TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik

TUD-FI05-05 — October 2005

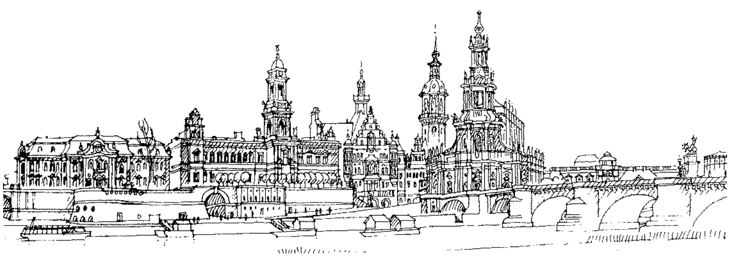
A. Maletti

Institute of Theoretical Computer Science

Compositions of Tree Series Transformations

Technische Berichte
Technical Reports

ISSN 1430-211X



Technische Universität Dresden
Fakultät Informatik
D-01062 Dresden
Germany

URL: http://www.inf.tu-dresden.de/

Compositions of Tree Series Transformations*

Andreas Maletti†

Technische Universität Dresden
Fakultät Informatik
D–01062 Dresden, Germany
e-mail: maletti@tcs.inf.tu-dresden.de

Abstract

Tree series transformations computed by bottom-up and top-down tree series transducers are called bottom-up and top-down tree series transformations, respectively. (Functional) compositions of such transformations are investigated. It turns out that the class of bottom-up tree series transformations over a commutative and complete semiring is closed under left-composition with linear bottom-up tree series transformations and right-composition with boolean deterministic bottom-up tree series transformations.

Moreover, it is shown that the class of top-down tree series transformations over a commutative and complete semiring is closed under right-composition with linear, nondeleting top-down tree series transformations. Finally, the composition of a boolean, deterministic, total top-down tree series transformation with a linear top-down tree series transformation is shown to be a top-down tree series transformation.

1 Introduction

Tree series transducers [21, 10, 15] were introduced as the transducing devices corresponding to weighted tree automata [2, 19, 4]. So far, the latter are applied in code selection and tree pattern matching [13, 3]. Weighted transducers on strings are applied in image manipulation [see, *e. g.*, 8], where the images are coded as weighted string automata, and speech processing [see, *e. g.*, 24]. Since natural language processing features many transformations on parse trees, which come equipped with a degree of certainty, it seems natural to consider finite-state devices capable of transforming weighted trees. For natural language processing, the potential of tree series transducers over the semiring of the positive real numbers was recently discovered [17].

Let us explain the scenario of natural language processing in some more detail. A tree bank is a collection of parse trees (of natural language sentences) each annotated with a weight (usually the relative frequency). When translating a natural language sentence from one language into another, we first have to parse the original sentence in order to obtain a parse tree. Since natural language is usually ambiguous we obtain a collection of parse trees each annotated with a probability. The probability is derived from the evidence found in the tree bank. Now the transformation stage translates the annotated parse trees into parse trees of the output language. Again there may be more than one possible translation for one parse tree, so that for each input parse tree we obtain a collection of annotated output parse trees. A tree bank containing parse trees of sentences in the output languages delivers the coefficients required to compute the probability.

Such collections of annotated parse trees are formal tree series; *i. e.*, mappings from a set of trees into a semiring. The translation stage can then be seen as a transformation which transforms tree series into tree series. Tree series transducers are finite-state devices computing such tree-series-to-tree-series transformations.

*This is an extended and revised version of: Andreas Maletti, "Compositions of bottom-up tree series transformations", Proc. 11th Int. Conf. Automata and Formal Languages, University of Szeged, 187–199, 2005

†Financially supported by the German Research Foundation (DFG, GK 334/3).

The complexity of the transformations involved in the translation stage is usually high (automata requiring several million states), so that modularity is of utmost importance. One designs small transducers that only deal with one phenomenon and then composes the transformations (*i. e.*, uses the output of the first transformation as the input of a second transformation) to obtain the final result. However, this approach is usually inefficient because many intermediate results are computed. By composing the transducers we can avoid these intermediate results. Moreover, the analysis of a single transducer is usually simpler than the analysis of a series of transducers. For example, an important problem in natural language processing is finding the most likely path (*i. e.*, the path that generates the highest probability) outputting a given parse tree. This problem is very difficult for compositions of transformations, so that composing the transducers that compute the transformations helps to reduce the complexity.

Since tree series transducers generalize tree transducers [26, 25, 9] by adding a cost component, we obtain top-down tree series transducers [21, 10, 15], where the input tree is processed from the root towards the leaves, and bottom-up tree series transducers [10, 15], where the input is processed from the leaves towards the root. In this paper, we deal with compositions of the transformations computed by both types of tree series transducers. Moreover, four notions of substitution on tree series are known. These are pure IO-substitution [6, 10], o-IO-substitution [15], [IO]-substitution [7], and OI-substitution [5, 21]. Here we deal with pure IO-substitution, since it seems to be the most appropriate choice for bottom-up tree series transducers (for top-down tree series transducers the choice of substitution is irrelevant).

Roughly speaking, a (bottom-up or top-down) tree series transducer is a (bottom-up or top-down) tree transducer [26, 25] in which the transitions carry a weight; a weight is an element of some semiring [18, 16]. The rewrite semantics works as follows. Along a successful computation on some input tree, the weights of the involved transitions are combined by means of the semiring multiplication; if there is more than one successful computation for some pair of input and output trees, then the weights of these computations are combined by means of the semiring addition.

In the unweighted case, bottom-up tree transformations are closed under left-composition with linear bottom-up tree transformations [9, Theorem 4.5] and right-composition with deterministic bottom-up tree transformations [9, Theorem 4.6] (see also [1, Theorem 6]). In this paper we try to extend these results to bottom-up tree series transformations. The first result was already generalized to bottom-up tree series transformations [21, 10]. Essentially the authors obtain that, for arbitrary commutative and complete semirings [18], bottom-up tree series transformations are closed under left-composition with nondeleting, linear bottom-up tree series transformations. We generalize this further by showing that the mentioned class of bottom-up tree series transformations is even closed under left-composition with linear bottom-up tree series transformations. The construction required to show this statement is mostly standard (*i. e.*, the transitions of the linear transducer are translated with the help of the second transducer) with one notable exception.

For commutative and complete semirings, the class of bottom-up tree series transformations is closed under right-composition with boolean homomorphism bottom-up tree series transformations [10, Corollary 5.5]. Using an adaptation of the standard construction, we also show that this class of bottom-up tree series transformations is actually closed under right-composition with boolean, deterministic bottom-up tree series transformations.

In the top-down case, we have that the class of top-down tree transformations is closed under right-composition with nondeleting, linear top-down tree transformations [1, Theorem 1]. Moreover, it is closed under left-composition with deterministic, total tree transformations [26, 25] (see also [1, Theorem 1]). These results were generalized for deterministic tree series transducers by [10, Theorem 5.18]. They showed that, for every commutative and complete semiring, the class of deterministic top-down tree series transformations is closed under right-composition with nondeleting, linear, deterministic tree series transformations and under left-composition with boolean, deterministic, total tree series transformations. We present a generalization of the former statement and a statement similar to the latter. More precisely, we show that the class of top-down tree series transformations is closed under right-composition with nondeleting, linear top-down tree series transformations. Secondly, we show that the composition of a boolean, deterministic, total top-down tree series transformation with a linear top-down tree series transformation is a top-down tree series transformation.

Together with this introduction the paper has 5 sections. Section 2 recalls general notions and notations. In particular, the definition of tree series transducers is presented. In Section 3 pure substitution is investigated with respect to basic properties such as distributivity, linearity, and associativity. Section 4 presents the composition results for bottom-up tree series transducers and Section 5 deals with compositions of top-down tree series transducers.

2 Preliminaries

We use $\mathbb{N}$ to represent the set of nonnegative integers $\{0, 1, 2, \ldots\}$, and we also use $\mathbb{N}_+ = \mathbb{N} \setminus \{0\}$. In the sequel, let $k, n \in \mathbb{N}$. We abbreviate $\{ i \in \mathbb{N} \mid 1 \leqslant i \leqslant k \}$ simply by $[k]$. Given sets A and I, we write A^I for the set of all mappings $f\colon I \longrightarrow A$. Occasionally, we use the family notation $(f(i))_{i \in I}$ for f, and moreover, if $I = [k]$, then we generally write $(f(1), \ldots, f(k))$ or just $f(1) \cdots f(k)$. A set Σ which is nonempty and finite is also called *alphabet*, and the elements thereof are called *symbols*. We use $\Sigma^* = \bigcup_{n \in \mathbb{N}} \Sigma^n$ for the set of all *words (over* Σ*)*. Given a word $w \in \Sigma^*$, we write $|w|$ for the unique $n \in \mathbb{N}$, also called *length* of w, such that $w \in \Sigma^n$.

Let A be a set. A *partition of* A is a family $(A_i)_{i \in I}$ of $A_i \subseteq A$ for some index set I such that: (i) $\bigcup_{i \in I} A_i = A$ and (ii) for every $i, j \in I$ with $i \neq j$ we have $A_i \cap A_j = \emptyset$. (Note that we do not require that $A_i \neq \emptyset$ for every $i \in I$.)

2.1 Trees

A *ranked alphabet* is an alphabet Σ together with a mapping $\mathrm{rk}_\Sigma\colon \Sigma \longrightarrow \mathbb{N}$ associating to each symbol its *rank*. We use the denotation Σ_k to represent the set of symbols (of Σ) having rank k; *i. e.*, $\Sigma_k = \{ \sigma \in \Sigma \mid \mathrm{rk}_\Sigma(\sigma) = k \}$. Furthermore, we use the sets $\mathrm{X} = \{ \mathrm{x}_i \mid i \in \mathbb{N}_+ \}$ and $\mathrm{Z} = \{ \mathrm{z}_i \mid i \in \mathbb{N}_+ \}$ of *(formal) variables* and the finite subsets $\mathrm{X}_k = \{ \mathrm{x}_i \mid i \in [k] \}$ and $\mathrm{Z}_k = \{ \mathrm{z}_i \mid i \in [k] \}$. Given a ranked alphabet Σ and $V \subseteq \mathrm{X} \cup \mathrm{Z}$, the set of Σ*-trees indexed by* V, denoted by $T_\Sigma(V)$, is inductively defined to be the smallest set T such that (i) $V \subseteq T$ and (ii) for every $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, and $t_1, \ldots, t_k \in T$ also $\sigma(t_1, \ldots, t_k) \in T$. Since we generally assume that $\Sigma \cap (\mathrm{X} \cup \mathrm{Z}) = \emptyset$, we write α instead of $\alpha()$ whenever $\alpha \in \Sigma_0$. Moreover, we also write T_Σ to denote $T_\Sigma(\emptyset)$.

We use variables of X to represent input trees and variables of Z to represent output trees. In particular, we never mix variables of X and Z; *i. e.*, any tree $t \in T_\Sigma(V)$ that we consider is either in $T_\Sigma(\mathrm{X})$ or $T_\Sigma(\mathrm{Z})$. So let (i) $V = \mathrm{X}$ and $v = \mathrm{x}$ or (ii) $V = \mathrm{Z}$ and $v = \mathrm{z}$. For every $t \in T_\Sigma(V)$, we denote by $|t|_i$ the number of occurrences of v_i in t, and in addition, we use $\mathrm{var}(t) = \{ i \in \mathbb{N}_+ \mid |t|_i \geqslant 1 \}$. Moreover, for every finite $I \subseteq \mathbb{N}_+$ and family $(t_i)_{i \in I}$ of $t_i \in T_\Sigma(V)$, the expression $t[t_i]_{i \in I}$ denotes the result of substituting in t every v_i by t_i for every $i \in I$. If $I = [n]$, then we simply write $t[t_1, \ldots, t_n]$. Let $I \subseteq \mathbb{N}_+$ be finite. We say that $t \in T_\Sigma(V)$ is *linear in* I (respectively, *nondeleting in* I), if v_i occurs at most once (respectively, at least once) in t for every $i \in I$.

Any subset $L \subseteq T_\Sigma(V)$ is called *tree language*. We define $\mathrm{var}(L) = \bigcup_{t \in L} \mathrm{var}(t)$ for every $L \subseteq T_\Sigma(V)$. Tree languages $L_1, L_2 \subseteq T_\Sigma(V)$ are called *variable-disjoint*, if $\mathrm{var}(L_1) \cap \mathrm{var}(L_2) = \emptyset$. Let $I \subseteq \mathbb{N}_+$ be finite and $L, L_i \subseteq T_\Sigma(V)$ for every $i \in I$. We lift substitution to tree languages by stating that $L[L_i]_{i \in I} = \{ t[t_i]_{i \in I} \mid t \in L, (\forall i \in I)\colon t_i \in L_i \}$.

2.2 Semirings

A *semiring* is an algebraic structure $\mathcal{A} = (A, +, \cdot, 0, 1)$ consisting of a commutative monoid $(A, +, 0)$ and a monoid $(A, \cdot, 1)$ such that (i) $\cdot$ distributes over $+$ and (ii) 0 is absorbing with respect to $\cdot$. The semiring is called commutative, if $\cdot$ is commutative. We say that $a \in A$ is *multiplicatively idempotent*, if $a^2 = a$. Clearly, the neutral elements 0 and 1 are always multiplicatively idempotent. As usual we use $\sum_{i \in I} a_i$ (respectively, $\prod_{i \in I} a_i$ for $I \subseteq \mathbb{N}$) for sums (respectively, products) of families $(a_i)_{i \in I}$ of $a_i \in A$ where for only finitely many $i \in I$ we have $a_i \neq 0$ (respectively, $a_i \neq 1$).

For products the order of the factors is given by the order $0 < 1 < \cdots$ on the index set I. We say that $\mathcal{A}$ is *complete*, whenever it is possible to define an infinitary sum operation $\sum_I$ for each index set I such that for every family $(\, a_i \,)_{i \in I}$ of $a_i \in A$ the following three conditions are satisfied.

(i) $\sum_I (\, a_i \,)_{i \in I} = a_j$, if $I = \{j\}$, and $\sum_I (\, a_i \,)_{i \in I} = a_{j_1} + a_{j_2}$, if $I = \{j_1, j_2\}$ with $j_1 \neq j_2$.

(ii) $\sum_I (\, a_i \,)_{i \in I} = \sum_J \big(\sum_{I_j} (\, a_i \,)_{i \in I_j} \big)_{j \in J}$ for all partitions $(\, I_j \,)_{j \in J}$ of I.

(iii) $\sum_I (\, a \cdot a_i \cdot a' \,)_{i \in I} = a \cdot \big(\sum_I (\, a_i \,)_{i \in I} \big) \cdot a'$ for all $a, a' \in A$.

In the sequel, we simply write the accustomed $\sum_{i \in I} a_i$ instead of the cumbersome $\sum_I (\, a_i \,)_{i \in I}$, and when speaking about a complete semiring, we implicitly assume $\sum_I$ to be given. For the rest of the paper, let $\mathcal{A} = (A, +, \cdot, 0, 1)$ be a commutative semiring with infinite summation $\sum_I$ such that $\mathcal{A}$ is complete with respect to $\sum_I$. Well-known complete semirings are the Boolean semiring $\mathbb{B} = (\{\bot, \top\}, \vee, \wedge, \bot, \top)$ with disjunction and conjunction and the semiring of the nonnegative real numbers $\mathbb{R}_+ = (\mathbb{R}_+ \cup \{\infty\}, +, \cdot, 0, 1)$.

2.3 Tree Series

Let S be a set. A *(formal) power series* φ is a mapping $\varphi \colon S \longrightarrow A$. Given $s \in S$, we denote $\varphi(s)$ also by (φ, s) and write φ as $\sum_{s \in S} (\varphi, s)\, s$. The *support* of φ is $\mathrm{supp}(\varphi) = \{\, s \in S \mid (\varphi, s) \neq 0 \,\}$. Power series with finite support are called *polynomials*, and power series with at most one support element are also called *monomials*. We denote the set of all power series $\varphi \colon S \longrightarrow A$ by $A\langle\!\langle S \rangle\!\rangle$. We call $\varphi \in A\langle\!\langle S \rangle\!\rangle$ *boolean*, if $(\varphi, s) = 1$ for every $s \in \mathrm{supp}(\varphi)$. The boolean monomial with empty support is denoted by $\widetilde{0}$. Power series $\varphi, \varphi' \in A\langle\!\langle S \rangle\!\rangle$ are summed componentwise; *i. e.*, $(\varphi + \varphi', s) = (\varphi, s) + (\varphi', s)$ for every $s \in S$. Finally, we also multiply the power series φ with a coefficient $a \in A$ componentwise; *i. e.*, $(a \cdot \varphi, s) = a \cdot (\varphi, s)$ for every $s \in S$.

In this paper, we only consider power series in which the set S is a set of trees. Such power series are also called *tree series*. A tree series $\varphi \in A\langle\!\langle T_\Sigma(V) \rangle\!\rangle$ is said to be *linear* (respectively, *nondeleting*) *in* $I \subseteq \mathbb{N}_+$, if every $t \in \mathrm{supp}(\varphi)$ is linear (respectively, nondeleting) in I. Finally, $\mathrm{var}(\varphi) = \bigcup_{t \in \mathrm{supp}(\varphi)} \mathrm{var}(t)$.

Let Δ be a ranked alphabet. Moreover, let $\varphi \in A\langle\!\langle T_\Delta(\mathrm{Z}) \rangle\!\rangle$, $I \subseteq \mathbb{N}_+$ be finite, and $\psi_i \in A\langle\!\langle T_\Delta(\mathrm{Z}) \rangle\!\rangle$ for every $i \in I$. The *pure tree series substitution* (for short: pure substitution) *(of* $(\, \psi_i \,)_{i \in I}$ *into* φ*)* [6, 10], denoted by $\varphi \longleftarrow (\, \psi_i \,)_{i \in I}$, is defined by

$$\varphi \longleftarrow (\, \psi_i \,)_{i \in I} = \sum_{\substack{t \in T_\Delta(\mathrm{Z}), \\ (\forall i \in I) \colon t_i \in T_\Delta(\mathrm{Z})}} (\varphi, t) \cdot \prod_{i \in I} (\psi_i, t_i)\; t[t_i]_{i \in I} \enspace .$$

2.4 Tree Series Transducers

Let Q be an alphabet, and Σ and Δ be ranked alphabets. We abbreviate $\{\, q(u) \mid q \in Q, u \in U \,\}$ by $Q(U)$ for every set U. A *tree representation* μ *(over* Q, Σ, Δ, *and* $\mathcal{A}$*)* [21, 10] is a family $(\, \mu_k(\sigma) \,)_{k \in \mathbb{N}, \sigma \in \Sigma_k}$ of matrices $\mu_k(\sigma) \in A\langle\!\langle T_\Delta(\mathrm{Z}) \rangle\!\rangle^{Q \times Q(\mathrm{X}_k)^*}$ such that (i) $\mu_k(\sigma)_{q,w} \neq \widetilde{0}$ for only finitely many $(q, w) \in Q \times Q(\mathrm{X}_k)^*$ and (ii) $\mu_k(\sigma)_{q,w} \in A\langle\!\langle T_\Delta(\mathrm{Z}_n) \rangle\!\rangle$ where $n = |w|$ for every $q \in Q$ and $w \in Q(\mathrm{X}_k)^*$. A tree representation μ is said to be:

- *polynomial* (respectively, *boolean*), if for every $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, $q \in Q$, and $w \in Q(\mathrm{X}_k)^*$ the tree series $\mu_k(\sigma)_{q,w}$ is polynomial (respectively, boolean);
- *input-nondeleting* (respectively, *input-linear*), if for every $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, $q \in Q$, and $w \in Q(\mathrm{X}_k)^*$ with $\mu_k(\sigma)_{q,w} \neq \widetilde{0}$ we have that w is nondeleting (respectively, linear) in $[k]$;
- *output-nondeleting* (respectively, *output-linear*), if for every $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, $q \in Q$, and $w \in Q(\mathrm{X}_k)^*$ the entry $\mu_k(\sigma)_{q,w}$ is nondeleting (respectively, linear) in $[n]$ where $n = |w|$;
- *nondeleting* (respectively, *linear*), if μ is input- and output-nondeleting (respectively, input- and output-linear);

- *bottom-up*, if for every $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, $q \in Q$, and $w \in Q(\mathrm{X}_k)^*$ with $\mu_k(\sigma)_{q,w} \neq \widetilde{0}$ we have that $w = q_1(\mathrm{x}_1) \cdots q_k(\mathrm{x}_k)$ for some $q_1, \ldots, q_k \in Q$;

- *top-down*, if μ is output-nondeleting and output-linear;

- *bu-deterministic* (respectively, *bu-total*), if for every $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, and $w \in Q(\mathrm{X}_k)^*$, there exists at most one (respectively, at least one) $(q, t) \in Q \times T_\Delta(\mathrm{Z})$ such that $t \in \mathrm{supp}(\mu_k(\sigma)_{q,w})$; and

- *td-deterministic* (respectively, *td-total*), if for every $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, and $q \in Q$, there exists at most one (respectively, at least one) $(w, t) \in Q(\mathrm{X}_k)^* \times T_\Delta(\mathrm{Z})$ such that $t \in \mathrm{supp}(\mu_k(\sigma)_{q,w})$.

Usually when we specify a tree representation μ, we just specify some entries of $\mu_k(\sigma)$ and implicitly assume the remaining entries to be $\widetilde{0}$. Moreover, when we are concerned with bottom-up tree representations we just write $\mu_k(\sigma)_{q,q_1\cdots q_k}$ instead of $\mu_k(\sigma)_{q,q_1(\mathrm{x}_1)\cdots q_k(\mathrm{x}_k)}$. A *tree series transducer* [10, 15] is a sextuple $M = (Q, \Sigma, \Delta, \mathcal{A}, F, \mu)$ consisting of:

- an alphabet Q of *states*;

- ranked alphabets Σ and Δ, also called *input* and *output ranked alphabet*, respectively;

- a complete semiring $\mathcal{A} = (A, +, \cdot, 0, 1)$;

- a vector $F \in A\langle\langle T_\Delta(\mathrm{Z}_1)\rangle\rangle^Q$ of nondeleting and linear tree series representing *top-most outputs*; and

- a tree representation μ over Q, Σ, Δ, and $\mathcal{A}$.

Tree series transducers inherit the properties from their tree representation; *e. g.*, a tree series transducer with a polynomial bottom-up tree representation would be called polynomial bottom-up tree series transducer. Moreover, we omit the prefix "bu" when we consider bottom-up tree series transducers and likewise we omit "td" when we consider top-down devices; *i. e.*, a deterministic bottom-up tree series transducer is a tree series transducer that is bottom-up and bu-deterministic. Finally, we say that the (bottom-up or top-down) tree series transducer M is a *homomorphism*, if $Q = \{\star\}$, $F_\star = 1\,\mathrm{z}_1$, and μ is deterministic and total.

Let $M = (Q, \Sigma, \Delta, \mathcal{A}, F, \mu)$ be a tree series transducer. Then the tree series transformation computed by M, typed $\|M\| \colon A\langle\langle T_\Sigma\rangle\rangle \longrightarrow A\langle\langle T_\Delta\rangle\rangle$, is defined as follows. For every $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, and $t_1, \ldots, t_k \in T_\Sigma$ we define the mapping $h_\mu \colon T_\Sigma \longrightarrow A\langle\langle T_\Delta\rangle\rangle^Q$ componentwise for every $q \in Q$ by

$$h_\mu\big(\sigma(t_1, \ldots, t_k)\big)_q = \sum_{\substack{w \in Q(\mathrm{X}_k)^*,\\ w = q_1(\mathrm{x}_{i_1})\cdots q_n(\mathrm{x}_{i_n})}} \mu_k(\sigma)_{q,w} \longleftarrow (\, h_\mu(t_{i_j})_{q_j}\,)_{j \in [n]} \ .$$

Moreover, $h_\mu(\varphi)_q = \sum_{t \in T_\Sigma} (\varphi, t) \cdot h_\mu(t)_q$ for every $\varphi \in A\langle\langle T_\Sigma\rangle\rangle$. Then for every $\varphi \in A\langle\langle T_\Sigma\rangle\rangle$ the *tree series transformation computed by* M is

$$\|M\|(\varphi) = \sum_{q \in Q} F_q \longleftarrow (h_\mu(\varphi)_q) \ .$$

By $\mathrm{BOT}(\mathcal{A})$ [respectively, $\mathrm{TOP}(\mathcal{A})$] we denote the class of tree series transformations computable by bottom-up (respectively, top-down) tree series transducers over $\mathcal{A}$. Similarly, we also use p–$\mathrm{BOT}(\mathcal{A})$ [respectively, b–$\mathrm{BOT}(\mathcal{A})$, l–$\mathrm{BOT}(\mathcal{A})$, n–$\mathrm{BOT}(\mathcal{A})$, d–$\mathrm{BOT}(\mathcal{A})$, and h–$\mathrm{BOT}(\mathcal{A})$] for the class of tree series transformations computable by polynomial (respectively, boolean, linear, nondeleting, deterministic, and homomorphism) bottom-up tree series transducers over $\mathcal{A}$. Combinations of restrictions are handled in the usual manner; *i. e.*, let x–$\mathrm{BOT}(\mathcal{A})$ and y–$\mathrm{BOT}(\mathcal{A})$ be two classes of tree series transformations, then

$$xy\text{–}\mathrm{BOT}(\mathcal{A}) = x\text{–}\mathrm{BOT}(\mathcal{A}) \cap y\text{–}\mathrm{BOT}(\mathcal{A}) \ .$$

Likewise we also use the corresponding classes of tree series transformations induced by restricted top-down tree series transducers.

According to custom, we write ; for function composition; so given two tree series transformations $\tau_1 \colon A\langle\langle T_\Sigma \rangle\rangle \longrightarrow A\langle\langle T_\Gamma \rangle\rangle$ and $\tau_2 \colon A\langle\langle T_\Gamma \rangle\rangle \longrightarrow A\langle\langle T_\Delta \rangle\rangle$, then for every $\varphi \in A\langle\langle T_\Sigma \rangle\rangle$ we have that $(\tau_1 \,;\tau_2)(\varphi) = \tau_2(\tau_1(\varphi))$. This composition is extended to classes of transformations in the standard manner.

In the sequel we use the notation $[y]$ where y is one of the abbreviations of restrictions (*i. e.*, $y \in \{\mathrm{p}, \mathrm{b}, \mathrm{l}, \mathrm{n}, \mathrm{d}, \mathrm{h}\}$) in equalities to mean that this restriction is optional; *i. e.*, throughout the statement $[y]$ can be substituted by the empty word or by y. For example,

$$[\mathrm{l}]\mathrm{p}\text{–}\mathrm{BOT}(\mathcal{A}) = \mathrm{nlp}\text{–}\mathrm{BOT}(\mathcal{A}) \,; [\mathrm{l}]\mathrm{h}\text{–}\mathrm{BOT}(\mathcal{A})$$

states that the class of tree series transformations computable by polynomial (respectively, linear, polynomial) bottom-up tree series transducers coincides with the composition of the class of tree series transformations computable by nondeleting, linear, polynomial bottom-up tree series transducers with the class of tree series transformations computable by homomorphism (respectively, linear, homomorphism) bottom-up tree series transducers.

3 Distributivity, Linearity, and Associativity

In this section we establish basic properties of pure substitution. In particular, we discuss distributivity, linearity, and associativity, which are the main properties required for our composition results. Distributivity and linearity are already handled in the literature [10, Propositions 2.8 and 2.9]. For the rest of this section, let $I \subseteq \mathbb{N}_+$ be a finite set, J a set, and J_i be a set for every $i \in I$. Moreover, let Δ be a ranked alphabet.

We first recall three properties of paramount importance from [15, Proposition 3.4]. In the sequel we use these basic properties without explicit mention.

Observation 3.1 (Proposition 3.4 of [15]) Let $\psi, \psi_i \in A\langle\langle T_\Delta(\mathrm{Z}) \rangle\rangle$ for every $i \in I$.

- If $I = \emptyset$, then $\psi \longleftarrow (\, \psi_i \,)_{i \in I} = \psi$.
- If $\psi = \widetilde{0}$, then $\psi \longleftarrow (\, \psi_i \,)_{i \in I} = \widetilde{0}$.
- If $\psi_i = \widetilde{0}$ for some $i \in I$, then $\psi \longleftarrow (\, \psi_i \,)_{i \in I} = \widetilde{0}$. □

For tree languages $L \subseteq T_\Delta(\mathrm{Z}_k)$ and $L_1, \ldots, L_k \subseteq T_\Delta$ we naturally have $L[L_i]_{i \in [k]} = L[L_i]_{i \in [k] \setminus \{j\}}$ for every $j \in [k]$ such that $j \notin \mathrm{var}(L)$ and $L_j \neq \emptyset$. A similar statement can be presented for pure substitution.

Observation 3.2 Let $\psi, \psi_i \in A\langle\langle T_\Delta(\mathrm{Z}) \rangle\rangle$ for every $i \in I$. Then for every $j \in I$ such that $j \notin \mathrm{var}(\psi)$ and $\psi_j = 1\; u$ for some $u \in T_\Delta(\mathrm{Z})$

$$\psi \longleftarrow (\psi_i)_{i \in I} = \psi \longleftarrow (\psi_i)_{i \in I \setminus \{j\}} \enspace .$$

Proof: The proof is straightforward and hence omitted. □

The first central result is that pure substitution is distributive and linear [10, Propositions 2.8 and 2.9]. We present the corresponding propositions of [10].

Proposition 3.3 (Proposition 2.9 of [10]) Let $\psi_j \in A\langle\langle T_\Delta(\mathrm{Z}) \rangle\rangle$ for every $j \in J$, and for every $i \in I$ and $j_i \in J_i$ let $\psi_{j_i} \in A\langle\langle T_\Delta(\mathrm{Z}) \rangle\rangle$.

$$\sum_{\substack{j \in J, \\ (\forall i \in I)\colon j_i \in J_i}} \psi_j \longleftarrow (\, \psi_{j_i} \,)_{i \in I} = \Big(\sum_{j \in J} \psi_j \Big) \longleftarrow \Big(\sum_{j_i \in J_i} \psi_{j_i} \Big)_{i \in I} \tag{1}$$

□

Proposition 3.4 (Proposition 2.8 of [10]) Let $a \in A$, and $\psi \in A\langle\langle T_\Delta(\mathrm{Z}) \rangle\rangle$. Moreover, let $\psi_i \in A\langle\langle T_\Delta(\mathrm{Z}) \rangle\rangle$ and $a_i \in A$ for every $i \in I$.

$$a \cdot \prod_{i \in I} a_i \cdot \big(\psi \longleftarrow (\, \psi_i \,)_{i \in I} \big) = (a \cdot \psi) \longleftarrow (\, a_i \cdot \psi_i \,)_{i \in I} \tag{2}$$

□

Next let us investigate associativity. Pure substitution generalizes IO-substitution on tree languages, which is not associative. Thus we cannot establish associativity in general. However, in [11, Lemma 2.4.3] it was shown that for every $k, n \in \mathbb{N}$ with $k \geqslant 1$ and $L \subseteq T_\Delta(\mathrm{Z}_k)$, $L_1, \ldots, L_k \subseteq T_\Delta(\mathrm{Z}_n)$, and $L'_1, \ldots, L'_n \subseteq T_\Delta(\mathrm{Z})$

$$\big(L[L_1, \ldots, L_k]\big)[L'_1, \ldots, L'_n] = L\big[L_1[L'_1, \ldots, L'_n], \ldots, L_k[L'_1, \ldots, L'_n]\big]$$

holds, whenever all $L'_1, \ldots, L'_n$ are singletons or $L_1, \ldots, L_k$ are pairwise variable-disjoint. For $k = 0$ to be eligible, we have to demand that $L'_i \neq \emptyset$ for every $i \in [n]$. Now we extend the variable-disjointness condition including the case $k = 0$ to tree series. Let $I, J \subseteq \mathbb{N}_+$ be finite and $\Psi = (\, \psi_j \,)_{j \in J}$ be a family of $\psi_j \in A\langle\langle T_\Delta(\mathrm{Z})\rangle\rangle$. Finally, let $\mathcal{I} = (\, I_j \,)_{j \in J}$ be a partition of I. The partition $\mathcal{I}$ is said to *conform to* Ψ, if for every $j \in J$ the condition $\mathrm{var}(\psi_j) \subseteq I_j$ holds. Note that for every family $\Psi = (\, \psi_j \,)_{j \in J}$ with $J \neq \emptyset$ of pairwise variable-disjoint tree series a partition of I conforming to Ψ exists. Further, if $J = \emptyset$ then such a partition only exists when $I = \emptyset$.

In [10, Proposition 2.10] an associativity-like law for monomials was proved and [14, Proposition 2.5] presents a generalized version. We present yet another straightforward generalization for pairwise variable-disjoint tree series. To increase the readability of the statements of this section, we assume a finite $I \subseteq \mathbb{N}_+$, $\psi \in A\langle\langle T_\Delta(\mathrm{Z})\rangle\rangle$, and a finite set $J \subseteq \mathbb{N}_+$ such that $\mathrm{var}(\psi) \subseteq J$. Moreover, let $(\, I_j \,)_{j \in J}$ be a family of $I_j \subseteq I$ such that $\bigcup_{j \in J} I_j = I$, $(\, \psi_j \,)_{j \in J}$ be a family of $\psi_j \in A\langle\langle T_\Delta(\mathrm{Z})\rangle\rangle$ such that $\mathrm{var}(\psi_j) \subseteq I_j$ for every $j \in J$, and $(\, \tau_i \,)_{i \in I}$ be a family of $\tau_i \in A\langle\langle T_\Delta(\mathrm{Z})\rangle\rangle$.

Proposition 3.5 (*cf.* Proposition 2.5 of [14]) If $(\, I_j \,)_{j \in J}$ is a partition of I conforming to $(\, \psi_j \,)_{j \in J}$, then

$$\big(\psi \longleftarrow (\, \psi_j \,)_{j \in J}\big) \longleftarrow (\, \tau_i \,)_{i \in I} = \psi \longleftarrow \big(\, \psi_j \longleftarrow (\, \tau_i \,)_{i \in I_j}\big)_{j \in J} \;. \tag{3}$$

Proof: Note that $J = \emptyset$ implies that $I = \emptyset$.

$$\begin{aligned}
&\big(\psi \longleftarrow (\, \psi_j \,)_{j \in J}\big) \longleftarrow (\, \tau_i \,)_{i \in I} \\
= &\sum_{\substack{u \in \mathrm{supp}(\psi),\\ (\forall j \in J):\; u_j \in \mathrm{supp}(\psi_j)}} \Big((\psi, u)\, u \longleftarrow \big((\psi_j, u_j)\, u_j\big)_{j \in J}\Big) \longleftarrow (\, \tau_i \,)_{i \in I} \\
&\text{(by Proposition 3.3)} \\
= &\sum_{\substack{u \in \mathrm{supp}(\varphi),\\ (\forall j \in J):\; u_j \in \mathrm{supp}(\psi_j)}} (\psi, u)\, u \longleftarrow \Big((\psi_j, u_j)\, u_j \longleftarrow (\, \tau_i \,)_{i \in I_j}\Big)_{j \in J} \\
&\text{(by [15, Proposition 2.4] and the fact that } (\, I_j \,)_{j \in J} \text{ conforms to } (\, \psi_j \,)_{j \in J}) \\
= &\;\psi \longleftarrow \big(\psi_j \longleftarrow (\, \tau_i \,)_{i \in I_j}\big)_{j \in J} \\
&\text{(by Proposition 3.3)}
\end{aligned}$$

□

This concludes our consideration of the case that the ψ_j are variable-disjoint. According to [11, Lemma 2.4.3] there is a second sufficient condition, namely that the τ_i are monomials. This case is considered in the next lemma.

Lemma 3.6 Let τ_i be monomial for every $i \in I$. If (τ_i, v_i) is multiplicatively idempotent for every $v_i \in T_\Delta(\mathrm{Z})$ and $i \in I$, then

$$\big(\psi \longleftarrow (\, \psi_j \,)_{j \in J}\big) \longleftarrow (\, \tau_i \,)_{i \in I} = \psi \longleftarrow \big(\, \psi_j \longleftarrow (\, \tau_i \,)_{i \in I_j}\big)_{j \in J} \;. \tag{4}$$

Proof: Firstly, let $J = \emptyset$. Then also $I = \emptyset$ and both sides of (4) are ψ. Secondly, let $\mathrm{supp}(\tau_i) = \emptyset$ for some $i \in I$. It follows that $J \neq \emptyset$ and hence both sides of (4) are $\widetilde{0}$. Finally, we assume that $J \neq \emptyset$, and for every $i \in I$ let $\mathrm{supp}(\tau_i) = \{v_i\}$ for some $v_i \in T_\Delta(\mathrm{Z})$.

$$\big(\psi \longleftarrow (\, \psi_j \,)_{j \in J}\big) \longleftarrow (\, \tau_i \,)_{i \in I}$$

$$= \sum_{\substack{u \in \mathrm{supp}(\psi),\\ (\forall j \in J):\ u_j \in \mathrm{supp}(\psi_j)}} \Big((\psi, u) \cdot \prod_{j \in J} (\psi_j, u_j) \cdot \prod_{i \in I} (\tau_i, v_i) \Big)\, u[u_j]_{j \in J}[v_i]_{i \in I}$$

$$= \sum_{\substack{u \in \mathrm{supp}(\psi),\\ (\forall j \in J):\ u_j \in \mathrm{supp}(\psi_j)}} \Bigg((\psi, u) \cdot \prod_{j \in J} \Big((\psi_j, u_j) \cdot \prod_{i \in I_j} (\tau_i, v_i) \Big) \Bigg)\, u\big[u_j[v_i]_{i \in I_j}\big]_{j \in J}$$

(because $\mathcal{A}$ is commutative, $J \neq \emptyset$, $\mathrm{var}(u_j) \subseteq \mathrm{var}(\psi_j) \subseteq I_j$ for every $j \in J$, and (τ_i, v_i) is multiplicatively idempotent for every $i \in I$)

$$= \psi \longleftarrow \big(\psi_j \longleftarrow (\,\tau_i\,)_{i \in I_j} \big)_{j \in J}$$ □

Note that if we set $I_j = I$ for every $j \in J$, then we obtain associativity. Moreover, if the tree series τ_i are boolean, then every (τ_i, u_i) is automatically multiplicatively idempotent.

4 Compositions of Bottom-up Tree Series Transformations

First let us review what is known about compositions of bottom-up tree series transformations. Bottom-up tree transformations (*i. e.*, polynomial bottom-up tree series transformations over the Boolean semiring, [see 10, Section 4]) are closed under left-composition with linear bottom-up tree transformations (see [1, Theorem 6] and [9, Theorem 4.5]); *i. e.*,

$$\mathrm{lp\text{–}BOT}(\mathbb{B}) \,;\, \mathrm{p\text{–}BOT}(\mathbb{B}) = \mathrm{p\text{–}BOT}(\mathbb{B}) \ .$$

This result was generalized to bottom-up tree series transformations over commutative and complete semirings in [20, 10]. More precisely, [20, Theorem 2.4] yields that

$$\mathrm{nl\text{–}BOT}(\mathcal{A}) \,;\, \mathrm{nl\text{–}BOT}(\mathcal{A}) = \mathrm{nl\text{–}BOT}(\mathcal{A}) \ .$$

In fact it is shown for nondeleting, linear top-down tree series transducers [10], but nondeleting, linear top-down tree series transducers and nondeleting, linear bottom-up tree series transducers are equally powerful [see 10, Theorem 5.24]. Moreover, $\mathrm{nl\text{–}BOT}(\mathcal{A}) \,;\, \mathrm{h\text{–}BOT}(\mathcal{A}) \subseteq \mathrm{BOT}(\mathcal{A})$ [10, Corollary 5.5]. So taking those results together and a decomposition [10, Lemma 5.6], we obtain the following result.

Theorem 4.1 For every commutative and complete semiring $\mathcal{A}$

$$\mathrm{nlp\text{–}BOT}(\mathcal{A}) \,;\, \mathrm{p\text{–}BOT}(\mathcal{A}) = \mathrm{p\text{–}BOT}(\mathcal{A}) \ . \tag{5}$$

Proof: The direction $\mathrm{p\text{–}BOT}(\mathcal{A}) \subseteq \mathrm{nlp\text{–}BOT}(\mathcal{A}) \,;\, \mathrm{p\text{–}BOT}(\mathcal{A})$ is trivial, so it remains to prove $\mathrm{nlp\text{–}BOT}(\mathcal{A}) \,;\, \mathrm{p\text{–}BOT}(\mathcal{A}) \subseteq \mathrm{p\text{–}BOT}(\mathcal{A})$.

$$\begin{aligned}
& \mathrm{nlp\text{–}BOT}(\mathcal{A}) \,;\, \mathrm{p\text{–}BOT}(\mathcal{A}) & \\
\subseteq\ & \mathrm{nlp\text{–}BOT}(\mathcal{A}) \,;\, \mathrm{nlp\text{–}BOT}(\mathcal{A}) \,;\, \mathrm{h\text{–}BOT}(\mathcal{A}) & \text{[10, Lemma 5.6]} \\
\subseteq\ & \mathrm{nlp\text{–}BOT}(\mathcal{A}) \,;\, \mathrm{h\text{–}BOT}(\mathcal{A}) & \text{[20, Theorem 2.4]} \\
\subseteq\ & \mathrm{p\text{–}BOT}(\mathcal{A}) & \text{[10, Corollary 5.5]}
\end{aligned}$$ □

We should like to obtain a result like $\mathrm{l\text{–}BOT}(\mathcal{A}) \,;\, \mathrm{BOT}(\mathcal{A}) = \mathrm{BOT}(\mathcal{A})$ for all commutative and complete semirings $\mathcal{A}$. We try to follow the classical (unweighted) construction, so we first extend h_μ such that it can treat variables (of X). We extend h_μ to $T_\Sigma(\mathrm{X})$ by supplying, for some $J \subseteq \mathbb{N}_+$, a mapping $\overline{q} \in Q^J$, which associates a state $\overline{q}(j)$, usually written as $\overline{q}_j$, to the variable x_j for $j \in J$. Intuitively speaking, the state $\overline{q}_j$ represents the initial state, with which the computation should be started at the leaves labeled x_j in the input tree. For all states $q \in Q$ different from $\overline{q}_j$ it should not be possible to start a (meaningful) computation at x_j (*i. e.*, $h_\mu^{\overline{q}}(\mathrm{x}_j)_q = \widetilde{0}$). This mapping is then extended to $T_\Sigma(\mathrm{X})$ in a manner analogous to h_μ.

Definition 4.2 Let $(Q, \Sigma, \Delta, \mathcal{A}, F, \mu)$ be a bottom-up tree series transducer. For every finite $J \subseteq \mathbb{N}_+$ and $\overline{q} \in Q^J$ we define the mapping $h_\mu^{\overline{q}} \colon T_\Sigma(\mathrm{X}) \longrightarrow A\langle\langle T_\Delta(\mathrm{Z})\rangle\rangle^Q$ componentwise for every $q \in Q$ as follows. For every $j \in J$, $n \in \mathbb{N}_+ \setminus J$, $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, and $t_1, \ldots, t_k \in T_\Sigma(\mathrm{X})$

$$h_\mu^{\overline{q}}(\mathrm{x}_n)_q = 1\ \mathrm{z}_n \tag{6}$$

$$h_\mu^{\overline{q}}(\mathrm{x}_j)_q = \begin{cases} 1\ \mathrm{z}_j & \text{if } q = \overline{q}_j \\ \widetilde{0} & \text{otherwise} \end{cases}, \tag{7}$$

$$h_\mu^{\overline{q}}(\sigma(t_1, \ldots, t_k))_q = \sum_{q_1, \ldots, q_k \in Q} \mu_k(\sigma)_{q, q_1 \cdots q_k} \longleftarrow (\, h_\mu^{\overline{q}}(t_i)_{q_i}\,)_{i \in [k]} \ . \tag{8}$$

The mapping $h_\mu^{\overline{q}} \colon A\langle\langle T_\Sigma(\mathrm{X})\rangle\rangle \longrightarrow A\langle\langle T_\Delta(\mathrm{Z})\rangle\rangle^Q$ is given for every $\varphi \in A\langle\langle T_\Sigma(\mathrm{X})\rangle\rangle$ by

$$h_\mu^{\overline{q}}(\varphi)_q = \sum_{t \in T_\Sigma(\mathrm{X})} (\varphi, t) \cdot h_\mu^{\overline{q}}(t)_q \ .$$

□

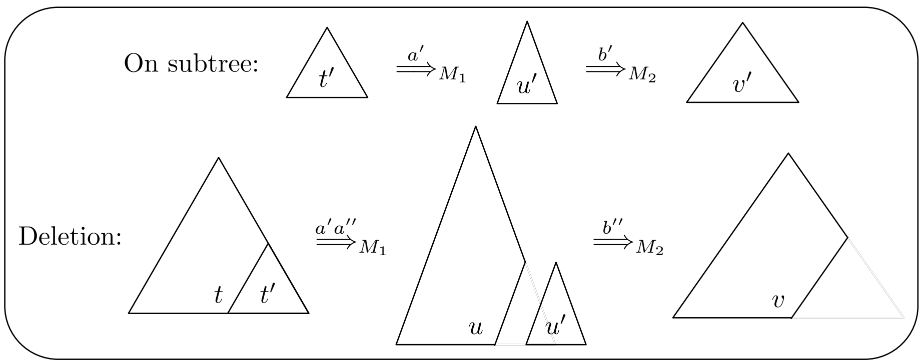


Figure 1: Computation of M' followed by M''.

Let $M' = (Q', \Sigma, \Gamma, \mathcal{A}, F', \mu')$ and $M'' = (Q'', \Gamma, \Delta, \mathcal{A}, F'', \mu'')$ be bottom-up tree series transducers. Then, similar to the (unweighted) product construction of bottom-up tree transducers, we translate the entries of μ' with the help of μ''. Let $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, $p, p_1, \ldots, p_k \in Q'$, and $q, q_1, \ldots, q_k \in Q''$. Roughly speaking, we obtain the entry $\mu_k(\sigma)_{(p,q),(p_1,q_1)\cdots(p_k,q_k)}$ in the tree representation μ of the composition of M' and M'' by applying the extended mapping $h_{\mu''}^{q_1 \cdots q_k}$ to the entry $\mu'_k(\sigma)_{p,p_1\cdots p_k}$. Thereby, we process the output trees of $\mathrm{supp}(\mu'_k(\sigma)_{p,p_1\cdots p_k})$ with the help of M'' starting the computation at the variables $\mathrm{x}_1, \ldots, \mathrm{x}_k$ in states $q_1, \ldots, q_k$.

However, there is a small problem which does not arise in the unweighted case. We depict the problem in Figures 1 and 2. Let us suppose that M' translates an input tree $t \in T_\Sigma$ into an output tree $u \in T_\Gamma$ with weight $a \in A$. During the translation, M' decides to delete the translation $u' \in T_\Gamma$ with weight $a' \in A$ of an input subtree $t' \in T_\Sigma$. Then due to the definition of pure substitution the weight a' of u' contributes to the weight a of u, whereas u' does not contribute to u. Furthermore, let us suppose that M'' would transform u into $v \in T_\Delta$ at weight $b \in A$ and u' into $v' \in T_\Delta$ at weight $b' \in A$. Since M'' does not process u', the weight b' does not contribute to b. However, the composition of M' and M'', when processing the input subtree t', transforms t' into u' at weight a' using the rules of M' and immediately also transforms u' into v' at weight b' using the rules of M''. If the composition tree series transducer now deletes the translation v' of t', then a' and b' still contribute to the weight of the overall transformation. This contrasts the situation encountered when M' and M'' run separately, because there only a' contributed to the weight of the overall transformation. In the classical case of tree transducers, b' could only be 0 or 1, so that one just had to avoid that $b' = 0$. In principle, this is achieved by requiring M'' to be total (however, by adjoining a dummy state, each bottom-up tree transducer can be turned into a total one computing the same tree transformation). The construction we propose here is similar, but has the major disadvantage that, for example, determinism is not preserved.

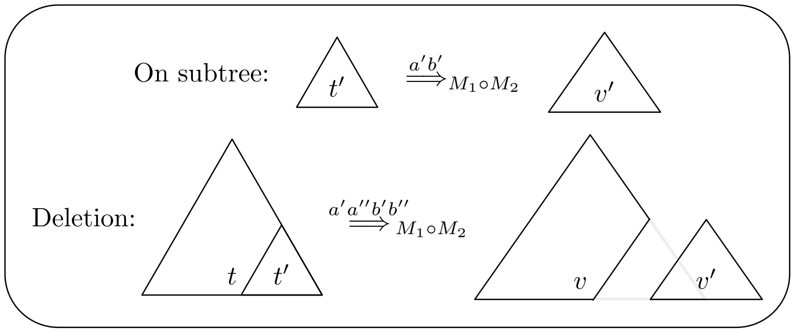


Figure 2: Computation of $M' \,; M''$.

Specifically, we address the aforementioned problem by manipulating the second transducer M'' such that it has a state $\perp$ which transforms each input tree into some output tree $\alpha \in \Delta_0$ at weight 1. Note that $\perp$ is no final state; *i. e.*, its top-most output is $\widetilde{0}$. Then we compose M' and M'' by processing those subtrees, which M' decided to delete, in the state $\perp$.

Definition 4.3 Let $M = (Q, \Sigma, \Delta, \mathcal{A}, F, \mu)$ be a bottom-up tree series transducer. A state $\perp \in Q$ is called *blind*, if there exists an $\alpha \in \Delta_0$ such that:

- $F_\perp = \widetilde{0}$;
- for every $k \in \mathbb{N}$ and $\sigma \in \Sigma_k$ we have $\mu_k(\sigma)_{\perp,\perp\cdots\perp} = 1\ \alpha$; and
- for every $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, $q, q_1, \dots, q_k \in Q$ with $\mu_k(\sigma)_{q,q_1\cdots q_k} \neq \widetilde{0}$

$$q = \perp \quad\iff\quad (\forall i \in [k])\colon q_i = \perp \ . \qquad \Box$$

To every bottom-up tree series transducer M we can adjoin a blind state $\perp$ and thereby obtain a bottom-up tree series transducer M'. It should be clear that $\|M\| = \|M'\|$.

Observation 4.4 Let M be a bottom-up tree series transducer. There exists a bottom-up tree series transducer M' with blind state $\perp$ such that $\|M\| = \|M'\|$.

Proof: Let $M = (Q, \Sigma, \Delta, \mathcal{A}, F, \mu)$ and $\perp \notin Q$ and $\alpha \in \Delta_0$. We construct $M' = (Q', \Sigma, \Delta, \mathcal{A}, F', \mu')$ with $Q' = Q \cup \{\perp\}$, $F'_q = F_q$ for every $q \in Q$ and $F'_\perp = \widetilde{0}$. The tree representation μ' is defined for every $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, and $q, q_1, \dots, q_k \in Q$ by

$$\mu'_k(\sigma)_{q,q_1\cdots q_k} = \mu_k(\sigma)_{q,q_1\cdots q_k} \qquad (9)$$
$$\mu'_k(\sigma)_{\perp,\perp\cdots\perp} = 1\ \alpha \ . \qquad (10)$$

Clearly, $\perp$ is a blind state of M' and also $\|M\| = \|M'\|$. $\Box$

Note that the construction does not preserve determinism.

Definition 4.5 Let $M' = (Q', \Sigma, \Gamma, \mathcal{A}, F', \mu')$ and $M'' = (Q'', \Gamma, \Delta, \mathcal{A}, F'', \mu'')$ be two bottom-up tree series transducers such that $\perp$ is a blind state of M''. The *composition of* M' *and* M'', denoted by $M' \,; M''$, is defined to be the bottom-up tree series transducer $M' \,; M'' = (Q' \times Q'', \Sigma, \Delta, \mathcal{A}, F, \mu)$ with

$$F_{(p,q)} = \sum_{q' \in Q''} F''_{q'} \longleftarrow \big(h^q_{\mu''}\big(F'_p\big)_{q'}\big) \qquad (11)$$

$$\mu_k(\sigma)_{(p,q),(p_1,q_1)\cdots(p_k,q_k)} = h^{q_1\cdots q_k}_{\mu''}\Big(\sum_{\substack{t \in T_\Gamma(\mathrm{Z}_k),\\ (\forall i \in [k])\colon i \notin \mathrm{var}(t) \iff q_i = \perp}} \big(\mu'_k(\sigma)_{p,p_1\cdots p_k}, t\big)\ t\Big)_q \qquad (12)$$

$$\mu_k(\sigma)_{(p,\perp),(p_1,\perp)\cdots(p_k,\perp)} = h_{\mu''}^{\perp\cdots\perp}\big(\mu'_k(\sigma)_{p,p_1\cdots p_k}\big)_\perp \tag{13}$$

for every $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, $p, p_1, \ldots, p_k \in Q'$, $q \in Q'' \setminus \{\perp\}$, and $q_1, \ldots, q_k \in Q''$. All the remaining entries in F and μ are $\widetilde{0}$. □

It is quite clear that the composition $M' \,;\, M''$ does not always compute $\|M'\| \,;\, \|M''\|$, because already for bottom-up tree transducers (*i. e.*, polynomial bottom-up tree series transducers over $\mathbb{B}$) it can be shown that the computed transformations are not closed with respect to composition. However, we have already mentioned that p–BOT($\mathbb{B}$) is closed under left-composition with lp–BOT($\mathbb{B}$) and under right-composition with d–BOT($\mathbb{B}$). The next proposition shows a central property of restricted bottom-up tree series transducers. Roughly speaking, it presents sufficient conditions that if imposed ensure that h_μ distributes over substitutions $t[u_1, \ldots, u_k]$ for $t \in T_\Sigma(\mathrm{X}_k)$ and $u_1, \ldots, u_k \in T_\Sigma$.

Proposition 4.6 Let $V \subseteq \mathrm{X}$ be a finite set, and let $M = (Q, \Sigma, \Delta, \mathcal{A}, F, \mu)$ be a bottom-up tree series transducer, $q \in Q$, $t \in T_\Sigma(V)$, and $u_i \in T_\Sigma$ for every $i \in \mathrm{var}(t)$.

$$h_\mu(t[u_i]_{i\in\mathrm{var}(t)})_q = \sum_{\overline{q}\in Q^{\mathrm{var}(t)}} h_\mu^{\overline{q}}(t)_q \longleftarrow \big(h_\mu(u_i)_{\overline{q}_i}\big)_{i\in\mathrm{var}(t)} \;,$$

provided that:

(a) M is boolean and deterministic; or

(b) t is linear.

Proof: We prove the statement by induction over t.
(i) First, let $t = \mathrm{x}_j$ for some $j \in \mathbb{N}_+$. Clearly, $\mathrm{var}(t) = \{j\}$.

$$\begin{aligned}
&h_\mu(\mathrm{x}_j[u_i]_{i\in\{j\}})_q \\
&= h_\mu(u_j)_q \\
&\quad \text{(by tree substitution)} \\
&= 1\ \mathrm{z}_j \longleftarrow (h_\mu(u_i)_q)_{i\in\{j\}} \\
&\quad \text{(by definition of pure substitution)} \\
&= \sum_{\overline{q}\in Q^{\{j\}}} h_\mu^{\overline{q}}(\mathrm{x}_j)_q \longleftarrow (h_\mu(u_i)_{q_i})_{i\in\{j\}} \\
&\quad \text{(because } h_\mu^{\overline{q}}(\mathrm{x}_j)_q = \widetilde{0} \text{ for every } \overline{q} \text{ such that } \overline{q}_j \neq q)
\end{aligned}$$

(ii) Let $t = \sigma(t_1, \ldots, t_k)$ for some $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, and $t_1, \ldots, t_k \in T_\Sigma(V)$.

$$\begin{aligned}
&h_\mu(\sigma(t_1, \ldots, t_k)[u_i]_{i\in\mathrm{var}(t)})_q \\
&= h_\mu(\sigma(t_1[u_i]_{i\in\mathrm{var}(t_1)}, \ldots, t_k[u_i]_{i\in\mathrm{var}(t_k)}))_q \\
&\quad \text{(by tree substitution)} \\
&= \sum_{q_1,\ldots,q_k\in Q} \mu_k(\sigma)_{q,q_1\cdots q_k} \longleftarrow \big(h_\mu(t_j[u_i]_{i\in\mathrm{var}(t_j)})_{q_j}\big)_{j\in[k]} \\
&\quad \text{(by definition of } h_\mu) \\
&= \sum_{q_1,\ldots,q_k\in Q} \mu_k(\sigma)_{q,q_1\cdots q_k} \longleftarrow \Big(\sum_{\overline{q}\in Q^{\mathrm{var}(t_j)}} h_\mu^{\overline{q}}(t_j)_{q_j} \longleftarrow \big(h_\mu(u_i)_{\overline{q}_i}\big)_{i\in\mathrm{var}(t_j)}\Big)_{j\in[k]} \\
&\quad \text{(by induction hypothesis)} \\
&= \sum_{q_1,\ldots,q_k\in Q} \sum_{(\forall j\in[k]):\ \overline{q(j)}\in Q^{\mathrm{var}(t_j)}} \mu_k(\sigma)_{q,q_1\cdots q_k} \longleftarrow \big(h_\mu^{\overline{q(j)}}(t_j)_{q_j} \longleftarrow (h_\mu(u_i)_{\overline{q(j)}_i})_{i\in\mathrm{var}(t_j)}\big)_{j\in[k]} \\
&\quad \text{(by Proposition 3.3)}
\end{aligned}$$

$$= \sum_{q_1,\dots,q_k \in Q} \sum_{\overline{q} \in Q^{\mathrm{var}(t)}} \mu_k(\sigma)_{q,q_1\cdots q_k} \longleftarrow \big(h_\mu^{\overline{q}}(t_j)_{q_j} \longleftarrow (h_\mu(u_i)_{\overline{q}_i})_{i\in \mathrm{var}(t_j)}\big)_{j\in[k]}$$

(because $\bigcup_{j\in[k]} \mathrm{var}(t_j) = \mathrm{var}(t)$ and by:

(a) determinism because there exists at most one $p \in Q$ such that $h_\mu(u_i)_p \neq \widetilde{0}$; or

(b) linearity of t because $\mathrm{var}(t_{j_1}) \cap \mathrm{var}(t_{j_2}) = \emptyset$ for $j_1 \neq j_2$)

$$= \sum_{\overline{q} \in Q^{\mathrm{var}(t)}} \sum_{q_1,\dots,q_k \in Q} \big(\mu_k(\sigma)_{q,q_1\cdots q_k} \longleftarrow (h_\mu^{\overline{q}}(t_j)_{q_j})_{j\in[k]}\big) \longleftarrow (h_\mu(u_i)_{\overline{q}_i})_{i\in \mathrm{var}(t)}$$

(by

(a) Lemma 3.6 because $h_\mu(u_i)_{\overline{q}_i}$ is a boolean monomial; or

(b) Proposition 3.5 because $(\mathrm{var}(t_j))_{j\in[k]}$ is the required partition)

$$= \sum_{\overline{q} \in Q^{\mathrm{var}(t)}} h_\mu^{\overline{q}}(\sigma(t_1,\dots,t_k))_q \longleftarrow (h_\mu(u_i)_{\overline{q}_i})_{i\in \mathrm{var}(t)}$$

(by definition of $h_\mu^{\overline{q}}$) □

With the help of this proposition we can show the correctness of the construction in Definition 4.5 for linear M'; *i. e.*, we can show that $\|M' \,; M''\| = \|M'\| \,; \|M''\|$ for linear M'.

Lemma 4.7 Let $\mathcal{A}$ be a commutative and complete semiring, $M' = (Q', \Sigma, \Gamma, \mathcal{A}, F', \mu')$ and $M'' = (Q'', \Gamma, \Delta, \mathcal{A}, F'', \mu'')$ be bottom-up tree series transducers, of which M' is linear and M'' has a blind state $\perp$. Moreover, let $M = (Q, \Sigma, \Delta, \mathcal{A}, F, \mu)$ be the composition of M' and M'' (see Definition 4.5). Then for every $t \in T_\Sigma$, $p \in Q'$, and $q \in Q''$

$$h_{\mu''}\big(h_{\mu'}(t)_p\big)_q = h_\mu(t)_{(p,q)}$$

and $\|M\| = \|M'\| \,; \|M''\|$.

Proof: We first claim that there exists an $\alpha \in \Delta_0$ such that $h_{\mu''}(u)_\perp = 1\ \alpha$ for every $u \in T_\Gamma$. The proof of this claim is straightforward and left to the reader. The remaining proof is done by induction on t and case analysis. Let $t = \sigma(t_1,\dots,t_k)$ for some $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, and $t_1,\dots,t_k \in T_\Sigma$.
(i) Let $q = \perp$.

$$h_{\mu''}\big(h_{\mu'}(\sigma(t_1,\dots,t_k))_p\big)_\perp$$

$$= \sum_{p_1,\dots,p_k \in Q'} \sum_{\substack{u \in T_\Gamma(\mathrm{Z}_k),\\ (\forall i \in [k]):\ u_i \in T_\Gamma}} (\mu'_k(\sigma)_{p,p_1\cdots p_k}, u) \cdot \prod_{i\in[k]} (h_{\mu'}(t_i)_{p_i}, u_i) \cdot h_{\mu''}(u[u_1,\dots,u_k])_\perp$$

(by definition of $h_{\mu'}$ and pure substitution)

$$= \sum_{p_1,\dots,p_k \in Q'} \sum_{\substack{u \in T_\Gamma(\mathrm{Z}_k),\\ (\forall i \in [k]):\ u_i \in T_\Gamma}} (\mu'_k(\sigma)_{p,p_1\cdots p_k}, u) \cdot \prod_{i\in[k]} (h_{\mu'}(t_i)_{p_i}, u_i)\ \alpha$$

(by $h_{\mu''}(u[u_1,\dots,u_k])_\perp = 1\ \alpha$; see claim)

$$= \sum_{p_1,\dots,p_k \in Q'} \sum_{\substack{u \in T_\Gamma(\mathrm{Z}_k),\\ (\forall i \in [k]):\ u_i \in T_\Gamma}} (\mu'_k(\sigma)_{p,p_1\cdots p_k}, u) \cdot \prod_{i\in[k]} (h_{\mu'}(t_i)_{p_i}, u_i) \cdot$$

$$\cdot\, h_{\mu''}^{\perp\cdots\perp}(u)_\perp \longleftarrow (h_{\mu''}(u_1)_\perp,\dots,h_{\mu''}(u_k)_\perp)$$

(by claim and pure substitution)

$$= \sum_{p_1,\dots,p_k \in Q'} h_{\mu''}^{\perp\cdots\perp}(\mu'_k(\sigma)_{p,p_1\cdots p_k})_\perp \longleftarrow \big(h_{\mu''}(h_{\mu'}(t_1)_{p_1})_\perp,\dots,h_{\mu''}(h_{\mu'}(t_k)_{p_k})_\perp\big)$$

(by Proposition 3.3 and Proposition 3.4)

$$= \sum_{p_1,\dots,p_k \in Q'} \mu_k(\sigma)_{(p,\perp),(p_1,\perp)\cdots(p_k,\perp)} \longleftarrow \big(h_\mu(t_1)_{(p_1,\perp)},\dots,h_\mu(t_k)_{(p_k,\perp)}\big)$$

(by definition of μ and induction hypothesis)

$$= \sum_{\substack{p_1,\dots,p_k \in Q',\\ q_1,\dots,q_k \in Q''}} \mu_k(\sigma)_{(p,\perp),(p_1,q_1)\cdots(p_k,q_k)} \longleftarrow \big(h_\mu(t_1)_{(p_1,q_1)}, \dots, h_\mu(t_k)_{(p_k,q_k)}\big)$$

(since $\mu_k(\sigma)_{(p,\perp),(p_1,q_1)\dots(p_k,q_k)} \neq \widetilde{0}$, only if $q_1 = \dots = q_k = \perp$)

$$= h_\mu(\sigma(t_1,\dots,t_k))_{p,\perp}$$

(by the definition of h_μ)

(ii) Now let $q \neq \perp$.

$$h_{\mu''}\big(h_{\mu'}(\sigma(t_1,\dots,t_k))_p\big)_q$$

$$= \sum_{p_1,\dots,p_k \in Q'} h_{\mu''}\big(\mu'_k(\sigma)_{p,p_1\cdots p_k} \longleftarrow (h_{\mu'}(t_1)_{p_1}, \dots, h_{\mu'}(t_k)_{p_k})\big)_q$$

(by definition of $h_{\mu'}$)

$$= \sum_{p_1,\dots,p_k \in Q'} \sum_{\substack{u \in T_\Gamma(\mathrm{Z}_k),\\ (\forall i \in [k])\colon u_i \in T_\Gamma}} (\mu'_k(\sigma)_{p,p_1\cdots p_k}, u) \cdot \prod_{i\in[k]} (h_{\mu'}(t_i)_{p_i}, u_i) \cdot h_{\mu''}(u[u_1,\dots,u_k])_q$$

(by definition of pure substitution)

$$= \sum_{p_1,\dots,p_k \in Q'} \sum_{\substack{u \in T_\Gamma(\mathrm{Z}_k),\\ (\forall i \in [k])\colon u_i \in T_\Gamma}} (\mu'_k(\sigma)_{p,p_1\cdots p_k}, u) \cdot \prod_{i\in[k]} (h_{\mu'}(t_i)_{p_i}, u_i) \cdot$$
$$\cdot\, h_{\mu''}(u[\mathrm{x}_1,\dots,\mathrm{x}_k]\,[u_1,\dots,u_k])_q$$

(by tree substitution)

$$= \sum_{p_1,\dots,p_k \in Q'} \sum_{\substack{u \in T_\Gamma(\mathrm{Z}_k),\\ (\forall i \in [k])\colon u_i \in T_\Gamma}} (\mu'_k(\sigma)_{p,p_1\cdots p_k}, u) \cdot \prod_{i\in[k]} (h_{\mu'}(t_i)_{p_i}, u_i) \cdot$$
$$\cdot \sum_{\overline{q} \in (Q'')^{\mathrm{var}(u)}} h^{\overline{q}}_{\mu''}(u)_q \longleftarrow \big(h_{\mu''}(u_i)_{\overline{q}_i}\big)_{i \in \mathrm{var}(u)}$$

(by Proposition 4.6)

$$= \sum_{\substack{p_1,\dots,p_k \in Q',\\ q_1,\dots,q_k \in Q''}} \sum_{\substack{u \in T_\Gamma(k),\\ (\forall i \in [k])\colon u_i \in T_\Gamma,\\ q_i = \perp \iff i \notin \mathrm{var}(u)}} (\mu'_k(\sigma)_{p,p_1\cdots p_k}, u) \cdot \prod_{i\in[k]} (h_{\mu'}(t_i)_{p_i}, u_i) \cdot$$
$$\cdot\, h^{q_1,\dots,q_k}_{\mu''}(u)_q \longleftarrow \big(h_{\mu''}(u_1)_{q_1}, \dots, h_{\mu''}(u_k)_{q_k}\big)$$

(by Observation 3.2 because $h_{\mu''}(u_i)_\perp = 1\,\alpha$)

$$= \sum_{\substack{p_1,\dots,p_k \in Q',\\ q_1,\dots,q_k \in Q''}} \mu_k(\sigma)_{(p,q),(p_1,q_1)\dots(p_k,q_k)} \longleftarrow \Big(h_{\mu''}\big(h_{\mu'}(t_1)_{p_1}\big)_{q_1}, \dots, h_{\mu''}\big(h_{\mu'}(t_k)_{p_k}\big)_{q_k}\Big)$$

(by definition of μ, Propositions 3.3 and 3.4)

$$= \sum_{\substack{p_1,\dots,p_k \in Q',\\ q_1,\dots,q_k \in Q''}} \mu_k(\sigma)_{(p,q),(p_1,q_1)\dots(p_k,q_k)} \longleftarrow \big(h_\mu(t_1)_{(p_1,q_1)}, \dots, h_\mu(t_k)_{(p_k,q_k)}\big)$$

(by induction hypothesis)

$$= h_\mu(\sigma(t_1,\dots,t_k))_{(p,q)}$$

(by definition of h_μ)

Now we can prove the main statement.

$$(\|M'\|\,;\|M''\|)(\varphi)$$

$$= \sum_{p \in Q', q' \in Q''} F''_{q'} \longleftarrow \big(h_{\mu''}\big(F'_p \longleftarrow (h_{\mu'}(\varphi)_p)\big)_{q'}\big)$$

(by the definition of $\|\cdot\|$ and Proposition 3.3)

$$= \sum_{p \in Q', q \in Q''} \Big(\sum_{q' \in Q''} F''_{q'} \longleftarrow \big(h^q_{\mu''}(F'_p)_{q'}\big)\Big) \longleftarrow \big(h_{\mu''}(h_{\mu'}(\varphi)_p)_q\big)$$

(see [12, Lemma 6.5] and [20, Lemma 2.2])

$$= \sum_{p \in Q', q \in Q''} F_{(p,q)} \longleftarrow (h_\mu(\varphi)_{(p,q)})$$

(by $h_{\mu''}\big(h_{\mu'}(t)_p\big)_q = h_\mu(t)_{(p,q)}$ and definition of $F_{(p,q)}$)

$$= \|M\|(\varphi)$$

(by definition of $\|\cdot\|$) □

It is easy to see that whenever M' and M'' are polynomial (respectively, nondeleting, linear), then also $M' \,; M''$ is polynomial (respectively, nondeleting, linear). Together with Lemma 4.7 this yields the first main theorem.

Theorem 4.8 Let $\mathcal{A}$ be a commutative and complete semiring.

$$\text{[p][n]l–BOT}(\mathcal{A}) \,; \text{[p][n][l]–BOT}(\mathcal{A}) = \text{[p][n][l]–BOT}(\mathcal{A}) \tag{14}$$

Proof: The statement follows directly from Lemma 4.7. □

We note that our construction does not preserve determinism [*cf.* 10, Corollary 5.5]. Thus, neither hl–BOT$(\mathcal{A})$; h–BOT$(\mathcal{A})$ = h–BOT$(\mathcal{A})$ nor hnl–BOT$(\mathcal{A})$; h–BOT$(\mathcal{A})$ = h–BOT$(\mathcal{A})$ follows from Lemma 4.7, because we introduce the blind state $\perp$ and thus our composition $M' \,; M''$, in general, has more than one state. The correctness of the latter two statements thus remains open.

Let us consider an example. Imagine a game to be played between two players. Player I moves first and the moves of the players alternate. Each player can play one out of three potential moves (called l, m, and r), however the second player may not play the same move as the first player just played. We model this scenario by a game tree which contains three types of nodes. First there are σ-nodes indicating that one of the players should make a move. Such a node has exactly three successors, which represent the remaining game to be played in case the moving player chooses to play l, m, and r, respectively. Second, there are α- and β-nodes indicating that Player I, respectively Player II, has won the game. Third, l-, m-, and r-nodes represent that the player played this option. (Randomized) strategies for both players can now be coded as bottom-up tree series transducers (in fact, it is easier to code them as linear top-down tree series transducers, but given such we can easily obtain a semantically equivalent linear bottom-up tree series transducer [15, Theorem 5.26]). The composition of the two bottom-up tree series transducers (*i. e.*, of the two strategies) can then be applied to compute, for example, the chances of winning the game for each player.

Example 4.9 Let $\Sigma = \Sigma_0 \cup \Sigma_3$ with $\Sigma_3 = \{\sigma\}$ and $\Sigma_0 = \{\alpha, \beta\}$, $\Gamma_1 = \{\mathrm{l}, \mathrm{m}, \mathrm{r}\}$, and $\Gamma = \Gamma_1 \cup \Sigma$. Moreover, let $M' = (Q', \Sigma, \Gamma, \mathbb{R}_+, F', \mu')$ be the bottom-up tree series transducer with $Q' = \{\perp, \top\}$, $F'_\top = 1\,\mathrm{z}_1$ and $F'_\perp = \widetilde{0}$ and

$$\begin{aligned}
&\mu'_0(\alpha)_\perp = \mu'_0(\alpha)_\top = 1\,\alpha \\
&\mu'_0(\beta)_\perp = \mu'_0(\beta)_\top = 1\,\beta \\
&\qquad \mu'_3(\sigma)_{\top,\perp\perp\perp} = 0.1\,\mathrm{l}(\mathrm{z}_1) + 0.3\,\mathrm{m}(\mathrm{z}_2) + 0.6\,\mathrm{r}(\mathrm{z}_3) \\
&\qquad \mu'_3(\sigma)_{\perp,\top\top\top} = 1\,\sigma(\mathrm{z}_1, \mathrm{z}_2, \mathrm{z}_3) \enspace .
\end{aligned}$$

The first player's strategy is modeled by M', and we represent a strategy of the second player by $M'' = (Q'', \Gamma, \Sigma, \mathbb{R}_+, F'', \mu'')$ with $Q'' = \Gamma_1 \cup \{\top\}$, $F''_\top = 1\,\mathrm{z}_1$, $F''_\gamma = \widetilde{0}$ for every $\gamma \in \Gamma_1$ and

$$\mu''_0(\alpha)_\gamma = \mu''_0(\alpha)_\top = 1\,\alpha$$

$$\begin{aligned}
\mu''_0(\beta)_\gamma = \mu''_0(\beta)_\top &= 1\,\beta \\
\mu''_1(\gamma)_{\top,\gamma} &= 1\,\mathrm{z}_1 \\
\mu''_3(\sigma)_{\mathrm{l},\top\top\top} &= 0.4\,\mathrm{z}_2 + 0.6\,\mathrm{z}_3 \\
\mu''_3(\sigma)_{\mathrm{m},\top\top\top} &= 0.5\,\mathrm{z}_1 + 0.5\,\mathrm{z}_3 \\
\mu''_3(\sigma)_{\mathrm{r},\top\top\top} &= 0.7\,\mathrm{z}_1 + 0.3\,\mathrm{z}_2 \enspace .
\end{aligned}$$

Now let us consider the game tree $t = \sigma\big(\sigma(\alpha, \beta, \alpha), \beta, \sigma(\alpha, \beta, \beta)\big)$. Then

$$\begin{aligned}
\|M'\|(1\,t) &= 0.1\,\mathrm{l}\big(\sigma(\alpha,\beta,\alpha)\big) + 0.3\,\mathrm{m}(\beta) + 0.6\,\mathrm{r}\big(\sigma(\alpha,\beta,\beta)\big) \\
\|M''\|\big(\|M'\|(1\,t)\big) &= 0.48\,\alpha + 0.52\,\beta \enspace ,
\end{aligned}$$

showing that for this particular game Player II has a slightly higher chance to win the game.

Let M_2 be the bottom-up tree series transducer that is obtained by adjoining a blind state to M''. Now let us compose M' and M_2. The composition $M' \,; M_2 = (Q, \Sigma, \Sigma, \mathbb{R}_+, F, \mu)$ is defined by $Q = Q' \times (Q'' \cup \{\perp\})$ and $F_{(\top,\top)} = 1\,\mathrm{z}_1$ and $F_q = \widetilde{0}$ for all $q \in Q \setminus \{(\top,\top)\}$. Finally, the tree representation μ is defined for every $p \in Q'$, $q \in Q''$, and $\gamma \in \Gamma_1$ by

$$\begin{aligned}
\mu_0(\alpha)_{(p,q)} = \mu_0(\alpha)_{(p,\perp)} = \mu_0(\beta)_{(p,\perp)} &= 1\,\alpha \\
\mu_0(\beta)_{(p,q)} &= 1\,\beta \\
\mu_3(\sigma)_{(\top,\top),(\perp,\mathrm{l})(\perp,\perp)(\perp,\perp)} &= 0.1\,\mathrm{z}_1 \\
\mu_3(\sigma)_{(\top,\top),(\perp,\perp)(\perp,\mathrm{m})(\perp,\perp)} &= 0.3\,\mathrm{z}_2 \\
\mu_3(\sigma)_{(\top,\top),(\perp,\perp)(\perp,\perp)(\perp,\mathrm{r})} &= 0.6\,\mathrm{z}_3 \\
\mu_3(\sigma)_{(\perp,\gamma),(\top,\top)(\top,\top)(\top,\top)} &= \begin{cases} 0.4\,\mathrm{z}_2 + 0.6\,\mathrm{z}_3 & \text{if } \gamma = \mathrm{l} \enspace , \\ 0.5\,\mathrm{z}_1 + 0.5\,\mathrm{z}_3 & \text{if } \gamma = \mathrm{m} \enspace , \\ 0.7\,\mathrm{z}_1 + 0.3\,\mathrm{z}_2 & \text{if } \gamma = \mathrm{r} \enspace , \end{cases} \\
\mu_3(\sigma)_{(\perp,\perp),(\top,\perp)(\top,\perp)(\top,\perp)} &= 1\,\alpha \enspace .
\end{aligned}$$

If we compute $\|M\|(1\,t)$, it shows the expected result $0.48\,\alpha + 0.52\,\beta$. □

Finally, let us consider the second result, which states that bottom-up tree transformations are closed under right-composition with deterministic bottom-up tree transformations [9, Theorem 4.6] and [1, Theorem 6]. This result was also generalized to $\mathrm{BOT}(\mathcal{A})\,; \mathrm{bh–BOT}(\mathcal{A}) = \mathrm{BOT}(\mathcal{A})$ [10, Corollary 5.5]. Since we have already seen that our previous construction destroys determinism, we simplify the construction to obtain a construction which is the analogue of the construction for the unweighted case. Note that without loss of generality we may assume a bottom-up tree series transducer to be total; the construction required to show this is the standard one (add a transition into a trap state, if no transition is present).

Definition 4.10 Let $M' = (Q', \Sigma, \Gamma, \mathcal{A}, F', \mu')$ and $M'' = (Q'', \Gamma, \Delta, \mathcal{A}, F'', \mu'')$ be tree series transducers, of which M'' is bottom-up. The *(simple) composition of M' and M''*, denoted by $M' \,;_\mathrm{s} M''$, is defined to be the tree series transducer $M' \,;_\mathrm{s} M'' = (Q' \times Q'', \Sigma, \Delta, \mathcal{A}, F, \mu)$ with

$$F_{(p,q)} = \sum_{q' \in Q''} F''_{q'} \longleftarrow \big(h^q_{\mu''}\big(F'_p\big)_{q'}\big) \tag{15}$$

$$\mu_k(\sigma)_{(p,q),(p_1,q_1)(\mathrm{x}_{i_1})\cdots(p_n,q_n)(\mathrm{x}_{i_n})} = h^{q_1\cdots q_n}_{\mu''}\big(\mu'_k(\sigma)_{p,p_1(\mathrm{x}_{i_1})\cdots p_n(\mathrm{x}_{i_n})}\big)_q \tag{16}$$

for every $k, n \in \mathbb{N}$, $\sigma \in \Sigma_k$, $p, p_1, \ldots, p_n \in Q'$, $q, q_1, \ldots, q_n \in Q''$, and $i_1, \ldots, i_n \in [k]$. □

It is easily seen that $M' \,;_\mathrm{s} M''$ is bu-deterministic, whenever M' and M'' are bu-deterministic and bottom-up. Moreover, $M' \,;_\mathrm{s} M''$ is a homomorphism bottom-up tree series transducer, if M' and M'' are homomorphism bottom-up tree series transducers and M'' is boolean. Note that, in general, the restriction that M'' is boolean is necessary in the last statement, because otherwise the composition $M' ;_{\mathrm{s}} M''$ might not be total.

The next observation shows that boolean, total, and deterministic bottom-up tree series transducers transform every input tree into an output tree with coefficient 1. This essentially means that such transducers (at the level of h_μ) cannot implement "checking"; *i. e.*, selective rejection of some input trees. They may still reject input trees by entering a state whose top-most output is $\widetilde{0}$.

Observation 4.11 (*cf.* Proposition 4.11 of [15]) Let $M = (Q, \Sigma, \Delta, \mathcal{A}, F, \mu)$ be a deterministic bottom-up tree series transducer. Then for every $t \in T_\Sigma$ there exists at most one $q \in Q$ such that $h_\mu(t)_q \neq \widetilde{0}$. Moreover, if M is additionally total and boolean, then there exists a unique $q \in Q$ such that $h_\mu(t)_q = 1\, u$ for some $u \in T_\Delta$.

Proof: Essentially the proof can be found in the proof of [15, Proposition 4.11]. Zero-divisor freeness is not required because M is boolean and it is straightforward to show that $h_\mu(t)_q$ is boolean. □

Now we are ready to show correctness of the simple composition $M' ;_{\mathrm{s}} M''$ provided that M' and M'' are bottom-up tree series transducers, of which M'' is boolean, total, and deterministic. Moreover, we prove the correctness also for particular top-down tree series transducers.

Lemma 4.12 Let $M' = (Q', \Sigma, \Gamma, \mathcal{A}, F', \mu')$ and $M'' = (Q'', \Gamma, \Delta, \mathcal{A}, F'', \mu'')$ be a tree series transducers, of which M'' is bottom-up. Moreover, let $M = (Q, \Sigma, \Delta, \mathcal{A}, F, \mu)$ be the simple composition of M' and M''. Then for every $t \in T_\Sigma$, $p \in Q'$, and $q \in Q''$

$$h_{\mu''}\big(h_{\mu'}(t)_p\big)_q = h_\mu(t)_{(p,q)}$$

and $\|M'\| ; \|M''\| = \|M\|$ provided that:

(a) M_1 is bottom-up and M_2 is boolean, total, and deterministic; or

(b) M_1 is top-down.

Proof: We prove the statement inductively, so let $t = \sigma(t_1, \ldots, t_k)$ for some $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, and $t_1, \ldots, t_k \in T_\Sigma$.

$$\begin{aligned}
& h_{\mu''}\big(h_{\mu'}(\sigma(t_1, \ldots, t_k))_p\big)_q \\
= & \sum_{\substack{w' \in Q'(\mathrm{X}_k)^*,\\ w' = p_1(\mathrm{x}_{i_1}) \cdots p_n(\mathrm{x}_{i_n})}} h_{\mu''}\big(\mu'_k(\sigma)_{p,w'} \longleftarrow (h_{\mu'}(t_{i_j})_{p_j})_{j \in [n]}\big)_q \\
& \text{(by definition of } h_{\mu'}) \\
= & \sum_{\substack{w' \in Q'(\mathrm{X}_k)^*,\\ w' = p_1(\mathrm{x}_{i_1}) \cdots p_n(\mathrm{x}_{i_n})}} \sum_{\substack{u \in \mathrm{supp}(\mu'_k(\sigma)_{p,w'}),\\ (\forall j \in [n]) \colon u_j \in T_\Gamma}} (\mu'_k(\sigma)_{p,w'}, u) \cdot \prod_{j \in [n]} (h_{\mu'}(t_{i_j})_{p_j}, u_j) \cdot h_{\mu''}(u[u_j]_{j \in [n]})_q \\
& \text{(by definition of pure substitution)} \\
= & \sum_{\substack{w' \in Q'(\mathrm{X}_k)^*,\\ w' = p_1(\mathrm{x}_{i_1}) \cdots p_n(\mathrm{x}_{i_n})}} \sum_{\substack{u \in \mathrm{supp}(\mu'_k(\sigma)_{p,w'}),\\ (\forall j \in [n]) \colon u_j \in T_\Gamma}} (\mu'_k(\sigma)_{p,w'}, u) \cdot \prod_{j \in [n]} (h_{\mu'}(t_{i_j})_{p_j}, u_j) \cdot \\
& \quad \cdot \sum_{\overline{q} \in (Q'')^{\mathrm{var}(u)}} h^{\overline{q}}_{\mu''}(u)_q \longleftarrow \big(h_{\mu''}(u_j)_{\overline{q}_j}\big)_{j \in \mathrm{var}(u)} \\
& \text{(by Proposition 4.6(a) in Case (a) and Proposition 4.6(b) otherwise)} \\
= & \sum_{\substack{w' \in Q'(\mathrm{X}_k)^*,\\ w' = p_1(\mathrm{x}_{i_1}) \cdots p_n(\mathrm{x}_{i_n})}} \sum_{\substack{u \in \mathrm{supp}(\mu'_k(\sigma)_{p,w'}),\\ (\forall j \in [n]) \colon u_j \in T_\Gamma}} (\mu'_k(\sigma)_{p,w'}, u) \cdot \\
& \quad \cdot \prod_{j \in [n]} (h_{\mu'}(t_{i_j})_{p_j}, u_j) \cdot \sum_{q_1, \ldots, q_n \in Q''} h^{q_1 \cdots q_n}_{\mu''}(u)_q \longleftarrow \big(h_{\mu''}(u_j)_{q_j}\big)_{j \in [n]}
\end{aligned}$$

(because
(a) Observation 3.2 is applicable due to Observation 4.11
(b) M' is top-down; *i. e.*, $\mathrm{var}(u) = [n]$)

$$= \sum_{\substack{w \in Q(\mathrm{X}_k)^*, \\ w = (p_1,q_1)(\mathrm{x}_{i_1})\cdots(p_n,q_n)(\mathrm{x}_{i_n})}} h^{q_1\cdots q_n}_{\mu''}(\mu'_k(\sigma)_{p,p_1(\mathrm{x}_{i_1})\cdots p_n(\mathrm{x}_{i_n})})_q \longleftarrow \big(h_{\mu''}(h_{\mu'}(t_{i_j})_{p_j})_{q_j}\big)_{j \in [n]}$$

(by Propositions 3.3 and 3.4)

$$= \sum_{\substack{w \in Q(\mathrm{X}_k)^*, \\ w = (p_1,q_1)(\mathrm{x}_{i_1})\cdots(p_n,q_n)(\mathrm{x}_{i_n})}} \mu_k(\sigma)_{(p,q),w} \longleftarrow \big(h_\mu(t_{i_j})_{(p_j,q_j)}\big)_{j \in [n]}$$

(by definition of $\mu_k(\sigma)_{(p,q),w}$ and induction hypothesis)

$$= h_\mu(\sigma(t_1, \ldots, t_k))_{(p,q)}$$

(by definition of h_μ)

The proof of the second statement is literally the same as the proof of the second statement of Lemma 4.7. □

Thus we obtain the following theorem for bottom-up tree series transducers [see 10, Corollary 5.5]. It remains open to prove stronger statements for restricted semirings; *e. g.*, for idempotent semirings [18].

Theorem 4.13 Let $\mathcal{A}$ be a commutative and complete semiring.

$$[\mathrm{p}][\mathrm{n}][\mathrm{l}][\mathrm{d}][\mathrm{h}]\text{–}\mathrm{BOT}(\mathcal{A})\,;[\mathrm{p}][\mathrm{n}][\mathrm{l}][\mathrm{h}]\mathrm{bd}\text{–}\mathrm{BOT}(\mathcal{A}) = [\mathrm{p}][\mathrm{n}][\mathrm{l}][\mathrm{d}][\mathrm{h}]\text{–}\mathrm{BOT}(\mathcal{A}) \tag{17}$$

Proof: The statement follows from Lemma 4.12. □

5 Compositions of Top-down Tree Series Transformations

Let us first review the known results about compositions of top-down tree series transformations. Note that top-down tree transducers are essentially polynomial top-down tree series transducers over $\mathbb{B}$ (see [10, Section 4.3]) In [1, Theorem 1] it is shown that

$$\mathrm{p}\text{–}\mathrm{TOP}(\mathbb{B})\,;\mathrm{pnl}\text{–}\mathrm{TOP}(\mathbb{B}) \subseteq \mathrm{p}\text{–}\mathrm{TOP}(\mathbb{B}) \qquad \mathrm{pt}\text{–}\mathrm{TOP}(\mathbb{B})\,;\mathrm{pl}\text{–}\mathrm{TOP}(\mathbb{B}) \subseteq \mathrm{p}\text{–}\mathrm{TOP}(\mathbb{B})$$
$$\mathrm{d}\text{–}\mathrm{TOP}(\mathbb{B})\,;\mathrm{pn}\text{–}\mathrm{TOP}(\mathbb{B}) \subseteq \mathrm{p}\text{–}\mathrm{TOP}(\mathbb{B}) \qquad \mathrm{dt}\text{–}\mathrm{TOP}(\mathbb{B})\,;\mathrm{p}\text{–}\mathrm{TOP}(\mathbb{B}) \subseteq \mathrm{p}\text{–}\mathrm{TOP}(\mathbb{B}) \ .$$

Some results were extended to arbitrary commutative and complete semirings $\mathcal{A}$ in [20, Theorem 2.4], which shows that

$$\mathrm{nl}\text{–}\mathrm{TOP}(\mathcal{A})\,;\mathrm{nl}\text{–}\mathrm{TOP}(\mathcal{A}) = \mathrm{nl}\text{–}\mathrm{TOP}(\mathcal{A}) \ ,$$

and in [10, Theorem 5.18], which shows that

$$[\mathrm{n}][\mathrm{l}]\mathrm{d}\text{–}\mathrm{TOP}(\mathcal{A})\,;\mathrm{dnl}\text{–}\mathrm{TOP}(\mathcal{A}) = [\mathrm{n}][\mathrm{l}]\mathrm{d}\text{–}\mathrm{TOP}(\mathcal{A})$$
$$[\mathrm{n}][\mathrm{l}]\mathrm{bdt}\text{–}\mathrm{TOP}(\mathcal{A})\,;[\mathrm{n}][\mathrm{l}]\mathrm{d}\text{–}\mathrm{TOP}(\mathcal{A}) = [\mathrm{n}][\mathrm{l}]\mathrm{d}\text{–}\mathrm{TOP}(\mathcal{A}) \ .$$

Without any additional construction we can already generalize the former statement of [10, Theorem 5.18]. We basically exploit the fact that nondeleting, linear top-down tree series transducers are as powerful as nondeleting, linear bottom-up tree series transducers [see 10, Theorem 5.24]. Thus given two top-down tree series transducers M' and M'', of which M'' is nondeleting and linear, we first construct a nondeleting, linear bottom-up tree series transducer M_2 such that $\|M_2\| = \|M''\|$. Note that M_2 is td-deterministic (but not necessarily bu-deterministic) whenever M'' is td-deterministic. Then we can apply the simple composition to M' and M_2 (see Definition 4.10) and obtain a tree series transducer M. It is easily seen that M is top-down, because M_2 is nondeleting and linear. Moreover, M is td-deterministic if M' and M_2 are td-deterministic.

Theorem 5.1 Let $\mathcal{A}$ be a commutative and complete semiring.

$$\text{[n][l][d]–TOP}(\mathcal{A})\,;\text{[d]nl–TOP}(\mathcal{A}) = \text{[n][l][d]–TOP}(\mathcal{A})$$

Proof: The decomposition is trivial, so it remains to check the composition. Let M' and M'' be top-down tree series transducers such that M'' is nondeleting and linear. By [10, Theorem 5.24] there exists a nondeleting, linear bottom-up tree series transducer M_2 such that $\|M_2\| = \|M''\|$. Moreover, the td-determinism property is preserved by this construction. Let $M = M' \,;_{\mathrm{s}} M_2$. By Lemma 4.12 we have $\|M\| = \|M'\|\,;\|M_2\|$. Moreover, it is easily observed that M is in fact top-down, because M_2 is nondeleting and linear. Moreover, M is td-deterministic (respectively, nondeleting, linear), if M' and M_2 are td-deterministic (respectively, nondeleting, linear). □

Using the same apparatus, we should also like to generalize the latter statement of [10, Theorem 5.18]; *i. e.*,

$$\text{[n][l]bdt–TOP}(\mathcal{A})\,;\text{[n][l]d–TOP}(\mathcal{A}) = \text{[n][l]d–TOP}(\mathcal{A}) \ .$$

So let M' and M'' be top-down tree series transducers. The first step is to construct a bottom-up tree series transducer M_2, which is semantically-equivalent to M''. However, if M'' is not linear, then, in general, such a tree series transducer need not exist [because p–TOP($\mathbb{B}$) $\not\subseteq$ p–BOT($\mathbb{B}$)]. Thus we restrict ourselves to linear M''. Consequently, let M' be boolean, deterministic, and total, and let M'' be linear. We first construct a linear bottom-up tree series transducer M_2 that computes the same tree series transformation as M'' (we follow the construction found in [15, Theorem 4.26]). The advantage of M_2 is that Proposition 4.6 is applicable to it. Then we apply the composition to M' and M_2 and obtain a tree series transducer M_1 that computes the tree series transformation $\|M_1\| = \|M'\|\,;\|M_2\|$. Finally, we observe an important property (namely, that "checking followed by deletion" is not possible) and manipulate M_1 such that we obtain a top-down tree series transducer M that computes $\|M\| = \|M_1\|$. First we need an easy observation.

Observation 5.2 (*cf.* Proposition 4.12 of [15]) Let $M = (Q, \Sigma, \Delta, \mathcal{A}, F, \mu)$ be a boolean, deterministic, and total top-down tree series transducer. Then for every $t \in T_\Sigma$ there exists a unique $q \in Q$ such that $h_\mu(t)_q = 1\, u$ for some $u \in T_\Delta$.

Proof: Essentially the proof can be found in the proof of [15, Proposition 4.12]. Zero-divisor freeness is not required because M is boolean and it is straightforward to show that $h_\mu(t)_q$ is boolean. □

Theorem 5.3 Let $\mathcal{A}$ be a commutative and complete semiring.

$$\text{bdt–TOP}(\mathcal{A})\,;\text{l–TOP}(\mathcal{A}) \subseteq \text{TOP}(\mathcal{A})$$

Proof: Let $M' = (Q', \Sigma, \Gamma, \mathcal{A}, F', \mu')$ be a boolean, deterministic, and total top-down tree series transducer, and let $M'' = (Q'', \Gamma, \Delta, \mathcal{A}, F'', \mu'')$ be a linear top-down tree series transducer. First we construct the linear bottom-up tree series transducer $M_2 = (Q_2, \Gamma, \Delta, \mathcal{A}, F_2, \mu_2)$ from M'' as presented in [15, Definition 4.24]. Clearly, $\|M_2\| = \|M''\|$ by [15, Lemma 5.25]. Moreover, it is noteworthy that we have the following two properties. There is a state $\perp \in Q_2$ and an $\alpha \in \Delta_0$ such that:

(a) $h_{\mu_2}(t)_\perp = 1\, \alpha$ for every $t \in T_\Gamma$; and

(b) for every $k \in \mathbb{N}$, $i \in [k]$, $\gamma \in \Gamma_k$, $q, q_1, \ldots, q_k \in Q_2$, and $u \in \mathrm{supp}((\mu_2)_k(\gamma)_{q,q_1\cdots q_k})$

$$i \in \mathrm{var}(u) \quad \Longleftrightarrow \quad q_i \neq \perp \ .$$

Now we may compose M' with M_2 using the simple composition (see Definition 4.10). We obtain the tree series transducer $M_1 = M' \,;_{\mathrm{s}} M_2$ (actually M_1 is a tree series transducer of type II [23]) with $M_1 = (Q_1, \Sigma, \Delta, \mathcal{A}, F_1, \mu_1)$. We show that M_1 has the following properties (*cf.* [23, Lemma 2]):

(i) $h_{\mu_1}(t)_{(p,\perp)} = 1\, \alpha$ for every $t \in T_\Sigma$ and $p \in Q'$;

(ii) $\mathrm{supp}((\mu_1)_k(\sigma)_{q,w})$ is linear for every $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, $q \in Q_1$, and $w \in Q_1(\mathrm{X}_k)^*$; and

(iii) for every $k \in \mathbb{N}$, $i \in [n]$, $\sigma \in \Sigma_k$, $(p,q) \in Q_1$, $w = (p_1,q_1)(\mathrm{x}_{i_1}) \cdots (p_n,q_n)(\mathrm{x}_{i_n}) \in Q_1(\mathrm{X}_k)^*$, and $u \in \mathrm{supp}((\mu_1)_k(\sigma)_{(p,q),w})$

$$i \in \mathrm{var}(u) \qquad \Longleftrightarrow \qquad q_i \neq \bot \ .$$

(i) By the proof of Lemma 4.12 we know that $h_{\mu_1}(t)_{(p,\bot)} = h_{\mu_2}(h_{\mu'}(t)_p)_\bot$. By Observation 5.2 we know that $h_{\mu'}(t)_p = 1\,u$ for some $u \in T_\Gamma$. Moreover, by Property (a) we have that $h_{\mu_2}(1\,u)_\bot = 1\,\alpha$; thus $h_{\mu_1}(t)_{(p,\bot)} = 1\,\alpha$.
(ii–iii) These properties are easily observed because M' is output-linear and output-nondeleting and M_2 is linear. For Property (iii) one also needs Statement (b).

Let $n \in \mathbb{N}$. We define $\mathrm{norm}_n \colon T_\Delta(\mathrm{Z}_n) \longrightarrow T_\Delta(\mathrm{Z}_n)$ by $\mathrm{norm}_n(u) = \mathrm{norm}_n(u,1)$ for every $u \in T_\Delta(\mathrm{Z}_n)$ where

$$\mathrm{norm}_n(u,n) = u$$
$$\mathrm{norm}_n(u,i) = \begin{cases} \mathrm{norm}_n(u,i+1) & \text{if } i \in \mathrm{var}(u), \\ \mathrm{norm}_{n-1}(u[z_{j-1}]_{j \in [n] \setminus [i]}, i) & \text{otherwise} \end{cases}$$

for every $i \in [n-1]$. Thus $\mathrm{norm}_3(\mathrm{z}_3) = \mathrm{z}_1$. Further, we define the mapping $\mathrm{del} \colon Q_1(\mathrm{X})^* \longrightarrow Q_1(\mathrm{X})^*$ for every $(p,q) \in Q_1$, $i \in \mathbb{N}_+$, and $w \in Q_1(\mathrm{X})^*$ by

$$\mathrm{del}(\varepsilon) = \varepsilon$$
$$\mathrm{del}((p,q)(\mathrm{x}_i) \cdot w) = \begin{cases} \mathrm{del}(w) & \text{if } q = \bot, \\ (p,q)(\mathrm{x}_i) \cdot \mathrm{del}(w) & \text{if } q \neq \bot \ . \end{cases}$$

We obtain $M = (Q_1, \Sigma, \Delta, \mathcal{A}, F_1, \mu)$ as follows. For every $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, $q \in Q_1$, and $w = q_1(\mathrm{x}_{i_1}) \cdots q_n(\mathrm{x}_{i_n}) \in Q_1(\mathrm{X}_k)^*$ let

$$\mu_k(\sigma)_{q,w} = \sum_{w' \in Q_1(\mathrm{X}_k)^*, \mathrm{del}(w') = w} \Big(\sum_{u' \in T_\Delta(\mathrm{Z})} ((\mu_1)_k(\sigma)_{q,w'}, u')\ \mathrm{norm}_{|w'|}(u') \Big) \ .$$

Clearly, M is a top-down tree series transducer. We prove

$$h_\mu(t)_{(p,q)} = h_{\mu_1}(t)_{(p,q)}$$

for every $t \in T_\Sigma$ and $(p,q) \in Q_1$ such that $q \neq \bot$. Let $t = \sigma(t_1, \dots, t_k)$ for some $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, and $t_1, \dots, t_k \in T_\Sigma$.

$$\begin{aligned}
& h_\mu(\sigma(t_1, \dots, t_k))_{(p,q)} \\
= & \sum_{\substack{w \in Q(\mathrm{X}_k)^*, \\ w = (p_1,q_1)(\mathrm{x}_{i_1}) \cdots (p_n,q_n)(\mathrm{x}_{i_n})}} \mu_k(\sigma)_{(p,q),w} \longleftarrow (h_\mu(t_{i_j})_{(p_j,q_j)})_{j \in [n]} \\
& \text{(by definition of } h_\mu) \\
= & \sum_{\substack{w \in Q(\mathrm{X}_k)^*, \\ w = (p_1,q_1)(\mathrm{x}_{i_1}) \cdots (p_n,q_n)(\mathrm{x}_{i_n})}} \mu_k(\sigma)_{(p,q),w} \longleftarrow (h_{\mu_1}(t_{i_j})_{(p_j,q_j)})_{j \in [n]} \\
& \text{(by induction hypothesis because } q_j \neq \bot) \\
= & \sum_{\substack{w \in Q(\mathrm{X}_k)^*, \\ w = (p_1,q_1)(\mathrm{x}_{i_1}) \cdots (p_n,q_n)(\mathrm{x}_{i_n})}} \Big(\sum_{w' \in Q_1(\mathrm{X}_k)^*, \mathrm{del}(w') = w} \\
& \quad \Big(\sum_{u' \in T_\Delta(\mathrm{Z})} ((\mu_1)_k(\sigma)_{(p,q),w'}, u')\ \mathrm{norm}_{|w'|}(u') \Big) \Big) \longleftarrow (h_{\mu_1}(t_{i_j})_{(p_j,q_j)})_{j \in [n]} \\
& \text{(by definition of } \mu_k(\sigma)_{(p,q),w})
\end{aligned}$$

$$= \sum_{\substack{w' \in Q_1(\mathrm{X}_k)^*, \\ \mathrm{del}(w') = (p_1,q_1)(\mathrm{x}_{i_1}) \cdots (p_n,q_n)(\mathrm{x}_{i_n})}} \Big(\sum_{u' \in T_\Delta(\mathrm{Z})} ((\mu_1)_k(\sigma)_{(p,q),w'}, u') \ \mathrm{norm}_{|w'|}(u') \Big)$$
$$\longleftarrow (h_{\mu_1}(t_{i_j})_{(p_j,q_j)})_{j \in [n]}$$
$$= \sum_{\substack{w' \in Q_1(\mathrm{X}_k)^*, \\ w' = (p_1,q_1)(\mathrm{x}_{i_1}) \cdots (p_n,q_n)(\mathrm{x}_{i_n})}} (\mu_1)_k(\sigma)_{(p,q),w'} \longleftarrow (h_{\mu_1}(t_{i_j})_{(p_j,q_j)})_{j \in [n]}$$
(by Observation 3.2 because $h_{\mu_1}(t_{i_j})_{(p_j,\perp)} = 1\ \alpha$)
$$= h_{\mu_1}(\sigma(t_1, \ldots, t_k))_{(p,q)}$$
(by definition of h_{μ_1})

It follows that $\|M\| = \|M_1\|$ and thus the main statement is proved. □

Acknowledgements: The author wishes to express his deepest gratitude to the referees of the conference version [22] of this paper. Their comments enabled the author to substantially improve the presentation of the results.

References

[1] B. S. Baker. Composition of top-down and bottom-up tree transductions. *Inform. Comput.*, 41(2):186–213, 1979.

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

[3] B. Borchardt. Code selection by tree series transducers. In *Proc. 9th Int. Conf. on Implementation and Application of Automata*, volume 3317 of LNCS, pages 57–67. Springer, 2004.

[4] B. Borchardt and H. Vogler. Determinization of finite state weighted tree automata. *J. Autom. Lang. Combin.*, 8(3):417–463, 2003.

[5] S. Bozapalidis. Equational elements in additive algebras. *Theory Comput. Systems*, 32(1): 1–33, 1999.

[6] S. Bozapalidis. Context-free series on trees. *Inform. Comput.*, 169(2):186–229, 2001.

[7] S. Bozapalidis and G. Rahonis. On the closure of recognizable tree series under tree homomorphism. In M. Droste and H. Vogler, editors, *Weighted Automata—Theory and Applications*, page 34. Technische Universität Dresden, 2004.

[8] K. Culik II and J. Kari. Digital images and formal languages. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3 — Beyond Words, chapter 10, pages 599–616. Springer, 1997.

[9] J. Engelfriet. Bottom-up and top-down tree transformations—a comparison. *Math. Systems Theory*, 9(3):198–231, 1975.

[10] J. Engelfriet, Z. Fülöp, and H. Vogler. Bottom-up and top-down tree series transformations. *J. Autom. Lang. Combin.*, 7(1):11–70, 2002.

[11] J. Engelfriet and E. M. Schmidt. IO and OI I. *J. Comput. System Sci.*, 15(3):328–353, 1977.

[12] Z. Ésik and W. Kuich. Formal tree series. *J. Autom. Lang. Combin.*, 8(2):219–285, 2003.

[13] C. Ferdinand, H. Seidl, and R. Wilhelm. Tree automata for code selection. *Acta Inform.*, 31 (8):741–760, 1994.

[14] Z. Fülöp, Z. Gazdag, and H. Vogler. Hierarchies of tree series transformations. *Theoret. Comput. Sci.*, 314:387–429, 2004.

[15] Z. Fülöp and H. Vogler. Tree series transformations that respect copying. *Theory Comput. Systems*, 36(3):247–293, 2003.

[16] J. S. Golan. *Semirings and their Applications.* Kluwer Academic, Dordrecht, 1999.

[17] J. Graehl and K. Knight. Training tree transducers. In S. Dumais, D. Marcu, and S. Roukos, editors, *Proc. of the Human Language Technology Conf. of the North American Chapter of the ACL*, pages 105–112. Association for Computational Linguistics, 2004.

[18] U. Hebisch and H. J. Weinert. *Semirings—Algebraic Theory and Applications in Computer Science.* World Scientific, Singapore, 1998.

[19] W. Kuich. Formal power series over trees. In S. Bozapalidis, editor, *Proc. 3rd Int. Conf. on Developments in Language Theory*, pages 61–101. Aristotle University of Thessaloniki, 1997.

[20] W. Kuich. Full abstract families of tree series I. In J. Karhumäki, H. A. Maurer, G. Paun, and G. Rozenberg, editors, *Jewels are Forever*, pages 145–156. Springer, 1999.

[21] W. Kuich. Tree transducers and formal tree series. *Acta Cybernet.*, 14(1):135–149, 1999.

[22] A. Maletti. Compositions of bottom-up tree series transformations. In Z. Ésik and Z. Fülöp, editors, *Proc. 11th Int. Conf. Automata and Formal Languages*, pages 187–199. University of Szeged, 2005.

[23] A. Maletti. The power of tree series transducers of type I and II. In C. de Felice and A. Restivo, editors, *Proc. 9th Int. Conf. Developments in Language Theory*, volume 3572 of LNCS, pages 338–349. Springer, 2005.

[24] M. Mohri. Finite-state transducers in language and speech processing. *Comput. Linguist.*, 23 (2):269–311, 1997.

[25] W. C. Rounds. Mappings and grammars on trees. *Math. Systems Theory*, 4(3):257–287, 1970.

[26] J. W. Thatcher. Generalized^2 sequential machine maps. *J. Comput. System Sci.*, 4(4):339–367, 1970.