Bottom-up and Top-down Tree Series Transformations

Zoltán Fülöp *
Department of Computer Science
University of Szeged
Árpád tér 2, H-6720 Szeged, Hungary
E-mail: fulop@inf.u-szeged.hu

Heiko Vogler
Department of Computer Science
Technical University of Dresden
D-01062 Dresden, Germany
E-mail: vogler@inf.tu-dresden.de

October 23, 2000

Abstract: We generalize bottom-up tree transducers and top-down tree transducers to the concept of bottom-up tree series transducer and top-down tree series transducer, respectively, by allowing formal tree series as output rather than trees, where a formal tree series is a mapping from output trees to some semiring. We associate two semantics with a tree series transducer: a mapping which transforms trees into tree series (for short: tree to tree series transformation or t-ts transformation), and a mapping which transforms tree series into tree series (for short: tree series transformation or ts-ts transformation).

We show that the standard case of tree transducers is reobtained by choosing the boolean semiring under the t-ts semantics. Also, for each of the two types of tree series transducers and for both types of semantics, we prove a characterization which generalizes in a straightforward way the corresponding characterization result for the underlying tree transducer class. More precisely, we prove that polynomial bottom-up tree series transducers can be characterized by the composition of finite state relabeling tree series transducers and homomorphism tree series transducers. Moreover, we prove that the total deterministic top-down tree series transducers can be characterized by the composition of homomorphism tree series transducers and linear total deterministic top-down tree series transducers.

1 Introduction

The investigation of this paper was inspired by [Kui99] in which a first attempt was made to integrate formal power series over trees into a restricted form of top-down tree transducer. The resulting tree series transducers transform an input tree into a formal tree series which is a mapping from the set of output trees into some semiring. By this means it is possible to measure the computation of output trees. Here we introduce the concepts of bottom-up tree series transducer and of (unrestricted) top-down tree series transducer. Before we discuss our investigation, we will now briefly review the origins of tree series transducers which are a) tree transducers and b) automata with multiplicity (or cost functions).

Tree transducers have been introduced in [Rou68, Rou70, Tha70]. They can be viewed as generalizations of generalized sequential machines [Ber79] to trees where the trees are either read and processed from their leaves towards their root (bottom-up tree transducer) or from their roots towards their leaves (top-down tree transducer). A rule (or: transition) of a bottom-up tree transducer looks like

$$\sigma(q_1(x_1), \ldots, q_k(x_k)) \rightarrow q(t)$$

and a rule of a top-down tree transducer has the form

$$q(\sigma(x_1, \ldots, x_k)) \rightarrow \zeta$$

where $q, q_1, \ldots, q_k$ are states of the tree transducers, $\sigma$ is an input symbol of rank $k$, $x_1, \ldots, x_k$ are variables ranging over trees (more precisely, over output trees in the bottom-up case, and over input trees

*Research of this author was supported by the Hungarian Scientific Foundation (OTKA) under Grant T 030084
in the top-down case), \( t \) is an output tree which may contain variables from the set \( X_k = \{ x_1, \ldots, x_k \} \), and \( \zeta \) is an output tree which may contain constructs of the form \( p(x_i) \) at its leaves where \( p \) is a state and \( 1 \leq i \leq k \). In the usual way, the rules of a tree transducer constitute a binary term rewriting relation (or: derivation relation) \( \Rightarrow_M \) by means of which the tree transformation \( \tau_M \subseteq T_S \times T_D \) computed by \( M \) can be defined (where \( T_S \) and \( T_D \) are the sets of input trees and output trees, respectively).

Since the seventies, tree transducers have been studied intensively. The first substantial papers dealt with (de-)composition and hierarchy results [Bak79, Eng75, Eng77, Eng82]. In [FV92] a method of deciding the equivalence of the compositions of classes of tree transformations is overviewed. Survey articles and books are [GS84, GS97, CDG+97, FY98]. Recently, a characterization of tree transformation classes in terms of monadic second order logic has been proved [BE00, EM99].

Now let us briefly review the second order of tree transducers: automata with multiplicities. Let \( M = (Q, \Sigma, \mu, q_0, Q_d) \) be a usual finite state string automaton where \( Q \) is the set of states, \( \Sigma \) the set of input symbols, \( \mu : Q \times \Sigma \rightarrow \mathcal{P}(Q) \) the transition function, \( q_0 \in Q \) the initial state, and \( Q_d \subseteq Q \) the set of final states. In the usual way, the transition function is extended to the function \( \mu : \mathcal{P}(Q) \times \Sigma^* \rightarrow \mathcal{P}(Q) \), and the language accepted by \( M \) is the language \( L(M) = \{ w \in \Sigma^* | \mu(q_0, w) \cap Q_d \neq \emptyset \} \).

In [Sch62], finite state string automata were generalized by associating with every transition an element \( a \in A \) of a semiring \( (A, +, \cdot, 0, 1) \) assumed to be the cost (or multiplicity) of this transition (also cf. [Eil74, SS78, BR88, KS86]). In this approach, the transition function \( \mu \) turns into a \( (Q \times Q) \)-matrix \( \mu \in (A(\Sigma))^{Q \times Q} \) where \( A(\Sigma) \) is the set of formal power series (in non-commuting variables), i.e., mappings of type \( \Sigma \rightarrow A \). Intuitively, \( \mu_{q, q'}(\sigma) \in A \) is the cost of making a transition from state \( q \) to state \( p \) while reading \( \sigma \) (note the order of the indices \( p \) and \( q \)). The usual automaton (without cost functions) is reobtained by considering the boolean semiring. Then \( \mu \) can be used to define the behaviour of a finite state automaton \( M \) with cost functions such that it is a formal power series \( \tau_M \in A(\Sigma^*) \) obtained by iterating the transition matrix \( \mu \).

This approach was generalized in [Kui97a] to tree automata over semirings (for a theoretical investigation on formal tree series, i.e., mappings of the type \( T_S \rightarrow A \), cf. [BR82]; in that paper, \( A \) is a vector space). Since the input symbols now have ranks, the transition matrix \( \mu \) has to be replaced by a family \( \mu = (\mu_k | k \geq 0) \) of transition matrices \( \mu_k \in (A(\Sigma(X_k)))^{Q^k \times Q^k} \) where \( \Sigma(X_k) = \{ \sigma(x_1, \ldots, x_k) | \sigma \in \Sigma^k \} \). Intuitively, if the tree automaton has already accepted the input trees \( t_1, \ldots, t_k \) in the states \( q_1, \ldots, q_k \), respectively, then \( (\mu_k)_{q_1q_2 \ldots q_k}(\sigma(x_1, \ldots, x_k)) \in A \) is the cost for making a transition from \( q_1, \ldots, q_k \) to \( q \) while reading the \( k \)-ary symbol \( \sigma \). The semantics of a tree automaton \( M \) with cost functions is a formal tree series \( \tau_M \in A(\langle T_S \rangle) \). We mention that tree automata with cost functions have been used in order to select code in compilers on the basis of certain cost criteria [FSW94] (also cf. [Sei92]).

In [Kui99] such tree automata with cost functions have been equipped with output and thereby the concept of tree transducer with cost functions has been introduced. In order to make this concept understandable on the background of the development explained above, let us reconsider the tree automata with cost functions. Such an automaton can be considered as a generating device: the trees together with their costs are built up and maintained in the entries of a matrix of type \( (A(\langle T_S \rangle))^Q \times 1 \).

An alternative point of view is to formalize a tree automaton with cost functions as a transducer which computes the partial identity. More precisely, it computes the function \( \tau_M : T_S \rightarrow A(\langle T_S \rangle) \) such that the formal tree series \( \tau_M(s) \) has the value 0 for every tree \( t \) such that \( t \neq s \). Then, the transitions are specified by a family \( \mu = (\mu_k | k \geq 0) \) of mappings \( \mu_k : \Sigma^k \rightarrow (A(\langle T_S \rangle))^{Q^k \times Q^k} \) for some output alphabet \( \Delta \). On the basis of these tree representations, in [Kui99] restricted types of top-down tree transducers with cost functions have been introduced and studied; the underlying transducer model is called nondeterministically simple top-down tree transducer in which, roughly speaking, to every subtree at most one state is applied (cf. [GS84]). Then a tree transducer \( M \) with cost functions computes the function \( \tau_M : T_S \rightarrow A(\langle T_D \rangle) \). This finishes the short review about the origins of our investigation.

In the present paper, we take a more liberal point of view on tree representations. We consider tree representations of the form \( \mu_k : \Sigma_k \rightarrow (A(\langle T_D \rangle))^{Q^k \times Q^k} \) such that only for finitely many indices \( (q, w) \in Q \times (Q(X_k))^n \), \( \mu_k(\sigma)_{qw} \neq 0 \), where 0 is the formal tree series which maps every tree to 0. The following table summarizes the explained development of the function \( \mu \).
Next we instantiate these tree representations in two different ways thereby obtaining the bottom-up and the top-down case. In the bottom-up case we additionally require that, for every index \( w \), if \( w \neq q_1(x_1) \ldots q_k(x_k) \) for some \( q_1, \ldots, q_k \in Q \), then \( \mu_\delta(q_\sigma) = 0 \). Moreover, if \( \mu_\delta(q_\sigma) \neq 0 \), then \( \mu_\delta(q_\sigma) \in A(\langle T_\Delta(X_k) \rangle) \). Then we call \( \mu \) a bottom-up tree representation. To show an example, let us consider the two rules

\[
\delta(q_1(x_1), q_2(x_2), q_3(x_3)) \rightarrow q_\gamma(q_\sigma(x_2, \sigma(x_3, x_2))), \quad p(\sigma(x_1, \sigma(\gamma(x_3), x_1)))
\]

of some bottom-up tree transducer, where \( \delta, \sigma, \gamma \) are input/output symbol of rank 3, 2, and 1, respectively, \( q_1, q_2, q_3 \) are states, and | \( p \) delimits the right-hand sides of the two rules. Let us assume that the costs for these two rules are \( a \) and \( b \), respectively, where \( a, b \in A \). Then these rules are represented by

\[
\mu_\delta(q_\sigma) = a \quad \mu_\gamma(q_\sigma) = b
\]

where \( w = q_1(x_1)q_2(x_2)q_3(x_3) \). We observe that the structure of \( w \) is very restricted: \( w \) is the left-hand side of the rule from which the input symbol (and the commas) have been stripped off. On the other hand, the structure of the trees \( \sigma(x_2, \sigma(x_3, x_2)) \) and \( \sigma(x_1, \sigma(\gamma(x_3), x_1)) \) to which the mapping \( \mu_\delta(q_\sigma) \) is applied, is liberal in the sense that variables of the left-hand side of the rule may occur an arbitrary number of times.

In the top-down case we require that, if \( \mu_\delta(q_\sigma) \neq 0 \), then \( \mu_\delta(q_\sigma) \in A(\langle T_\Delta(X_1) \rangle) \), where \( l = |w| \) and \( T_\Delta(X_1) \) denotes the set of all trees \( t \) over \( \Delta \cup X_1 \) such that, for every \( 1 \leq i \leq k \), the variable \( x_i \) occurs exactly once in \( t \) and, reading the leaves of \( t \) from left to right, the variables occur in the order \( x_1 < x_2 < \ldots < x_n \). Then \( \mu \) is a top-down tree representation. As an example, let us consider the two rules

\[
q_\delta(q_1(x_1, x_2, x_3)) \rightarrow q_\gamma(p(x_2), q(x_1), p(x_1))), \quad \gamma(q(x_2), p(x_2))
\]

with costs \( c \) and \( d \), respectively. These rules are represented by

\[
\mu_\delta = c \quad \mu_\gamma = d
\]

where \( v = p(x_2)q(x_1) \) and \( v' = q(x_2)p(x_2) \). We observe that now \( v \) and \( v' \) are more flexible: they are obtained by scanning the frontier of the right-hand side of the rules and listing all objects of the form \( p(x_i) \). On the other hand, the trees \( \gamma(x_1, \gamma(x_2, x_3)) \) and \( \gamma(x_1) \), \( x_2 \) to which the mappings \( \mu_\delta(q_\sigma) \) and \( \mu_\gamma(q_\sigma) \) are applied, respectively, have a much more restricted form than in the bottom-up case: they are linear in the variables and the variables occur ordered.

The bottom-up tree representation and the top-down tree representation define the bottom-up tree series transducer and the top-down tree series transducer, respectively, via the concept of the initial homomorphism \( h_\mu : T_\Sigma \rightarrow A(\langle T_\Sigma \rangle) \). With every tree series transducer, we associate two semantics: the tree to tree series semantics (t-ts semantics) which is a mapping of type \( T_\Sigma \rightarrow A(\langle T_\Delta \rangle) \), and the tree series to tree series semantics (ts-ts semantics), which is a mapping of type \( A(\langle T_\Sigma \rangle) \rightarrow A(\langle T_\Delta \rangle) \). As in
the case of tree transducers (cf. [Eng75]) we define certain important subclasses, like linear, relabeling, and homomorphism tree series transducers.

The contents of this paper is as follows. In Section 2 we develop the notions and basic definitions of bottom-up and top-down tree transducers, semirings, formal tree series; we tried to make this paper self contained. In Section 3 we introduce the concepts of bottom-up tree series transducers and of top-down tree series transducers, and we discuss an example in detail. In Section 4 we show that, using the boolean semiring, there is a one-to-one correspondence between tree series transducers and tree transducers (of the same type). In Section 5 we prove two characterization results: 1) polynomial bottom-up tree series transducers can be characterized by the composition of finite state relabeling tree series transducers and homomorphism tree series transducers, and 2) deterministic top-down tree series transducers can be characterized by the composition of homomorphism tree series transducers and linear deterministic top-down tree series transducers.

The proofs of the correctness of constructions is usually done by an induction on the tree structure. In order to save space, we only show the induction step, and not the induction base.

2 Preliminaries

2.1 Sets and relations

Let \( H \) be a set. Then \( \mathcal{P}(H) \) is the powerset of \( H \). The empty set is denoted by \( \emptyset \). Let \( F, G \), and \( H \) be sets and let \( \varrho \subseteq F \times G \) and \( \tau \subseteq G \times H \) be two binary relations. The composition of \( \varrho \) and \( \tau \) is the binary relation \( \varrho \circ \tau = \{ (a, c) \subseteq F \times H \mid \text{there is a } b \in G \text{ such that } (a, b) \in \varrho \text{ and } (b, c) \in \tau \} \). The notion of composition is extended to classes of relations. For two classes \( C_1 \) and \( C_2 \) of relations we define \( C_1 \circ C_2 = \{ \varrho \circ \tau \mid \varrho \in C_1 \text{ and } \tau \in C_2 \} \).

2.2 Strings

If \( \Sigma \) is an alphabet, then \( \Sigma^* \) denotes the set of strings over \( \Sigma \), while the empty string is denoted by \( \varepsilon \). The length of a string \( w \in \Sigma^* \) is defined in a standard way and denoted by \( |w| \).

2.3 Trees and tree transducers

Let \( \Sigma \) be a ranked alphabet. By \( \Sigma^{(k)} \) we denote the set of all symbols of \( \Sigma \) which have rank \( k \). Moreover, let \( H \) be a set disjoint with \( \Sigma \). The set of (finite, labeled and ordered) trees over \( \Sigma \) indexed by \( H \), denoted by \( T_\Sigma(H) \), is the smallest subset \( T \) of \( (\Sigma \cup H \cup \{ \}, \{ \})^* \) such that (i) \( H \subseteq T \) and (ii) if \( \sigma \in \Sigma^{(k)} \) with \( k \geq 0 \) and \( s_1, \ldots, s_k \in T \), then \( \sigma(s_1, \ldots, s_k) \in T \). In case \( k = 0 \), we identify \( \sigma(\varepsilon) \) with \( \sigma \). Moreover, \( T_\Sigma(\emptyset) \) is denoted by \( T_\Sigma \). It should be clear that \( T_\Sigma = \emptyset \) if and only if \( \Sigma^{(0)} = \emptyset \). Since we are not interested in this particular case, we assume that \( \Sigma^{(0)} \neq \emptyset \) for every ranked alphabet \( \Sigma \) appearing as input or output ranked alphabet of some tree transducer in this paper.

We will need the set \( X = \{ x_1, x_2, \ldots \} \) of variable symbols. For every \( k \geq 0 \), we define \( X_k = \{ x_1, \ldots, x_k \} \), thus \( X_0 = \emptyset \). We use the variables to occur in trees, so we will frequently consider the sets \( T_\Sigma(X), T_\Sigma(X_k), \) etc. of trees where \( \Sigma \) is a ranked alphabet.

The tree substitution is defined as follows. Let \( t \in T_\Sigma(X_k) \) for some ranked alphabet \( \Sigma \) and \( k \geq 0 \). Moreover, let \( t_1, \ldots, t_k \) be also trees over (maybe other) ranked alphabets. Then \( t[x_1 \leftarrow t_1, \ldots, x_k \leftarrow t_k] \), or shortly just \( t[t_1, \ldots, t_k] \), stands for the tree which is obtained from \( t \) by substituting, for every \( 1 \leq i \leq k \), the tree \( t_i \) for each occurrence of \( x_i \).

A tree \( t \in T_\Sigma(X_k) \) is called linear if every variable \( x_i \), \( 1 \leq i \leq k \) occurs at most once in it.

We distinguish a subset \( T_\Sigma(X_k) \) of \( T_\Sigma(X_k) \) as follows. Let a tree \( t \in T_\Sigma(X_k) \) be in \( T_\Sigma(X_k) \) if for every \( 1 \leq i \leq k \), the variable \( x_i \) occurs exactly once in \( t \) and, reading the leaves of \( t \) from left to right, the variables occur in the order \( x_1 < x_2 < \ldots < x_k \). Note that all elements of \( T_\Sigma(X_k) \) are linear.

Let \( t \in T_\Sigma(H) \). The linearization of \( t \) with respect to \( H \), denoted by \( \text{lin}_H(t) \), is defined as the unique pair \( (t', w) \) where \( t' \in T_\Sigma(X_k) \) and \( w = a_1 \ldots a_k \in H^* \) such that \( t = t'[x_1 \leftarrow a_1, \ldots, x_k \leftarrow a_k] \).

If \( Q \) is a unary ranked alphabet, i.e., the rank of all symbols in \( Q \) is \( 1 \), then \( Q(X_k) \) stands for the set \( \{ q(x_i) \mid q \in Q \text{ and } 1 \leq i \leq k \} \). A string \( w \in (Q(X_k))^* \) is called linear if every variable \( x_i \), \( 1 \leq i \leq k \) occurs at most once in it.

Next we shortly recall the concept of the bottom-up tree transducer and the top-down tree transducer. For more terminology and details about tree transducers the reader is advised to consult [Eng75].
A \textit{bottom-up tree transducer} is a tuple $M = (Q, \Sigma, \Delta, Q_d, R)$ where $Q$ is a unary ranked alphabet (of states), $\Sigma$ and $\Delta$ are ranked alphabets (of input symbols and output symbols, resp.), $Q_d \subseteq Q$ is the set (of final states), and $R$ is a finite set of rules of the form $\sigma(q_1(x_1), \ldots, q_k(x_k)) \to q(t)$, where $k \geq 0$, $\sigma \in \Sigma^{(k)}$, $q_1, \ldots, q_k, q \in Q$, and $t \in T_\Delta(X_k)$. The \textit{derivation relation (induced by $M$)} is the binary relation $\Rightarrow_M \subseteq TQ \times TQ \times \Delta$ defined by $\varphi \Rightarrow_M \psi$ iff there is a $\beta \in T_\Delta(X_1)$, there is a rule $\sigma(q_1(x_1), \ldots, q_k(x_k)) \to q(t)$ in $R$, there are $t_1, \ldots, t_k \in T_\Delta$ such that $\varphi = \beta[x_1 \leftarrow \sigma(q_1(t_1), \ldots, q_k(t_k))]$ and $\psi = \beta[x_1 \leftarrow q(t[x_1 \leftarrow t_1, \ldots, x_k \leftarrow t_k])]$. The \textit{tree transformation computed by $M$} is the set $\tau_M = \{ (s, t) \mid s \Rightarrow_M q(t) \text{ for some } q \in Q_d \}$. The \textit{class of all tree transformations computed by bottom-up tree transducers} is denoted by $BOT_\Delta$.

A \textit{top-down tree transducer} is a tuple $M = (Q, \Sigma, \Delta, Q_d, R)$ where $Q$ is a unary ranked alphabet (of states), $\Sigma$ and $\Delta$ are ranked alphabets (of input symbols and output symbols, resp.), $Q_d \subseteq Q$ is the set (of initial states), and $R$ is a finite set of rules of the form $q(\sigma(x_1, \ldots, x_k)) \to t$, where $q \in Q$, $k \geq 0$, $\sigma \in \Sigma^{(k)}$, and $t \in T_\Delta(Q(X_k))$. The \textit{derivation relation (induced by $M$)} is the binary relation $\Rightarrow_M \subseteq TQ \times TQ \times \Delta$ defined by $\varphi \Rightarrow_M \psi$ iff there is a $\beta \in T_\Delta(X_1)$, there is a rule $q(\sigma(x_1, \ldots, x_k)) \to t$ in $R$, there are $s_1, \ldots, s_k \in T_\Sigma$ such that $\varphi = \beta[x_1 \leftarrow q(\sigma(s_1, \ldots, s_k))]$ and $\psi = \beta[x_1 \leftarrow t[x_1 \leftarrow s_1, \ldots, x_k \leftarrow s_k]]$. The \textit{tree transformation computed by $M$} is the set $\tau_M = \{ (s, t) \mid q(s) \Rightarrow_M q(t) \text{ for some } q \in Q_d \}$. The \textit{class of all tree transformations computed by top-down tree transducers} is denoted by $TOP_\Sigma$.

2.4 Semirings

For a survey paper on the relevance of semirings and formal power series to formal languages and automata cf. [Kui97b].

A \textit{semiring} is an algebraic structure $\mathbb{A} = (\mathbb{A}, +, \cdot, 0, 1)$ with two operations sum $+$ and product $\cdot$ such that $(\mathbb{A}, +, 0)$ is a commutative monoid, $(\mathbb{A}, \cdot, 1)$ is a monoid, and for every $a, b, c \in \mathbb{A}$ the following laws hold: $(a + b) \cdot c = a \cdot c + b \cdot c$, $a \cdot (b + c) = a \cdot b + a \cdot c$, and $a \cdot 0 = 0 \cdot a = 0$. Whenever the operations $(+$ and $\cdot)$ and the neutral elements (0 and 1) are clear from the context, then we denote the semiring $(\mathbb{A}, +, \cdot, 0, 1)$ just by $\mathbb{A}$. We give some examples of semirings.

- The \textit{boolean semiring} is the semiring $\mathbb{B} = \{0, 1\}$ with disjunction and conjunction as sum and product, resp.

- Let $\Delta$ be an alphabet. The \textit{semiring of formal languages (over $\Delta$)}, denoted by $\text{Lang}(\Delta)$, is the semiring $\left( \mathbb{P}(\Delta^*), \cup, \cdot, 0, \varepsilon \right)$ where $\cdot$ is the usual concatenation of string languages.

- The \textit{semiring of natural numbers} is the semiring $\mathbb{N} = (\mathbb{N}, +, \cdot, 0, 1)$ with the usual addition and multiplication operations.

Let $(\mathbb{A}, +, \cdot, 0, 1)$ be a semiring. $\mathbb{A}$ is \textit{commutative} if, for every $a, b \in \mathbb{A}$, the equation $a \cdot b = b \cdot a$ holds.

$\mathbb{A}$ is \textit{naturally ordered} if the binary relation $\preceq$ on $\mathbb{A}$ is a partial order on $\mathbb{A}$, where $\preceq$ is defined by $a \preceq b$ iff there is a $c \in \mathbb{A}$ such that $a + c = b$.

$\mathbb{A}$ is \textit{complete} if it is possible to define sums for all families $(a_i \mid i \in I)$ of elements of $\mathbb{A}$, where $I$ is an arbitrary index set, such that the following three conditions are satisfied:

(i) $\sum_{i \in \emptyset} a_i = 0$, $\sum_{i \in \emptyset} a_i = a_j$, $\sum_{i \in \emptyset} a_i = a_j + a_k$ for $j \neq k$.

(ii) $\sum_{j \in J} (\sum_{i \in I_j} a_i) = \sum_{i \in I} a_i$ if $\bigcup_{j \in J} I_j = I$ and $I_j \cap I_k = \emptyset$ for $j \neq k$.

(iii) $\sum_{i \in I} (c \cdot a_i) = c \cdot (\sum_{i \in I} a_i)$, $\sum_{i \in I} (a_i \cdot c) = (\sum_{i \in I} a_i) \cdot c$.

$\mathbb{A}$ is \textit{\omega-continuous} if the following three conditions holds.

(i) $\mathbb{A}$ is naturally ordered.

(ii) $\mathbb{A}$ is complete.

(iii) If $\sum_{0 \leq i \leq n} a_i \preceq c$ for all $n \in \mathbb{N}$, then $\sum_{i \in \mathbb{N}} a_i \preceq c$ for all sequences $(a_i \mid i \in \mathbb{N})$ in $\mathbb{A}$ and all $c \in \mathbb{A}$.

5
2.5 Matrices

We recall some notions concerning matrices (cf. [Ku97b]).

Let \( I, J \) and \( S \) be arbitrary sets. An \( I \times J \) matrix over \( S \) is a mapping \( M : I \times J \rightarrow S \). The set of all \( I \times J \) matrices over \( S \) is denoted by \( S^{I \times J} \). An element \( M(i, j) \in S \) is also denoted by \( M_{i,j} \). If \( I \) (or \( J \)) is a singleton, then we identify it by \( \{1\} \) and write just \( 1 \) for \( I \) (or \( J \)). Moreover in that case an element of the form \( M_{1,1} \) (or \( M_{i,j} \)) is written just \( M \) (or \( M_{i,j} \)).

Let \( A \) be some structure in which the operations (infinite) sum and multiplication are defined. Let \( M : I \times J \rightarrow A \) be a matrix and \( a \in A \). Then \( a \cdot M \) is the \( I \times J \) matrix over \( A \) such that for every \((i, j) \in I \times J\), \((a \cdot M)_{i,j} = a \cdot M_{i,j} \) where \( \cdot \) is the multiplication of \( A \). Moreover, for a family \( \{M^{(k)} \in K \) of \( I \times J \) matrices over \( A \) we define the \( I \times J \) matrix \( \sum_{k \in K} M^{(k)} \) by

\[
(\sum_{k \in K} M^{(k)})_{i,j} = \sum_{k \in K} (M^{(k)})_{i,j}.
\]

2.6 Formal tree series

Let \( \Delta \) be ranked alphabet and let \((A, +, 0, 1)\) be a semiring. A formal tree series (over \( \Delta \) and \( A \)) is a mapping \( r : T_\Delta \rightarrow A \). For every \( t \in T_\Delta \), the element \( r(t) \in A \) is called coefficient of \( t \) and it is also denoted by \( (r,t) \); then the mapping itself is written as the formal sum \( \sum_{t \in T_\Delta} r(t) t \). The set of all formal tree series over \( \Delta \) and \( A \) is denoted by \( A \langle T_\Delta \rangle \).

Note that in this paper \( A \) stands for a commutative and continuous semiring. Let \( a \in A \) and let \( r_1, r_2 \in A \langle T_\Delta \rangle \). The formal tree series \( a \cdot r_1 \) is the formal tree series \( \sum_{t \in T_\Delta} (a \cdot r_1)(t) t \). Moreover, \( r_1 + r_2 \) denotes the formal tree series \( \sum_{t \in T_\Delta} (r_1 + r_2)(t) t \) and, for every family \( (r_i)_{i \in I} \) of formal tree series over \( \Delta \) and \( A \), \( \sum_{i \in I} r_i = \sum_{t \in T_\Delta} (\sum_{i \in I} r_i)(t) t \).

Let \( r : T_\Delta \rightarrow A \) be a formal tree series. If there is an \( a \in A \) such that for every \( t \in T_\Delta \) we have \( (r,t) = a \), then \( r \) is called the constant \( a \) and is denoted by \( \bar{a} \). The support of \( r \) is defined as the set \( \text{supp}(r) = \{ t \in T_\Delta \mid (r,t) \neq 0 \} \). Moreover, the formal tree series \( r \) is called a polynomial if \( \text{supp}(r) \) is finite and is called a singleton if \( \text{supp}(r) \) is a singleton. Hence a singleton is a polynomial. The set of all polynomials over \( \Delta \) and \( A \) is denoted by \( A \langle T_\Delta \rangle \).

A polynomial \( r \in A \langle T_\Delta \rangle \) is written as \( a_1 t_1 + \ldots + a_n t_n \), where \( \text{supp}(r) = \{ t_1, \ldots, t_n \} \) and, for every \( 1 \leq i \leq n \), \( a_i = (r, t_i) \). Moreover, if \( a_i = 1 \), then \( a_i = t_i \) is written as \( t_i \). Hence for \( t \in T_\Delta \) either \( t \) denotes the tree in \( T_\Delta \) or it denotes the formal tree series \( r : T_\Delta \rightarrow A \) such that \( (r, t) = 1 \) and \( (r, s) = 0 \) for every \( s \in T_\Delta - \{ t \} \). However, it will always be clear from the context which meaning is intended.

Let \( \Delta \) be a ranked alphabet. We define the binary relation \( \text{pick}_\Delta = \{(r,t) \mid r \in B(T_\Delta), t \in \text{supp}(r)\} \subseteq B \langle T_\Delta \rangle \times T_\Delta \). Note that \( \text{pick}_\Delta \) is a set of pairs and not a set of coefficients in \( B \). The class of relations \( \text{pick}_\Delta \) for an arbitrary ranked alphabet \( \Delta \) is denoted by \( \text{PICK} \).

Now we define the substitution of formal tree series. This is a straightforward generalization of the \( IO \)-substitution of languages [ES77, ES78, EV85].

Let \( t \in T_\Delta(X) \) and \( \bar{s} = (s_1, \ldots, s_l) \in (A \langle T_\Delta(X) \rangle)^l \). Define \( t \leftrightarrow \bar{s} \in A \langle T_\Delta(X) \rangle \) as follows:

(i) If \( t = x_j \), \( 1 \leq j \leq l \), then \( t \leftrightarrow \bar{s} = s_j \).

(ii) If \( t = \alpha \in \Delta^{(0)} \), then \( t \leftrightarrow \bar{s} = \alpha \).

(iii) If \( t \notin \Delta^{(0)} \cup X_1 \), then \( t \leftrightarrow \bar{s} = \sum_{t_1, t_2, \ldots, t_l \in T_\Delta(X_1)} (s_1, t_1 \ldots, s_l, t_l) (t[t_1, \ldots, t_l]) \).

Note that, for every \( t \in T_\Delta \), \( t \leftrightarrow (t) = t \). Moreover, if \( s_i = 0 \) for some \( 1 \leq i \leq l \), then also \( t \leftrightarrow \bar{s} = 0 \) no matter whether \( x_i \) occurs in \( t \) or not.
Now let \( r \in A(\langle T_\Delta(X_1) \rangle) \) and \( \bar{s} = (s_1, \ldots, s_l) \in (A(\langle T_\Delta(X) \rangle))^l \). Define \( r \bar{s} \in A(\langle T_\Delta(X) \rangle) \) by \( r \bar{s} = \sum_{(r,t) \in T_\Delta(X)} (r,t) (t \bar{s}) \). Certainly \( 0 \bar{s} = 0 \).

If the semiring \( A \) is the boolean semiring \( B \), then the substitution of formal tree series is the same as the \( IO \)-substitution on formal languages as defined in [ES77, ES78].

We note that in [Kui99] an \( OI \)-substitution of formal tree series is defined. The reason why we work with \( IO \)-substitution will be discussed at the end of Section 4.

2.7 Tree series transformations

A tree to tree series transformation (for short: t-ts transformation) is a mapping \( \tau : T_2 \to A(\langle T_\Delta \rangle) \). A tree series to tree series transformation (for short: ts-ts transformation) is a mapping \( \rho : A(\langle T_2 \rangle) \to A(\langle T_\Delta \rangle) \).

Every t-ts transformation can be extended in a natural way to a ts-ts transformation. Let \( \tau : T_2 \to A(\langle T_\Delta \rangle) \) be a t-ts transformation. The \textit{extension} of \( \tau \) is the mapping \( \tilde{\tau} : A(\langle T_2 \rangle) \to A(\langle T_\Delta \rangle) \) such that for every \( r \in A(\langle T_2 \rangle) \), \( \tilde{\tau}(r) = \sum_{s \in T_\Delta} ((r,s) \tau(s)) \).

We also extend a mapping \( \tau : T_2 \to A(\langle T_\Delta \rangle)^{I \times J} \) to \( \tilde{\tau} : A(\langle T_2 \rangle) \to A(\langle T_\Delta \rangle)^{I \times J} \). In this case for an \( r \in A(\langle T_2 \rangle) \) the extension \( \tilde{\tau}(r) \) is defined by the same equation but now it will be an \( I \times J \) matrix over \( A(\langle T_\Delta \rangle) \).

Let \( \tau_1 : T_2 \to A(\langle T_\Delta \rangle) \) and \( \tau_2 : T_\Delta \to A(\langle T_I \rangle) \) be two t-ts transformations. Then define the \textit{composition of} \( \tau_1 \) \textit{and} \( \tau_2 \), denoted by \( \tau_1 \circ \tau_2 \), as the t-ts transformation \( \tau_1 \circ \tau_2 : T_2 \to A(\langle T_I \rangle) \) where \( (\tau_1 \circ \tau_2)(s) = \tilde{\tau}_1 (\tau_2(s)) \) for every \( s \in T_\Delta \). We note that \( \circ \) is not a composition of relations in the sense we defined it in Subsection 2.1. The \textit{composition of two classes} \( C_1 \) and \( C_2 \) of t-ts transformations, denoted by \( C_1 \circ C_2 \) is the class \( \{ \tau_1 \circ \tau_2 \mid \tau_i \in C_i, 1 \leq i \leq 2 \} \) of t-ts transformations.

There is a close relationship between the composition of t-ts transformations and ts-ts transformations which will be made clear in the next lemma.

**Lemma 2.2** Let \( \tau_1 : T_2 \to A(\langle T_\Delta \rangle) \), \( \tau_2 : T_\Delta \to A(\langle T_I \rangle) \), and \( \tau : T_2 \to A(\langle T_I \rangle) \) be three t-ts transformations. Then, if \( \tau_1 \circ \tau_2 = \tau \), then \( \tilde{\tau}_1 \circ \tilde{\tau}_2 = \tilde{\tau} \).

**Proof.** Let \( r \in A(\langle T_\Sigma \rangle) \). Then

\[
\tilde{\tau}_1 \circ \tilde{\tau}_2 (r) = \sum_{s \in T_\Delta} (r,s) \tilde{\tau}_2 (s) = \sum_{s \in T_\Delta} \sum_{t \in T_\Delta} (r,s) \tau_1 (\tau_2 (s), t) \tau_2 (t) = \sum_{s \in T_\Delta} \sum_{t \in T_\Delta} (\tau_1 (s), t) \tau_2 (t) = \sum_{s \in T_\Delta} \sum_{t \in T_\Delta} (\tau_1 (s), t) \tau_2 (t) = (\tilde{\tau}_1 \circ \tilde{\tau}_2) (r).
\]

\( \Box \)

For every ranked alphabet \( \Sigma \) there is a particular tree series transformation \( \iota_\Sigma : T_\Sigma \to A(\langle T_\Delta \rangle) \) defined by \( \iota_\Sigma (s) = 1 \cdot s \) for every \( s \in T_\Sigma \). We call \( \iota_\Sigma \) the identity tree series transformation over \( \Sigma \) and \( A \).

3 Tree series transducers

In this section we define the notion of a bottom-up tree series transducer and top-down tree series transducer which are generalizations of the notions of bottom-up tree transducers and top-down tree transducers, resp., as they occur in the literature (cf. e.g. [Eng75]). The generalization consists of allowing the transducers to compute a formal tree series as output rather than only trees. Also, a top-down tree series transducer generalizes the concept of the tree transducer as it is defined in [Kui99], because the latter can only cover the nondeterministically simple case of top-down tree transducers [GS84] (cf. Fig. 1 where going north-east along an edge means to generalize from the boolean semiring to an arbitrary semiring, and going north-west along an edge means to generalize from the nondeterministically simple case to the arbitrary non-deterministic case).
In the following let $\Sigma$ and $\Delta$ be two ranked alphabets, and $Q$ be a unary ranked alphabet. Moreover, let $(A, +, \cdot, 0, 1)$ be a commutative continuous semiring.

**Definition 3.1** A tree representation (over $Q$, $\Sigma$, $\Delta$, and $A$) is a family $\mu = (\mu_k | k \geq 0)$ of mappings $\mu_k : \Sigma^{(k)} \to (A \langle T_\Delta (X) \rangle)^{Q \times (Q(X_k))^*}$ such that only for finitely many indices $(q, w) \in Q \times (Q(X_k))^*$, $\mu_k(\sigma)_{q,w} \neq 0$.

A tree representation is bottom-up if the following additional conditions hold. For every $\sigma \in \Sigma^{(k)}$, $q \in Q$, and $w \in (Q(X_k))^*$, if $w \neq q_1(x_1) \ldots q_k(x_k)$ for some $q_1, \ldots, q_k \in Q$, then $\mu_k(\sigma)_{q,w} = 0$. Moreover, if $\mu_k(\sigma)_{q,w} \neq 0$, then $\mu_k(\sigma)_{q,w} \in A \langle T_\Delta (X_k) \rangle$.

A tree representation is top-down if the following additional condition holds. For every $\sigma \in \Sigma^{(k)}$, $q \in Q$, and $w \in (Q(X_k))^*$, if $\mu_k(\sigma)_{q,w} \neq 0$, then $\mu_k(\sigma)_{q,w} \in A \langle T_\Delta (X_l) \rangle$, where $l = |w|$.

**Definition 3.2** Let $\mu$ be either a bottom-up tree representation or a top-down tree representation. For every $k \geq 0$ and $\sigma \in \Sigma^{(k)}$, $\mu$ induces the mapping

$$\mu_k(\sigma) : (A \langle T_\Delta (X) \rangle)^{Q \times 1} \times \ldots \times (A \langle T_\Delta (X) \rangle)^{Q \times 1} \to (A \langle T_\Delta (X) \rangle)^{Q \times 1}$$

with $k$ arguments. The mapping $\mu_k(\sigma)$ is defined for every $P_1, \ldots, P_k \in (A \langle T_\Delta (X) \rangle)^{Q \times 1}$ and $q \in Q$ as:

$$\mu_k(\sigma)(P_1, \ldots, P_k)_q = \sum_{w = q_1(x_1) \ldots q_k(x_k) \in (Q(X_k))^*} \mu_k(\sigma)_{q,w} \cdot ((P_1)_{q_1}, \ldots, (P_k)_{q_k}).$$

Note that, for $k = 0$, $(Q(X_k))^* = \{ \varepsilon \}$. Hence, according to our convention about matrix notation, in case $\sigma \in \Sigma^{(0)}$ we write $\mu_0(\sigma)_{q,l}$ as $\mu_0(\sigma)_q$.

Since $(\langle A \langle T_\Delta (X) \rangle \rangle)^{Q \times 1}, (\mu_k(\sigma) | k \geq 0, \sigma \in \Sigma^{(k)})$ and also $(\langle A \langle T_\Delta \rangle \rangle)^{Q \times 1}, (\mu_k(\sigma) | k \geq 0, \sigma \in \Sigma^{(k)})$ are $\Sigma$-algebras, there is a unique homomorphism

$$h_\mu : T_\Sigma \to (A \langle T_\Delta \rangle)^{Q \times 1}_\Sigma$$

defined for every $\sigma \in \Sigma^{(k)}$, $t_1, \ldots, t_k \in T_\Sigma$ by

$$h_\mu(\sigma(t_1, \ldots, t_k)) = \mu_k(\sigma)(h_\mu(t_1), \ldots, h_\mu(t_k)).$$

The mapping $h_\mu$ extends to $\overline{h_\mu} : A \langle T_\Sigma \rangle \to (A \langle T_\Delta \rangle)^{Q \times 1}$, for the definition see Subsection 2.7.
Definition 3.3 A bottom-up tree series transducer (over $A$) is a tuple $M = (Q, \Sigma, \Delta, A, Q_d, \mu)$ where $Q$ is a unary ranked alphabet (of states), $\Sigma$ and $\Delta$ are ranked alphabets (of input symbols and output symbols, resp.), $A$ is a semiring, $Q_d \subseteq Q$ (set of final states), and $\mu$ is a bottom-up tree representation over $Q, \Sigma, \Delta$, and $A$.

Definition 3.4 A top-down tree series transducer (over $A$) is a tuple $M = (Q, \Sigma, \Delta, A, Q_d, \mu)$ where $Q$ is a unary ranked alphabet (of states), $\Sigma$ and $\Delta$ are ranked alphabets (of input symbols and output symbols, resp.), $A$ is a semiring, $Q_d \subseteq Q$ (set of initial states), and $\mu$ is a top-down tree representation over $Q, \Sigma, \Delta$, and $A$.

We associate with every bottom-up (and top-down) tree series transducer $M$ two transformations, the t-ts transformation computed by $M$ and the ts-ts transformation computed by $M$. These are defined as follows.

Definition 3.5 Let $M = (Q, \Sigma, \Delta, A, Q_d, \mu)$ be a bottom-up tree series transducer or a top-down tree series transducer.

1. The t-ts transformation computed by $M$ is the mapping $\tau_M : T_\Sigma \rightarrow A \langle T_\Delta \rangle$ which is defined by

$$\tau_M(s) = \sum_{q \in Q_d} h_\mu(s)_q$$

for every $s \in T_\Sigma$.

2. The ts-ts transformation computed by $M$ is the extension $\tilde{\tau}_M : A \langle T_\Sigma \rangle \rightarrow A \langle T_\Delta \rangle$ of $\tau_M$, cf. Subsection 2.7.

Note that $\tilde{\tau}_M(s) = \sum_{q \in Q_d} h_\mu(s)_q$ for every $s \in T_\Sigma$. Sometimes we use the notion of t-ts semantics (and ts-ts semantics) of $M$ to speak about the t-ts transformation (and ts-ts transformation, respectively) computed by $M$.

Now we give an example for a bottom-up tree series transducer. In the theory of term rewriting systems one is interested among others in sufficient conditions for the property of termination (cf. e.g. [Jan97, BN98]). One such condition is expressed in the following implication. Let $R$ be a term rewriting system, then $R$ is terminating if, for every derivation $t_1 \Rightarrow_R t_2 \Rightarrow_R t_3 \ldots$, there is no pair $i, j$ with $i < j$ such that $t_i$ can be embedded homeomorphically into $t_j$. For two trees $s$ and $t$, $t$ can be embedded homeomorphically into $s$ (for short: $s \geq_{emb} t$) if $t$ is obtained from $s$ by dropping an arbitrary number of occurrences of symbols in $s$ (cf. Definition 5.4.2 of [BN98]).

Example 3.6 For the particular ranked alphabet $\Sigma = \{\sigma(2), \gamma(1), \alpha(0)\}$ and $s, t \in T_\Sigma$, we have $s \geq_{emb} t$ if $t$ is obtained from $s$ by dropping an arbitrary number of occurrences of $\sigma$’s or $\gamma$’s.

The relation $\geq_{emb}$ as tree transformation, can be computed by the bottom-up tree transducer $M_{drop} = (Q, \Sigma, Q_d, R_{drop})$, i.e., $\tau_{M_{drop}} = \geq_{emb}$, where $Q = Q_d = \{\ast\}$ and $R_{drop}$ consists of the following rules:

\[
\begin{align*}
\sigma(\ast(x_1), \ast(x_2)) & \rightarrow \ast(\sigma(x_1, x_2)) \\
\sigma(\ast(x_1), \ast(x_2)) & \rightarrow \ast(x_1) \\
\sigma(\ast(x_1), \ast(x_2)) & \rightarrow \ast(x_2) \\
\gamma(\ast(x_1)) & \rightarrow \ast(\gamma(x_1)) \\
\gamma(\ast(x_1)) & \rightarrow \ast(x_1) \\
\alpha & \rightarrow \ast(\alpha).
\end{align*}
\]

Clearly, for a given tree $s \in T_\Sigma$ there are (finitely) many trees $t \in T_\Sigma$ such that $s \geq_{emb} t$. Also it is clear that, for a given tree $t \in T_\Sigma$ there are (infinitely) many trees $s \in T_\Sigma$ such that $s \geq_{emb} t$.

Now we might be interested in the following question: given two trees $s, t \in T_\Sigma$, how many possibilities are there to embed $t$ into $s$? For instance, consider the trees $s = \sigma(\alpha, \gamma(\gamma(\alpha)))$ and $t = \sigma(\alpha, \gamma(\alpha))$. Hence, $s \geq_{emb} t$. One way of obtaining $t$ from $s$ is to drop the $\gamma$ at occurrence 2 of $s$, another way to obtain $t$ is to drop the $\gamma$ at occurrence 21. As another example, consider $t' = s$. Then $s \geq_{emb} t'$ and there is one way of obtaining $t'$ from $s$, viz. dropping no occurrence of a symbol. Finally, consider $t'' = \sigma(\gamma(\alpha), \alpha)$. Clearly, $s \not\geq_{emb} t''$ and hence, the number of different ways in which $t''$ can be obtained from $s$ is zero.

Thus, we define the t-ts transformation $\#emb : T_\Sigma \rightarrow \text{Nat}(\langle T_\Sigma \rangle)$ such that, for every $s, t \in T_\Sigma$,

\[
(\#emb(s), t) = \begin{cases} n & \text{if } s \geq_{emb} t \text{ and there are } n \text{ different ways to embed } t \text{ into } s \\ 0 & \text{otherwise} \end{cases}
\]
Note that, \( (#emb(s), t) \neq 0 \) if and only if \( s \geq_{emb} t \). Thus, e.g.,

\[
#emb(\sigma(\alpha, \gamma(\gamma(\alpha)))) = \sigma(\alpha, \gamma(\gamma(\alpha))) + 2\sigma(\alpha, \gamma(\alpha)) + \sigma(\alpha, \alpha) + \gamma(\gamma(\alpha)) + 2\gamma(\alpha) + 2\alpha.
\]

Now, based on \( M_{drop} \), we construct a bottom-up tree series transducer \( M_{\#}^{\#}_{drop} = (Q, \Sigma, \Sigma, Nat, Q_d, \mu) \) such that \( \tau_{M_{\#}^{\#}_{drop}} = \#emb \), where \( \mu \) is defined as follows.

\[
\begin{align*}
& \mu_2(\sigma(\alpha, \gamma(\gamma(\alpha)))) = \sigma(\alpha, \gamma(\gamma(\alpha))) + 2\sigma(\alpha, \gamma(\alpha)) + \sigma(\alpha, \alpha) + \gamma(\gamma(\alpha)) + 2\gamma(\alpha) + 2\alpha. \\
& \mu_1(\gamma(\alpha)) = \gamma(\alpha). \\
& \mu_0(\alpha) = \alpha.
\end{align*}
\]

Now let us compute \( \tau_{M_{\#}^{\#}_{drop}}(s) \) for our example tree \( s = \sigma(\alpha, \gamma(\gamma(\alpha))) \).

\[
\begin{align*}
& h_\mu(\alpha)_s \\
& = \frac{\mu_0(\alpha)}{\mu_0(\alpha)} \\
& = \sum_{w=\alpha} \mu_0(\alpha)_{s, w} \leftarrow () \\
& = \mu_0(\alpha)_{s, t} \leftarrow () \\
& = \alpha \\
& h_\mu(\gamma(\alpha))_s \\
& = \frac{\mu_1(\gamma)}{h_\mu(\alpha)} \\
& = \sum_{w=\alpha} \mu_1(\gamma)_{s, w} \leftarrow (h_\mu(\alpha)_{s, w}) \\
& = \mu_0(\gamma)_{s, \alpha} \leftarrow (h_\mu(\alpha)_s) \\
& = (\gamma(\alpha))_{s, \alpha} \leftarrow (\alpha) \\
& = \sum_{t \in T_{\Sigma}(\alpha)} (\gamma(\alpha))(t) = (\gamma(\alpha))_s \\
& = (\sum_{s \in T_{\Sigma}(\alpha)} (\gamma(\alpha))_{s, \alpha} + (\sum_{s \in T_{\Sigma}(\alpha)} (\gamma(\alpha))_{s, \alpha})) \\
& = (\gamma(\alpha))(t) + \gamma(\alpha) + \alpha \\
& h_\mu(\gamma(\gamma(\alpha)))_s \\
& = \mu_2(\sigma(\alpha, \gamma(\gamma(\alpha))))_{s, \alpha} \\
& = \sum_{w=\alpha} \mu_2(\sigma)_{s, w} \leftarrow (h_\mu(\alpha)_{s, w}, h_\mu(\gamma(\gamma(\alpha))))_{s, w} \\
& = \mu_2(\sigma)_{s, \alpha} \leftarrow (h_\mu(\alpha)_{s, \alpha}, h_\mu(\gamma(\gamma(\alpha))))_{s, \alpha} \\
& = (\sigma(\alpha, \gamma(\gamma(\alpha))))_{s, \alpha} \leftarrow (\alpha, \gamma(\gamma(\alpha)) + 2\gamma(\alpha) + \alpha) \\
& = \sum_{t \in T_{\Sigma}(\alpha)} (\sigma(\alpha, \gamma(\gamma(\alpha))))_{s, \alpha} \leftarrow (\alpha, \gamma(\gamma(\alpha)) + 2\gamma(\alpha) + \alpha).
\end{align*}
\]
\[
\begin{align*}
&= (\sigma(x_1, x_2) - (\alpha, \gamma(\alpha)) + 2\gamma(\alpha) + \alpha)) \\
&+ (x_1 \leftarrow (\alpha, \gamma(\alpha) + 2\gamma(\alpha) + \alpha)) \\
&+ (x_2 \leftarrow (\alpha, \gamma(\alpha) + 2\gamma(\alpha) + \alpha)) \\
&= \sum_{x_1, x_2 \in T_2(x_1, x_2)} (\alpha, \gamma(\alpha) + 2\gamma(\alpha) + \alpha) \\
&+ \sum_{x_1, x_2 \in T_2(x_1, x_2)} (\alpha, \gamma(\alpha) + 2\gamma(\alpha) + \alpha) \\
&+ \sum_{x_1, x_2 \in T_2(x_1, x_2)} (\alpha, \gamma(\alpha) + 2\gamma(\alpha) + \alpha) \\
&= (\sigma(\alpha, \gamma(\alpha)) + 2\sigma(\alpha, \gamma(\alpha)) + \sigma(\alpha, \gamma(\alpha)) + 2\gamma(\alpha) + \alpha) \\
&+ \sigma(\alpha, \gamma(\alpha)) + 2\sigma(\alpha, \gamma(\alpha)) + \sigma(\alpha, \gamma(\alpha)) + 2\gamma(\alpha) + \alpha) \\
&= \sigma(\alpha, \gamma(\alpha)) + 2\sigma(\alpha, \gamma(\alpha)) + \sigma(\alpha, \gamma(\alpha)) + 2\gamma(\alpha) + \alpha \\
\end{align*}
\]

Thus, we have verified that \(\tau_{\mathcal{G}_{\text{drop}}}^\#(s) = \#\text{emb}(s)\) for the tree \(s = \sigma(\alpha, \gamma(\alpha))\). \(\square\)

Next we define some restricted versions of bottom-up tree series transducers. These are the same restrictions which are imposed on tree transducers, see e.g. [Eng75, Bak79, GS84].

**Definition 3.7** Let \(M = (Q, \Sigma, \Delta, A, Q_d, \mu)\) be a bottom-up tree series transducer.
1. \(M\) is **deterministic** if for every \(k \geq 0\), \(\sigma \in \Sigma^k\), \(q_1, \ldots, q_k \in Q\), there is at most one \(q \in Q\) such that \(\mu_k(\sigma)_{q,w} \neq 0\) where \(w = q_1(x_1) \ldots q_k(x_k)\), and if \(\mu_k(\sigma)_{q,w} \neq 0\) for some \(q \) and \(w\), then \(\mu_k(\sigma)_{q,w}\) is a singleton.
2. \(M\) is **total** if \(Q_d = Q\) and for every \(k \geq 0\), \(\sigma \in \Sigma^k\), \(q_1, \ldots, q_k \in Q\), there is at least one \(q \in Q\) such that \(\mu_k(\sigma)_{q,w} \neq 0\).
3. \(M\) is a **bottom-up homomorphism tree series transducer** if it is total and the set \(Q\) of states is a singleton.
4. \(M\) is **linear** if for every \(k \geq 0\), \(\sigma \in \Sigma^k\), \(q, q_1, \ldots, q_k \in Q\), every \(t \in \text{supp}(\mu_k(\sigma)_{q,q_1(x_1) \ldots q_k(x_k)})\) is linear.
5. \(M\) is a **bottom-up finite state relabeling tree series transducer** if for every \(k \geq 0\), \(\sigma \in \Sigma^k\) and \(q, q_1, \ldots, q_k \in Q\), every \(t \in \text{supp}(\mu_k(\sigma)_{q,q_1(x_1) \ldots q_k(x_k)})\) has the form \(\delta(x_1, \ldots, x_k)\) for some \(\delta \in \Delta^k\).
6. \(M\) is **polynomial** if for every \(k \geq 0\), \(\sigma \in \Sigma^k\), \(q, q_1, \ldots, q_k \in Q\), \(\mu_k(\sigma)_{q,q_1(x_1) \ldots q_k(x_k)} \in A(\text{supp}(\Delta(X)))\).
\(\square\)

Notice that bottom-up finite state relabeling tree series transducers are polynomial.

The class of **t-ts transformations induced by bottom-up tree series transducers** is denoted by \(\text{BOT}^\#_{t-ts}\), the classes of t-ts transformations which correspond to the syntactic subclasses of Definition 3.7 1.-6. are denoted by \(d-\text{BOT}^\#_{t-ts}\), \(t-\text{BOT}^\#_{t-ts}\), \(h-\text{BOT}^\#_{t-ts}\), \(l-\text{BOT}^\#_{t-ts}\), \(r-\text{BOT}^\#_{t-ts}\), and \(p-\text{BOT}^\#_{t-ts}\), resp. If more than one syntactic restriction holds, then \(\text{BOT}^\#_{t-ts}\) is prefixed by the string of corresponding letters, e.g. \(ltd-\text{BOT}^\#_{t-ts}\) denotes the class of t-ts transformations induced by linear, total, deterministic bottom-up tree series transducers. If all the considered bottom-up tree series transducers are over the same semiring \(A\), then the corresponding class of t-ts transformations is indexed by \(A\) as e.g. \(ltd-\text{BOT}^\#_{t-ts}(A)\). The class of **ts-ts transformations induced by bottom-up tree series transducers** is denoted by \(\text{BOT}^\#_{t_{ts}-t_{ts}}\). The denotation of the classes of ts-ts transformations computed by subclasses of bottom-up tree series transducers is obtained from the denotation of the corresponding classes of t-ts transformations by replacing \(t - ts\ by ts - ts\).

Note that the bottom-up tree series transducer \(M^\#_{\text{drop}}\) considered in Example 3.6 is in fact a bottom-up homomorphism tree series transducer. In what follows, like in case of \(M^\#_{\text{drop}}\), we will denote the only state of a homomorphism tree series transducer by \(\ast\).

Now we introduce the corresponding restrictions for top-down tree series transducers.

**Definition 3.8** Let \(M = (Q, \Sigma, \Delta, A, Q_d, \mu)\) be a top-down tree series transducer.
1. \(M\) is **deterministic** if the following conditions hold. \(Q_d\) is a singleton. For every \(k \geq 0\), \(\sigma \in \Sigma^k\), \(q \in Q\), there is at most one \(w \in (Q(X_k))^\ast\) such that \(\mu_k(\sigma)_{q,w} \neq 0\), and if \(\mu_k(\sigma)_{q,w} \neq 0\) for some \(q\) and \(w\), then \(\mu_k(\sigma)_{q,w}\) is a singleton.
2. \(M\) is **total** if for every \(k \geq 0\), \(\sigma \in \Sigma^k\), \(q \in Q\), there is at least one \(w \in (Q(X_k))^\ast\) such that \(\mu_k(\sigma)_{q,w} \neq 0\), and \(Q_d \neq 0\).
3. \( M \) is a \textit{top-down homomorphism tree series transducer} if it is total deterministic and the set \( Q \) of states is a singleton.

4. \( M \) is \textit{linear} if for every \( k \geq 0 \), \( \sigma \in \Sigma^{(k)} \), \( q \in Q \), \( w \in (Q(X_k))^* \), if \( \mu_k(\sigma)_{q,w} \neq 0 \), then \( w \) is linear.

5. \( M \) is a \textit{top-down finite state relabeling tree series transducer} if for every \( k \geq 0 \), \( \sigma \in \Sigma^{(k)} \), \( q \in Q \), \( w \in (Q(X_k))^* \), if \( \mu_k(\sigma)_{q,w} \neq 0 \), then \( w = q_1(x_1) \ldots q_k(x_k) \) and every \( t \in \text{supp}(\mu_k(\sigma)_{q,w}) \) has the form \( \delta(x_1, \ldots, x_k) \) for some \( \delta \in \Delta^{(k)} \).

6. \( M \) is \textit{polynomial} if for every \( k \geq 0 \), \( \sigma \in \Sigma^{(k)} \), \( q \in Q \), \( w \in (Q(X_k))^* \), if \( \mu_k(\sigma)_{q,w} \neq 0 \), then \( w \in A(T_\Delta(X)) \).

Again like in the bottom-up case, top-down finite state relabeling tree series transducers are polynomial.

The \textit{class of t-ts transformations induced by top-down tree series transducers} is denoted by \( \text{TOP}_t \).

The classes of t-ts transformations which correspond to the syntactic subclasses of Definition 3.8.1-6. are denoted by \( \text{d-TOP}_t \), \( \text{t-TOP}_t \), \( \text{h-TOP}_t \), \( \text{l-TOP}_t \), \( \text{r-TOP}_t \), \( \text{p-TOP}_t \), resp.

For combinations of syntactic restrictions and for the restriction to one semiring \( \Lambda \) we use a notation similar to that of the bottom-up case. The \textit{class of t-ts transformations induced by top-down tree series transducers} is denoted by \( \text{TOP}_t \) and also here we use the same notation system as in the bottom-up case.

We note that a tree transducer in the sense of [Kui97a] is a top-down tree series transducer in which the elements \( \mu_k \) of the tree representation \( \mu \) have the particular form

\[
\mu_k : \Sigma_k \rightarrow (A(T_\Delta(X)))^{Q \times Q_k}
\]

where \( Q_k = Q(\{x_1\}) \times \ldots \times Q(\{x_k\}) \).

Finally, we show that, like in case of conventional tree transducers, the difference between the bottom-up and the top-down approaches disappears for certain basic classes of tree series transducers (cf. Lemma 3.2 of [Eng75] and the remark after Definition 3.14 of [Eng75]).

**Lemma 3.9**

1) \( h\text{-TOP}_t = h\text{-BOT} \) and \( h\text{-TOP}_t = h\text{-BOT} \).

2) \( r\text{-TOP}_t \) and \( r\text{-TOP}_t \) are tree transducers.

**Proof.** First we show that 1) holds. We prove only the first statement because the second one follows immediately from the definition of the t-ts semantics.

Let \( M = (\{\lambda\}, \Sigma, \Delta, A, \{\lambda\}, \mu) \) be a top-down homomorphism tree series transducer. We construct the bottom-up homomorphism tree series transducer \( M' = (\{\lambda\}, \Sigma, \Delta, A, \{\lambda\}, \mu') \) in the following way. For every \( k \geq 0 \), \( \sigma \in \Sigma^{(k)} \), and \( t \in T_{\Sigma}(X_k) \), let \( \text{lin}_X(t) = (t', x_{i_1}, \ldots, x_{i_k}) \). Now let us define \( (\mu_k(\sigma)_{*,*,t}(t')) = (\mu_k(\sigma)_{*,*,(x_{i_1}),\ldots,(x_{i_k})}(t')) \). It is easy to see that \( \tau_M = \tau_{M'} \), which proves the inclusion \( h\text{-TOP}_t \subseteq h\text{-BOT}_t \).

To show the converse of the inclusion, let us take a top-down homomorphism tree series transducer \( M = (\{\lambda\}, \Sigma, \Delta, A, \{\lambda\}, \mu) \). Now we construct the bottom-up homomorphism tree series transducer \( M' = (\{\lambda\}, \Sigma, \Delta, A, \{\lambda\}, \mu') \) as follows. For every \( k \geq 0 \), \( \sigma \in \Sigma^{(k)} \), sequence \( (x_{i_1}, \ldots, x_{i_k}) \) where \( 1 \leq i_1, \ldots, i_k \leq k \), and tree \( t' \in T_{\Sigma}(X_k) \), \( t = t'[x_i = x_{i_1}, \ldots, x_i = x_{i_k}] \).

Now let us define \( (\mu_k(\sigma)_{*,*,t}(t')) = (\mu_k(\sigma)_{*,*,(x_{i_1}),\ldots,(x_{i_k})}(t')) \). Again it is easy to see that \( \tau_M = \tau_{M'} \), which proves \( h\text{-BOT}_t \subseteq h\text{-TOP}_t \).

The proof of 2) is left to the reader, cf. the corresponding statement for tree transducers in [Eng75].

From now on we call a bottom-up homomorphism tree series transducer as well as a top-down homomorphism tree series transducer a \textit{homomorphism tree series transducers}. Similarly we drop the notions "bottom-up" and "top-down" from bottom-up and top-down finite state relabeling tree series transducers. Let us denote the classes \( h\text{-TOP}_t \) and \( r\text{-TOP}_t \) (and \( h\text{-TOP}_t \) and \( r\text{-TOP}_t \)) by \( \text{HOM}_t \) and \( \text{QREL}_t \) (\( \text{HOM}_t \) and \( \text{QREL}_t \)), respectively.

It is an exercise to show that for every ranked alphabet \( \Sigma \) and semiring \( A = \Sigma \) is in any of the classes of tree to tree series transformations defined in this section. We will use this fact in Section 5.

### 4 Tree series transducers over \( \mathcal{B} \)

In this section we show that, under the t-ts semantics, every bottom-up tree transducer can be simulated by a polynomial bottom-up tree series transducer over \( \mathcal{B} \) and vice versa, every polynomial bottom-up tree series transducer over \( \mathcal{B} \) can be simulated by a bottom-up tree transducer. The same correspondence holds for the top-down case.
4.1 The bottom-up case

Definition 4.1 Let $M = (Q, \Sigma, \Delta, Q_d, R)$ be a bottom-up tree transducer and $M' = (Q, \Sigma, \Delta, B, Q_d, \mu)$ be a polynomial bottom-up tree transducer over $B$. $M$ and $M'$ are related if the following equivalence is true: for every $k \geq 0$, $\sigma \in \Sigma^k$, $t \in T_\Delta(X_k)$, and $q, q_1, \ldots, q_k \in Q$,

$$(\mu_k(\sigma)_{q, q_1(x_1), \ldots, q_k(x_k)}, t) = 1 \text{ iff there is a rule } \sigma(q_1(x_1), \ldots, q_k(x_k)) \rightarrow q(t) \text{ in } R.$$ 

\hfill \Box

Lemma 4.2 Let $M$ be a bottom-up tree transducer with $\Sigma$ and $\Delta$ as input and output ranked alphabet, respectively, and $M'$ be a polynomial bottom-up tree transducer such that $M$ and $M'$ are related. Then, $\tau_M = \tau_{M'} \circ \text{pick}_\Delta$.

Proof. Let $M$ and $M'$ be specified as in Definition 4.1. First, we can prove by induction on $s$ that, for every $s \in T_\Sigma$, $t \in T_\Delta$, and $q \in Q$ the following equivalence holds: $(h_\mu(s)_q, t) = 1$ iff $s \Rightarrow_M q(t)$.

$s = \sigma(s_1, \ldots, s_k)$ with $k \geq 1$:

$s \Rightarrow_M q(t)$

iff $\sigma(s_1, \ldots, s_k) \Rightarrow_M q(t)$

iff there are $q_1, \ldots, q_k \in Q$ and $t_1, \ldots, t_k \in T_\Delta$ such that $\sigma(s_1, \ldots, s_k) \Rightarrow_M \sigma(q_1(t_1), \ldots, q_k(t_k)) \Rightarrow_M q(t)$

iff there are $q_1, \ldots, q_k \in Q$, $t_1, \ldots, t_k \in T_\Delta$, $t' \in T_\Delta(X_k)$, and there is a rule $\sigma(q_1(x_1), \ldots, q_k(x_k)) \rightarrow q(t')$ in $R$ such that $s_i \Rightarrow_M q_i(t_i)$ and $t = t'[t_1, \ldots, t_k]$

iff there are $q_1, \ldots, q_k \in Q$, $t_1, \ldots, t_k \in T_\Delta$, $t' \in T_\Delta(X_k)$ such that

$$(\mu_\sigma(q_1(x_1), \ldots, q_k(x_k)) \sum_{t' \in T_\Delta(X_k)} t'(h_\mu(s_1, t_1), \ldots, h_\mu(s_k, t_k)) q, t) = 1$$

iff $$(\sum_{w = q_1(x_1), q_2(x_2)} \mu_\sigma(w, q, t) (h_\mu(s_1, t_1), \ldots, h_\mu(s_k, t_k)) q, t) = 1$$

iff $$(\sum_{w = q_1(x_1), q_2(x_2)} \mu_\sigma(w, q, t) (h_\mu(s_1, t_1), \ldots, h_\mu(s_k, t_k)) q, t) = 1$$

iff $$(\sum_{w = q_1(x_1), q_2(x_2)} \mu_\sigma(w, q, t) (h_\mu(s_1, t_1), \ldots, h_\mu(s_k, t_k)) q, t) = 1$$

iff $$(\sum_{w = q_1(x_1), q_2(x_2)} \mu_\sigma(w, q, t) (h_\mu(s_1, t_1), \ldots, h_\mu(s_k, t_k)) q, t) = 1$$

Second, we can prove that $\tau_M = \tau_{M'} \circ \text{pick}_\Delta$. Let $s \in T_\Sigma$ and $t \in T_\Delta$.

$s \in \tau_M$

iff $\exists q \in Q_d : s \Rightarrow_M q(t)$

iff $\exists q \in Q_d : (h_\mu(s)_q, t) = 1$

iff $\exists q \in Q_d : (h_\mu(s)_q, t) \in \text{pick}_\Delta$

iff $\sum_{q \in Q_d} h_\mu(s)_q, t \in \text{pick}_\Delta$

iff $(\tau_{M'}(s), t) \in \text{pick}_\Delta$

iff $(s, t) \in \tau_{M'} \circ \text{pick}_\Delta$ \hfill \Box

Lemma 4.3 For every bottom-up tree transducer $M$ there is a polynomial bottom-up tree transducer $M'$ such that $M$ and $M'$ are related.

Proof. Let $M = (Q, \Sigma, \Delta, Q_d, R)$ be a bottom-up tree transducer. Construct the bottom-up tree transducer $M' = (Q, \Sigma, \Delta, B, Q_d, \mu)$ with the bottom-up tree representation $\mu$ over $Q$, $\Sigma$, $\Delta$, and $B$ as follows: for every $k \geq 0$, $\sigma \in \Sigma^k$, $t \in T_\Delta(X_k)$, and $q, q_1, \ldots, q_k \in Q$, $(\mu_\sigma(q_1(x_1), \ldots, q_k(x_k)), t) = 1$ if there is a rule $\sigma(q_1(x_1), \ldots, q_k(x_k)) \rightarrow q(t)$ in $R$, and $(\mu_\sigma(q_1(x_1), \ldots, q_k(x_k)), t) = 0$ if there is no such rule. Obviously, $M'$ is a polynomial, and $M$ and $M'$ are related. \hfill \Box

Example 4.4 (Cf. Example 2.1 of [Eng75].) Consider the bottom-up tree transducer $M = (Q, \Sigma, \Delta, Q_d, R)$ where $Q = \{q\}$, $\Sigma = \{\sigma^{(1)}, \sigma^{(2)}, \alpha^{(0)}\}$, $\Delta = \{\sigma^{(1)}, \alpha^{(1)}, \alpha^{(2)}\}$, $Q_d = \{q\}$, and $R$ consists of the following rules:

\begin{align*}
\end{align*}

13
\[
\begin{align*}
\bar{\gamma}(q(x_1)) & \rightarrow q(\sigma(x_1, x_1)) \\
\gamma(q(x_1)) & \rightarrow q(\gamma_1(x_1)) \\
\gamma(q(x_1)) & \rightarrow q(\gamma_2(x_1)) \\
\alpha & \rightarrow q(\alpha)
\end{align*}
\]

This is an example of Property (B1) in [Eng75]: a bottom-up tree transducer is able to process a subtree of an input tree in a nondeterministic way and copy the results of this process.

Using Lemma 4.3 we construct the bottom-up tree series transducer \( M' = (Q, \Sigma, \Delta, B, Q_d, \mu) \) as follows:

\[
\begin{align*}
\mu_1(\bar{\gamma})_{q, q(x_1)} &= \sigma(x_1, x_1) \\
\mu_1(\gamma)_{q, q(x_1)} &= \gamma_1(x_1) + \gamma_2(x_1) \\
\mu_0(\alpha)_{q, t} &= \alpha
\end{align*}
\]

Then, according to Lemmas 4.3 and 4.2, \( M' \) is the bottom-up tree series transducer such that \( \tau_M = \tau_{M'} \circ \text{pick}_\Delta \). Note that \( M' \) is not deterministic, hence is not a homomorphism tree series transducer.

Let us compute now in detail \( h_{\mu}(\alpha), h_{\mu}(\gamma \alpha) \), and \( h_{\mu}(\bar{\gamma} \gamma \alpha) \), and eventually \( \tau_{M'}(r) \) for the polynomial formal tree series \( r = \bar{\gamma} \gamma \alpha \).

\[
\begin{align*}
h_{\mu}(\alpha)_q &= \overline{\mu_0(\alpha)(\alpha)}_q = \mu_0(\alpha)_q, t = \alpha \\
h_{\mu}(\gamma \alpha)_q &= \\
&= \mu_1(\gamma)(h_{\mu}(\alpha))_q \\
&= \sum_{w=p_1(x_1), \ldots, p_l(x_1) \in Q((x_1))} \mu_1(\gamma)_{q, w} \leftarrow (h_{\mu}(\alpha)_{p_1}, \ldots, h_{\mu}(\alpha)_{p_l}) \\
&= \mu_1(\gamma)_{q, q(x_1)} \leftarrow (h_{\mu}(\alpha)_q) \\
&= (\gamma_1(x_1) + \gamma_2(x_1), \sigma(x_1, x_1)) \leftarrow (\alpha) \\
&= \gamma_1(\alpha) + \gamma_2(\alpha).
\end{align*}
\]

Finally,

\[
\begin{align*}
\tau_{M'}(\bar{\gamma} \gamma \alpha) &= \sum_{q \in Q_d} h_{\mu}(\bar{\gamma} \gamma \alpha)_q \\
&= h_{\mu}(\bar{\gamma} \gamma \alpha)_q \\
&= \sigma(\gamma_1(\alpha), \gamma_1(\alpha)) + \sigma(\gamma_2(\alpha), \gamma_2(\alpha)). \hfill \square
\end{align*}
\]

**Lemma 4.5** For every polynomial bottom-up tree series transducer \( M \) over \( B \) there is a bottom-up tree transducer \( M' \) such that \( M' \) and \( M \) are related.

**Proof.** Let \( M = (Q, \Sigma, \Delta, B, Q_d, \mu) \) be a polynomial bottom-up tree series transducer. Construct the bottom-up tree transducer \( M' = (Q, \Sigma, \Delta, Q_d, R) \) as follows. For every \( k \geq 0, \sigma \in \Sigma^k, t \in T_\Delta(X_k) \), and \( q, q_1, \ldots, q_k \in Q \), the rule \( \sigma(q_1(x_1), \ldots, q_k(x_k)) \rightarrow q(t) \) is in \( R \) if \( (\mu_k(\sigma)_{q_1(x_1), \ldots, q_k(x_k)}, t) = 1 \). Note that, since \( M \) is polynomial, \( R \) is a finite set. Obviously, \( M \) and \( M' \) are related. \hfill \square

From Lemmas 4.2, 4.3, and 4.5 immediately a characterization of bottom-up tree transducers in terms of polynomial bottom-up tree series transducers follows.

**Theorem 4.6** \( p-BOT_{t-t_d}(B) \circ PICK = BOT_{t_t} \).
4.2 The top-down case

Next we show that top-down tree transducers (as they occur in the literature) are characterized by polynomial top-down tree series transducers over the boolean semiring $\mathbb{B}$.

**Definition 4.7** Let $M = (Q, \Sigma, \Delta, Q_d, R)$ be a top-down tree transducer and let $M' = (Q, \Sigma, \Delta, B, Q_d, \mu)$ be a polynomial top-down tree series transducer over $\mathbb{B}$.  

$M$ and $M'$ are related if the following equivalence is true: for every $k \geq 0$, $\sigma \in \Sigma^{(k)}$, $q \in Q$, $w \in (Q(X_k))^*$, $t' \in \widehat{T_{\Delta}(X)}$ where $l = |w|$, 

$$(\mu_k(\sigma), q, w, t') = 1 \text{ if there is a rule } q(\sigma(x_1, \ldots, x_k)) \rightarrow t \in R \text{ such that } \lim_{Q(X_k)}(t) = (t', w).$$

\[\square\]

**Lemma 4.8** Let $M$ be a top-down tree transducer with $\Sigma$ and $\Delta$ as input and as output ranked alphabets, respectively, and $M'$ be a polynomial top-down tree series transducer such that $M$ and $M'$ are related. Then, $\tau_M = \tau_{M'} \circ \text{pick}_\Delta$ holds.

**Proof.** Let $M$ and $M'$ be specified as in Definition 4.7. We can prove that, for every $s \in T_\Sigma$, $t \in T_\Delta$, and $q \in Q$ the following equivalence holds: $(\mu(s), q, t) = 1$ if $q(s) \Rightarrow_M t$.

$s = \sigma(s_1, \ldots, s_k)$ with $k \geq 1$:

$q(s) \Rightarrow_M t$

<table>
<thead>
<tr>
<th>Condition</th>
<th>$\Rightarrow_M$</th>
<th>$\Rightarrow_M$</th>
</tr>
</thead>
<tbody>
<tr>
<td>if there are $q_1, \ldots, q_l \in Q$, $i_1, \ldots, i_l \in {1, \ldots, k}$, $t_1, \ldots, t_l \in T_\Delta$, and $t' \in \widehat{T_{\Delta}(X)}$ such that $q(\sigma(x_1, \ldots, x_k)) \rightarrow t' \left[ q_1(x_{i_1}), \ldots, q_l(x_{i_l}) \right] \Rightarrow_M t_1, \ldots, t_l$</td>
<td>$t'$ is reducible to $t$</td>
<td>$t'$ is reducible to $t$</td>
</tr>
<tr>
<td>if there are $q_1, \ldots, q_l \in Q$, $i_1, \ldots, i_l \in {1, \ldots, k}$, $t_1, \ldots, t_l \in T_\Delta$, and $t' \in \widehat{T_{\Delta}(X)}$ such that $q(\sigma(x_1, \ldots, x_k)) \rightarrow t' \left[ q_1(x_{i_1}), \ldots, q_l(x_{i_l}) \right] \in R$ and $q_j(s_{i_j}) \Rightarrow_M t_j$ and $t'[t_1, \ldots, t_l] = t$</td>
<td>$t'$ is reducible to $t$</td>
<td>$t'$ is reducible to $t$</td>
</tr>
<tr>
<td>if there are $q_1, \ldots, q_l \in Q$, $i_1, \ldots, i_l \in {1, \ldots, k}$, $t_1, \ldots, t_l \in T_\Delta$, and $t' \in \widehat{T_{\Delta}(X)}$ such that $(\mu_k(\sigma), q_1(x_{i_1}), \ldots, q_l(x_{i_l})) = 0$ if $t' \notin \widehat{T_{\Delta}(X)}$. The equation $\tau_M = \tau_{M'} \circ \text{pick}_\Delta$ can be verified in the same way as in the proof of Lemma 4.2.</td>
<td>$t'$ is reducible to $t$</td>
<td>$t'$ is reducible to $t$</td>
</tr>
</tbody>
</table>

At the step which is marked by * we have used the fact that the top-down tree representation $(\mu_k(\sigma), q_1(x_{i_1}), \ldots, q_l(x_{i_l})) = 0$ if $t' \notin \widehat{T_{\Delta}(X)}$. The equation $\tau_M = \tau_{M'} \circ \text{pick}_\Delta$ can be verified in the same way as in the proof of Lemma 4.2. \[\square\]

**Lemma 4.9** For every top-down tree transducer $M$ there is a polynomial tree transducer $M'$ such that $M$ and $M'$ are related.

**Proof.** Let $M = (Q, \Sigma, \Delta, Q_d, R)$ be a top-down tree transducer. Construct the top-down tree series transducer $M' = (Q, \Sigma, \Delta, B, Q_d, \mu)$ with the top-down tree representation $\mu$ over $Q$, $\Sigma$, $\Delta$, and $\mathbb{B}$ as follows: for every $k \geq 0$, $\sigma \in \Sigma^{(k)}$, $q \in Q$, $w \in (Q(X_k))^*$, $t' \in \widehat{T_{\Delta}(X)}$ where $l = |w|$, define $(\mu_k(\sigma), q, w, t') = 1$ if there is a rule $q(\sigma(x_1, \ldots, x_k)) \rightarrow t$ in $R$ such that $\lim_{Q(X_k)}(t) = (t', w)$, otherwise $(\mu_k(\sigma), q, w, t') = 0$. Obviously, $M'$ is and polynomial. Moreover, $M$ and $M'$ are related. \[\square\]

**Example 4.10** (cf. Example 2.2 of [Eng75].) Consider the top-down tree transducer $M = (Q, \Sigma, \Delta, Q_d, R)$ where $Q = \{q\}$, $\Sigma = \{\tilde{\sigma}(1), \gamma(1), \alpha(0)\}$, $\Delta = \{\sigma(2), \gamma(1), \alpha(0)\}$, $Q_d = \{q\}$, and $R$ consists of the following rules:

$q(\tilde{\sigma}(x_1)) \rightarrow \sigma(q(x_1), t(x_1))$
$q(\gamma(x_1)) \rightarrow \gamma(x_1)$
$q(\gamma(x_1)) \rightarrow \gamma(z(q(x_1)))$
$q(\alpha) \rightarrow \alpha$
This exhibits an example of the property (T) of [Eng77]: a top-down tree transducer is able to copy a subtree of its input tree and process the copies differently.

By Lemma 4.9 we construct the top-down tree representation $\mu$ over $Q$, $\Sigma$, $\Delta$, and $BB$ as follows:

$$
\begin{align*}
\mu_1(\vec{\gamma})_{q,q(x_1)q(x_2)} &= \sigma(x_1, x_2) \\
\mu_2(\vec{\gamma})_{q,q(x_1)} &= \gamma_1(x_1) + \gamma_2(x_1) \\
\mu_{0}(\alpha)_{q,t} &= \alpha
\end{align*}
$$

Then, according to Lemmas 4.8 and 4.9, $M' = (Q, \Sigma, \Delta, B; Q, \mu)$ is the top-down tree series transducer such that $\tau_M = \tau_{M'} \circ p\text{ick}_\Delta$.

Let us again compute in detail $h_\mu(\alpha)$, $h_\mu(\gamma \alpha)$, and $h_\mu(\vec{\gamma} \gamma \alpha)$, and eventually $\tau_{M'}(r)$ for the polynomial formal tree series $r = \vec{\gamma} \gamma \alpha$.

$$
\begin{align*}
\mu_1(\vec{\gamma})_{q} &= \mu_1(\vec{\gamma})_{h_\mu(\gamma \alpha)q} \\
&= \sum_{q = q(x_1), \ldots, q(x_n) \in \Sigma^*, \mu_1(\vec{\gamma})_{q} = \gamma_1(x_1) + \gamma_2(x_1)} \\
&= \sigma(\gamma_1, \gamma_2) + \sigma(\gamma_1, \gamma_2)
\end{align*}
$$

Finally,

$$
\begin{align*}
\tau_{M'}(\vec{\gamma} \gamma \alpha) &= \sum_{q \in Q_0} h_\mu(\vec{\gamma} \gamma \alpha)_q \\
&= \sigma(\gamma_1, \gamma_1) + \sigma(\gamma_1, \gamma_2) + \sigma(\gamma_2, \gamma_1) + \sigma(\gamma_2, \gamma_2).
\end{align*}
$$

\textbf{Lemma 4.11} For every polynomial top-down tree series transducer $M$ over $B$ there is a top-down tree transducer $M'$ such that $M$ and $M'$ are related.

\textbf{Proof.} Let $M = (Q, \Sigma, B, Q_d, \mu)$ be a polynomial top-down tree series transducer. Construct the top-down tree transducer $M' = (Q, \Sigma, \Delta, Q_d, \mu)$ as follows. Let $R$ be the smallest set of rules satisfying the condition that for every $k \geq 0$, $\sigma \in \Sigma^k$, $q \in Q$, $w = q_1(x_{i_1}) \ldots q_k(x_{i_k}) \in (Q(X_k))^*$, and $t' \in T_\Delta(X_i)$ with $(\mu_2(\sigma)_{q,w}, t') = 1$, the rule $q(\sigma(x_1, \ldots, x_k)) \rightarrow t$ is in $R$, where $t = t'[x_1 \rightarrow q_1(x_{i_1}), \ldots, x_k \rightarrow q_k(x_{i_k})]$. (Note that $\text{lin}(q(X))(t) = (t', w)$.) Again, since $M$ is polynomial, $R$ is a finite set. Obviously, $M$ and $M'$ are related. 

Thus, it follows from Lemmas 4.8, 4.9, and 4.11 that also for the top-down case we obtain a characterization of top-down tree transducers in terms of polynomial top-down tree series transducers.

\textbf{Theorem 4.12} $\text{pTOP}_{t \rightarrow t}(B) \circ \text{PICK} = \text{TOP}_{t}$.

We would like to finish this section by comparing the embedding of top-down tree transducers into top-down tree series transducers as done in our Lemma 4.9 with the embedding of nondeterministically simple root-to-frontier tree transducers into tree transducers as done on page 139 of [Kui99]. This comparison is reasonable, because nondeterministically simple root-to-frontier transducers are particular top-down tree transducers, and tree transducers of [Kui99] are particular top-down tree series transducers.

16
Consider the top-down tree transducer in Example 4.10. In fact, this is a nondeterministically simple root-to-frontier transducer in the sense of [Kui99]. In the construction of [Kui99], the tree representation \( \mu \) (using our notation) has the following entry:

\[
(\mu_{1}(\bar{x}))[q,q(x_1), t] = 1
\]

where \( t = \sigma(x_1, x_1) \). Compare this with the corresponding entry of the top-down tree representation in our Example 4.10:

\[
(\mu_{1}(\bar{x}))[q,q(x_1), q(x_1), t] = 1
\]

where \( t = \sigma(x_1, x_2) \).

We observe that in Kuich’s approach the copying of the input subtree is done in the term \( t \), whereas in our approach the copying is done in the index \( w \) of the top-down tree representation. Both solutions are possible. Also, it is clear that, if the solution of Kuich is chosen, then the tree series in Definition 3.2 have to be substituted in an \( OI \)-like way (as it was done in [Kui99] on page 136) and not in an \( IO \)-like way as we have defined it on page 6. On the other hand, bottom-up tree series transductions cannot be described by using \( OI \)-substitution. Thus, in order to deal with one substitution only, we choose the \( IO \)-substitution.

5 Two characterization results

In this section we characterize the classes \( p\cdot BOT_{t\to t} \) and \( td\cdot TOP_{t\to t} \). Using Lemma 2.2, the corresponding characterizations of the classes \( p\cdot BOT_{ts\to ts} \) and \( td\cdot TOP_{ts\to ts} \) will follow immediately. So let us concentrate now on the classes of \( t\)-ts transformations.

5.1 The characterization for the bottom-up case

First we characterize the class \( p\cdot BOT_{t\to ts} \) in terms of the composition of finite state relabeling tree series transducers and homomorphism tree series transducers. The proof is based on one decomposition lemma (cf. Theorem 3.15 of [Eng75]) and one composition lemma (cf. Lemma 4.1 of [Eng75]).

**Lemma 5.1**

1) \( p\cdot BOT_{t\to ts} \subseteq QREL_{t\to ts} \circ p\cdot HOM_{t\to ts} \)

2) \( dp\cdot BOT_{t\to ts} \subseteq dQREL_{t\to ts} \circ p\cdot HOM_{t\to ts} \)

**Proof.** First we prove 1). For this, let \( M = (Q, \Sigma, \Delta, A, Q_d, \mu) \) be a polynomial bottom-up tree series transducer. We will construct a finite state relabeling tree series transducer \( M_1 \) and a polynomial homomorphism tree series transducer \( M_2 \), such that \( \tau_M = \tau_{M_1} \circ \tau_{M_2} \).

Define the ranked alphabet \( \Omega = \bigcup_{\sigma \in \Sigma} \Omega_{\sigma} \), where, for every \( k \geq 0 \), \( \sigma \in \Sigma^{(k)} \) we let \( \Omega_{\sigma} = \{ [\sigma, q, w, t]^{(k)} | q \in Q, w = q_1(x_1) \ldots q_k(x_k) \in (Q(X_k))^*, t \in supp(\mu_k(\sigma,q,w)) \} \). Note that, since \( M \) is polynomial, \( \Omega \) is a finite set.

Now let \( M_1 = (Q, \Sigma, \Delta, A, Q_d, \mu_1) \) be such that for every \( k \geq 0 \), \( \sigma \in \Sigma^{(k)} \), \( q \in Q \), \( w = q_1(x_1) \ldots q_k(x_k) \in (Q(X_k))^* \) and \( t' \in T_D(X_k) \)

\[
((\mu_1)_k(\sigma,q,w,t')) = \begin{cases} 
\mu_k(\sigma,q,w,t) & \text{if } t' = [\sigma, q, w, t](x_1, \ldots, x_k) \\
0 & \text{otherwise.} 
\end{cases}
\]

Let \( M_2 = (\{*, \}, \Omega, \Delta, A, \{*, \}, \mu_2) \) be such that for every \( k \geq 0 \), \( [\sigma, q, w, t] \in \Omega^{(k)} \), and \( t' \in T_D(X_k) \) define

\[
((\mu_2)_k([\sigma, q, w, t], *, x_1, \ldots, x_k, t')) = \begin{cases} 
1 & \text{if } t' = t \\
0 & \text{otherwise.} 
\end{cases}
\]

Now we prove that \( \tau_M = \tau_{M_1} \circ \tau_{M_2} \).
\[ \tau_M(s) = \sum_{q \in Q_M} h_{\mu}(s)_q \]
\[ = \sum_{q \in Q_M} h_{\mu}(s)_q \]
\[ = h_{\mu}(\tau_M(s))_s = (\tau_M \circ \tau_M)(s) \]

The equation marked by + has not been proved yet. In order to verify it, we extend \( \mu_2 \) to \( \mu'_2 \) by letting \( \mu'_2 = ((\mu'_2)_0, (\mu'_2)_k) \quad \text{for} \quad k \geq 1 \), where \((\mu'_2)_0 : (\Delta(0) \cup X_{mx}) \rightarrow A\{T_{\Delta_{\ast}}(X_{mx})\}^{\ast \times \{\ast\}} \). Here \( mx = \text{max}\{k | \Omega^{(k)} \neq \emptyset \} \). The mapping \((\mu'_2)_0(\alpha) = (\mu_2)_0(\alpha) \) for every \( \alpha \in \Omega^{(0)} \), and \((\mu'_2)_0(x_i) = *(x_i) \) for every \( x_i \in X_{mx} \).

In this extension the elements of \( X_{mx} \) and of \( \{\ast\}(X_{mx}) \) are considered as 0-ary symbols, therefore \( \mu'_2 \) is a homomorphism tree representation over \( \{\ast\}, \Omega \cup X_{mx}, \Delta \cup \{\ast\}(X_{mx}), \) and \( A \).

Now we can verify the equation + by showing that for every \( q \in Q \) and \( s \in T_{\Sigma} \) the equality \( h_{\mu}(s)_q = h_{\mu'_2}(h_{\mu}(s)_q)_s \) holds. This proof will be performed by induction on \( s \).

\[ s = \sigma(s_1, \ldots, s_k) : \]
\[ h_{\mu}(\sigma(s_1, \ldots, s_k))_q = \mu_2(\sigma(h_{\mu}(s_1), \ldots, h_{\mu}(s_k)))_q \]
\[ = \mu_2(\sigma_q(h_{\mu}(s_1), \ldots, h_{\mu}(s_k)))_q \]
\[ = \sum_{w=q_1(s_1) \ldots q_k(s_k)} \mu_2(\sigma)(q,w) \leftarrow (h_{\mu}(s_1), \ldots, h_{\mu}(s_k))_q \]
\[ = \sum_{w=q_1(s_1) \ldots q_k(s_k)} \mu_2(\sigma)(q,w) \leftarrow (h_{\mu}(s_1), \ldots, h_{\mu}(s_k))_q \]
\[ = \sum_{w=q_1(s_1) \ldots q_k(s_k)} \mu_2(\sigma)(q,w) \leftarrow (h_{\mu}(s_1), \ldots, h_{\mu}(s_k))_q \]
\[ = \mu_2(\sum_{w=q_1(s_1) \ldots q_k(s_k)} \mu_2(\sigma)(q,w) \leftarrow (h_{\mu}(s_1), \ldots, h_{\mu}(s_k))_q) \]
\[ = h_{\mu'}(\mu_2(\sigma)(s_1), \ldots, s_k))_q \]

Here equation ++ holds by definition of \( \mu_1 \) and \( \mu'_2 \). This can be checked easily. Moreover, equation +++ also holds because the result of the computation of a homomorphism tree series transducer on an input tree \( t[t_1, \ldots, t_k] \) is of the form \( t'[t'_1, \ldots, t'_k] \), where \( t', t'_1, \ldots, t'_k \) are the result of the computations on the input trees \( t, t_1, \ldots, t_k \), respectively. In fact, the extension desired above is necessary to make the tree series transducer \( M_2 \) able to handle the input \( t \), which may contain variables.

It is clear that, if \( M \) is deterministic, then also \( M_1 \) is deterministic, hence the inclusion 2) is also proved.

In the following lemma, we compose a (deterministic) polynomial bottom-up tree series transducer with a polynomial bottom-up homomorphism tree series transducer. The construction involved is closely related to the direct product construction [Bak79] for bottom-up tree transducers.

**Lemma 5.2** 1) \( p\text{-BOT}_{\tau_0 \text{-} t} \circ p\text{-HOM}_{\tau_0 \text{-} t} = p\text{-BOT}_{\tau_0 \text{-} t} \)
2) \( dp\text{-BOT}_{\tau_0 \text{-} t} \circ p\text{-HOM}_{\tau_0 \text{-} t} = dp\text{-BOT}_{\tau_0 \text{-} t} \)

**Proof.** First we prove 1). The inclusion \( p\text{-BOT}_{\tau_0 \text{-} t} \subseteq p\text{-BOT}_{\tau_0 \text{-} t} \circ p\text{-HOM}_{\tau_0 \text{-} t} \) follows from the obvious fact that for every \( \tau \in p\text{-BOT}_{\tau_0 \text{-} t} \), we have \( \tau = \tau \circ \tau \) where \( \tau \) is a suitable identity tree to tree series transformation. Moreover, \( \tau \in p\text{-HOM}_{\tau_0 \text{-} t} \), see our note at the end of Section 3.

In order to prove the converse inclusion, let \( M_1 = (Q_1, \Sigma, \Delta, A, Q_{d_1}, \mu_1) \) be a polynomial bottom-up and \( M_2 = (\{\ast\}, \Delta, \Gamma, A, \{\ast\}, \mu_2) \) be a homomorphism tree series transducer.

Construct the bottom-up tree series transducer \( M = (Q_1, \Sigma, \Gamma, A, Q_{d_1}, \mu) \) where, for every \( k \geq 0, \sigma \in \Sigma^{(k)} \), and \( q, q_1, \ldots, q_k \in Q_1 \),

\[ \mu_k(\sigma)(q_1(s_1) \ldots q_k(s_k)) = \mu_{\mu_1, \mu_2}(\mu_1)(\mu_2)(\sigma)(q_1(s_1) \ldots q_k(s_k)) \]

where \( \mu_{\mu_1, \mu_2} \) is the following extension of \( \mu_2 \): \( \mu_{\mu_1, \mu_2} = ((\mu_{\mu_1, \mu_2})_0, (\mu_{\mu_1, \mu_2})_i) \quad \text{for} \quad i \geq 1 \) and \( (\mu_{\mu_1, \mu_2})_0 : \Delta(0) \cup X_{mx} \rightarrow A\{T_{\Delta_{\ast}}(X_{mx})\}^{\{\ast\}} \) such that,

- for every \( \alpha \in \Delta^{(0)} \), \( (\mu_{\mu_1, \mu_2})_0(\alpha) = (\mu_2)_0(\alpha) \) and
- for every \( 1 \leq i \leq k \), \( (\mu_{\mu_1, \mu_2})_0(x_i)_\ast = *(x_i) \),
- for every \( j \not\in \{1, \ldots, k\} \), \( (\mu_{\mu_1, \mu_2})_0(x_j)_\ast = 0 \).
It should be clear that $M$ is also a polynomial tree series transducer.

Now we prove the equality $\tau_M = \tau_{M_1} \circ \tau_{M_2}$ which shows the correctness of the construction. Let $s \in T_\Sigma$.

\[
\begin{align*}
& (\tau_{M_1} \circ \tau_{M_2})(s) \\
& = \tau_{M_2}(\tau_{M_1}(s)) \\
& = \tau_{M_2}(\sum_{q \in Q_{M_1}} h_{\mu_1}(s)_q) \\
& = h_{\mu_2}(\sum_{q \in Q_{M_1}} h_{\mu_1}(s)_q) \\
& = \sum_{q \in Q_{M_1}} h_{\mu_2}(h_{\mu_1}(s)_q) \\
& = \tau_M(s)
\end{align*}
\]

At the equation + we have used the fact that, for every $q \in Q_1$ and $s \in T_\Sigma$, the equality $h_{\mu_2}(h_{\mu_1}(s)_q) = h_{\mu}(s)_q$ holds. This fact we will prove now by induction on $s$.

\[
\sigma(s_1, \ldots, s_k):
\]

\[
\begin{align*}
& h_{\mu}(\sigma(s_1, \ldots, s_k))_q \\
& = \mu_k(\sigma)(h_{\mu_1}(s_1), \ldots, h_{\mu_k}(s_k))_q \\
& = \sum_{u = q_1(s_1) \ldots q_k(s_k)} \mu_k(\sigma)_{q, u} \rightarrow (h_{\mu_1}(s_1)_q, \ldots, h_{\mu_k}(s_k)_q)
\end{align*}
\]

In this way we get:

\[
\begin{align*}
& \sum_{u = q_1(s_1) \ldots q_k(s_k)} \mu_k(\sigma)_{q, u} \rightarrow \left(\overline{h_{\mu_1}(h_{\mu_1}(s_1)_q), \ldots, \overline{h_{\mu_2}(h_{\mu_1}(s_k)_q)}}\right)
\end{align*}
\]

The equation ++ holds by definition of $\mu$. Moreover, equation +++ holds because of the way in which the bottom-up homorphism tree series transducer $M_2$ computes and because of the definition of $\mu_{s, k}$ (cf. also Lemma 1.1 of [Eng75]). This finishes the proof of 1).

If $M_1$ is deterministic in the above construction, then $M$ is also deterministic, hence 2) is also proved.

Now the characterization of the class $p\text{-}BOT_{t-ts}$ follows immediately from Lemmas 5.1 and 5.2 and the fact that every finite state relabeling tree series transducer is polynomial.

**Theorem 5.3**

1) $p\text{-}BOT_{t-ts} = QREL_{t-ts} \circ p\text{-}HOM_{t-ts}$

2) $dp\text{-}BOT_{t-ts} = d-QREL_{t-ts} \circ p\text{-}HOM_{t-ts}$

It follows immediately from Lemma 2.2 that the classes $p\text{-}BOT_{ts-ts}$ and $dp\text{-}BOT_{ts-ts}$ can be characterized in the same way.

**Corollary 5.4**

1) $p\text{-}BOT_{ts-ts} = QREL_{ts-ts} \circ p\text{-}HOM_{ts-ts}$

2) $dp\text{-}BOT_{ts-ts} = d-QREL_{ts-ts} \circ p\text{-}HOM_{ts-ts}$

### 5.2 The characterization for the top-down case

Second we characterize the class $(t)\text{-}TOP_{t-ts}$ (cf. Theorem 5.7). We prepare this characterization by proving two lemmas (cf. Lemma 3.6 of [Eng75]).

**Lemma 5.5**

1) $TOP_{t-ts} \subseteq HOM_{t-ts} \circ l\text{-}TOP_{t-ts}$

2) $d\text{-}TOP_{t-ts} \subseteq HOM_{t-ts} \circ ld\text{-}TOP_{t-ts}$

3) $td\text{-}TOP_{t-ts} \subseteq HOM_{t-ts} \circ ltd\text{-}TOP_{t-ts}$. 

19
Proof. First we prove 1. Therefore, let $M = (Q, \Sigma, \Delta, A, Q_0, \mu)$ be a top-down tree series transducer. Construct the homomorphism tree series transducer $H = (\{\ast\}, \Sigma, \Delta, A, \{\ast\}, \mu_H)$ and the linear top-down tree series transducer $M' = (Q, \Sigma', \Delta, A, Q_0, \mu'_H)$ such that $\tau_M = \tau_H \circ \tau_{M'}$.

Let $n = \max\{\#occ(x_i, w)\mid k \geq 0, \sigma \in \Sigma^{(k)}, q \in Q, w \in (Q(X_k))^*, \mu_k(\sigma)_{q,w} \neq 0, 1 \leq i \leq k\}$ where $\#occ(x_i, w)$ denotes the number of occurrences of $x_i$ in $w$.

Define $\Sigma' = \{\sigma'(k-n)\mid \sigma \in \Sigma^{(k)}\}$ and for every $k \geq 0$ and and $\sigma \in \Sigma^{(k)}$, define

$$(\mu_H)_{\sigma,w} = \sigma'(x_1, \ldots, x_n, \ldots, x_{(k-1)n+1}, \ldots, x_{kn})$$

where $\tilde{w} = *\sigma(x_1, \ldots, \ast(x_1) \ldots \ast(x_k) \ldots \ast(x_k))$ and for every $w \neq \tilde{w}$ define

$$(\mu_H)_{\sigma,w} = 0.$$ 

Thus, $H$ is a homomorphism tree series transducer.

Now we turn to the construction of $M'$. Let $w = q_1(x_i_1) \ldots q_l(x_i_l) \in (Q(X_k))^*$. Define $w' = q_1(x_{j_1}) \ldots q_l(x_{j_l})$ where, for every $1 \leq d \leq l$, $x_{j_d} = x_{(u-1)n+k}$ if $x_{i_d} = x_{u}$ and $q_d(x_{i_d})$ contains the $k$th occurrence of $x_u$ in $w$. (Cf. the proof of Lemma 3.6 in [Engl1].) Then define for every $k \geq 0$, $\sigma \in \Sigma^{(k)}$, $q \in Q$, $w \in (Q(X_k))^*$:

$${\mu'_H}_{\sigma,w} = \begin{cases} \mu_k(\sigma)_{q,w} & \text{if there is a } v \in (Q(X_k))^* \text{ such that } v' = w \\ 0 & \text{otherwise} \end{cases}$$

It is easy to see that $M'$ is linear. Moreover, if $M$ is (total) deterministic, then it is easy to check that $M'$ is also (total) deterministic.

Before we prove that $\tau_M = \tau_H \circ \tau_{M'}$, we define the mapping $copy: T_{\Sigma} \to T_{\Sigma'}$ and show its connection with $\tau_H$. The mapping $copy$ is defined by induction as follows: $copy(\sigma(s_1, \ldots, s_k)) = \sigma'(s_1', \ldots, s_k')$ where for every $1 \leq j \leq k-n$, let $s_j' = copy(s_j)$ where $i$ is determined by the conditions $j = (i-1)n+k$ and $1 \leq k < n$. We extend the mapping $copy$ in a natural way to the t-ts transformation $copy: T_{\Sigma} \to A(T_{\Sigma'})$ by defining, for every $s \in T_{\Sigma}$, $copy(s) = 1 \cdot copy(s)$. Then, for every $s \in T_{\Sigma}$, $copy(s) = \tau_H(s)$.

Now we prove that $\tau_M = \tau_H \circ \tau_{M'}$. Let $s \in T_{\Sigma}$.

We can verify equation + by proving the following statement by induction on $s$: for every $s \in T_{\Sigma}$, $q \in Q$, $\mu(s,q) = \mu(s, copy(q))$.

At equation ++ we used the property that $\mu'_H(\sigma',q,w) = 0$ if $u \neq w$ for some $w \in (Q(X_k))^*$ with $\mu_k(\sigma,q,u) = 0$.

Now if $M$ is deterministic (and total) in the above proof, then $M'$ will also be deterministic (and total). This verifies 2) and 3).

\[\square\]
Lemma 5.6  1) \( td\cdot \text{TOP}_{\tau_{\text{ts}}} \circ d\cdot \text{TOP}_{\tau_{\text{ts}}} = d\cdot \text{TOP}_{\tau_{\text{ts}}} \)
2) \( td\cdot \text{TOP}_{\tau_{\text{ts}}} \circ td\cdot \text{TOP}_{\tau_{\text{ts}}} = td\cdot \text{TOP}_{\tau_{\text{ts}}} \).

Proof. First we prove 1). The inclusion \( d\cdot \text{TOP}_{\tau_{\text{ts}}} \subseteq td\cdot \text{TOP}_{\tau_{\text{ts}}} \circ d\cdot \text{TOP}_{\tau_{\text{ts}}} \) follows from that for every \( \tau \in d\cdot \text{TOP}_{\tau_{\text{ts}}} \) we have \( \tau = \iota \circ \tau \), where \( \iota \) is a suitable identity tree to tree series transformation.

The converse inclusion can be seen as follows. Let \( M_1 = (Q_1, \Sigma, \Delta, q_1, A, \mu_1) \) and \( M_2 = (Q_2, \Delta, \Gamma, q_2, A, \mu_2) \) be deterministic top-down tree series transducers and assume also that \( M_1 \) is total.

We construct a third deterministic top-down tree series transducer \( M = (Q, \Sigma, \Gamma, q_0, A, \mu) \) such that \( \tau_M = \tau_{M_1} \circ \tau_{M_2} \). Our construction is the generalization of the usual direct product construction decrived among others in [Bak79, Eng75, FV98].

First we extend \( \mu \) to \( \mu' \) by letting \( \mu'_\alpha = ((\mu_\alpha)'_0, (\mu_\alpha)'_k \mid k \geq 1) \), where \( (\mu_\alpha)'_0 : (\Delta(0) \cup X_{m_\alpha}) \to A(T_{\text{TS}}(\Sigma))(X_{m_\alpha}) \). Here \( m_\alpha = \max \{ k \mid \Sigma^{(k)}(0) \neq \emptyset \} \). The mapping \( (\mu_\alpha)'_0 \) is defined such that \( (\mu_\alpha)'_0(\alpha) = (\mu_\alpha)'_0(\alpha) \) for every \( \alpha \in \Delta(0) \), and \( (\mu_\alpha)'_0(x_i)_p = p(x_i) \) for every \( x_i \in X_{m_\alpha} \) and \( p \in Q_1 \).

In this extension the elements of \( X_{m_\alpha} \) and of \( Q_2(X_{m_\alpha}) \) are considered as 0-ary symbols, therefore \( \mu' \) is a top-down tree representation over \( Q_2, \Delta \cup X_{m_\alpha}, \Gamma \cup Q_2(X_{m_\alpha}), \) and \( A \).

Now we can construct \( M \) as follows. Let \( Q = Q_1 \times Q_1 \) and \( q_0 = (q_2, q_1) \). Moreover, the mapping \( \mu \) is defined as follows. For every \( k \geq 0, \sigma \in \Sigma^{(k)} \), \( (p, q) \in Q, v \in (Q(X_k))^* \), \( t' \in T_{\text{TR}}(X_m) \), where \( m = |v| \), and \( a, a \in A \).

\((\mu_{k_\alpha}(\sigma)_{p,q}, t') = a\) if

there exist \( w \in (Q_1(X_k))^* \), \( t \in \overline{T_\Delta}(X_i) \), and \( i \in T_\Gamma(Q_2(X_i)) \) such that the following conditions hold:

- \( l = |w| \),
- \( ((\mu_\alpha)'_0(\sigma), q, w, t) = a \),
- \( (h_{\mu'}(t)_p, l) = a \),
- \( \text{lin}_{Q_2(x_i)}(i) = (t', u) \), and
- \( v = u(w) \).

The notion \( u(w) \) is defined in the following way. Let \( w \in (Q_1(X_k))^* \) with \( |w| = l \) and let \( u \in (Q_2(X_i))^* \). Then \( u(w) \) is a word in \((Q(X_k))^* \) obtained by replacing, for every \( p \in Q_2 \) and \( x_i \in X_i \), the expression \( p(x_i) \) in \( u \) by \( (p, q)(x_j) \) where \( q(x_j) \) is the \( i \)th letter in \( v \).

Now we prove that \( \tau_N = \tau_{M_1} \circ \tau_{M_2} \).

Hence it is sufficient to show that for every \( s \in T_\Sigma, p \in Q_1 \) and \( q \in Q_2 \) the equation \( h_\mu(s)_{p,q} = h_\mu(s)_{p,q} \). We prove by induction on \( s \).

\( s = \sigma(s_1, \ldots, s_k) \):

\[ h_\mu(\sigma(s_1, \ldots, s_k))_{p,q} = h_\mu_1(h_\mu(s_1), \ldots, h_\mu(s_k))_{p,q} \]

\[ \mu_\alpha(\sigma)_{p,q} \leftarrow (h_\mu(s_1)_{p_1,q_1}, \ldots, h_\mu(s_m)_{p_m,q_m}) \]

for some \( v \in (p_1, q_1)(x_1), \ldots, (p_m, q_m)(x_m) \in (Q_2 \times Q_1(X_k))^* \)

\( s_1 \ldots s_k \):

\[ h_\mu_1((\mu_\alpha)_{p,q})_{p,q} \leftarrow (h_\mu_1(h_\mu_1(s_1, q_1), \ldots, h_\mu_1(s_m, q_m))_{p_m}) \]

for some \( w \in (Q_1(X_k))^* \) with \( w = q'_1(x_1), \ldots, q'_k(x_j) \) and \( u \in (Q_2(X_i))^* \)

such that the following conditions hold:

\[ h_\mu_2((\mu_\alpha)'_0(\sigma), q, w, t) = a, \text{lin}_{Q_2(\Delta)}(i) = (t', u) \text{ and } v = u(w) \].

\[ h_\mu_2((\mu_\alpha)'_0(\sigma), q, w, t) = a, \text{lin}_{Q_2(\Delta)}(i) = (t', u) \text{ and } v = u(w) \].

21
The equation \( + \) holds because of the construction of \( M \) while \( ++ \) holds because it describes how a top-down tree series transducers works on an input of the form \( f[t_1, \ldots, t_k] \). Moreover, at \( ++ \) we also used that \( M_1 \) is total. In fact, it is possible that some of the variables \( x_{ji_1}, \ldots, x_{ji_m} \) do not occur among the \( x_{i_1}, \ldots, x_{i_m} \) (because \( M_2 \) may delete them). Hence some of the formal tree series 
\[
 h_{\mu_1}(s_1)^{q_1'} \cdots h_{\mu_2}(s_1)^{q_2'} \ldots h_{\mu_1}(s_1)^{q_1''} \cdots h_{\mu_2}(s_1)^{q_2''} 
\]
may not be among 
\[
 h_{\mu_1}(s_1)^{q_1'} \cdots h_{\mu_2}(s_1)^{q_2'} \ldots h_{\mu_1}(s_1)^{q_1''} \cdots h_{\mu_2}(s_1)^{q_2''} . \]
However, if \( M_1 \) is total, then each of 
\[
 h_{\mu_1}(s_1)^{q_1'} \cdots h_{\mu_2}(s_1)^{q_2'} \ldots h_{\mu_1}(s_1)^{q_1''} \cdots h_{\mu_2}(s_1)^{q_2''} . 
\]
exists. This finishes the proof of 1).

Certainly, if \( M_1 \) is deterministic, then \( M \) will also be deterministic. Hence 2) is also shown. □

From Lemmas 5.5 and 5.6 we obtain the following characterization immediately.

**Theorem 5.7** \( HOM_{t-t} \circ ld-TOP_{t-t} = d-TOP_{t-t} \)

As a corollary from Lemma 5.6, Theorem 5.7 and Lemma 2.2 we obtain characterizations for the corresponding classes of \( t-s \) transformations.

**Corollary 5.8**

1) \( td-TOP_{t-t} \circ d-TOP_{t-t} = d-TOP_{t-t} \)

2) \( td-TOP_{t-t} \circ td-TOP_{t-t} = td-TOP_{t-t} \)

3) \( HOM_{t-t} \circ ld-TOP_{t-t} = d-TOP_{t-t} \)

### 6 Further research topics

Clearly, the present paper is only the starting point for further investigations on tree series transducers.

For instance, the homomorphism \( \mu \) can be considered as a homomorphism tree transducer with external functions [FHV93] in which the algebra does not have trees as carrier sets but formal tree series. Which results can be achieved by generalizing from homomorphism tree transducers to top-down, attributed or macro tree transducers? Clearly, for this we would need an appropriate definition of tree representation.

The extension \( \tilde{\tau} \) of a tree series transformation \( \tau \) in, say \( BOT_{t-t} \), is a transformation of formal tree series. Which results can be proved if the arguments of such transformations are themselves computed by some tree transducer device?

In a natural way, one can define bottom-up (and top-down) finite tree series automata, e.g., a bottom-up tree series transducer \( M \) is a bottom-up finite-state tree series automaton if \( \Sigma = \Delta \) and, for every \( k \geq 0 \), \( \sigma \in \Sigma^{(k)} \), \( q, q_1, \ldots, q_k \in Q \), if \( \mu_k(\sigma_q q_1(x_1), \ldots, q_k(x_k)) \neq 0 \), then \( \text{supp}(\mu_k(\sigma_q q_1(x_1), \ldots, q_k(x_k))) = \{ \sigma(x_1, \ldots, x_k) \} \). Let \( BOT_{t-s} \) denote the class of \( t-s \) transformations computed by bottom-up finite-state tree series automata. Now it is a natural question whether - as it holds in the tree transducer case - deterministic polynomial bottom-up finite-state tree series automata are as powerful as their nondeterministic version, i.e., whether \( p-BOT_{t-s} \subseteq dp-BOT_{t-s} \). Unfortunately, the naive adaptation of the powerset construction does not work, because then also coefficients are summed up which do not count in the nondeterministic case. However, we think that \( p-BOT_{t-s} \subseteq \text{REL} \circ dp-BOT_{t-s} \) where \( \text{REL} \) denotes a class of relabelings that add to every node the set \( P \) of states such that \( p \in P \) iff the nondeterministic automaton can reach this node in state \( q \), and \( q \) is successful (for the notion of successful cf. [BE97]).

**Acknowledgement**

The second author would like to thank Manfred Droste and Dietrich Kuske for fruitful discussions on this topic.

**References**


