT is closed under composition at power 2. Note that [3] expresses this result in terms of bimorphisms, and 'strict' is abbreviated by 'e' there. In fact, $\not\in$snl-XT = B(lcs, lcs) by [2] and [19, Theorem 4].

Theorem 12 ([3, Theorem 6.2]). For every $n \geq 2$,

$$\not\in\text{snl-XT} \subsetneq \not\in\text{snl-XT}^2 = \not\in\text{snl-XT}^n \ .$$

Now we establish our first composition result, which is analogous to the classical composition result for linear top-down tree transducers with regular look-ahead [8]. The only difference is that our first transducer has extended left-hand sides (i.e., it is an l-xt$^{\mathrm{R}}$ instead of just an l-t$^{\mathrm{R}}$). Since this fact does not affect the original composition construction [7], we can use the original result to obtain our first result.

Lemma 13 (composition on the right).

$$[\not\in][\mathrm{s}][\mathrm{d}][\mathrm{n}]\text{l-XT}^{\mathrm{R}} \,;\, [\mathrm{s}][\mathrm{d}][\mathrm{n}]\text{l-T}^{\mathrm{R}} = [\not\in][\mathrm{s}][\mathrm{d}][\mathrm{n}]\text{l-XT}^{\mathrm{R}}$$

Proof. By Theorem 11, $[\not\in][\mathrm{s}][\mathrm{d}][\mathrm{n}]\text{l-XT}^{\mathrm{R}} = \mathrm{lc}[\mathrm{s}]\text{-H}^{-1} \,;\, [\mathrm{s}][\mathrm{d}][\mathrm{n}]\text{l-T}^{\mathrm{R}}$, where the option $[\not\in]$ in the left-hand side corresponds to the option [s] in lc[s]-H^{-1}. Thus, we obtain the chain of equalities

$$\begin{aligned}
[\not\in][\mathrm{s}][\mathrm{d}][\mathrm{n}]\text{l-XT}^{\mathrm{R}} \,;\, [\mathrm{s}][\mathrm{d}][\mathrm{n}]\text{l-T}^{\mathrm{R}} &= \mathrm{lc}[\mathrm{s}]\text{-H}^{-1} \,;\, [\mathrm{s}][\mathrm{d}][\mathrm{n}]\text{l-T}^{\mathrm{R}} \,;\, [\mathrm{s}][\mathrm{d}][\mathrm{n}]\text{l-T}^{\mathrm{R}} \\
&= \mathrm{lc}[\mathrm{s}]\text{-H}^{-1} \,;\, [\mathrm{s}][\mathrm{d}][\mathrm{n}]\text{l-T}^{\mathrm{R}} \\
&= [\not\in][\mathrm{s}][\mathrm{d}][\mathrm{n}]\text{l-XT}^{\mathrm{R}} \ ,
\end{aligned}$$

where in the second step we applied the following well-known composition result for linear top-down tree transducers with regular look-ahead [8]:[3]

$$[\mathrm{s}][\mathrm{d}][\mathrm{n}]\text{l-T}^{\mathrm{R}} \,;\, [\mathrm{s}][\mathrm{d}][\mathrm{n}]\text{l-T}^{\mathrm{R}} = [\mathrm{s}][\mathrm{d}][\mathrm{n}]\text{l-T}^{\mathrm{R}} \ . \qquad \square$$

In the next result we present a decomposition that corresponds to property P of [6]. It demonstrates how to simulate an l-xt$^{\mathrm{R}}$ by a delabeling l-d$^{\mathrm{R}}$ and an $\not\in$snl-xt, for which we have the composition closure result in Theorem 12.

Lemma 14 (decomposition).

$$\not\in[\mathrm{s}]\text{l-XT}^{\mathrm{R}} \subseteq [\mathrm{s}]\mathrm{dl}\text{-T}^{\mathrm{R}} \,;\, \not\in\text{snl-XT}$$

Proof. Let $M = (Q, \Sigma, \Delta, I, R, c)$ be an arbitrary ε-free l-xt$^{\mathrm{R}}$. Moreover, let $m \in \mathbb{N}$ be such that $m \geq |\mathrm{var}(r)|$ for every $(\ell, q, r) \in R$; i.e., the integer m is larger than the maximal number of states in the right-hand side of the rules. For every rule $\rho = (\ell, q, r) \in R$ and non-state position $w \in \mathrm{pos}_\Sigma(\ell)$ in its left-hand side, let

$$\mathrm{used}_\rho(w) = \{i \in \mathbb{N} \mid wi \in \mathrm{pos}(\ell), \mathrm{var}(\ell|_{wi}) \cap \mathrm{var}(r) \neq \emptyset\}$$

be the indices of the direct subtrees below w in ℓ, which still contain states that occur in r. We construct a delabeling l-xt$^{\mathrm{R}}$

$$M_1 = (Q_1, \Sigma, R \cup \{@_i \mid 0 \leq i \leq m\}, I_1, R_1, c_1)$$

such that

- $Q_1 = \{\langle \rho, w\rangle \mid \rho = (\ell, q, r) \in R, w \in \mathrm{pos}(\ell)\}$,
- $\mathrm{rk}(\rho) = |\mathrm{used}_\rho(\varepsilon)|$ for every rule $\rho \in R$ and $\mathrm{rk}(@_i) = i$ for every $0 \leq i \leq m$,

[3] The abbreviation 'd' has a completely different meaning in [8].

- $I_1 = \{\langle \rho, \varepsilon\rangle \mid q \in I, \rho \in R_q\}$, and
- for every rule $\rho = (\ell, q, r) \in R$ and non-state position $w \in \mathrm{pos}_\Sigma(\ell)$ we construct the following rule of R_1:

$$\ell(w)(\langle \rho, w1\rangle, \ldots, \langle \rho, wk\rangle) \xrightarrow{\langle \rho, w\rangle} \begin{cases} \langle \rho, wi_1\rangle & \text{if } r \in Q \\ \rho(\langle \rho, wi_1\rangle, \ldots, \langle \rho, wi_n\rangle) & \text{if } r \notin Q, w = \varepsilon \\ @_n(\langle \rho, wi_1\rangle, \ldots, \langle \rho, wi_n\rangle) & \text{otherwise,} \end{cases}$$

  where $k = \mathrm{rk}(\ell(w))$ and $\{i_1, \ldots, i_n\} = \mathrm{used}_\rho(w)$ with $i_1 < \cdots < i_n$,
- for every rule $\rho = (\ell, q, r) \in R$, state position $w \in \mathrm{pos}_Q(\ell)$ and rule $\rho' \in R_{\ell(w)}$ we construct the following rule of R_1:

$$\langle \rho', \varepsilon\rangle \xrightarrow{\langle \rho, w\rangle} \langle \rho', \varepsilon\rangle \ ,$$

- $c_1(\langle \rho, w\rangle) = (\ell|_w)c$ for every $\rho = (\ell, q, r) \in R$ and $w \in \mathrm{pos}(\ell)$.

To obtain the desired l-t$^{\mathrm{R}}$ we simply eliminate the ε-rules using standard methods. The elimination is successful because the ε-rules are non-cyclic.[4] Intuitively speaking, the transducer M_1 processes the input and deletes subtrees that are not necessary for further processing. Moreover, it marks the positions in the input where a rule application would be possible. Finally, it also already executes all erasing rules of M.

Let $m' \in \mathbb{N}$ be such that $m' \geq |\ell|$ for all $(\ell, q, r) \in R$. We construct the l-xt

$$M_2 = (Q, R \cup \{@_i \mid 0 \leq i \leq m\}, \Delta, I, R_2)$$

such that R_2 contains all valid rules $(\rho(t_1, \ldots, t_k), q, r)$ with $\rho = (\ell, q, r) \in R$ of a strict ln-xt with $\mathrm{pos}_R(t_i) = \emptyset$ and $|t_i| \leq m'$ for every $1 \leq i \leq k$, where $k = \mathrm{rk}(\rho)$. □

Example 15. *A full example for the construction of Lemma 14 would be quite lengthy, so let us only illustrate the construction of M_1 on the example rule ρ:*

$$\sigma(p, \sigma(\alpha, q)) \xrightarrow{q} \sigma(\alpha, \sigma(q, \alpha)) \ ,$$

for which the only non-trivial look-ahead is $c(p) = L$. For this rule we construct the following rules in M_1:

$$\sigma(\langle \rho, 1\rangle, \langle \rho, 2\rangle) \xrightarrow{\langle \rho, \varepsilon\rangle} \rho(\langle \rho, 2\rangle) \qquad\qquad \langle \rho', \varepsilon\rangle \xrightarrow{\langle \rho, 1\rangle} \langle \rho', \varepsilon\rangle$$

$$\sigma(\langle \rho, 21\rangle, \langle \rho, 22\rangle) \xrightarrow{\langle \rho, 2\rangle} @_1(\langle \rho, 22\rangle) \qquad \alpha \xrightarrow{\langle \rho, 21\rangle} @_0 \qquad \langle \rho'', \varepsilon\rangle \xrightarrow{\langle \rho, 22\rangle} \langle \rho'', \varepsilon\rangle$$

for all rules $\rho' \in R_p$ and $\rho'' \in R_q$. Moreover, the look-ahead c_1 of M_1 is such that

$$c_1(\langle \rho, 1\rangle) = L \qquad\qquad c_1(\langle \rho, 21\rangle) = \{\alpha\} \ .$$

Now we are able to prove that the class $\not\in$l-XT$^{\mathrm{R}}$ is closed under composition at the third power.

Theorem 16. For every $n \geq 1$,

$$(\not\in[\mathrm{s}]\text{l-XT}^{\mathrm{R}})^n \subseteq [\mathrm{s}]\text{dl-T}^{\mathrm{R}} \,;\, \not\in\text{snl-XT}^2 \subseteq (\not\in[\mathrm{s}]\text{l-XT}^{\mathrm{R}})^3 \ .$$

Proof. The second inclusion is trivial, so we prove the first inclusion by induction over n. The statement is a trivial consequence of Lemma 14 for $n = 1$. In the induction step, we obtain

$$\begin{aligned}(\not\in[\mathrm{s}]\text{l-XT}^{\mathrm{R}})^{n+1} &= \not\in[\mathrm{s}]\text{l-XT}^{\mathrm{R}} \,;\, (\not\in[\mathrm{s}]\text{l-XT}^{\mathrm{R}})^n \\ &\subseteq \not\in[\mathrm{s}]\text{l-XT}^{\mathrm{R}} \,;\, [\mathrm{s}]\text{dl-T}^{\mathrm{R}} \,;\, \not\in\text{snl-XT}^2\end{aligned}$$

[4] Note that due to the ε-freeness of M, we have $w \neq \varepsilon$ in the ε-rules of the second item. Since these rules are the only constructed ε-rules, we cannot chain two ε-rules.

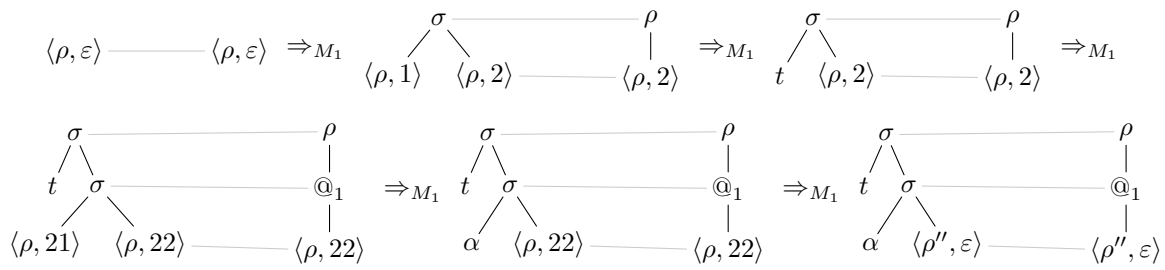

Figure 3: Derivation using M_1 of Example 15 for some $t \in L$ and $\rho'' \in R_q$.

$$\begin{aligned} &= \not\in\text{[s]l-XT}^{\text{R}} \,;\, \not\in\text{snl-XT}^2 \\ &\subseteq \text{[s]dl-T}^{\text{R}} \,;\, \not\in\text{snl-XT} \,;\, \not\in\text{snl-XT}^2 \\ &= \text{[s]dl-T}^{\text{R}} \,;\, \not\in\text{snl-XT}^2 \end{aligned}$$

where we used the induction hypothesis in the second step, Lemma 13 in the third step, Lemma 14 in the fourth step, and finally, Theorem 12 in the last step. □

It is known that we can simulate an l-t$^{\text{R}}$ (with look-ahead) by a composition of two l-t (without look-ahead). This fact allows us to obtain the following corollary for the closure of the class $\not\in$l-XT under composition at power four.

Corollary 17 (without look-ahead). For every $n \geq 1$

$$\not\in\text{[s]l-XT}^n \subseteq \text{QR} \,;\, \text{[s]dl-T} \,;\, \not\in\text{snl-XT}^2 \subseteq \not\in\text{[s]l-XT}^4 \ .$$

Proof. The second inclusion is trivial, and for the first inclusion we use [s]dl-T$^{\text{R}} \subseteq$ QR ; [s]dl-T and Theorem 16. □

Up to now, we have shown that the classes $\not\in$l-XT$^{\text{R}}$ and $\not\in$l-XT are closed under composition at the third and fourth power, respectively. In the rest of the section, we will show that the (strict) classes $\not\in$sl-XT$^{\text{R}}$ and $\not\in$sl-XT are closed under composition already at the second power. We start with a lemma that proves the converse of Lemma 14 in the strict case.

Lemma 18 (composition on the left).

$$\text{lds-H} \,;\, \not\in\text{sl-XT}^{\text{R}} \subseteq \not\in\text{sl-XT}^{\text{R}}$$

Proof. Let $d\colon T_\Sigma \to T_\Delta$ be a strict delabeling linear tree homomorphism. Recall that d also defines a tree transformation $d\colon T_\Sigma(Q) \to T_\Delta(Q)$, which acts as an identity on states; i.e., $d(q) = q$ for every $q \in Q$. Moreover, let $M = (Q, \Delta, \Gamma, I, R, c)$ be a strict l-xt$^{\text{R}}$ and $m \in \mathbb{N}$ be such that $m \geq |\ell'|$ for every $\ell' \in d^{-1}(\ell)$ and $(\ell, q, r) \in R$. Note that the set $d^{-1}(\ell)$ is finite because d is strict. We construct the l-xt$^{\text{R}}$ $M' = (Q \cup \{1, \ldots, m\}, \Sigma, \Gamma, I, R', c')$ such that for every rule $(\ell, q, r) \in R$ we have each valid rule (ℓ', q, r) in R' where $\ell' \in d^{-1}(\ell)$ and $|\text{pos}_\Sigma(\ell')| = |\text{pos}_\Delta(\ell)|$. Moreover, $c'(q) = d^{-1}(c(q))$ for all $q \in Q$ and $c'(i) = T_\Sigma$ for every $1 \leq i \leq m$. Clearly, $d^{-1}(c(q))$ is a regular tree language because $c(q)$ is a regular tree language and regular tree languages are closed under inverse tree homomorphisms [8, Lemma 1.2]. Finally, we observe that M' is also strict since it has the same right-hand sides as M. □

Now we state and prove the mentioned result for the class $\not\in$sl-XT$^{\text{R}}$. The general approach is the same as in Theorem 16, but we now use Lemma 18 in the main step.

Theorem 19. For every $n \geq 1$

$$(\not\in\text{sl-XT}^{\text{R}})^n \subseteq \not\in\text{snl-XT} \,;\, \not\in\text{sl-XT}^{\text{R}} \subseteq (\not\in\text{sl-XT}^{\text{R}})^2 \ .$$

Class	Closed at power	Proved in
$\not\ell$-XT$^{\mathrm{R}}$	3	Theorem 16
$\not\ell$-XT	4	Corollary 17
$\not$sl-XT$^{\mathrm{R}}$	2	Theorem 19
$\not$sl-XT	2	Corollary 20

Table 2: Summary of the results of Section 4.

Proof. Again, the second inclusion is trivial. For the first inclusion, the induction basis is also trivial, and we prove the induction step as follows

$$\begin{aligned}
(\not\text{sl-XT}^{\mathrm{R}})^{n+1} &= (\not\text{sl-XT}^{\mathrm{R}})^{n} \,;\, \not\text{sl-XT}^{\mathrm{R}} \\
&\subseteq \not\text{snl-XT} \,;\, \not\text{sl-XT}^{\mathrm{R}} \,;\, \not\text{sl-XT}^{\mathrm{R}} \\
&= \not\text{snl-XT} \,;\, \not\text{snl-XT} \,;\, \text{lds-H} \,;\, \not\text{sl-XT}^{\mathrm{R}} \\
&\subseteq \not\text{snl-XT}^2 \,;\, \not\text{sl-XT}^{\mathrm{R}} \\
&= \not\text{snl-XT}^2 \,;\, \not\text{snl-XT} \,;\, \text{lds-H} \\
&= \not\text{snl-XT}^2 \,;\, \text{lds-H} \\
&= \not\text{snl-XT} \,;\, \not\text{sl-XT}^{\mathrm{R}}
\end{aligned}$$

using the induction hypothesis in the second step, Lemma 18 in the fourth step, and Theorem 12 in the sixth step. The final step composes the tree homomorphism with the second tree transducer using the result of Lemma 13. □

Again, we derive the "without look-ahead"-version of the general result. In contrast to Corollary 17, the power of closedness does not increase by one for strict $\not\ell$-xt.

Corollary 20 (without look-ahead). For every $n \geq 1$

$$\not\text{sl-XT}^{n} \subseteq \not\text{snl-XT} \,;\, \not\text{sl-XT} \subseteq \not\text{sl-XT}^{2} \ .$$

Proof. The second inclusion is trivial. For the first inclusion, we observe that

$$\begin{aligned}
\not\text{sl-XT}^{n} &\subseteq \not\text{snl-XT} \,;\, \not\text{sl-XT}^{\mathrm{R}} \\
&= \text{lcs-H}^{-1} \,;\, \text{snl-T} \,;\, \text{QR} \,;\, \not\text{sl-XT} \\
&= \text{lcs-H}^{-1} \,;\, \text{snl-T} \,;\, \not\text{sl-XT} \\
&= \not\text{snl-XT} \,;\, \not\text{sl-XT}
\end{aligned}$$

using first Theorem 19, and then Theorem 11 and [8, Theorem 2.6] in the second step. Next, we use the composition result [4, Theorem 1] for top-down tree transducers and Theorem 11 again. □

Table 2 summarizes our results concerning the powers at which the considered classes are closed under composition.

5. Least power of closedness

In this section, we will determine the least power at which each of the classes $\not\ell$-XT$^{\mathrm{R}}$, $\not\ell$-XT, $\not$sl-XT$^{\mathrm{R}}$, and $\not$sl-XT is closed under composition.

Theorem 21. For every $n \geq 2$

$$\not\text{sl-XT} \subsetneq \not\text{sl-XT}^2 = \not\text{sl-XT}^n \qquad (\not\text{sl-XT}^{\mathrm{R}}) \subsetneq (\not\text{sl-XT}^{\mathrm{R}})^2 = (\not\text{sl-XT}^{\mathrm{R}})^n.$$

Proof. By [3, Section 3.4], the classes $\not$sl-XT and $\not$sl-XT$^{\mathrm{R}}$ are not closed under composition at power 1. Moreover, by Theorem 19 and Corollary 20 both classes are closed at power 2. □

In the following, we will use the computed dependencies, for which we observe some important properties next. For the rest of this section, let $M = (Q, \Sigma, \Delta, I, R, c)$ be the considered l-xt$^{\mathrm{R}}$. To simplify the development, we disregard the actual input and output trees in the computed dependencies and will consider only the set links(M) = links(dep(M)). We say that M computes the linking structures of links(M).

Definition 22 ([20, Definition 8]). A linking structure $D \in \mathcal{L}$ is *input hierarchical*[5] if for all links $(v_1, w_1), (v_2, w_2) \in D$

- if $v_1 \prec v_2$, then $w_1 \preceq w_2$, and
- if $v_1 = v_2$, then $w_1 \preceq w_2$ or $w_2 \preceq w_1$.

Input hierarchical linking structures have no crossing links, which are links $(v_1, w_1), (v_2, w_2) \in D$ such that $v_1 \prec v_2$ and $w_2 \prec w_1$. We define *output hierarchical* using the same conditions as in Definition 22 for the output side; i.e., D is output hierarchical if and only if D^{-1} is input hierarchical. Moreover, D is *hierarchical* if it is both input and output hierarchical. Finally, a set $\mathcal{D} \subseteq \mathcal{L}$ of linking structures is hierarchical if each element $D \in \mathcal{D}$ is hierarchical.

We also need a property that enforces the existence of certain links. Roughly speaking, for a set of linking structures $\mathcal{D} \subseteq \mathcal{L}$ there should be an integer that limits the distance between links in each linking structure $D \in \mathcal{D}$.

Definition 23 ([20, Definition 10]). A set $\mathcal{D} \subseteq \mathcal{L}$ of link structures has *bounded distance in the input* if there exists an integer $k \in \mathbb{N}$ such that for every $D \in \mathcal{D}$ and all $(v, w), (vv'', w'') \in D$ there exists $(vv', w') \in D$ with $v' \preceq v''$ and $|v'| \leq k$.

In other words, in a set of link structures with bounded distance k in the input, we know for any of its link structures that between any two source-nested links there exists a link such that the distance of its source to the smaller link's source is at most k. This yields that the distance to the source of the next nested link (if such a link does exist) can be at most k. Note however, that the above property does not require a link every k symbols. As before, a set $\mathcal{D} \subseteq \mathcal{L}$ of linking structures has *bounded distance in the output* if $\mathcal{D}^{-1} = \{D^{-1} \mid D \in \mathcal{D}\}$ has bounded distance in the input. Finally, $\mathcal{D}$ has *bounded distance* if it has bounded distance in both the input and the output.

Lemma 24. For every l-xt$^{\mathrm{R}}$ M, the set links(M) is hierarchical with bounded distance.

Proof. This lemma follows trivially from Definition 6.

Next we consider the problem whether a tree transformation can be computed by an l-xt$^{\mathrm{R}}$. For this we specify certain links that are intuitively clear and necessary between nodes of input-output tree pairs. Hereby we obtain the specification sentential forms. Then we consider whether this specification can be implemented by an l-xt$^{\mathrm{R}}$. Often we cannot identify the nodes of a link exactly. In such cases, we use splines with inverted arrow heads, which indicate that there is a link to some position of the subtree pointed to. For example, the splines in Figure 4 indicate that a node of t_1 (resp. t_2) on the left is linked to a node of t_1 (resp. t_2) on the right.

Theorem 25. $(\not\text{l-XT}^{\mathrm{R}})^2 \subsetneq (\not\text{l-XT}^{\mathrm{R}})^3 = (\not\text{l-XT}^{\mathrm{R}})^n$ for every $n \geq 3$.

[5] This notion is called *strictly input hierarchical* in [20].

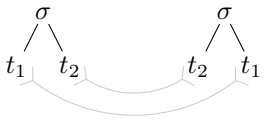

Figure 4: Links with inverted arrow.

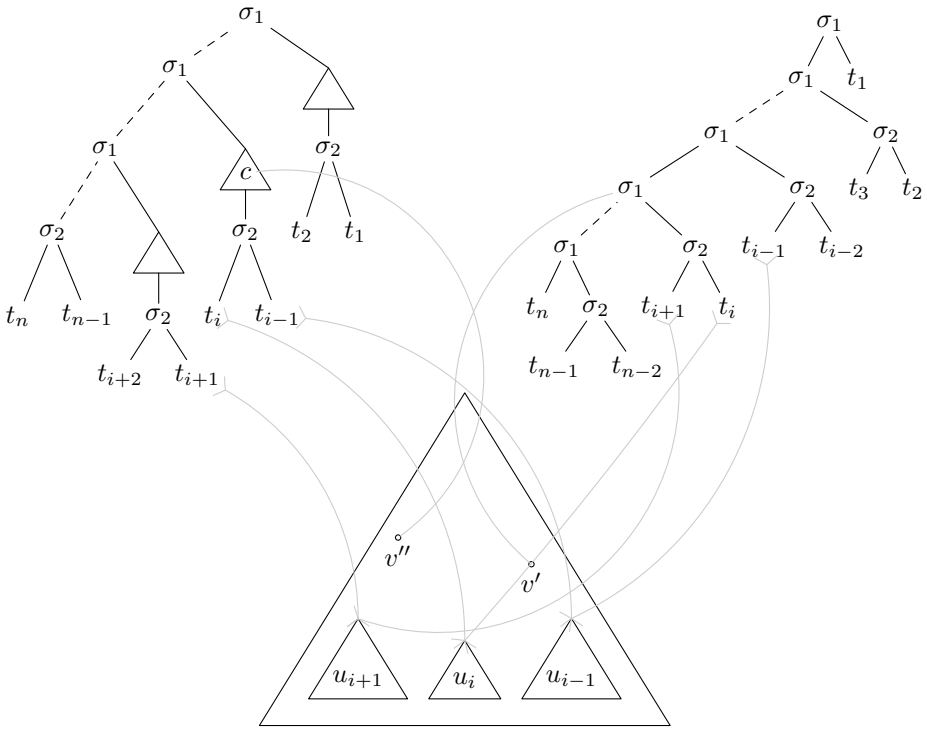

Figure 5: Illustration of the relevant part of the specification used in the proof of Theorem 25.

Proof. The inclusion is trivial and the equality follows from Theorem 16, so we only have to prove strictness. Recall the l-xt$^{\mathrm{R}}$ M_1 of Example 2. In addition, we use the two bimorphisms B_2 and B_3 of [6, Section II.2.2.3.1], which are in B(lcs,lcs).[6] As mentioned, $\not\in$snl-XT = B(lcs, lcs), hence B_2 and B_3 can also be defined by some $\not\in$snl-xt M_2 and M_3, respectively. For convenience, we present M_2 and M_3 explicitly before we show that $\tau = M_1 \,;\, M_2 \,;\, M_3$ cannot be computed by a composition of two $\not\in$l-xt$^{\mathrm{R}}$.

Let $M_2 = (Q_2, \Delta, \Gamma, \{\star\}, R_2)$ be the $\not\in$snl-xt with

- $Q_2 = \{\star, \mathrm{id}, \mathrm{id}'\}$ and $\Gamma = \{\sigma^{(2)}, \gamma^{(1)}, \alpha^{(0)}\}$, and
- R_2 is the set of the rules

$$\sigma_1(\star, \sigma_2(\mathrm{id}, \mathrm{id}')) \xrightarrow{\star} \sigma(\sigma(\star, \mathrm{id}), \mathrm{id}') \qquad \sigma_2(\mathrm{id}, \mathrm{id}') \xrightarrow{\star} \sigma(\mathrm{id}, \mathrm{id}')$$
$$\gamma(\mathrm{id}) \xrightarrow{\mathrm{id},\mathrm{id}'} \gamma(\mathrm{id}) \qquad \alpha \xrightarrow{\mathrm{id},\mathrm{id}'} \alpha \ .$$

Moreover, let $M_3 = (Q_3, \Gamma, \Delta, \{\star\}, R_3)$ be the $\not\in$snl-xt with

- $Q_3 = \{\star, p, \mathrm{id}, \mathrm{id}'\}$,

[6] In [6] strictness is denoted by 'e'.

- R_3 is the set of the rules

$$\sigma(p,\mathrm{id}) \xrightarrow{\star} \sigma_1(p,\mathrm{id}) \qquad \sigma(\sigma(p,\mathrm{id}),\mathrm{id}') \xrightarrow{p} \sigma_1(p,\sigma_2(\mathrm{id},\mathrm{id}')) \qquad \gamma(\mathrm{id}) \xrightarrow{\mathrm{id},\mathrm{id}'} \gamma(\mathrm{id})$$
$$\gamma(\mathrm{id}) \xrightarrow{p} \gamma(\mathrm{id}) \qquad \alpha \xrightarrow{\mathrm{id},\mathrm{id}'} \alpha \ .$$

We present a proof by contradiction, hence we assume that $\tau = N_1 \,;\, N_2$ for some $\not\in$l-xt N_1 and N_2. With the help of Lemma 24, we conclude that links(N_1) and links(N_2) are hierarchical with bounded distance k_1 and k_2, respectively. We consider $(t,u) \in \tau$ such that the left σ_1-spines of t and u are longer than k_1 and k_2, respectively.[7] Moreover, by assumption, there exists an intermediate tree s and linking structures $D_1, D_2 \in \mathcal{L}$ such that $(t, D_1, s) \in \mathrm{dep}(N_1)$ and $(s, D_2, u) \in \mathrm{dep}(N_2)$. We specify some links that, due to the last two inclusions, should be necessarily present in D_1 and D_2 and show that they lead to a contradiction (see Figure 5).

First, we consider the links D_2 generated by N_2. Since the left σ_1-spine in u is longer than k_2 and there are links at the root (i.e., $(\varepsilon,\varepsilon) \in D_2$) and inside t_n, there must be a linking point at position $w \in \mathrm{pos}_{\sigma_1}(u)$ along the left σ_1-spine with $w \neq \varepsilon$, which links to position v' in the intermediate tree s (i.e., $(v', w) \in D_2$). Let $u|_w = \sigma_1(u', \sigma_2(t_{i+1}, t_i))$. We assume that every t_j with $j \leq n$ is high enough (i.e., $\mathrm{ht}(t_j) > k_2$). This yields that there exist linking points in t_{i+1}, t_i, and t_{i-1} of u. Let $(v'_{i+1}, w_{i+1}), (v'_i, w_i), (v'_{i-1}, w_{i-1}) \in D_2$ be those linking points. Since D_2 is hierarchical and w_{i+1} and w_i are below w in u, we know that v'_{i+1} and v'_i are below v' in s (i.e., $v' \preceq v'_{i+1}, v'_i$), whereas $v' \not\preceq v'_{i-1}$. Next, we locate t_i in the input tree t. By the general shape of t, the subtree t_i occurs in a subtree $\sigma_1(t', c[\sigma_2(t_i, t_{i-1})])$ for some tree $c \in T_\Sigma(X_1)$ with exactly one occurrence of x_1. Again we assume that c is suitably large[8], which forces a linking point inside c in addition to those in t_{i+1}, t_i, and t_{i-1} because $(t,s) \in N_1$. Using the same arguments for N_1 as before for N_2, we now locate the links $(y, v'') \in D_1$ linking into c, which dominates the links $(y_i, v''_i), (y_{i-1}, v''_{i-1}) \in D_1$ linking to t_i and t_{i-1}, respectively. Thus, $v'' \preceq v''_i, v''_{i-1}$. In addition, there is a link $(y_{i+1}, v''_{i+1}) \in D_1$ linking into t_{i+1}, which is not dominated by v''. Consequently, $v'' \not\preceq v''_{i+1}$. Finally, let

$$v_{i-1} = \mathrm{lcp}(v'_{i-1}, v''_{i-1}) \qquad v_i = \mathrm{lcp}(v'_i, v''_i) \qquad v_{i+1} = \mathrm{lcp}(v'_{i+1}, v''_{i+1}) \ .$$

It is easy to see that the relevant links can be chosen such that

$$\begin{array}{lll} v' \not\preceq v_{i-1} & v' \preceq v_i & v' \preceq v_{i+1} \\ v'' \preceq v_{i-1} & v'' \preceq v_i & v'' \not\preceq v_{i+1} \ . \end{array}$$

Now, we have either $\mathrm{lcp}(v_{i-1}, v_i) \preceq \mathrm{lcp}(v_i, v_{i+1})$ or $\mathrm{lcp}(v_i, v_{i+1}) \preceq \mathrm{lcp}(v_{i-1}, v_i)$. In the first case we get

$$v'' \preceq \mathrm{lcp}(v_{i-1}, v_i) \preceq \mathrm{lcp}(v_i, v_{i+1}) \preceq v_{i+1} \ ,$$

and in the second case we get

$$v' \preceq \mathrm{lcp}(v_i, v_{i+1}) \preceq \mathrm{lcp}(v_{i-1}, v_i) \preceq v_{i-1} \ ,$$

which are contradictions. □

Fortunately, we can use the proof of the previous theorem to conclude that four is the least power at which the class $\not\in$l-XT is closed under composition.

Corollary 26. $\not\in$l-XT$^3 \subsetneq \not\in$l-XT$^4 = \not\in$l-XT^n for every $n \geq 4$.

[7] Some additional requirements on t and u are developed in the proof later on.
[8] More precisely, we assume that the only element of $\mathrm{pos}_{x_1}(c)$ is longer than k_1.

Class	Least power of closedness	Proved in
B(l, l)	4	[6, Section II-2-2-3-3]
$\not\in$snl-XT = $\not\in$snl-XT$^{\mathrm{R}}$	2	[3, Theorem 6.2]
$\not\in$sl-XT$^{\mathrm{R}}$, $\not\in$sl-XT	2	Theorem 21
$\not\in$l-XT$^{\mathrm{R}}$	3	Theorem 25
$\not\in$l-XT	4	Corollary 26

Table 3: Summary of the results of Sections 4 and 5.

Proof. The inclusion is trivial and the equality follows from Corollary 17. For the strictness, the proof of Theorem 25 essentially shows that in the first step we must delete the contexts indicated by triangles (such as c) in Figure 5 because otherwise we can apply the method used in the proof to derive a contradiction (it relies on the existence of a linking point inside such a context c). Thus, in essence we must first implement a variant of the l-xt$^{\mathrm{R}}$ M_1 of Example 2. It is a simple exercise to show that the deletion of the excess material cannot be done by a single l-xt as it cannot reliably determine the left-most occurrence of σ_2 without the look-ahead. Thus, if we only have l-xt to achieve the transformation, then we already need a composition of two l-xt to perform the required deletion of the contexts, which proves the main statement. □

In Table 3 we summarize the results of this and the previous section, which allow us to present the least power at which the closure of the considered composition hierarchies is achieved. For the sake of completeness, we also present the corresponding results for the classes $\not\in$snl-XT and B(l, l) that were obtained in [3, 6]. Recall that B(l, l) is the class of all tree transformations computable by bimorphisms, in which both tree homomorphisms are linear.

6. Infinite composition hierarchies

To complete the picture, we will need one further result showing the infiniteness of the composition hierarchy for a large number of classes. In order to obtain a result that is as general as possible, we use bimorphisms [3] instead of l-xt$^{\mathrm{R}}$ in this section. We conclude several results for various tree transducer classes from the result for bimorphisms.

The main auxiliary notion used in the proof of the infiniteness of the composition hierarchy is a notion assigning levels to positions in a tree. Let $t \in T_\Sigma$ and $\ell \in \mathbb{N}$ be an arbitrary integer. Since branching positions (i.e., those that are labeled by symbols of rank at least 2) will play an essential role, we define the set of all branching positions of t together with two different successors and the branching positions along a given path. For every $w, ww'' \in \mathrm{pos}(t)$, let

$$\begin{aligned} \mathrm{br}_t &= \{\langle w, i, j\rangle \mid w \in \mathrm{pos}(t),\, t(w) \notin \Sigma_0 \cup \Sigma_1,\, 1 \le i, j \le \mathrm{rk}(t(w)),\, i \neq j\} \\ \mathrm{br}_t(w, w'') &= \{ww' \mid w' \preceq w'' \text{ and } (ww', i, j) \in \mathrm{br}_t \text{ for some } i, j\} \ . \end{aligned}$$

We inductively define the sets $\mathrm{SP}^\ell_n(t) \subseteq \mathrm{pos}(t) \times \mathbb{N} \times \mathbb{N}$ and $\mathrm{P}^\ell_n(t) \subseteq \mathrm{pos}(t)$ for every $t \in T_\Sigma$ and $n \in \mathbb{N}$ as follows:

$$\begin{aligned} \mathrm{SP}^\ell_0(t) &= \mathrm{br}_t \qquad \text{and} \qquad \mathrm{P}^\ell_0(t) = \{w \mid \langle w, i, j\rangle \in \mathrm{SP}^\ell_0(t)\} \\ \mathrm{SP}^\ell_{n+1}(t) &= \{\langle w, i, j\rangle \in \mathrm{br}_t \mid \exists w_1 \in \mathrm{P}^\ell_n(t|_{wi})\colon |\mathrm{br}_t(wi, w_1) \cap \mathrm{P}^\ell_n(t)| \ge \ell^{n+1}, \\ &\qquad\qquad \exists w_2 \in \mathrm{P}^\ell_n(t|_{wj})\colon |\mathrm{br}_t(wj, w_2) \cap \mathrm{P}^\ell_n(t)| \ge \ell^{n+1}\} \\ \mathrm{P}^\ell_{n+1}(t) &= \{w \mid \langle w, i, j\rangle \in \mathrm{SP}^\ell_{n+1}(t)\} \ . \end{aligned}$$

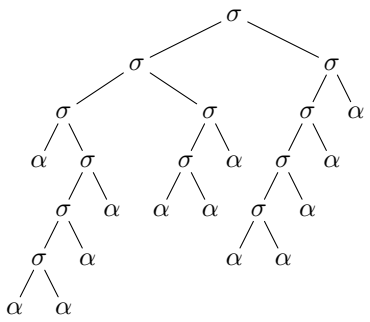

Figure 6: Tree used in Example 27.

Intuitively, all branching positions have level 0 (for any distance ℓ) and a branching position w has level $n+1$ if there are 2 paths in different direct subtrees below w that both have at least ℓ^{n+1} branching positions of level n along the path. Clearly, $\mathrm{SP}^\ell_{n+1}(t) \subseteq \mathrm{SP}^\ell_n(t)$ and $\mathrm{P}^\ell_{n+1}(t) \subseteq \mathrm{P}^\ell_n(t)$ for all $n \in \mathbb{N}$ and $t \in T_\Sigma$.

Example 27. *Let t be the tree depicted in Figure 6. Then*

$$\begin{aligned}
\mathrm{P}^2_0(t) &= \{\varepsilon, 1, 11, 112, 1121, 11211, 12, 121, 2, 21, 211, 2111\} = \mathrm{br}_t \\
\mathrm{P}^2_1(t) &= \{\varepsilon, 1\} \\
\mathrm{P}^2_2(t) &= \emptyset \ .
\end{aligned}$$

Lemma 28. Let $\varphi\colon T_\Gamma \to T_\Delta$ be a complete tree homomorphism. Moreover, let $t \in T_\Gamma$ and $\ell, n \in \mathbb{N}$. If $\langle \varepsilon, i, j\rangle \in \mathrm{SP}^\ell_n(t)$, then there exists $\langle v, i', j'\rangle \in \mathrm{SP}^\ell_n(\varphi(t))$ such that

- $v \in \mathrm{pos}(\varphi(t(\varepsilon)))$,
- $vi' \preceq v_1$ for some $v_1 \in \mathrm{pos}_{x_i}(\varphi(t(\varepsilon)))$, and
- $vj' \preceq v_2$ for some $v_2 \in \mathrm{pos}_{x_j}(\varphi(t(\varepsilon)))$.

Proof. Let $t = \gamma(t_1, \ldots, t_k)$ with $\gamma \in \Gamma_k$ and $t_1, \ldots, t_k \in T_\Gamma$, and let $u = \varphi(t)$. We prove the statement by induction on n. In the induction base, we have $n = 0$ and $\langle \varepsilon, i, j\rangle \in \mathrm{SP}^\ell_0(t)$. We arbitrarily select $v_1 \in \mathrm{pos}_{x_i}(\varphi(\gamma))$ and $v_2 \in \mathrm{pos}_{x_j}(\varphi(\gamma))$, which are positions of the variables x_i and x_j in $\varphi(\gamma)$, respectively. Since φ is complete, such positions exist. Let $v = \mathrm{lcp}(v_1, v_2)$ be the longest common prefix, and let $v_1 = vi'v'_1$ and $v_2 = vj'v'_2$. Clearly, $v \in \mathrm{pos}(\varphi(\gamma))$ and $\langle v, i', j'\rangle \in \mathrm{br}_{\varphi(t)}$ because v_1 and v_2 are in the distinct direct subtrees i' and j' below v. Trivially, $vi' \preceq v_1$ and $vj' \preceq v_2$, which completes the induction base.

In the induction step, let $\langle \varepsilon, i, j\rangle \in \mathrm{SP}^\ell_{n+1}(t)$ and $w_1 \in \mathrm{P}^\ell_n(t_i)$ and $w_2 \in \mathrm{P}^\ell_n(t_j)$ be the required positions of level n such that

$$|\mathrm{br}_t(i, w_1) \cap \mathrm{P}^\ell_n(t)| \geq \ell^{n+1} \qquad \text{and} \qquad |\mathrm{br}_t(j, w_2) \cap \mathrm{P}^\ell_n(t)| \geq \ell^{n+1} \ .$$

We observe that

$$u = \varphi(t) = \varphi(\gamma)[\varphi(t_1), \ldots, \varphi(t_k)] \ .$$

Now, we follow a similar approach as in the induction base. Since φ is complete, the tree u contains the subtrees $\varphi(t_i)$ and $\varphi(t_j)$ at any of the positions $v_1 \in \mathrm{pos}_{x_i}(\varphi(\gamma))$ and $v_2 \in \mathrm{pos}_{x_j}(\varphi(\gamma))$, respectively. As before, we let $v = \mathrm{lcp}(v_1, v_2) \in \mathrm{pos}(\varphi(\gamma))$ be the longest common prefix, and let $v_1 = vi'v'_1$ and $v_2 = vj'v'_2$. Clearly, $\langle v, i', j'\rangle \in \mathrm{br}_u$ is a branching position. It remains to show that $\langle v, i', j'\rangle \in \mathrm{SP}^\ell_{n+1}(u)$ because the remaining conditions of the lemma are already fulfilled.

Due to the fact that $w_1 \in \mathrm{P}^\ell_n(t_i)$, we have $\langle \varepsilon, i'', j''\rangle \in \mathrm{SP}^\ell_n(t_i|_{w_1})$ for some $i'', j'' \in \mathbb{N}$. Moreover, since φ is complete, the subtree $\varphi(t_i|_{w_1})$ occurs in $u|_{v_1}$. By the induction hypothesis, there exist $v''_1 \in \mathrm{P}^\ell_n(\varphi(t_i|_{w_1}))$. Moreover, $v_1v'v''_1 \in \mathrm{P}^\ell_n(u)$ for some $v' \in \mathrm{pos}(u|_{v_1})$ such that $u|_{v'} = \varphi(t_i|_{w_1})$. We will only verify the condition

$$|\mathrm{br}_u(vi', v'_1v'v''_1) \cap \mathrm{P}^\ell_n(u)| \geq \ell^{n+1} \tag{1}$$

because the proof for the second path works analogously.

Let $w_1' \in \mathrm{br}_t(i, w_1) \cap \mathrm{P}_n^\ell(t)$ be any position of level n along the path from i to iw_1 such that $w_1' \prec iw_1$. Let $i''' \in \mathbb{N}$ be the unique integer such that $w_1' i''' \preceq iw_1$. Since $w_1', iw_1 \in \mathrm{P}_n^\ell(t)$ we can conclude that there exist $i'', j'' \in \mathbb{N}$ such that $i''' \neq j''$ and that $\langle w_1', i'', j'' \rangle \in \mathrm{SP}_n^\ell(t)$. Then $\langle w_1', i''', j'' \rangle \in \mathrm{SP}_n^\ell(t)$ because $iw_1 \in \mathrm{P}_n^\ell(t)$. In other words, $\langle \varepsilon, i''', j'' \rangle \in \mathrm{SP}_n^\ell(t|_{w_1'})$. Since φ is complete, the translation $\varphi(t|_{w_1'})$ occurs in $\varphi(t_i)$. By the induction hypothesis, there exists $\langle v''', i_1, j_1 \rangle \in \mathrm{SP}_n^\ell(\varphi(t|_{w_1'}))$, which yields $\langle v'' v''', i_1, j_1 \rangle \in \mathrm{SP}_n^\ell(\varphi(t_i))$ for some position $v'' \in \mathrm{pos}(\varphi(t_i))$ with $\varphi(t_i)|_{v''} = \varphi(t_{w_1'})$, such that $v' v'' v''' i_1 \preceq v_1'$. Clearly,

$$v_1 v' v'' v''' \in \mathrm{br}_u(vi', v_1' v' v_1'') \cap \mathrm{P}_n^\ell(u) \ .$$

Moreover, the two conditions that $v''' \in \mathrm{pos}(\varphi(t(w_1')))$ and that φ is complete guarantee that for each selection of w_1' we obtain a different position $v_1 v' v'' v''' \in \mathrm{br}_u(vi', v_1' v' v_1'') \cap \mathrm{P}_n^\ell(u)$. This verifies (1), because there are at least ℓ^{n+1} possible selections of w_1'. □

Lemma 29. Let $\psi \colon T_\Gamma \to T_\Sigma$ be a linear tree homomorphism. Moreover, let $t \in T_\Gamma$ and $\ell, n \in \mathbb{N}$ be such that $\ell > \mathrm{ht}(\psi(\gamma'))$ for all symbols $\gamma' \in \Gamma$. If $\langle v, i', j' \rangle \in \mathrm{SP}_{n+1}^\ell(\psi(t))$ with $v \in \mathrm{pos}(\psi(t(\varepsilon)))$, then there exists $\langle \varepsilon, i, j \rangle \in \mathrm{SP}_n^\ell(t)$ such that

- $vi' \preceq v_1$ for some $v_1 \in \mathrm{pos}_{x_i}(\psi(t(\varepsilon)))$ and
- $vj' \preceq v_2$ for some $v_2 \in \mathrm{pos}_{x_j}(\psi(t(\varepsilon)))$.

Proof. Let $t = \gamma(t_1, \ldots, t_k)$ with $\gamma \in \Gamma_k$ and $t_1, \ldots, t_k \in T_\Gamma$, and let $u = \psi(t)$ be its image. Moreover, let $\langle v, i', j' \rangle \in \mathrm{SP}_{n+1}^\ell(u)$ with $v \in \mathrm{pos}(\psi(\gamma))$. By definition, there exist long paths in u starting at vi' and vj', which are longer than ℓ, which in turn is longer than any path in $\psi(\gamma')$ for every $\gamma' \in \Gamma$. Consequently, there exist $1 \leq i, j \leq k$ and

$$v_1 \in \mathrm{pos}_{x_i}(\psi(\gamma)) \qquad \text{and} \qquad v_2 \in \mathrm{pos}_{x_j}(\psi(\gamma))$$

such that $vi' \preceq v_1$ and $vj' \preceq v_2$. Since ψ is linear, we know that $i \neq j$. It remains to prove that $\langle \varepsilon, i, j \rangle \in \mathrm{SP}_n^\ell(t)$ as the additional properties of the lemma are already satisfied. We prove this remaining property by induction on n, and the induction base is already proven as $\langle \varepsilon, i, j \rangle \in \mathrm{br}_t$.

We proceed with the induction step. By definition, there exist positions $v_1' \in \mathrm{P}_n^\ell(u|_{vi'})$ and $v_2' \in \mathrm{P}_n^\ell(u|_{vj'})$ such that

$$|\mathrm{br}_u(vi', v_1') \cap \mathrm{P}_n^\ell(u)| \geq \ell^{n+1} \qquad \text{and} \qquad |\mathrm{br}_u(vj', v_2') \cap \mathrm{P}_n^\ell(u)| \geq \ell^{n+1} \ .$$

Let $w_1 \in \mathrm{pos}(t_i)$ be the position that creates the symbol $u(vi'v_1')$. We only prove the required property for the path $\mathrm{br}_t(i, w_1)$.

Since w_1 creates the symbol $u(vi'v_1')$, there exists a position $v'' \in \mathrm{pos}(u)$ such that $vi' \preceq v'' \preceq vi'v_1'$ and $u|_{v''} = \psi(t|_{w_1})$. Moreover, by $v_1' \in \mathrm{P}_n^\ell(u|_{vi'})$ there exists $\langle v_1'', i_1', j_1' \rangle \in \mathrm{SP}_n^\ell(u|_{v''})$ with $vi'v_1' = v''v_1''$, which yields that $v_1'' \in \mathrm{pos}(\psi(t(w_1)))$. With the help of the induction hypothesis we conclude the existence of $\langle \varepsilon, i_1, j_1 \rangle \in \mathrm{SP}_n^\ell(t|_{w_1})$, and thus $w_1 \in \mathrm{P}_n^\ell(t)$. Finally, for every $v' \in \mathrm{br}_u(vi', v_1') \cap \mathrm{P}_n^\ell(u)$. Let $w' \in \mathrm{pos}(t_i)$ with $i \preceq w' \preceq w_1$ be the position that creates the symbol $u(v')$, and let $v'' \in \mathrm{pos}(u)$ be the position such that $vi' \preceq v'' \preceq v'$ and $u|_{v''} = \psi(t|_{w'})$. Since $v' \in \mathrm{P}_n^\ell(u)$ there exists $\langle v_1'', i_1', j_1' \rangle \in \mathrm{SP}_n^\ell(u|_{v''})$ with $v' = v''v_1''$, which yields that $v_1'' \in \mathrm{pos}(\psi(t(w')))$. Moreover, let $i_1'' \in \mathbb{N}$ be the unique integer such that $v'i_1'' \preceq vi'v_1'$. Consequently, $\langle v_1'', i_1'', j_1' \rangle \in \mathrm{SP}_n^\ell(u|_{v''})$ because $vi'v_1' \in \mathrm{P}_n^\ell(u)$. By the induction hypothesis, there exists $\langle \varepsilon, i_1, j_1 \rangle \in \mathrm{SP}_n^\ell(t|_{w'})$ such that

- $v_1'' i_1'' \preceq v_1'''$ for some $v_1''' \in \mathrm{pos}_{x_{i_1}}(\psi(t(w')))$ and
- $v_1'' j_1' \preceq v_2'''$ for some $v_2''' \in \mathrm{pos}_{x_{j_1}}(\psi(t(w')))$.

Moreso, we have $w' \in \mathrm{br}_t(i, w_1) \cap \mathrm{P}_n^\ell(t)$, and $\psi(t|_{w'i_1}) = u|_{v'''}$ for some position $vi' \preceq v''' \preceq vi'v_1'$, which determines another position of $\mathrm{br}_u(vi', v_1')$. Since at most ℓ positions of $\mathrm{br}_u(vi', v_1')$ can be created by a single symbol of t and $|\mathrm{br}_u(v_1, v_1') \cap \mathrm{P}_n^\ell(u)| \geq \ell^{n+1} - \ell + 1 = (\ell^n - 1 + \ell^{-1})\ell$. Consequently, we obtain at least ℓ^n positions in $\mathrm{br}_t(i, w_1) \cap \mathrm{P}_n^\ell(t)$, which completes the induction step and the proof. □

Next, we combine the previous two lemmas into the main statement that will be used to prove the infinity of several composition hierarchies. In essence, we show that a bimorphism in $\mathrm{B}(\mathrm{l}, \mathrm{c})$ can reduce the level by at most 1 (while keeping the distance ℓ).

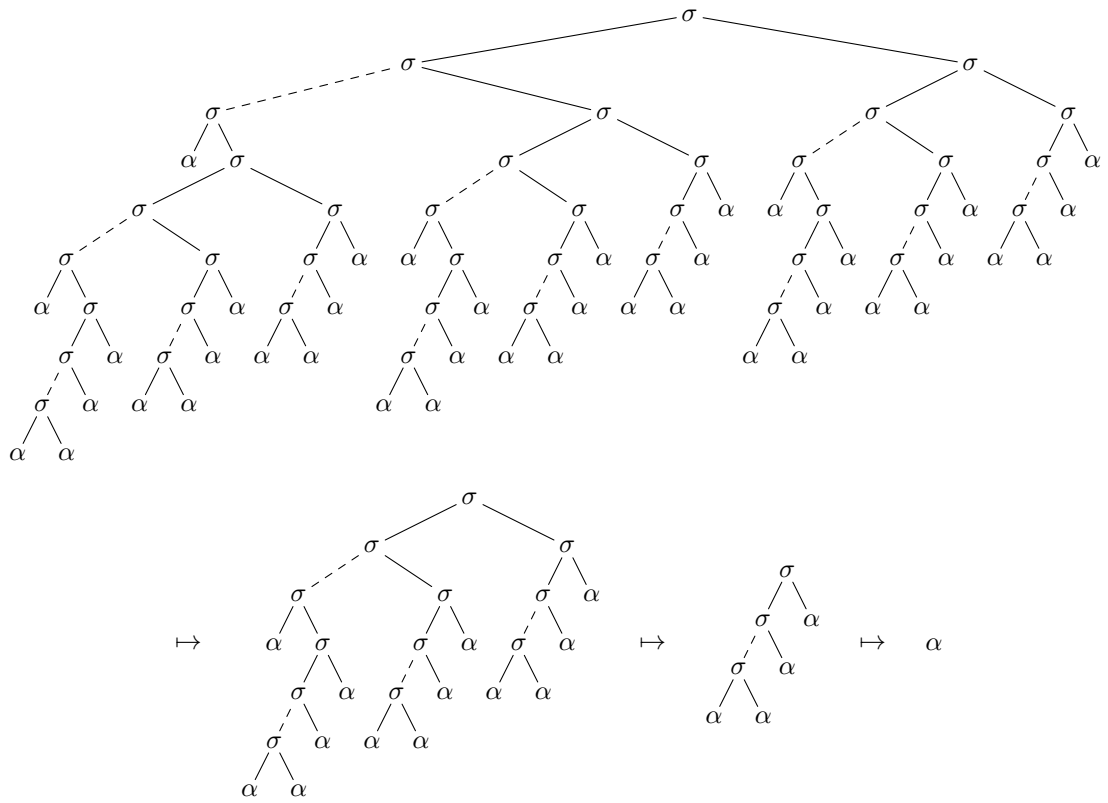

Figure 7: Illustration of the 3-fold composition of the tree transformation computed by the ∉nl-xt M of Example 32.

Theorem 30. Let $B = (\psi, L, \varphi)$ be a bimorphism such that $\psi\colon T_\Gamma \to T_\Sigma$ is linear and $\varphi\colon T_\Gamma \to T_\Delta$ is complete. Moreover, let $(t, u) \in B$, and let $\ell \in \mathbb{N}$ be such that $\ell > \mathrm{ht}(\psi(\gamma))$ for every $\gamma \in \Gamma$. If $\mathrm{P}^\ell_{n+1}(t) \neq \emptyset$, then $\mathrm{P}^\ell_n(u) \neq \emptyset$.

Proof. Since $(t, u) \in B$, there exists $s \in L$ such that $\psi(s) = t$ and $\varphi(s) = u$. By assumption, we have that $\mathrm{P}^\ell_{n+1}(\psi(s)) \neq \emptyset$, which by Lemma 29 yields that $\mathrm{P}^\ell_n(s) \neq \emptyset$. Using the completeness of φ and Lemma 28, we obtain that $\mathrm{P}^\ell_n(u) \neq \emptyset$ as desired. □

Now we can simply chain Theorem 30 to show that an n-fold composition of tree transformations of $\mathrm{B}(\mathrm{l}, \mathrm{c})$ can decrease the level by at most n (for a suitable distance ℓ).

Corollary 31 (of Theorem 30). Let $n \geq 1$, and for every $1 \leq i \leq n$, let $B_i = (\psi_i, L_i, \varphi_i)$ be a bimorphism such that ψ_i is linear and φ_i is complete. Moreover, let $\varphi_i\colon T_{\Gamma_i} \to T_{\Delta_i}$ and $\psi_{i+1}\colon T_{\Gamma_{i+1}} \to T_{\Delta_i}$ for every $1 \leq i < n$. Finally, let $\ell \in \mathbb{N}$ be such that $\ell > \mathrm{ht}(\psi_i(\gamma))$ for every $1 \leq i \leq n$ and $\gamma \in \Gamma_i$, and let $(t, u) \in B_1 \,;\cdots;\, B_n$. If $\mathrm{P}^\ell_{n+1}(t) \neq \emptyset$, then $\mathrm{P}^\ell_1(u) \neq \emptyset$.

It remains to demonstrate a tree transformation that can be computed by $(n+1)$ ∉nl-xt and that reduces the level of positions from $n + 1$ to 0. Clearly, this tree transformation cannot be computed by an n-fold composition of tree transformations from $\mathrm{B}(\mathrm{l}, \mathrm{c})$ because the output tree should contain a position of level 1 by Corollary 31. We make sure that the assumptions of Corollary 31 are satisfied.

Example 32. *Let $M = (Q, \Sigma, \Sigma, \{\star\}, R)$ be the ∉nl-xt such that*

- *$Q = \{\star, q\}$ and $\Sigma = \{\sigma^{(2)}, \alpha^{(0)}\}$, and*

- *R contains exactly the following rules*

$$\sigma(\star,\alpha) \xrightarrow{\star,q} \star \qquad \sigma(\star,q) \xrightarrow{\star,q} \sigma(\star,q) \qquad \alpha \xrightarrow{\star} \alpha$$

The 3-fold composition $M\,;M\,;M$ can compute the transformation indicated in Figure 7.

We use the tree transformation computed by the ⧸nl-xt M of Example 32, and show that n transformations from B(l, c) cannot compute the tree transformation M^{n+1}.

Lemma 33. For every $n \geq 1$,

$$\not\text{nl-XT}^{n+1} \not\subseteq \text{B(l,c)}^n \ .$$

Proof. Let $L_0 = \{\alpha\}$. For every $i \geq 1$, let C_i and L_i be the smallest sets such that
- $\sigma(x_1, t) \in C_i$ for every $t \in L_{i-1}$ and $c[c'] \in C_i$ for all $c, c' \in C_i$, and
- $L_i = \{c[\alpha] \mid c \in C_i\}$.

It is easy to see that $n+1$ compositions of the tree transformation M computed by the ⧸nl-xt M of Example 32 can relate L_{n+2} and L_1 (see Figure 7); more precisely, for every $t \in L_{n+2}$ there exists $u \in L_1$ such that $(t,u) \in M^{n+1}$. With the help of Corollary 31 we can complete the proof if $\mathrm{P}^{\ell}_{n+1}(t) \neq \emptyset$ for some $t \in L_{n+2}$ and suitably large $\ell \in \mathbb{N}$.

Let $\ell \in \mathbb{N}$ be fixed. Thus, we now prove that for every $n \in \mathbb{N}$ there exists a tree $t \in L_{n+1}$ such that $\mathrm{P}^{\ell}_{n}(t) \neq \emptyset$ by induction on n. In fact, we prove the stronger statement that there exists $t \in L_{n+1}$ and $w \in \mathrm{pos}(t)$ such that $|\mathrm{br}_t(\varepsilon, w) \cap \mathrm{P}^{\ell}_{n}(t)| \geq \ell^{n+1}$. For $n = 0$, we select the tree $c^{\ell}[\alpha] \in L_1$, where $c = \sigma(x_1, \alpha)$, and the position $w = 1^{\ell-1}$. Since $\mathrm{P}^{\ell}_{0}(t) = \mathrm{br}_t(\varepsilon, w)$, the selection of t and w fulfills the requirements. In the induction step, there exist a tree $t \in L_{n+1}$ and $w \in \mathrm{pos}(t)$ such that $|\mathrm{br}_t(\varepsilon, w) \cap \mathrm{P}^{\ell}_{n}(t)| \geq \ell^{n+1}$. We consider the tree $t' = c^{\ell^{n+2}}[c[\alpha]]$ with $c = \sigma(x_1, t)$ and the position $w' = 1^{\ell^{n+2}-1}$. Obviously, $t' \in L_{n+2}$ and $w'' \in \mathrm{P}^{\ell}_{n+1}(t')$ for every $w'' \preceq w'$ (via the paths $12w$ and $2w$), which completes our induction and proof. □

Now we are able to prove that the composition hierarchy of ⧸nl-XT and several other classes is infinite.

Theorem 34. For every $n \geq 1$,

$$\text{B(l,c)}^n \subsetneq \text{B(l,c)}^{n+1} \qquad \text{[s][n]l-XT}^n \subsetneq \text{[s][n]l-XT}^{n+1}$$
$$\not\text{nl-XT}^n \subsetneq \not\text{nl-XT}^{n+1} \qquad (\text{[s][n]l-XT}^{\mathrm{R}})^n \subsetneq (\text{[s][n]l-XT}^{\mathrm{R}})^{n+1} \ .$$

Proof. First, we note that all inclusions are trivial. Thus, we only need to argue the strictness. By [19, Theorem 17] we have $\text{B(lcs, lc)} = \not\text{nl-XT}$, hence

$$\not\text{nl-XT}^n \subseteq \text{B(l,c)}^n \qquad \text{and} \qquad \not\text{nl-XT}^{n+1} \subseteq \text{B(l,c)}^{n+1} \ .$$

These facts together with Lemma 33 imply the strictness of the two inclusions on the left. Moreover,

$$\cdots \subseteq (\text{[s][n]l-XT}^{\mathrm{R}})^n \subseteq \text{[s][n]l-XT}^{n+1} \subseteq (\text{[s][n]l-XT}^{\mathrm{R}})^{n+1} \subseteq \text{[s][n]l-XT}^{n+2} \subseteq \cdots$$

for all $n \geq 1$, which shows that the composition hierarchy of [s][n]l-XT is infinite if and only if the composition hierarchy of $\text{[s][n]l-XT}^{\mathrm{R}}$ is infinite. Thus, it remains to show the latter property.

Using simple symmetry, we observe that $\text{snl-XT} = \not\text{nl-XT}^{-1}$, which together with the symmetric version of Lemma 33 yields $\text{snl-XT}^{n+1} \not\subseteq \text{B(c,l)}^n$. Furthermore, $\text{B(lc, l)} = \text{l-XT}^{\mathrm{R}}$ by [19, Theorem 4], so naturally,

$$(\text{[s][n]l-XT}^{\mathrm{R}})^n \subseteq \text{B(c, l)}^n \qquad \text{and} \qquad \text{snl-XT}^{n+1} \subseteq (\text{[s][n]l-XT}^{\mathrm{R}})^{n+1}$$

for all $n \geq 1$. Taken together with $\text{snl-XT}^{n+1} \not\subseteq \text{B(c,l)}^n$ we obtain the desired strictness. □

Table 4 summarizes our results of this section. For the sake of completeness, we mention some additional results from the literature, where T stands for the class of all tree transformations computable by top-down tree transducers [7], and ⧸-XT stands for the class of tree transformations computable by ε-free extended top-down tree transducers [16]. The mentioned result $\not\text{-XT} \subseteq \mathrm{T}^2$ can be concluded from [16, Theorem 4.8].

Class with infinite composition hierarchy	Proved in
B(l,c), $\not\in$ln-XT, [s][n]l-XT, [s][n]l-XT^{R}	Theorem 34
T	[9, Theorem 3.14]
$\not\in$-XT	$\not\in$-XT $\subseteq \mathrm{T}^2$ and $(\mathrm{T}^n \mid n \geq 1)$ is infinite
B(c,c)	[1] and [6, Section II-2-2-3-4]

Table 4: Summary of the results of Section 6.

References

[1] Arnold, A., Dauchet, M.: Transductions inversibles de forêts. Thèse 3ème cycle M. Dauchet, Université de Lille (1975)
[2] Arnold, A., Dauchet, M.: Bi-transductions de forêts. In: ICALP. pp. 74–86. Edinburgh University Press (1976)
[3] Arnold, A., Dauchet, M.: Morphismes et bimorphismes d'arbres. Theoret. Comput. Sci. 20(1), 33–93 (1982)
[4] Baker, B.S.: Composition of top-down and bottom-up tree transductions. Inform. and Control 41(2), 186–213 (1979)
[5] Chiang, D.: An introduction to synchronous grammars. In: ACL. Association for Computational Linguistics (2006), part of a tutorial given with K. Knight
[6] Dauchet, M.: Transductions de forêts bimorphismes de magmoïdes. Première thèse, Université de Lille (1977)
[7] Engelfriet, J.: Bottom-up and top-down tree transformations—a comparison. Math. Systems Theory 9(3), 198–231 (1975)
[8] Engelfriet, J.: Top-down tree transducers with regular look-ahead. Math. Systems Theory 10(1), 289–303 (1977)
[9] Engelfriet, J.: Three hierarchies of transducers. Math. Systems Theory 15(2), 95–125 (1982)
[10] Engelfriet, J., Maneth, S.: Macro tree translations of linear size increase are MSO definable. SIAM J. Comput. 32(4), 950–1006 (2003)
[11] Fülöp, Z., Maletti, A., Vogler, H.: Preservation of recognizability for synchronous tree substitution grammars. In: ATANLP. pp. 1–9. Association for Computational Linguistics (2010)
[12] Fülöp, Z., Maletti, A., Vogler, H.: Weighted extended tree transducers. Fundam. Inform. 111(2), 163–202 (2011)
[13] Fülöp, Z., Vogler, H.: Syntax-Directed Semantics—Formal Models Based on Tree Transducers. EATCS Monographs on Theoret. Comput. Sci., Springer (1998)
[14] Gécseg, F., Steinby, M.: Tree Automata. Akadémiai Kiadó, Budapest (1984)
[15] Gécseg, F., Steinby, M.: Tree languages. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, vol. 3, chap. 1, pp. 1–68. Springer (1997)
[16] Graehl, J., Hopkins, M., Knight, K., Maletti, A.: The power of extended top-down tree transducers. SIAM J. Comput. 39(2), 410–430 (2009)
[17] Graehl, J., Knight, K., May, J.: Training tree transducers. Comput. Linguist. 34(3), 391–427 (2008)
[18] Knight, K., Graehl, J.: An overview of probabilistic tree transducers for natural language processing. In: CICLing. LNCS, vol. 3406, pp. 1–24. Springer (2005)
[19] Maletti, A.: Compositions of extended top-down tree transducers. Inform. and Comput. 206(9–10), 1187–1196 (2008)
[20] Maletti, A.: Tree transformations and dependencies. In: MOL. LNAI, vol. 6878, pp. 1–20. Springer (2011)
[21] May, J., Knight, K., Vogler, H.: Efficient inference through cascades of weighted tree transducers. In: ACL. pp. 1058–1066. Association for Computational Linguistics (2010)
[22] Rounds, W.C.: Mappings and grammars on trees. Math. Systems Theory 4(3), 257–287 (1970)
[23] Thatcher, J.W.: Generalized^2 sequential machine maps. J. Comput. System Sci. 4(4), 339–367 (1970)