Theoretical Computer Science 93 (1992) 43–74
Elsevier

Fundamental Study

Top-down tree transducers with two-way tree walking look-ahead

Sándor Vágvölgyi

Research Group on Theory of Automata, Hungarian Academy of Sciences, Aradi tér 1, H-6720 Szeged, Hungary

Communicated by M. Nivat
Received May 1989

Abstract

S. Vágvölgyi, Top-down tree transducers with two-way tree walking look-ahead, Theoretical Computer Science 93 (1992) 43–74.

We consider top-down tree transducers with deterministic, nondeterministic and universal two-way tree walking look-ahead and compare the transformational powers of their deterministic and strongly deterministic versions by giving the inclusion diagram of the induced tree transformation classes. We also study the closure properties of these transformation classes with respect to composition.

Contents

1. Introduction

Top-down tree transducers were introduced by Rounds [21] and Thatcher [23] to give an algebraic formalism for the theory of syntax-directed translation. The class of top-down tree transformations is not closed under composition, in fact, the increasing powers lead to an infinite hierarchy with respect to inclusion (cf. [4]).

Engelfriet [2] defined the notion of a top-down tree transducer with regular look-ahead in order to get nice closure properties with respect to composition. It was shown that both classes of linear and deterministic top-down tree transformations with regular look-ahead are closed under composition. Moreover, the concept of look-ahead turned out to be an elegant and efficient tool in tree language theory. In [3] Engelfreit showed that the class of deterministic top-down tree transformations with regular look-ahead equals the class of partial functions that can be realized by compositions of bottom-up or top-down tree transformations. Furthermore, Engelfriet et al. [6] proved that the deterministic checking tree pushdown tree transducer is equivalent to the deterministic top-down tree-to-string transducer with regular look-ahead. Finally, Engelfriet and Vogler [7] showed that the OI-macro tree transducer with regular look-ahead is equivalent to the OI-macro tree transducer. The concept of look-ahead for storage types was studied in [5, 8, 9, 10].

A top-down tree transducer was equipped with deterministic top-down look-ahead capability in [17]. It means that the look-ahead sets of this type of transducers are restricted to tree languages recognizable by deterministic top-down tree automata. The notion of determinism in the sense of [2] was also specialized and called strong determinism in [17]. As a result, it was shown that the class of all strongly deterministic top-down tree transformations with deterministic top-down look-ahead is also closed under composition and it is the closure of the class of deterministic top-down tree transformations under composition.

So, it turned out from [17] that both strong determinism and deterministic top-down look-ahead are of theoretical interest. Therefore, in [12] we combined these notions with the original concept of determinism and regular look-ahead (see [2]) and obtained new tree transducers such as strongly deterministic top-down tree transducer with look-ahead, etc. In [13] the look-ahead tree languages for deterministic top-down tree automata were iterated and in this way an infinite hierarchy of tree language classes, between the class of tree languages recognized by deterministic top-down tree automata and the class of regular tree languages, was obtained. Finally, a characterization of irreducible sets modulo left linear term rewriting systems by one-state deterministic top-down tree automata with prefix look-ahead was provided in [14].

A top-down tree transducer may be equipped with other type of look-ahead capacity as well. Two-way tree walking languages seem to be reasonable look-ahead sets. Kamimura and Slutzki [19, 20, 22] introduced and studied three proper subclasses of recognizable tree languages: the class of deterministic two-way tree walking languages, denoted by 2DTL, the class of nondeterministic two-way tree walking

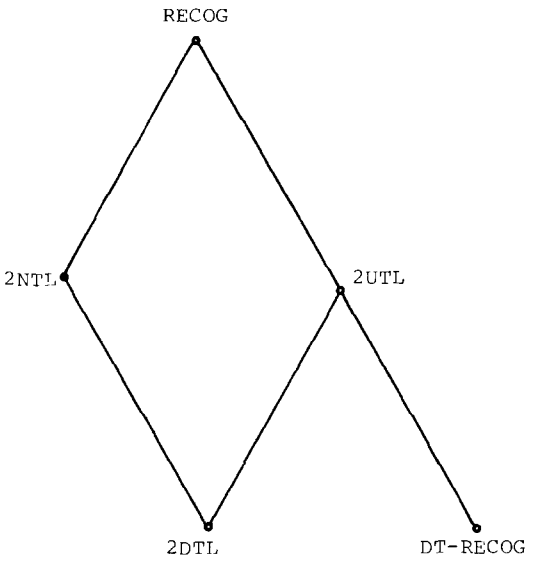

Fig. 1.

languages (2NTL), and the class of universal two-way tree walking languages (2UTL). They have compared these tree language classes, the class of recognizable tree languages and the class of tree languages recognizable by deterministic top-down tree automata and presented an inclusion diagram for them (cf. Fig. 1).

We continue the research carried out in [12, 13, 14, 17]: we equip a top-down tree transducer with deterministic, nondeterministic and universal two-way tree walking look-ahead. The corresponding tree transformation classes are denoted by T^{2DTL}-FST, T^{2NTL}-FST and T^{2UTL}-FST, respectively, and their deterministic and strongly deterministic subclasses are denoted by prefixing the denotations by D and D_s, respectively. The goal of this work is to study systematically the deterministic and strongly deterministic top-down tree transducers with some type of two-way tree walking look-ahead and in this way to establish new connections between fundamental concepts and tools in tree language theory.

In Section 2, we provide some notations and notions of the theory of tree languages which will be used in the paper.

In Section 3, we describe the recognizing capability of two-way tree walking automata from a new point of view by showing that for any modifier $Y \in \{2DTT, 2NTT, 2UTT\}$, $RECOG = dom(DT\text{-}FST \circ Id(Y))$, where RECOG denotes the class of all recognizable tree languages and DT-FST stands for the class of tree transformations induced by deterministic top-down tree transducers. This result intuitively means that the computation of a nondeterministic top-down tree automaton A can be simulated in two steps. In the first step, a deterministic top-down tree transducer computes in a parallel way all computations of A on the input tree, and in

the second step, a two-way tree walking automaton looks for an accepting computation of A. Using this result we characterize the transformational power of top-down tree transducers with two-way tree walking look-ahead. We obtain the following decomposition results: for any modifiers $X \in \{\ , D\}$ and $Y \in \{2DTL, 2NTL, 2UTL\}$, $XT^R\text{-FST} = DT\text{-FST} \circ XT^Y\text{-FST}$ and $FTA \circ DT\text{-FST} = DT\text{-FST} \circ D_sT^Y\text{-FST}$, where $(D)T^R$-FST denotes the class of (deterministic) top-down tree transformations with regular look-ahead, FTA stands for the class of tree transformations induced by top-down finite tree automata and FTA ∘ DT-FST is the class of transformational systems of Rounds (cf. [21]).

In Section 4, we present the inclusion diagram for the deterministic and strongly deterministic top-down tree transformation classes with some type of look-ahead.

Section 5 is concerned with the closure properties of top-down tree transductions with some type of two-way tree walking look-ahead under composition. We shall prove the following results. For each modifier $Y \in \{2DTL, 2NTL, 2UTL\}$, $D_sT^Y\text{-FST} \subset (D_sT^Y\text{-FST})^2 = (D_sT^Y\text{-FST})^3 = FTA \circ DT\text{-FST}$, $DT^Y\text{-FST} \subset (DT^Y\text{-FST})^2 = (DT^Y\text{-FST})^3 = DT^R\text{-FST}$ and, for every $k \geqslant 1$, $(T^Y\text{-FST})^k \subset (T^Y\text{-FST})^{k+1}$.

2. Preliminaries

We recall and invent some notations, basic definitions and terminology which will be used in the paper. Nevertheless, the reader is assumed to be familiar with the basic concepts of tree language theory (see, e.g. [1, 2, 18, 21]).

2.1. General notations

We identify the singleton set $\{a\}$ with its unique element a. The set of nonnegative integers is denoted by N. For every $k, n \in N$ with $k \leqslant n$, $[k, n]$ denotes the set $\{k, \ldots, n\}$. For a set A, $|A|$ denotes the cardinality of A. We write ω for $|N|$. We need a countably infinite set $X = \{x_1, x_2, \ldots\}$ of variable symbols, that will be kept fixed throughout the paper. The set of the first m elements $x_1, \ldots, x_m$ of X is denoted by X_m.

$A \subseteq B$ means that the set A is a subset of the set B. Proper inclusion is denoted by $\subset$. $A \nsubseteq B$ abbreviates the formula $\neg(A \subseteq B)$.

Given any finite number of sets $A_1, \ldots, A_n$, the Cartesian product $A_1 \times \cdots \times A_n$ is the set of ordered n-tuples $\{(a_1, \ldots, a_n) \mid a_i \in A_i,\ i \in [1, n]\}$.

A binary relation from A to B is any subset R of $A \times B$. Given a binary relation R between A and B, the set $\{x \in A \mid (x, y) \in R \text{ for some } y \in B\}$ is called the domain of R and denoted by $\mathrm{dom}(R)$. The composition of two relations R_1 and R_2, denoted by $R_1 \circ R_2$, is the set of pairs $\{(x, z) \mid (x, y) \in R_1 \text{ and } (y, z) \in R_2 \text{ for some } y\}$. R^n is called the n-fold composition of R with itself (R^0 is the identity relation). The reflexive, transitive closure of R is denoted by R^*. All these notions can easily be transferred to classes of relations for which we use the same notations.

By an inclusion diagram we mean the Hasse diagram of a poset, of which the order is the inclusion.

2.2. Trees

A ranked alphabet Σ is a finite set in which each symbol has a unique rank in N. The subset Σ_m of Σ is the set of symbols of rank m ($m \in \mathrm{N}$). Note that, for $i \neq j$, Σ_i and Σ_j are disjoint. The maximal rank of Σ is the largest integer n such that Σ_n is nonempty.

Given a ranked alphabet Σ and some set S, the set of all terms or trees over Σ indexed by S, denoted by $T_\Sigma(S)$, is defined inductively as follows.

(a) $S \subseteq T_\Sigma(S)$.

(b) If $l \in \mathrm{N}$, $\sigma \in \Sigma_l$ and $t_1, \ldots, t_l \in T_\Sigma(S)$, then $\sigma(t_1, \ldots, t_l) \in T_\Sigma(S)$.

Terms of the form $\sigma(\ )$ are identified with σ. If $Y = \emptyset$, then $T_\Sigma(Y)$ is written as T_Σ. $T_\Sigma(X_m)$ is written as $T_{\Sigma,m}$; thus, $T_{\Sigma,0} = T_\Sigma$. It is well known that T_Σ is in an effective one-to-one correspondence with the set of all finite, rooted, ordered and directed trees, whose nodes are labeled with the elements of Σ, in such a way that a node with l descendants is labeled by a symbol from Σ_l. Nodes labeled by elements of Σ_0 are referred to as leaves. A node is called internal if it is neither leaf nor root. The height $\mathrm{hg}(t)$ and the set of subtrees $\mathrm{sub}(t)$ of a tree $t \in T_\Sigma$ are defined as follows.

(a) If $t \in \Sigma_0 \cup S$, then $\mathrm{hg}(t) = 0$ and $\mathrm{sub}(t) = \{t\}$.

(b) If $t = \sigma(t_1, \ldots, t_l)$ ($l \in \mathrm{N}$), then $\mathrm{hg}(t) = 1 + \max(\mathrm{hg}(t_i) \mid i \in [1, l])$ and $\mathrm{sub}(t) = \bigcup(\mathrm{sub}(t_i) \mid i \in [1, l]) \cup \{t\}$.

Let $m \in \mathrm{N}$, $t \in T_{\Sigma,m}$ and $t_1, \ldots, t_m \in T_\Sigma$. We denote by $t[t_1, \ldots, t_m]$ the tree obtained from t by substituting t_i for each occurrence of x_i in t for $i \in [1, m]$. We obviously have $t[t_1, \ldots, t_m] \in T_\Sigma$.

Now let Σ and Δ be ranked alphabets. A tree language L is a subset of T_Σ. A tree transformation τ is a subset of $T_\Sigma \times T_\Delta$, i.e. a binary relation from T_Σ to T_Δ. For each $L \subseteq T_\Sigma$, $\mathrm{Id}(L)$ is the identical tree transformation $\{(t, t) \mid t \in L\}$ on L. For a class $\mathscr{L}$ of tree languages, $\mathrm{Id}(\mathscr{L}) = \{\mathrm{Id}(L) \mid L \in \mathscr{L}\}$.

2.3. Tree transducers

We adopt the definition of a top-down finite state tree transducer (t-fst) from [1]. A t-fst is a system $\mathscr{T} = \langle \Sigma, \Delta, Q, Q_{\mathrm{in}}, R \rangle$, where

(a) Σ and Δ are ranked alphabets;

(b) Q, the state set of $\mathscr{T}$, is a ranked alphabet consisting of 1-ary function symbols with $Q \cap (\Sigma \cup \Delta \cup X) = \emptyset$;

(c) Q_{in} is a subset of Q, the set of initial states;

(d) R is a finite set of rules of the form $q(\sigma(x_1, \ldots, x_m)) \to \bar{t}$, where $m \in \mathrm{N}$, $\sigma \in \Sigma_m$, $q \in Q$ and $\bar{t} \in T_\Delta(Q(X_m))$. (Here and in what follows, for a unary ranked alphabet Q and a set L of trees, $Q(L)$ denotes the set $\{q(t) \mid q \in Q \text{ and } t \in L\}$.)

$\mathscr{T}$ can be used to induce a tree transformation from T_Σ to T_Δ in the way described as follows. Define the relation $\Rightarrow_{\mathscr{T}}$ on the set $T_\Delta(Q(T_\Sigma))$ so that for any $t, r \in T_\Delta(Q(T_\Sigma))$,

$t \Rightarrow_{\mathscr{T}} r$ if and only if the next two conditions hold:

(a) there is a rule of the form $q(\sigma(x_1, \ldots, x_m)) \to \bar{t}$ in R;

(b) r can be obtained from t by substituting a subtree $q(\sigma(t_1, \ldots, t_m))$ of t by $\bar{t}[t_1, \ldots, t_m]$. It should be clear that the relation $\Rightarrow_{\mathscr{T}}$ can be interpreted as a method of rewriting trees into trees.

The tree transformation induced by the t-fst $\mathscr{T}$ is defined as

$$\tau(\mathscr{T}) = \{(t, r) \mid t \in T_\Sigma,\ r \in T_\Delta \text{ and } q(t) \Rightarrow_{\mathscr{T}}^* r \text{ for some } q \in Q_{\text{in}}\}.$$

If $q(\sigma(x_1, \ldots, x_m)) \to \bar{t}$ is a rule of $\mathscr{T}$, then it is called a (q, σ)-rule, and $q(\sigma(x_1, \ldots, x_m))$ is called the left-hand side of this rule. We write $n_{\mathscr{T}}(q, \sigma)$ to denote the number of (q, σ)-rules of $\mathscr{T}$. If there is no danger of confusion, then we simply write $n(q, \sigma)$ rather than $n_{\mathscr{T}}(q, \sigma)$.

For any $q \in Q$, we denote by $\mathscr{T}(q)$ the tree transducer $\mathscr{T}(q) = \langle \Sigma, \Delta, Q, \{q\}, R \rangle$.

Now we introduce three special types of t-fst's. Let, to this end, $\mathscr{T} = \langle \Sigma, \Delta, Q, Q_{\text{in}}, R \rangle$ be a t-fst. We say that $\mathscr{T}$ is

(a) deterministic t-fst (dt-fst) if Q_{in} is a singleton set and there are no two different rules in R with the same left-hand side.

(b) a top-down finite tree automaton (fta) if $\Sigma = \Delta$ and each rule in R is of the form $q(\sigma(x_1, \ldots, x_m)) \to \sigma(q_1(x_1), \ldots, q_m(x_m))$, where $q, q_1, \ldots, q_m \in Q$ (if $\mathscr{T}$ is an fta, then the tree transformation $\tau(\mathscr{T})$ is a partial identity on T_Σ);

(c) a deterministic top-down finite tree automaton (dfta) if it is deterministic and it is an fta.

A standard construction shows that above (from (a) to (c)) we may assume without loss of generality that $\mathscr{T}$ has only one initial state, i.e. $Q_{\text{in}} = \{q_0\}$ holds. This is used in the paper without any reference. Moreover, we write q_0 rather than $\{q_0\}$.

The tree language recognized by an fta $\mathscr{T} = \langle \Sigma, \Sigma, Q, q_0, R \rangle$ is

$$L(\mathscr{T}) = \{t \in T_\Sigma \mid q_0(t) \Rightarrow_{\mathscr{T}}^* t\}\ (= \text{dom}(\tau(\mathscr{T}))).$$

Moreover, the tree language recognized by $\mathscr{T}$ with state $q \in Q$ as initial state is

$$L(\mathscr{T}(q)) = \{t \in T_\Sigma \mid q(t) \Rightarrow_{\mathscr{T}}^* t\}.$$

Now let x be any modifier taken from {t-fst, dt-fst, fta, dfta}. A relation is called an x transformation if it can be induced by some tree transducer of type x. The class of all x transformations will be denoted by X; hence, we can speak about T-FST, DT-FST, FTA and DFTA.

We introduce the notations RECOG = dom(FTA), DT-RECOG = dom(DFTA). In most works related to this topic, RECOG is called the class of all recognizable (or regular) tree languages, whereas there are no widespread names for the class DT-RECOG to the best of our knowledge.

A bottom-up finite tree automaton (bfta, cf. [1]) is a construct $\mathscr{B} = \langle \Sigma, Q, Q_f, R \rangle$ where

(a) Σ is the input, output ranked alphabet;

(b) Q, the state set of $\mathscr{B}$, is a ranked alphabet consisting of 1-ary function symbols with $Q \cap (\Sigma \cup X) = \emptyset$;

(c) Q_f is a subset of Q, the set of final states;

(d) R is a finite set of rules of the form $\sigma(q_1(x_1), \ldots, q_l(x_l)) \to q(\sigma(x_1, \ldots, x_l))$, where $\sigma \in \Sigma_l$, $l \in \mathbf{N}$, $q_1, \ldots, q_l, q \in Q$.

The rules in R turn into tree rewriting rules, as usual, by substituting trees in T_Σ for the x's. Define the relation $\Rightarrow_{\mathscr{B}}$ on the set $T_\Sigma(Q(T_\Sigma))$ so that for any $t, r \in T_\Sigma(Q(T_\Sigma))$, $t \Rightarrow_{\mathscr{B}} r$ if and only if the next two conditions hold.

(a) There is a rule of the form $\sigma(q_1(x_1), \ldots, q_l(x_l)) \to q(\sigma(x_1, \ldots, x_l))$ in R.

(b) r can be obtained from t by substituting a subtree $\sigma(q_1(t_1), \ldots, q_l(t_l))$ of t by $q(\sigma(t_1, \ldots, t_l))$. The tree language recognized by $\mathscr{B}$ is

$$L(\mathscr{B}) = \{t \in T_\Sigma \mid t \Rightarrow_{\mathscr{B}}^* q(t) \text{ for some } q \in Q_f\}.$$

Moreover, the tree language recognized by $\mathscr{B}$ with state $q \in Q$, is

$$L(\mathscr{B}(q)) = \{t \in T_\Sigma \mid t \Rightarrow_{\mathscr{B}}^* q(t)\}.$$

Thus,

$$L(\mathscr{B}) = \bigcup (L(\mathscr{B}(q)) \mid q \in Q_f).$$

$\mathscr{B}$ is deterministic (dbfta) if there are no two different rules in R with the same left-hand side. In this case, for any $p, q \in Q$, $p \neq q$ if and only if $L(\mathscr{B}(p)) \cap L(\mathscr{B}(q)) = \emptyset$.

$\mathscr{B}$ is total (tbfta) if, for each $\sigma \in \Sigma_l$ ($l \geqslant 0$) and $q_1, \ldots, q_l \in Q$, there is at least one rule with left-hand side $\sigma(q_1(x_1), \ldots, q_l(x_l))$.

It should be clear what a tdbfta is.

It is well known from the literature [1, 18] that the recognizing capabilities of fta's dbfta's tbfta's and tdbfta's are the same. On the other hand, the recognition power of dfta's is more limited, i.e. there are recognizable tree languages that cannot be recognized by any dfta.

Proposition 2.1. *The class of tree languages recognized by tdbfta's is the same as* RECOG; *moreover,* DT-RECOG $\subset$ RECOG (*see* [1, 18]).

We shall need the following observation.

Proposition 2.2. *For any bfta* $\mathscr{B} = (\Sigma, Q, Q_f, R)$, *one can construct an fta* $\mathscr{T} = \langle \Sigma, \Sigma, Q, Q_f, R' \rangle$ *such that for each state* $q \in Q$, $L(\mathscr{B}(q)) = L(\mathscr{T}(q))$.

Proof. R' is constructed as follows. For any $\sigma \in \Sigma_l$, $l \in \mathbf{N}$ and $q, q_1, \ldots, q_l \in Q$, the rule

$$q(\sigma(x_1, \ldots, x_l)) \to \sigma(q_1(x_1), \ldots, q_l(x_l))$$

is in R' if and only if the rule

$$\sigma(q_1(x_1), \ldots, q_l(x_l)) \rightarrow q(\sigma(x_1, \ldots, x_l))$$

is in R.

In the proofs of Theorems 2.7 and 2.8 in [18, ch. IV] it was shown that, for each state $q \in Q$, $L(\mathscr{B}(q)) = L(\mathscr{T}(q))$. □

2.4. *Two-way tree walking automata*

A finite two-way tree walking automaton (2-fta, cf. [22]) $\mathscr{A}$ is a construct $\langle Q, \Sigma, \delta, q_0, F \rangle$, where

(a) Q is the finite, nonempty set of states;

(b) Σ is the input ranked alphabet;

(c) $q_0 \in Q$ is the initial state;

(d) $F \subseteq Q$ is the set of final states;

(e) δ is a mapping from $Q \times \Sigma$ to the finite subsets of $Q \times [0, m]$ with m being the maximal rank of Σ.

$\mathscr{A}$ is deterministic (2-dfta) if $\delta(q, \sigma)$ has at most one element for every $q \in Q$ and $\sigma \in \Sigma$.

An instantaneous description (ID) of $\mathscr{A}$ on a tree t in T_Σ is a triple of the form (q, n, t), where $q \in Q$ and n is a node of t or a special symbol α. The initial ID of $\mathscr{A}$ on t is (q_0, r, t) with r being the root of t. The ID $I = (q, n, t)$ is said to be accepting if $q \in F$ and $n = \alpha$.

We define a relation $\vdash_{\mathscr{A}}$ between IDs of $\mathscr{A}$ on a tree $t \in T_\Sigma$ as follows. For $n \neq \alpha$, $(q, n, t) \vdash_{\mathscr{A}} (p, m, t)$ if $(p, i) \in \delta(q, \sigma)$, where σ is the label of n and either of the following holds.

(1) $i = 0$, n is the root of t and $m = \alpha$,

(2) $i = 0$, n is not the root of t and m is the father of n,

(3) $i > 0$, m is the ith son of n.

Note if $n = \alpha$, then there is no ID J with $I \vdash_{\mathscr{A}} J$.

A sequence of IDs $P = \langle I_i \mid 0 \leqslant i < k \leqslant \omega \rangle$ with I_0 being the initial ID of $\mathscr{A}$ on t is called a computation path of $\mathscr{A}$ on t if $I_i \vdash_{\mathscr{A}} I_{i+1}$ for every $i \in \mathbf{N}$ such that $i + 1 < k$. P is accepting if $k \in \mathbf{N}$ and I_{k-1} is an accepting ID; otherwise, it is nonaccepting. A computation path is maximal if either it is infinite or there is no ID I for its last ID, I_{k-1}, such that $I_{k-1} \vdash_{\mathscr{A}} I$. Obviously, every computation path can be extended to a maximal computation path.

We now introduce two different ways to define the recognition by a 2-fta $\mathscr{A}$. The first one is the ordinary notion of acceptance by a nondeterministic device. The nondeterministic two-way tree walking language recognized by $\mathscr{A}$ is

$$L_{\mathrm{N}}(\mathscr{A}) = \{t \in T_\Sigma \mid \text{there is an accepting computation path of } \mathscr{A} \text{ on } t\}.$$

The class of all nondeterministic two-way tree walking languages is denoted by 2NTL.

In the second type of recognition, we require that every computation path must be accepting. The universal two-way tree walking language recognized by $\mathscr{A}$ is

$$L_U(\mathscr{A}) = \{t \in T_\Sigma \mid \text{every computation path of } \mathscr{A} \text{ on } t \text{ is accepting}\}.$$

The class of all universal two-way tree walking languages is denoted by 2UTL.

If $\mathscr{A}$ is a 2-dfta, then $L_U(\mathscr{A}) = L_N(\mathscr{A})$, which is called the deterministic two-way tree walking language recognized by $\mathscr{A}$ and is denoted simply by $L(\mathscr{A})$.

2DTL stands for the class of deterministic two-way tree walking languages.

We recall the inclusion diagram of Fig. 1 from [22].

2.5. *Top-down tree transducers with look-ahead*

First we recall the definition of the regular look-ahead from [2]. A t-fst with regular look-ahead (t^r-fst) is a system $\mathscr{T} = \langle \Sigma, \Delta, Q, Q_{in}, R \rangle$, where Σ, Δ, Q and Q_{in} are the same as for a t-fst, while R is now a finite set of rules of the form $\langle q(\sigma(x_1, \ldots, x_m)) \to \bar{t}; L_1, \ldots, L_m \rangle$ such that $q(\sigma(x_1, \ldots, x_m)) \to \bar{t}$ is an ordinary t-fst rule and L_i is a recognizable subset of T_Σ, for every $i \in [1, m]$.

The relation $\Rightarrow_{\mathscr{T}}$ on the set $T_\Delta(Q(T_\Sigma))$ is now defined in the following way: for $t, r \in T_\Delta(Q(T_\Sigma))$, $t \Rightarrow_{\mathscr{T}} r$ if and only if

(a) there is a rule $\langle q(\sigma(x_1, \ldots, x_m)) \to \bar{t}; L_1, \ldots, L_m \rangle$ in R, and

(b) r can be obtained from t by substituting a subtree $q(\sigma(t_1, \ldots, t_m))$ of t by $\bar{t}[t_1, \ldots, t_m]$ such that $t_i \in L_i$ for each $i \in [1, m]$.

The tree transformation induced by $\mathscr{T}$ is

$$\tau(\mathscr{T}) = \{(t, r) \mid t \in T_\Sigma, r \in T_\Delta \text{ and } q(t) \Rightarrow_{\mathscr{T}}^* r \text{ for some } q \in Q_{in}\}.$$

We define the following versions of a t^r-fst. We say that a t^r-fst $\mathscr{T} = \langle \Sigma, \Delta, Q, Q_{in}, R \rangle$ is

(a) strongly deterministic (d$_s$t^r-fst) if Q_{in} is a singleton set and there are no two different rules in R with the same left-hand side;

(b) deterministic (dt^r-fst) if Q_{in} is a singleton set and $L_i \cap L_i'$ holds for some $i \in [1, m]$ whenever $\langle q(\sigma(x_1, \ldots, x_m)) \to t; L_1, \ldots, L_m \rangle$ and $\langle q(\sigma(x_1, \ldots, x_m)) \to \bar{t}; L_1', \ldots, L_m' \rangle$ are different rules in R;

(c) a t-fst with deterministic top-down look-ahead (t^{dtr}-fst) if $L_i \in$ DT-RECOG for each rule $\langle q(\sigma(x_1, \ldots, x_m)) \to t; L_1, \ldots, L_m \rangle$ and $i \in [1, m]$;

(d) a t-fst with deterministic two-way tree walking look-ahead (t^{2dtl}-fst) if $L_i \in$ 2DTL for each rule $\langle q(\sigma(x_1, \ldots, x_m)) \to t; L_1, \ldots, L_m \rangle$ and $i \in [1, m]$.

The meaning of the abbreviations t^{2ntl}-fst and t^{2utl}-fst should now be clear.

For any modifiers $x \in$ {deterministic, strongly deterministic} and $y \in$ {deterministic top-down, deterministic two-way tree walking, nondeterministic two-way tree walking, universal two-way tree walking}, an xt-fst with y look-ahead is defined and denoted in the natural way. For example, a strongly deterministic t-fst with deterministic two-way tree walking look-ahead is denoted by d$_s$t^{2dtl}-fst.

As in the definition of a top-down tree transducer, a standard construction shows that, for a t^r-fst $\mathscr{T}=\langle \Sigma, \Delta, Q, Q_{in}, R\rangle$ of any type defined above, we may assume without loss of generality that $\mathscr{T}$ has only one initial state, i.e. $Q_{in}=\{q_0\}$ holds. This is used hereafter without any reference. Moreover, we write q_0 rather than $\{q_0\}$.

Let z be any type of tree transducers that is defined above. The class of all tree transformations that can be induced by tree transducers of type z will be denoted by the capital-letter correspondent Z. Thus, we have the classes T^R-FST, DT^R-FST, D_sT^R-FST, T^{DTR}-FST, DT^{DTR}-FST, D_sT^{DTR}-FST, T^{2DTL}-FST, DT^{2DTL}-FST, D_sT^{2DTL}-FST, T^{2NTL}-FST, DT^{2NTL}-FST, D_sT^{2NTL}-FST, T^{2UTL}-FST, DT^{2UTL}-FST and D_sT^{2UTL}-FST as classes of tree transformations.

We recall the inclusion diagram of Fig. 2 from [12].

Note that FTA ∘ DT-FST is the class of Rounds' transformational systems (cf. [21]).

Engelfriet and Vogler [7] stated the following result, but the proof was left to the reader (see the remark after Definition 4.18 in [7]).

Proposition 2.3. *Let $\mathscr{T}=\langle \Sigma, \Delta, Q, q_0, R\rangle$ be a dt^r-fst. Then there are tdbfta $\mathscr{A}=\langle \Sigma, S, S_f, \tilde{R}\rangle$ and dt^r-fst $\mathscr{T}'=\langle \Sigma, \Delta, Q, q_0, R'\rangle$ such that $\tau(\mathscr{T})=\tau(\mathscr{T}')$ and that, for each look-ahead set L that appears in some rule of $\mathscr{T}'$, there is a state $s\in S$ with $L=L(\mathscr{A}(s))$.*

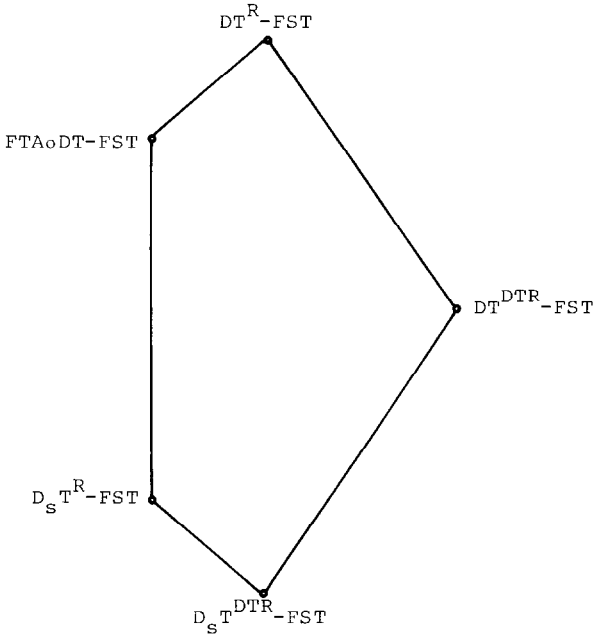

Fig. 2.

Proof. We only outline the construction of $\mathscr{A}$ and $\mathscr{T}'$. Let $L_1, \ldots, L_n$ be all the tree languages that appear as look-ahead sets in some rule of $\mathscr{T}$. Assume that, for each $j \in [1, n]$, the tdbfta $\mathscr{A}_j = (S^{(j)}, \Sigma, S_f^{(j)}, R_j)$ recognizes L_j. Consider the tdbfta $\mathscr{A} = (\Sigma, S, S_f, \tilde{R})$, where

(i) $S = \{\langle s_1, \ldots, s_n \rangle \mid s_1 \in S^{(1)}, \ldots, s_n \in S^{(n)}\}$,

(ii) $S_f = \emptyset$,

(iii) $\tilde{R}$ is defined as follows. For every $\sigma \in \Sigma_l$, $l \in \mathbf{N}$ and $s_1, s_{11}, \ldots, s_{l1} \in S^{(1)}$, $\ldots$, $s_n, s_{1n}, \ldots, s_{ln} \in S^{(n)}$, the rule

$$\sigma(\langle s_{11}, \ldots, s_{1n} \rangle(x_1), \ldots, \langle s_{l1}, \ldots, s_{ln} \rangle(x_l)) \to \langle s_1, \ldots, s_n \rangle(\sigma(x_1, \ldots, x_l))$$

is in $\tilde{R}$ if and only if the rule

$$\sigma(s_{1j}(x_1), \ldots, s_{lj}(x_l)) \to s_j(\sigma(x_1, \ldots, x_l))$$

is in R_j for each $j \in [1, n]$.

The rules of the dt^r-fst $\mathscr{T}' = \langle \Sigma, \Delta, Q, q_0, R' \rangle$ are defined in the following manner. The rule

$$\langle q(\sigma(x_1, \ldots, x_l)) \to t;\ L(\mathscr{A}(\langle s_{11}, \ldots, s_{1n} \rangle)), \ldots, L(\mathscr{A}(\langle s_{l1}, \ldots, s_{ln} \rangle)) \rangle$$

is in R', where $s_{11}, \ldots, s_{l1} \in S^{(1)}, \ldots, s_{1n}, \ldots, s_{ln} \in S^{(n)}$ if and only if the rule

$$\langle q(\sigma(x_1, \ldots, x_l)) \to t;\ L_{j_1}, \ldots, L_{j_l} \rangle$$

is in R with $j_1, \ldots, j_l \in [1, n]$ and

$$s_{1j_1} \in S_f^{(j_1)}, \ldots, s_{lj_l} \in S_f^{(j_l)}.$$

Now it can be easily seen that the tdbfta $\mathscr{A}$ and the dt^r-fst $\mathscr{T}'$ satisfy the conditions of the proposition. □

From Propositions 2.2 and 2.3 we immediately obtain the following result.

Proposition 2.4. *Let $\mathscr{T} = \langle \Sigma, \Delta, Q, q_0, R \rangle$ be a dt^r-fst. Then there are fta $\mathscr{A} = (\Sigma, \Sigma, S, s_0, R\rangle$ and dt^r-fst $\mathscr{T}' = \langle \Sigma, \Delta, Q, q_0, R' \rangle$ with $\tau(\mathscr{T}) = \tau(\mathscr{T}')$ such that, for each look-ahead set L that appears in some rule of $\mathscr{T}'$, there is a state $s \in S$ with $L = L(\mathscr{A}(s))$; moreover, for any different states $s, s' \in S$, $L(\mathscr{A}(s)) \cap L(\mathscr{A}(s')) = \emptyset$.*

3. Top-down tree transducers with two-way tree walking look-ahead

In this section, we show how to simulate a finite top-down tree automaton by the composition of a deterministic top-down tree transducer and a deterministic two-way tree walking automaton. Using this result we decompose a (deterministic) top-down tree transformation with regular look-ahead into a deterministic top-down tree

transformation and a (deterministic) top-down tree transformation with deterministic two-way tree walking look-ahead. Moreover, we show that a transformational system of Rounds appears as a composition of a deterministic top-down tree transformation and a strongly deterministic top-down tree transformation with deterministic two-way tree walking look-ahead.

Convention 3.1. We assume that, for any fta $\mathcal{T}=\langle \Sigma, \Sigma, Q, q_0, R\rangle$, there is given an order on the set R and that whenever we list rewriting rules in R having some common property we do that with respect to this order.

Definition 3.2. Let $\mathcal{T}=\langle \Sigma, \Sigma, Q, q_0, R\rangle$ be an fta. Consider the ranked alphabet

$$\Gamma=\{*\}\cup\{\sigma^{(d)} \mid \sigma\in\Sigma_l,\ d=l\cdot n(q,\sigma) \text{ for some } q\in Q\}.$$

The rank of $*$ is 0 and the rank of $\sigma^{(d)}$ is d.

The parallel computation tree $\mathrm{PC}(\mathcal{T}, q, t)\in T_\Gamma$ of $\mathcal{T}$ on t starting from the state q is defined in the following way. Assume that $t=\sigma(t_1,\ldots,t_l)$ and let $k=n(q,\sigma)$.

(i) If $k=0$, then $\mathrm{PC}(\mathcal{T}, q, t)=*$.

(ii) If $k>0$ and the (q,σ)-rules are

$$q(\sigma(x_1,\ldots,x_l))\to\sigma(q_{11}(x_1),\ldots,q_{1l}(x_l)),$$
$$\vdots$$
$$q(\sigma(x_1,\ldots,x_l))\to\sigma(q_{k1}(x_1),\ldots,q_{kl}(x_l)),$$

then $\mathrm{PC}(\mathcal{T}, q, t)=\sigma^{(d)}(\mathrm{PC}(\mathcal{T}, q_{11}, t_1),\ldots,\mathrm{PC}(\mathcal{T}, q_{1l}, t_l),\ldots,\mathrm{PC}(\mathcal{T}, q_{k1}, t_1),\ldots,\mathrm{PC}(\mathcal{T}, q_{kl}, t_l))$, where $d=k\cdot l$.

Note that, according to Convention 3.1, the definition of $\mathrm{PC}(\mathcal{T}\ q, t)$ is correct. The parallel computation tree $\mathrm{PC}(\mathcal{T}, t)$ of $\mathcal{T}$ on t is defined to be $\mathrm{PC}(\mathcal{T}, q_0, t)$. Intuitively, $\mathrm{PC}(\mathcal{T}, q, t)$ contains all information about all computations of $\mathcal{T}$ on t starting from the state q. Now we present two examples of parallel computation trees because the definition seems to be involved.

Example 3.3. Consider the fta $\mathcal{T}=\langle \Sigma, \Sigma, Q, \mathrm{even}, R\rangle$, where

(i) $\Sigma=\Sigma_0\cup\Sigma_2$, $\Sigma_0=\{a\}$ and $\Sigma_2=\{\sigma\}$,

(ii) $Q=\{\mathrm{even}, \mathrm{odd}\}$,

(iii) R consists of the following rules, listed with respect to the order on R.

$$\mathrm{even}(\sigma(x_1,x_2))\to\sigma(\mathrm{even}(x_1),\mathrm{even}(x_2)),$$
$$\mathrm{even}(\sigma(x_1,x_2))\to\sigma(\mathrm{odd}(x_1),\mathrm{odd}(x_2)),$$
$$\mathrm{odd}(\sigma(x_1,x_2))\to\sigma(\mathrm{even}(x_1),\mathrm{odd}(x_2)),$$
$$\mathrm{odd}(\sigma(x_1,x_2))\to\sigma(\mathrm{odd}(x_1),\mathrm{even}(x_2)),$$
$$\mathrm{odd}(a)\to a.$$

It should be clear that $L(\mathscr{T})$ consists of all trees that contain an even number of a's. The parallel computation tree of $\mathscr{T}$ on $\sigma(a, a)$ is $\mathrm{PC}(\mathscr{T}, \sigma(a, a)) = \sigma^{(4)}(*, *, a, a)$ and the parallel computation tree of $\mathscr{T}$ on $\sigma(a, \sigma(a, a))$ is

$$\mathrm{PC}(\mathscr{T}, \sigma(a, \sigma(a, a))) = \sigma^{(4)}(*, \sigma^{(4)}(*, *, a, a), a, \sigma^{(4)}(*, a, a, *)).$$

Definition 3.4. We introduce the special deterministic top-down tree transformation ind_Σ for each ranked alphabet Σ (cf. [19]). It changes a label $\sigma \in \Sigma$ of a node of an input tree to $\langle \sigma, i \rangle$ (i.e. attaches the subscript i to σ), where $i \in [0, m]$ and m is the maximal rank of Σ, if that node is the ith son (from left to right) of its father, in particular, $i = 0$ if the node is the root of the tree. The tree $\mathrm{ind}_\Sigma(t)$ is called the indexed copy of t. We shall omit the subscript Σ from $\mathrm{ind}_\Sigma(t)$ and simply write $\mathrm{ind}(t)$.

It should be clear that $\mathrm{ind}(t)$ is induced by the dt-fst $\mathscr{T} = \langle \Sigma, \Delta, Q, 0, R \rangle$, where

(1) $\Delta = \{\langle \sigma, i \rangle \mid \sigma \in \Sigma, i \in [0, m]\}$ with m being the maximal rank of Σ;

(2) $Q = [0, m]$;

(3) R consists of the rules $i(\sigma(x_1, \ldots, x_l)) \to \langle \sigma, i \rangle (1(x_1), \ldots, l(x_l))$, where $i \in Q$ and $\sigma \in \Sigma_l$.

The indexed copy $\mathrm{ind}(\mathrm{PC}(\mathscr{T}, q, t))$ of the parallel computation tree $\mathrm{PC}(\mathscr{T}, q, t)$ is called indexed parallel computation tree, is denoted by $\mathrm{IPC}(\mathscr{T}, q, t)$, and can be computed by a deterministic top-down tree transducer; moreover, $\mathrm{ind}(\mathrm{PC}(\mathscr{T}, t))$ is denoted by $\mathrm{IPC}(\mathscr{T}, t)$.

Proposition 3.5. *Let $\mathscr{T} = \langle \Sigma, \Sigma, Q, q_0, R \rangle$ be an arbitrary fta. Then there is a dt-fst $\mathscr{T}_1$ such that $\tau(\mathscr{T}_1) = \{(t, \mathrm{IPC}(\mathscr{T}, t)) \mid t \in T_\Sigma\}$.*

Proof. Set $m = \max(l \cdot n(q, \sigma) \mid q \in Q, \ \sigma \in \Sigma_l, l \in \mathbf{N})$. The dt-fst $\mathscr{T}_1 = \langle \Sigma, \Delta, Q^{(1)}, (q_0, 0), R_1 \rangle$ is constructed as follows.

(1) Δ is determined by (a) and (b).

(a) $\langle \sigma^{(d)}, i \rangle \in \Delta_d$ whenever $\sigma \in \Sigma_l$, $l \in \mathbf{N}$, $i \in [0, m]$ and $d = l \cdot n(q, \sigma)$ for some $q \in Q$.

(b) $\langle *, i \rangle \in \Delta_0$ whenever $i \in [0, m]$.

Note that the maximal rank of Δ is m.

(2) $Q^{(1)} = \{\langle q, i \rangle \mid q \in Q, i \in [1, m]\} \cup \{\langle q_0, 0 \rangle\}$.

(3) R_1 is defined as follows. For every $q \in Q$, $\sigma \in \Sigma_l$, $l \in \mathbf{N}$, $i \in [0, m]$ and for $k = n(q, \sigma)$, $d = k \cdot l$,

(i) if $k = 0$, then the rule

$$\langle q, i \rangle (\sigma(x_1, \ldots, x_l)) \to \langle *, i \rangle$$

is in R_1, and

(ii) if $k > 0$ and the (q, σ)-rules are

$$q(\sigma(x_1, \ldots, x_l)) \to \sigma(q_{11}(x_1), \ldots, q_{1l}(x_l)),$$
$$\vdots$$
$$q(\sigma(x_1, \ldots, x_l)) \to \sigma(q_{k1}(x_1), \ldots, q_{kl}(x_l)),$$

then the rule

$$\langle q, i\rangle(\sigma(x_1, \ldots, x_l)) \to \langle \sigma^{(d)}, i\rangle(\langle q_{11}, 1\rangle(x_1), \ldots,$$

$$\langle q_{1l}, l\rangle(x_1), \ldots, \langle q_{k1}, (k-1)\cdot l+1\rangle(x_1), \ldots, \langle q_{kl}, k\cdot l\rangle(x_l))$$

is in R_1.

It is obvious that $\tau(\mathscr{T}_1)=\{(t, \mathrm{IPC}(\mathscr{T}, t)) \mid t\in T_\Sigma\}$. □

In the proof of the next theorem we show how to simulate the computation of a top-down finite tree automaton by the composition of a deterministic top-down tree transducer and the identical tree transformation on a two-way tree walking language.

Theorem 3.6. *For each modifier* $X\in\{\mathrm{2DTT}, \mathrm{2NTT}, \mathrm{2UTT}\}$, $\mathrm{RECOG}=\mathrm{dom}(\mathrm{DT\text{-}FST}\circ \mathrm{Id}(X))$.

Proof. First we prove that $\mathrm{dom}(\mathrm{DT\text{-}FST}\circ \mathrm{Id}(X))\subseteq \mathrm{RECOG}$. By the inclusion $X\subseteq \mathrm{FTA}$ (see Fig. 1), we obtain that

$$\mathrm{dom}(\mathrm{DT\text{-}FST}\circ \mathrm{Id}(X))\subseteq \mathrm{dom}(\mathrm{DT\text{-}FST}\circ \mathrm{FTA})\subseteq \mathrm{dom}(\mathrm{T\text{-}FST})^2=\mathrm{RECOG}$$

(see [1, 2]).

Now we prove that $\mathrm{RECOG}\subseteq \mathrm{dom}(\mathrm{DT\text{-}FST}\circ \mathrm{Id}(X))$ holds as well. Let $\mathscr{T}$, $\mathscr{T}_1$ and m be the same as in the proof of Proposition 3.5. Thus, we have $\tau(\mathscr{T}_1)=\{(t, \mathrm{IPC}(\mathscr{T}, t)) \mid t\in T_\Sigma\}$. We define a 2-dfta $\mathscr{T}_2$ such that $L(\mathscr{T})=\mathrm{dom}(\tau(\mathscr{T}_1)\circ \mathrm{Id}(L(\mathscr{T}_2)))$.

Let $t\in T_\Sigma$ be an arbitrary tree. Intuitively, a subtree $\bar{p}$ of $\mathrm{IPC}(\mathscr{T}, t)$, can be obtained by indexing some subtree p of $\mathrm{PC}(\mathscr{T}, t)$, where $p=\mathrm{PC}(\mathscr{T}, q, r)$ for some subtree r of t and state $q\in Q$. $\mathscr{T}_2$ begins traversing $\bar{p}$ in state start and "falls off" the subtree $\bar{p}$ (i.e. moves up from its root) in state $\langle \mathrm{yes}, j\rangle$ (where $j>0$ and the root of $\bar{p}$ is the jth son of its father or $j=0$ and $\bar{p}=\mathrm{IPC}(\mathscr{T}, t)$) if and only if $\bar{p}$ contains a successful computation $q(r)\Rightarrow^*_{\mathscr{T}} r$. Otherwise, $\mathscr{T}_2$ "falls off" $\bar{p}$ in state $\langle \mathrm{no}, j\rangle$.

The formal construction of $\mathscr{T}_2=\langle \Delta, Q^{(2)}, \delta, \mathrm{start}, \{\langle \mathrm{yes}, 0\rangle\}\rangle$ is as follows:

(1) The ranked alphabet Δ and the nonnegative integer m are the same as in the proof of Proposition 3.5.

(2) $Q^{(2)}=\{\mathrm{start}\}\cup\{\langle \mathrm{yes}, i\rangle, \langle \mathrm{no}, i\rangle \mid i\in[0, m]\}$.

(3) The mapping δ is defined by the following conditions:

(i) $\delta(\mathrm{start}, \langle \sigma^{(d)}, j\rangle)=(\mathrm{start}, 1)$ for any $\langle \sigma^{(d)}, j\rangle\in\Delta_d$ such that $d>0$.

($\mathscr{T}_2$ begins traversing the subtree at the current node by visiting its first son.)

(ii) $\delta(\mathrm{start}, \langle \sigma^{(0)}, j\rangle)=(\langle \mathrm{yes}, j\rangle, 0)$ for any $\langle \sigma^{(0)}, j\rangle\in\Delta_0$.

($\mathscr{T}_2$ reaches a leaf that is different from $\langle *, i\rangle$, where $i\in[0, m]$ and, thus, finds a piece of some successful computation of $\mathscr{T}$. Then, in order to complete it, $\mathscr{T}_2$ goes back to the father of the leaf. In particular, $\mathscr{T}_2$ "falls off" the input tree $\mathrm{IPC}(\mathscr{T}, t)$ in state $\langle \mathrm{yes}, 0\rangle$ if $j=0$.)

(iii) $\delta(\text{start}, \langle *, j\rangle) = (\langle \text{no}, j\rangle, 0)$ for any $\langle *, j\rangle \in \Delta_0$.
($\mathscr{T}_2$ arrives at a leaf which indicates the dead end of some derivation of $\mathscr{T}$. Then $\mathscr{T}_2$ goes back to the father of this leaf indicating the failure of $\mathscr{T}$. In particular, $\mathscr{T}_2$ "falls off" the input tree $\text{IPC}(\mathscr{T}, t)$ in state $\langle \text{no}, 0\rangle$ if $j=0$.)

(iv) $\delta(\langle \text{yes}, i\rangle, \langle \sigma^{(d)}, j\rangle) = (\text{start}, i+1)$ whenever $\langle \sigma^{(d)}, j\rangle \in \Delta_d$, $\sigma \in \Sigma_l$, $d = l \cdot k$, $k \in \mathbb{N}$ and $(u-1)\cdot l < i < u \cdot l$ for some $u \in [1, k]$.
($\mathscr{T}_2$ has found a piece of successful derivation of $\mathscr{T}$ in the subtree at the ith son of the current node. Now, to complete it, $\mathscr{T}_2$ goes to the $i+1$th son.)

(v) $\delta(\langle \text{yes}, i\rangle, \langle \sigma^{(d)}, j\rangle) = (\langle \text{yes}, j\rangle, 0)$ whenever $\langle \sigma^{(d)}, j\rangle \in \Delta_d$, $\sigma \in \Sigma_l$, $d = l \cdot k$, $k \in \mathbb{N}$ and $i = u \cdot l$ for some $u \in [1, k]$.
($\mathscr{T}_2$ has found a piece of successful computation of $\mathscr{T}$ in the subtree at the current node. Now, to complete it, $\mathscr{T}_2$ goes back to the father of the current node.)

(vi) $\delta(\langle \text{no}, i\rangle, \langle \sigma^{(d)}, j\rangle) = (\text{start}, u \cdot l + 1)$, where $\langle \sigma^{(d)}, j\rangle \in \Delta_d$, $d > 0$, $\delta \in \Sigma_l$, $d = k \cdot l$, $k \in \mathbb{N}$ and $i \in [(u-1)\cdot l + 1, u \cdot l]$ for some $1 \leqslant u < k$.
(The subtree at the current node was obtained by indexing some subtree p of $\text{PC}(\mathscr{T}, t)$, which is equal to $\text{PC}(\mathscr{T}, q, r)$ for some subtree r of t and state $q \in Q$. $\mathscr{T}_2$ has found no piece of successful derivation of $\mathscr{T}$ in the subtree at the ith son. This means that the application of the uth (q, σ)-rule at the root of r results in $\mathscr{T}$ failing on r. Now $\mathscr{T}_2$ checks if the application of the next (q, σ)-rule (with respect to the order on R) at the root of r leads to a derivation $q(r) \Rightarrow_{\mathscr{T}}^* r$. This is being checked in the subtrees at the $u \cdot l + 1$th, ..., $(u+1)\cdot l$th sons of the current node.)

(vii) $\delta(\langle \text{no}, i\rangle, \langle \sigma^{(d)}, j\rangle) = (\langle \text{no}, j\rangle, 0)$, where $\langle \sigma^{(d)}, j\rangle \in \Delta_d$, $d > 0$, $\sigma \in \Sigma_l$, $d = k \cdot l$, $k \in \mathbb{N}$ and $i \in [(k-1)\cdot l + 1, k \cdot l]$.
($\mathscr{T}_2$ has traversed the subtree at the current node and realized that it contains only failing computations of $\mathscr{T}$. If $j > 0$, then $\mathscr{T}_2$ goes back to the father of the current node in state $\langle \text{no}, j\rangle$. If $j = 0$, then the current node is the root of the input tree $\text{IPC}(\mathscr{T}, t)$ and $\mathscr{T}_2$ "falls off" $\text{IPC}(\mathscr{T}, t)$ in state $\langle \text{no}, 0\rangle$.)

(viii) $\delta(\langle \text{yes}, i\rangle, \langle *, j\rangle) = \emptyset$ for any $\langle \text{yes}, i\rangle \in Q^{(2)}$ and $\langle *, j\rangle \in \Delta_0$.

(ix) $\delta(\langle \text{no}, i\rangle, \langle *, j\rangle) = \emptyset$ for every $\langle \text{no}, i\rangle \in Q^{(2)}$ and $\langle *, j\rangle \in \Delta_0$.

It is sufficient to prove that $\text{dom}(\tau(\mathscr{T})) = \text{dom}(\tau(\mathscr{T}_1) \circ \text{Id}(L(\mathscr{A}))$. To this end we prove that, for all $q \in Q$ and $t \in T_\Sigma$, the equivalence

$$(*) \qquad q(t) \Rightarrow_{\mathscr{T}}^* t \text{ if and only if } (\text{start}, n, \text{IPC}(\mathscr{T}, q, t)) \vdash_{\mathscr{T}_2}^* (\langle \text{yes}, 0\rangle, \alpha, \text{IPC}(\mathscr{T}, q, t))$$

holds with n being the root of $\text{IPC}(\mathscr{T}, q, t)$.

We proceed by induction on the height of t.

If $\text{hg}(t) = 0$, then $(*)$ obviously holds.

Assume that $t = \sigma(t_1, \ldots, t_l)$ with $\sigma \in \Sigma_l$ and $l > 0$ and that $(*)$ has been proved for all trees from T_Σ of height less than $\text{hg}(t)$. In the rest of the proof we denote by n_i the ith son of the node n.

First we show that the left-hand side of $(*)$ implies its right-hand side. For this assume that we have

$$q(\sigma(t_1, \ldots, t_l)) \Rightarrow_{\mathscr{T}} \sigma(q_1(t_1), \ldots, q_l(t_l)) \Rightarrow_{\mathscr{T}}^* \sigma(t_1, \ldots, t_l).$$

Then by the induction hypothesis, for each $i\in[1, l]$,

$$(\text{start}, \bar{n}_i, \text{IPC}(\mathscr{T}, q_i, t_i)) \vdash^*_{\mathscr{T}_2} (\langle \text{yes}, 0\rangle, \alpha, \text{IPC}(\mathscr{T}, q_i, t_i))$$

with $\bar{n}_i$ being the root of $\text{IPC}(\mathscr{T}, q_i, t_i)$.

Assume that

$$\begin{aligned} &q(\sigma(x_1, \ldots, x_l)) \to \sigma(q_{11}(x_1), \ldots, q_{1l}(x_l)),\\ &\vdots\\ &q(\sigma(x_1, \ldots, x_l)) \to \sigma(q_{k1}(x_1), \ldots, q_{kl}(x_l)) \end{aligned}$$

are all the (q, σ)-rules in R. Then for $d = k \cdot l$,

$$\begin{aligned} \text{PC}(\mathscr{T}, q, \sigma(t_1, \ldots, t_l)) = \sigma^{(d)}(&\text{PC}(\mathscr{T}, q_{11}, t_1), \ldots, \text{PC}(\mathscr{T}, q_{1l}, t_l), \ldots,\\ &\text{PC}(\mathscr{T}, q_{k1}, t_1), \ldots, \text{PC}(\mathscr{T}, q_{kl}, t_l)). \end{aligned}$$

Without loss of generality we may assume that $q_1 = q_{i1}, \ldots, q_l = q_{il}$ for some $i\in[1, k]$; moreover, that, for $j\in[1, i-1]$, it does not hold that $\sigma(q_{j1}(t_1), \ldots, q_{jl}(t_l)) \Rightarrow^*_{\mathscr{T}} \sigma(t_1, \ldots, t_l)$, i.e. $q(\sigma(x_1, \ldots, x_l)) \to \sigma(q_{i1}(x_1), \ldots, q_{il}(x_l))$ is the least rule, with respect to the order on R, of which the application at the root of t results in a successful computation of $\mathscr{T}$ on t.

Now we conclude with the accepting computation path of $\mathscr{T}_2$ on $\text{IPC}(\mathscr{T}, q, t)$. For some integers $j_1\in[1, l]$, $j_2\in[l+1, 2\cdot l], \ldots, j_{i-1}\in[(i-2)\cdot l+1, (i-1)\cdot l]$,

$$\begin{aligned} (\text{start}, n, \text{IPC}(\mathscr{T}, q, t)) &\vdash^*_{\mathscr{T}_2} (\text{start}, n_{j_1}, \text{IPC}(\mathscr{T}, q, t))\\ &\vdash^*_{\mathscr{T}_2} (\langle \text{no}, j_1\rangle, n, \text{IPC}(\mathscr{T}, q, t))\\ &\vdash^*_{\mathscr{T}_2} (\text{start}, n_{j_2}, \text{IPC}(\mathscr{T}, q, t))\\ &\vdash^*_{\mathscr{T}_2} (\langle \text{no}, j_2\rangle, n, \text{IPC}(\mathscr{T}, q, t))\\ &\vdash^*_{\mathscr{T}_2} (\text{start}, n_{j_{i-1}}, \text{IPC}(\mathscr{T}, q, t))\\ &\vdash^*_{\mathscr{T}_2} (\langle \text{no}, j_{i-1}\rangle, n, \text{IPC}(\mathscr{T}, q, t))\\ &\vdash^*_{\mathscr{T}_2} (\text{start}, n_{(i-1)\cdot l+1}, \text{IPC}(\mathscr{T}, q, t))\\ &\vdash^*_{\mathscr{T}_2} (\langle \text{yes}, (i-1)\cdot l+1\rangle, n, \text{IPC}(\mathscr{T}, q, t))\\ &\vdash^*_{\mathscr{T}_2} (\text{start}, n_{(i-1)\cdot l+2}, \text{IPC}(\mathscr{T}, q, t))\\ &\vdash^*_{\mathscr{T}_2} (\langle \text{yes}, (i-1)\cdot l+2\rangle, \text{IPC}(\mathscr{T}, q, t))\\ &\vdash^*_{\mathscr{T}_2} (\text{start}, n_{i\cdot l}, \text{IPC}(\mathscr{T}, q, t))\\ &\vdash^*_{\mathscr{T}_2} (\langle \text{yes}, i\cdot l\rangle, n, \text{IPC}(\mathscr{T}, q, t))\\ &\vdash_{\mathscr{T}_2} (\langle \text{yes}, 0\rangle, \alpha, \text{IPC}(\mathscr{T}, q, t)). \end{aligned}$$

We now prove that the right-hand side of (∗) implies its left-hand side. To this end, consider the above accepting computation path of $\mathscr{T}_2$. We denote the ith (q, σ) rule of $\mathscr{T}$ with respect to the order on R by $q(\sigma(x_1, \ldots, x_l)) \to \sigma(q_1(x_1), \ldots, q_l(x_l))$. According to the induction hypothesis, the derivations

$$q_1(t_1) \Rightarrow_{\mathscr{T}}^* t_1,$$

$$q_2(t_2) \Rightarrow_{\mathscr{T}}^* t_2,$$

$$\vdots$$

$$q_l(t_l) \Rightarrow_{\mathscr{T}}^* t_l$$

hold. Thus, we obtain that

$$q(\sigma(t_1, \ldots, t_l)) \Rightarrow_{\mathscr{T}}^* \sigma(q_1(t_1), \ldots, q_l(t_l)) \Rightarrow_{\mathscr{T}}^* \sigma(t_1, \ldots, t_l). \qquad \square$$

Later we shall need the 2-dfta $\mathscr{T}_3 = \langle \Delta, Q, \delta, \text{start}, \{\langle \text{no}, 0\rangle\}\rangle$ that we define from $\mathscr{T}_2$ in the proof of Theorem 3.6 by changing the set of final states into $\{\langle \text{no}, 0\rangle\}$. It should be clear that $L(\mathscr{T}_2) \cap L(\mathscr{T}_3) = \emptyset$.

In the next theorem we decompose any top-down tree transduction with regular look-ahead into a deterministic top-down tree transformation and a top-down tree transformation with two-way tree walking look-ahead.

Theorem 3.7. *For any values of the modifiers* $X \in \{\ , \mathrm{D_s}, \mathrm{D}\}$ *and* $Y \in \{2\mathrm{DTL}, 2\mathrm{NTL}, 2\mathrm{UTL}\}$, $X\mathrm{T}^{\mathrm{R}}\text{-FST} \subseteq \text{DT-FST} \circ X\mathrm{T}^{\mathrm{Y}}\text{-FST}$.

Proof. First we show that $\mathrm{T}^{\mathrm{R}}\text{-FST} \subseteq \text{DT-FST} \circ \mathrm{T}^{2\mathrm{DTL}}\text{-FST}$. Let $\mathscr{A} = \langle \Sigma, \Gamma, Q, q_0, R\rangle$ be a t^{r}-fst. Let $L_1, \ldots, L_n$ be all the tree languages that appear as look-ahead sets in some rule of $\mathscr{A}$. Let the tree language L_i be recognized by the fta $\mathscr{T}_i = \langle \Sigma, \Sigma, S^{(i)}, s_i, R_i\rangle$ for each $i \in [1, n]$. Now we define a dt-fst $\mathscr{A}_1$ and a $\mathrm{t}^{2\mathrm{dtl}}$-fst $\mathscr{A}_2$ such that $\tau(\mathscr{A}) = \tau(\mathscr{A}_1) \circ \tau(\mathscr{A}_2)$. For each subtree p of the input tree with $p = \sigma(p_1, \ldots, p_l)$ and $l > 0$, $\mathscr{A}_1$ computes the indexed parallel computation trees $\mathrm{IPC}(\mathscr{T}_1, p_1), \ldots, \mathrm{IPC}(\mathscr{T}_n, p_1), \ldots, \mathrm{IPC}(\mathscr{T}_1, p_l), \ldots, \mathrm{IPC}(\mathscr{T}_n, p_l)$ and joins them to the root of p. Then $\mathscr{A}_2$, using its deterministic two-way tree walking look-ahead capability on the indexed parallel computation trees, is able to count the look-ahead of $\mathscr{A}$.

Hereafter for every $i \in [1, n]$, $s \in S^{(i)}$ and $\sigma \in \Sigma$, we write $n_i(s, \sigma)$ to denote the number of (s, σ)-rules rather than $n_{\mathrm{T}_i}(s, \sigma)$. Set

$$m = \max(l \cdot n_i(s, \sigma) \mid i \in [1, n],\ s \in S^{(i)},\ \sigma \in \Sigma_l,\ l \in \mathrm{N}).$$

The dt-fst $\mathscr{A}_1 = \langle \Sigma, \Omega, Q^{(1)}, q_1, R_1\rangle$ is defined as follows:

(1) The ranked alphabet Ω is the least set satisfying the following three conditions:

(i) for every $\sigma\in\Sigma_l$, $l\in\mathbf{N}$ and for $d=(n+1)\cdot l$, $\sigma^{(d)}\in\Omega_d$;

(ii) for every $\sigma\in\Sigma_l$, $l\in\mathbf{N}$, $j\in[0,m]$, $s\in S^{(i)}$ and for $d=l\cdot n_i(s,\sigma)$, if $n_i(s,\sigma)>0$, then $\langle\sigma^{(d)},j\rangle\in\Omega_d$;

(iii) for each $j\in[0,m]$, the symbol $\langle *,j\rangle\in\Omega_0$.

(2) The state set for $\mathscr{A}_1$ is

$$Q^{(1)}=\{\langle s,i\rangle \mid s\in S^{(v)},\ v\in[1,n],\ i\in[0,m]\}\cup\{q_1\}.$$

(3) The rules set R_1 contains the following rules:

(a) For each $\sigma\in\Sigma_l$, $l\in\mathbf{N}$ and for $d=(n+1)\cdot l$,

$$q_1(\sigma(x_1,\ldots,x_l))\to\sigma^{(d)}(q_1(x_1),\ldots,q_1(x_l),\langle s_1,0\rangle(x_1),\ldots,$$

$$\langle s_n,0\rangle(x_1),\ldots,\langle s_1,0\rangle(x_l),\ldots,\langle s_n,0\rangle(x_l))$$

is a rule in R_1.

(If $l>0$, then the application of this rule at the root of a subtree $\sigma(p_1,\ldots,p_l)$ results in the computation of the indexed parallel computation trees $\mathrm{IPC}(\mathscr{A}_1,p_1),\ldots,\mathrm{IPC}(\mathscr{A}_n,p_1),\ldots,\mathrm{IPC}(\mathscr{A}_1,p_l),\ldots,\mathrm{IPC}(\mathscr{A}_n,p_l)$. These trees are being substituted in the $(l+1)$th, …, $(l+n)$th, …, $l+(l-1)\cdot n+1$th, …, $l\cdot(n+1)$th arguments of $\sigma^{(d)}$, respectively. If $l=0$, then we simply have the rule $q_1(\sigma)\to\sigma$.)

(b) For every $s\in S^{(i)}$, $i\in[0,n]$, $\sigma\in\Sigma_l$, $l\in\mathbf{N}$, $j\in[0,m]$ and for $k=n_i(s,\sigma)$, $d=k\cdot l$,

(i) if $k=0$, then $\langle s,j\rangle(\sigma(x_1,\ldots,x_l))\to\langle *,j\rangle\in R_1$;

(ii) if $k>0$ and the (s,σ)-rules in R_i are

$$s(\sigma(x_1,\ldots,x_l))\to\sigma(s_{11}(x_1),\ldots,s_{1l}(x_l)),$$

$$\vdots$$

$$s(\sigma(x_1,\ldots,x_l))\to\sigma(s_{k1}(x_1)),\ldots,s_{kl}(x_l)),$$

then the rule

$$\langle s,j\rangle(\sigma(x_1,\ldots,x_l))\to\langle\sigma^{(d)},j\rangle(\langle s_{11},1\rangle(x_1),\ldots,\langle s_{1l},l\rangle(x_l),\ldots,$$

$$\langle s_{k1},(k-1)\cdot l+1\rangle(x_1),\ldots,\langle s_{kl},k\cdot l\rangle(x_l))$$

is in R_1.

Now for $i\in[1,n]$, construct the 2-dfta $\mathscr{T}_i'$ to $\mathscr{T}_i$ in the same way as $\mathscr{T}_2$ was constructed to $\mathscr{T}$ in the proof of Theorem 3.6. The tree languages $L(\mathscr{T}_i')$, $i\in[1,n]$, will appear as look-ahead sets of $\mathscr{A}_2$, in fact, $\mathscr{A}_2$ counts the regular look-ahead of $\mathscr{A}$ on the indexed parallel computation trees joined to the nodes of the input tree for $\mathscr{A}$.

The rewriting rules for the $\mathrm{t}^{2\mathrm{dtl}}$-fst $\mathscr{A}_2=(\Omega,\Gamma,Q,q_0,R_2)$ are defined in the following manner.

(i) For every $q\in Q$ and $\sigma\in\Sigma_0$, $q(\sigma)\to\sigma$ is in R_2 if and only if $q(\sigma)\to\sigma$ is in R_1.

(ii) For every $q\in Q$, $\sigma\in\Sigma_l$ with $l>0$ and for $d=(n+1)\cdot l$, $\langle q(\sigma^{(d)}(x_1,\ldots,x_d))\to t;\ N_1,\ldots,N_d\rangle\in R_2$ if and only if $\langle q(\sigma(x_1,\ldots,x_l))\to t;\ L_{i_1},\ldots,L_{i_l}\rangle\in R_1$ with $i_1,\ldots,i_l\in[1,n]$ and

$$N_u=\begin{cases} L(\mathscr{T}'_j) & \text{if } u=l+i_j+(j-1)\cdot n \text{ for some } j\in[1,l],\\ T_{\Sigma\cup\Omega} & \text{if } u\in[1,l],\\ T_\Omega & \text{otherwise.}\end{cases}$$

It can be seen that $\tau(\mathscr{A})=\tau(\mathscr{A}_1)\circ\tau(\mathscr{A}_2)$ and that the construction of $\mathscr{A}_2$ preserves strong determinism. Thus, we have proved that

$$\mathrm{T^R\text{-}FST}\subseteq\mathrm{DT\text{-}FST}\circ\mathrm{T^{2DTL}\text{-}FST}$$

and that

$$\mathrm{D_sT^R\text{-}FST}\subseteq\mathrm{DT\text{-}FST}\circ\mathrm{D_sT^{2DTL}\text{-}FST}.$$

Now we show that $\mathrm{DT^R\text{-}FST}\subseteq\mathrm{DT\text{-}FST}\circ\mathrm{DT^{2DTL}\text{-}FST}$. Let, to this end, $\mathscr{B}=\langle\Sigma,\Gamma,Q,q_0,R\rangle$ be a $\mathrm{dt^r}$-fst and let $L_1,\ldots,L_n$ be all the tree languages that appear as look-ahead sets in some rule of $\mathscr{B}$. By Proposition 2.4 we may assume, without loss of generality, that there is an fta $\mathscr{A}=(\Sigma,\Sigma,S,s_0,R)$ such that, for each look-ahead set L_i with $i\in[1,n]$, there is a state $s_i\in S$ with $L_i=L(\mathscr{A}(s_i))$ and such that, for any two different states $s,s'\in S$, $L(\mathscr{A}(s))\cap L(\mathscr{A}(s'))=\emptyset$. Then put

$$m=\max(l\cdot n(s,\sigma)\mid s\in S,\ \sigma\in\Sigma_l,\ l\in\mathbf{N}).$$

We define a dt-fst $\mathscr{B}_1$ and a $\mathrm{t^{2dtl}}$-fst $\mathscr{B}_2$ such that $\tau(\mathscr{B})=\tau(\mathscr{B}_1)\circ\tau(\mathscr{B}_2)$. For each subtree p of the input tree with $p=\sigma(p_1,\ldots,p_l)$ and $l>0$, $\mathscr{B}_1$ computes the indexed parallel computation trees $\mathrm{IPC}(\mathscr{A}(s_1),p_1),\ldots,\mathrm{IPC}(\mathscr{A}(s_n),p_1),\ldots,\mathrm{IPC}(\mathscr{A}(s_1),p_l),\ldots,\mathrm{IPC}(\mathscr{A}(s_n),p_l)$ and joins them to the root of p. Then $\mathscr{B}_2$, using its deterministic two-way tree walking look-ahead capability on these indexed parallel computation trees, is able to count the look-ahead of $\mathscr{B}$.

The dt-fst $\mathscr{B}_1=\langle\Sigma,\Omega,Q^{(1)},q_1,R_1\rangle$ is defined as follows.

(1) The ranked alphabet Ω is the least set satisfying the following three conditions:

(i) For every $\sigma\in\Sigma_l$, $l\in\mathbf{N}$ and for $d=(n+1)\cdot l$, $\sigma^{(d)}\in\Omega_d$.

(ii) For every $\sigma\in\Sigma_l$, $l\in\mathbf{N}$, $j\in[0,m]$, $s\in S$ and for $d=l\cdot n(s,\sigma)$, if $n(s,\sigma)>0$, then $\langle\sigma^{(d)},j\rangle\in\Omega_d$.

(iii) For each $j\in[0,m]$, the symbol $\langle *,j\rangle\in\Omega_0$.

(2) The state set for $\mathscr{B}_1$ is

$$Q^{(1)}=\{\langle s,j\rangle\mid s\in S,\ j\in[0,m]\}\cup\{q_1\}.$$

(3) The rule set R_1 contains the following rules:

(a) For each $\sigma\in\Sigma_l$, $l\in\mathbf{N}$ and for $d=(n+1)\cdot l$,

$$q_1(\sigma(x_1,\ldots,x_l))\to\sigma^{(d)}(q_1(x_1),\ldots,q_1(x_l),\langle s_1,0\rangle(x_1),\ldots,$$
$$\langle s_n,0\rangle(x_1),\ldots,\langle s_1,0\rangle(x_l),\ldots,\langle s_n,0\rangle(x_l))$$

is a rule in R_1.

(If $l>0$, then the application of this rule at the root of a subtree $\sigma(p_1, \ldots, p_l)$ results in the computation of the indexed parallel computation trees $\mathrm{IPC}(\mathscr{A}_1, p_1), \ldots, \mathrm{IPC}(\mathscr{A}_n, p_1), \ldots, \mathrm{IPC}(\mathscr{A}_1, p_l), \ldots, \mathrm{IPC}(\mathscr{A}_n, p_l)$. These trees are being substituted in the $(l+1)$th, $\ldots, (l+n)$th, $\ldots, l+(l-1)\cdot n$th, $\ldots, l\cdot(n+1)$th arguments of $\sigma^{(d)}$, respectively. If $l=0$, then we simply have the rule $q_1(\sigma)\to\sigma$.)

(b) For every $s\in S$, $\sigma\in\Sigma_l$, $l\in\mathbf{N}$, $j\in[0, m]$ and for $k=n(s,\sigma)$, $d=k\cdot l$,

(i) if $k=0$, then $\langle s, j\rangle(\sigma(x_1, \ldots, x_l))\to\langle *, j\rangle$ is in R_1.

(ii) if $k>0$ and the (s,σ)-rules are

$$s(\sigma(x_1, \ldots, x_l)) \to \sigma(s_{11}(x_1), \ldots, s_{1l}(x_l)),$$
$$\vdots$$
$$s(\sigma(x_1, \ldots, x_1)) \to \sigma(s_{k1}(x_1), \ldots, s_{kl}(x_l)),$$

then

$$\langle s, j\rangle(\sigma(x_1, \ldots, x_l)) \to \langle \sigma^{(d)}, j\rangle(\langle s_{11}, 1\rangle(x_1), \ldots,$$
$$\langle s_{1l}, l\rangle(x_l), \ldots, \langle s_{k1}, (k-1)\cdot l+1\rangle(x_1), \ldots, \langle s_{kl}, k\cdot l\rangle(x_l))$$

is a rule in R_1.

Construct the 2-dfta's $\mathscr{T}_2$ and $\mathscr{T}_3$ to $\mathscr{T}$ as in the proof of Theorem 3.6 and as in the remark after the proof of Theorem 3.6, respectively. The tree languages $L(\mathscr{T}_2)$ and $L(\mathscr{T}_3)$ will appear as look-ahead sets of $\mathscr{B}_2$, in fact, $\mathscr{B}_2$ counts the regular look-ahead of $\mathscr{B}$ on the indexed parallel computation trees that are joined to the nodes of the input tree for $\mathscr{B}$.

The rewriting rules of the $\mathrm{t}^{\mathrm{2dtl}}$-fst $\mathscr{B}_2=(\Delta, \Gamma, Q, q_0, R_2)$ are defined in the following manner.

(i) For every $q\in Q$ and $\sigma\in\Sigma_0$, the rule $q(\sigma)\to\sigma$ is in R_2 if and only if the rule $q(\sigma)\to\sigma$ is in R_1.

(ii) For every $q\in Q$, $\sigma\in\Sigma_l$ with $l>0$ and for $d=(n+1)\cdot l$,

$$\langle q(\sigma^{(d)}(x_1, \ldots, x_d))\to t;\ N_1, \ldots, N_d\rangle\in R_2$$

if and only if

$$\langle q(\sigma(x_1, \ldots, x_l))\to t;\ L_{i_1}, \ldots, L_{i_l}\rangle\in R_1$$

with $i_1, \ldots, i_l\in[1, n]$ and

$$N_u=\begin{cases} L(\mathscr{T}_2) & \text{if } u=l+i_j+(j-1)\cdot n \text{ for some } j\in[1, l],\\ T_{\Sigma\cup\Omega} & \text{if } u\in[1, l],\\ L(\mathscr{T}_3) & \text{otherwise.}\end{cases}$$

Here we show that $\mathscr{B}_2$ is deterministic.

Consider two different $(q, \sigma^{(d)})$-rules

$$\langle q(\sigma^{(d)}(x_1, \ldots, x_d)) \to t;\ N_1, \ldots, N_d \rangle$$

and

$$\langle q(\sigma^{(d)}(x_1, \ldots, x_d)) \to t';\ N'_1, \ldots, N'_d \rangle$$

of $\mathscr{B}_2$ that are constructed from the different rules

$$\langle q(\sigma(x_1, \ldots, x_l)) \to t;\ L_{v_1}, \ldots, L_{v_l} \rangle$$

and

$$\langle q(\sigma(x_1, \ldots, x_l)) \to t';\ L_{z_1}, \ldots, L_{z_l} \rangle$$

of $\mathscr{T}$, respectively. Since $\mathscr{T}$ is deterministic, $L_{v_i} \cap L_{z_i} = \emptyset$ for some $i \in [1, l]$. Let $u = l + v_i + (i-1) \cdot n$. Then $N_u = L(\mathscr{T}_2)$ and $N'_u = L(\mathscr{T}_3)$. After the proof of Theorem 3.6 we noted that $L(\mathscr{T}_2) \cap L(\mathscr{T}_3) = \emptyset$, implying $N_u \cap N'_u = \emptyset$. Thus, we have proved that $\mathscr{B}_2$ is a $\mathrm{dt}^{\mathrm{2dtl}}$-fst. It is left to the reader to show that $\tau(\mathscr{B}) = \tau(\mathscr{B}_1) \circ \tau(\mathscr{B}_2)$.

Thus, we have verified the inclusion $\mathrm{DT^R}$-FST $\subseteq$ DT-FST $\circ$ $\mathrm{DT^{2DTL}}$-FST. Since 2DTL $\subseteq$ 2NTL $\cap$ 2UTL (see Fig. 1), all the other inclusions that are stated by the theorem are true. □

In the next theorem we show how to simulate the composition of a deterministic top-down tree transducer and of a top-down tree transducer with two-way tree walking look-ahead by a top-down tree transducer with regular look-ahead.

Theorem 3.8. *For any values of the modifiers* $X \in \{\ , \mathrm{D}\}$ *and* $Y \in \{\mathrm{2DTL}, \mathrm{2NTL}, \mathrm{2UTL}\}$, DT-FST $\circ$ $X\mathrm{T}^Y$-FST $\subseteq$ $X\mathrm{T}^{\mathrm{R}}$-FST; *moreover,* DT-FST $\circ$ $\mathrm{D_sT}^Y$-FST $\subseteq$ FTA $\circ$ DT-FST.

Proof.

(i) DT-FST $\circ$ T^Y-FST $\subseteq$ $\mathrm{DT^R}$-FST $\circ$ $\mathrm{T^R}$-FST $\subseteq$ $\mathrm{T^R}$-FST (by Theorem 2.11 in [2]);

(ii) DT-FST $\circ$ DT^Y-FST $\subseteq$ $\mathrm{DT^R}$-FST $\circ$ $\mathrm{DT^R}$-FST $\subseteq$ $\mathrm{DT^R}$-FST (by Theorem 2.11 in [2]);

(iii) DT-FST $\circ$ $\mathrm{D_sT}^Y$-FST $\subseteq$ DT-FST $\circ$ $\mathrm{D_sT^R}$-FST (by Fig. 1)
= FTA $\circ$ DT-FST (by the proof of Lemma 6.3 in [12]).

□

Here we give a new decomposition of Rounds' transformational system.

Theorem 3.9. *For each* $Y \in \{\mathrm{2DTL}, \mathrm{2NTL}, \mathrm{2UTL}\}$, FTA $\circ$ DT-FST $\subseteq$ DT-FST $\circ$ $\mathrm{D_sT}^Y$-FST.

Proof. Let $\mathscr{A}_1 = (\Sigma, \Sigma, Q^{(1)}, q_1, R_1)$ be an fta and let $\mathscr{A}_2 = (\Sigma, \Gamma, Q^{(2)}, q_2, R_2)$ be a dt-fst. We define the dt-fst $\mathscr{A}_3 = (\Sigma, \Gamma, Q^{(3)}, q_3, R_3)$ and the $\mathrm{d_st}^{\mathrm{2dtl}}$-fst $\mathscr{A}_4 = (\Gamma, \Delta, Q^{(4)}, q_4, R_4)$ such that $\tau(\mathscr{A}_1) \circ \tau(\mathscr{A}_2) = \tau(\mathscr{A}_3) \circ \tau(\mathscr{A}_4)$.

Intuitively, for an input tree t, $\mathscr{A}_3$ computes $\mathrm{IPC}(\mathscr{A}_1, t)$ and joins it to the root of t. Then $\mathscr{A}_4$, using its deterministic two-way tree walking look-ahead capability on $\mathrm{IPC}(\mathscr{A}_1, t)$, decides whether $t \in L(\mathscr{A}_1)$. If $t \in L(\mathscr{A}_1)$, then $\mathscr{A}_4$ simulates $\mathscr{A}_2$ on t.

Set $m = \max(l \cdot n(q, \sigma) \mid q \in Q,\ \sigma \in \Sigma_l,\ l \in \mathbf{N})$. The dt-fst $\mathscr{A}_3 = \langle \Sigma, \Delta, Q^{(3)}, q_3, R_3 \rangle$ is defined as follows:

(1) Δ is the least ranked alphabet satisfying (a), (b), (c) and (d).

(a) For every $\sigma \in \Sigma_l$, $l \in \mathbf{N}$, $i \in [0, m]$, $q \in Q$ such that $n(q, \sigma) > 0$ and for $d = l \cdot n(q, \sigma)$, $\langle \sigma^{(d)}, i \rangle \in \Delta_d$.

(b) For each $i \in [0, m]$, $\langle *, i \rangle \in \Delta_0$.

(c) For every $\sigma \in \Sigma_l$, $\bar{\sigma} \in \Delta_{l+1}$.

(d) $\Sigma \subseteq \Delta$.

(2) $Q^{(3)} = \{q_3, \mathrm{id}\} \cup \{\langle q, i \rangle \mid q \in Q^{(1)},\ i \in [0, m]\}$.

(3) R_3 is defined in the following way.

(i) For every $q \in Q^{(1)}$, $\sigma \in \Sigma_l$, $l \in \mathbf{N}$, $i \in [0, m]$ and for $k = n(q, \sigma)$, $d = k \cdot l$,

(a) if $k = 0$, then $\langle q, i \rangle (\sigma(x_1, \ldots, x_l)) \to \langle *, i \rangle \in R_1$

(b) if $k > 0$, and the (q, σ)-rules are

$$q(\sigma(x_1, \ldots, x_l)) \to \sigma(q_{11}(x_1), \ldots, q_{1l}(x_l)),$$
$$\vdots$$
$$q(\sigma(x_1, \ldots, x_l)) \to \sigma(q_{k1}(x_1), \ldots, q_{kl}(x_l)),$$

then the rule

$$\langle q, i \rangle (\sigma(x_1, \ldots, x_l)) \to \langle \sigma^{(d)}, i \rangle (\langle q_{11}, 1 \rangle (x_1), \ldots,$$
$$\langle q_{1l}, l \rangle (x_l), \ldots, \langle q_{k1}, (k-1) \cdot l + 1 \rangle (x_1), \ldots, \langle q_{kl}, k \cdot l \rangle (x_l))$$

is in R_3.

(ii) For every $\sigma \in \Sigma_l$ such that $l > 0$ and for $k = n(q, \sigma)$, $d = k \cdot l$, if the (q_1, σ)-rules of R_1 are

$$q_1(\sigma(x_1, \ldots, x_l)) \to \sigma(q_{11}(x_1), \ldots, q_{1l}(x_l)),$$
$$\vdots$$
$$q_1(\sigma(x_1, \ldots, x_l)) \to \sigma(q_{k1}(x_1), \ldots, q_{kl}(x_l)),$$

then the rule

$$q_3(\sigma(x_1, \ldots, x_l)) \to \bar{\sigma}(\mathrm{id}(x_1), \ldots, \mathrm{id}(x_l), \langle \sigma^{(d)}, 0 \rangle (\langle q_{11}, 1 \rangle (x_1), \ldots,$$
$$\langle q_{1l}, l \rangle (x_l), \ldots, \langle q_{k1}, (k-1) \cdot l + 1 \rangle (x_1), \ldots, \langle q_{kl}, k \cdot l \rangle (x_l)))$$

is in R_3.

(iii) For each $\sigma \in \Sigma_0$, if the rule $q_1(\sigma) \to \sigma$ is in R_1, then the rule $q_3(\sigma) \to \sigma$ is in R_3.

(iv) For every $\sigma \in \Sigma_l$ and $l \in \mathbf{N}$, $\mathrm{id}(\sigma(x_1, \ldots, x_l)) \to \sigma(\mathrm{id}(x_1), \ldots, \mathrm{id}(x_l))$ is in R_3.

It is an easy observation that $\tau(\mathscr{A}_3) = \{(t, t) \mid t \in L(\mathscr{A}_1) \cap \Sigma_0\} \cup \{(\sigma(t_1, \ldots, t_l), \bar{\sigma}(t_1, \ldots, t_l, \mathrm{IPC}(\mathscr{A}_1, \sigma(t_1, \ldots, t_l))) \mid l > 0,\ \sigma \in \Sigma_l,\ t_1, \ldots, t_l \in T_\Sigma\}$.

The $d_s t^{2dtl}$-fst $\mathscr{A}_4 = \langle \Delta, \Gamma, Q^{(4)}, q_4, R_4 \rangle$ is constructed as follows:

(1) $Q^{(4)} = \{q_4\} \cup Q^{(2)}$.

(2) The rule set R_4 is determined by (i) and (ii).

(i) Define $\mathscr{B}$ to $\mathscr{A}_1$ in the same way as $\mathscr{T}_2$ was defined to $\mathscr{T}$ in the proof of Theorem 3.6. If $\bar{\sigma} \in \Sigma_{l+1}$ with $l > 0$ and $q_2(\sigma(x_1, \ldots, x_l)) \to r$ is in R_2, then the rule $\langle q_4(\bar{\sigma}(x_1, \ldots, x_{l+1})) \to r; T_\Sigma, \ldots, T_\Sigma, L(\mathscr{B}) \rangle$ is in R_4. (This rule can be applied at the root of the tree $\bar{\sigma}(t_1, \ldots, t_l, \mathrm{IPC}(\mathscr{A}_1, \sigma(t_1, \ldots, t_l)))$ if $\mathrm{IPC}(\mathscr{A}_1, \sigma(t_1, \ldots, t_l)) \in L(\mathscr{B})$, i.e. if $\sigma(t_1, \ldots, t_l) \in L(\mathscr{A}_1)$.)

(ii) If $\sigma \in \Sigma_0 \cap L(\mathscr{A}_1)$ and $q_2(\sigma) \to r$ is in R_2, then $q_4(\sigma) \to r$ is in R_4.

($\mathscr{A}_4$ simulates the computation of $\mathscr{A}_2$ on the tree σ of height 0 if $\sigma \in L(\mathscr{A}_1)$.)

(iii) For every $\sigma \in \Sigma_l$ and $q \in Q^{(2)}$, if $q(\sigma(x_1, \ldots, x_l)) \to r$ is in R_2, then $\langle q(\sigma(x_1, \ldots, x_l)) \to r; T_\Sigma, \ldots, T_\Sigma \rangle$ is in R_4. (After coming down from the root of the input tree $\sigma(t_1, \ldots, t_l, \bar{\sigma}(t_1, \ldots, t_l, \mathrm{IPC}(\mathrm{A}_1, \sigma(t_1, \ldots, t_l))))$, $\mathscr{A}_4$ simulates $\mathscr{A}_2$ in the natural way.)

It is easy to see that $\tau(\mathscr{A}_1) \circ \tau(\mathscr{A}_2) = \tau(\mathscr{A}_3) \circ \tau(\mathscr{A}_4)$. Thus, we have the inclusion $\mathrm{FTA} \circ \mathrm{DT\text{-}FST} \subseteq \mathrm{DT\text{-}FST} \circ \mathrm{D_s T^{2DTL}\text{-}FST}$. The other two inclusions of the theorem are simple consequences of the above result. □

The main result of this section gives a characterization for the translation power of top-down tree transducers with two-way tree walking look-ahead.

Theorem 3.10. *For any value of the modifiers* $X \in \{\ , \mathrm{D}\}$ *and* $Y \in \{2\mathrm{DTL}, 2\mathrm{NTL}, 2\mathrm{UTL}\}$, *we have*

$$X\mathrm{T^R\text{-}FST} = \mathrm{DT\text{-}FST} \circ X\mathrm{T^Y\text{-}FST}$$

and

$$\mathrm{FTA} \circ \mathrm{DT\text{-}FST} = \mathrm{DT\text{-}FST} \circ \mathrm{D_s T^Y\text{-}FST}.$$

Proof. It follows from Theorems 3.7, 3.8 and 3.9. □

4. An inclusion diagram for deterministic and strongly deterministic classes of top-down tree transformations with look-ahead

In this section we gather together the deterministic and strongly deterministic classes of top-down tree transformations with some type of look-ahead and Rounds' transformational system. We prove that Fig. 3 shows the inclusion diagram of these classes.

To this end, we prove that all inclusions shown are proper and that all unrelated classes are incomparable. Along the proof we verify the following five formulae.

(i) $\mathrm{DT\text{-}FST} \subset \mathrm{D_s T^{DTR}\text{-}FST}$;

(ii) for any modifier $Y \in \{2\mathrm{DTL}, 2\mathrm{NTL}, 2\mathrm{UTL}\}$,
$\mathrm{D_s T^Y\text{-}FST} \subset \mathrm{DT^Y\text{-}FST}$ (i.e. the vertical lines indicate proper inclusions);

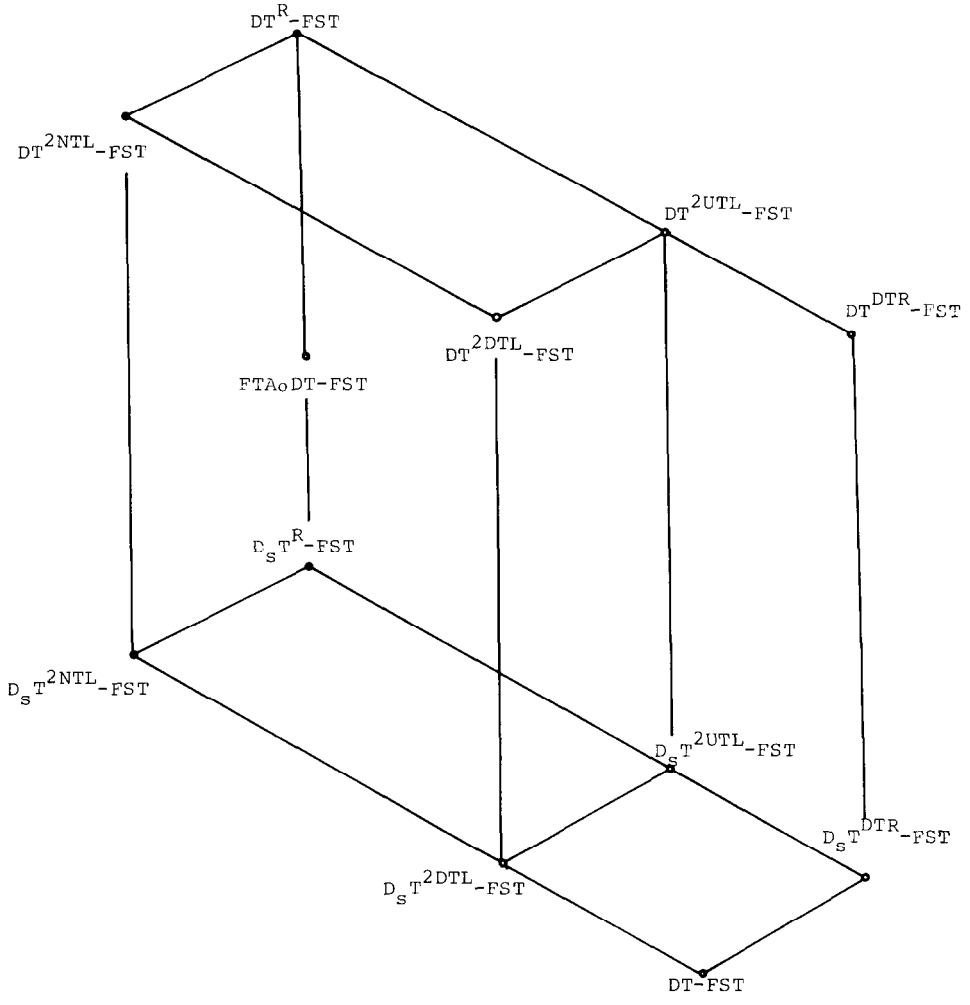

Fig. 3.

(iii) $D_sT^{DTR}\text{-FST} \not\subseteq DT^{2NTL}\text{-FST}$;

(iv) $D_sT^{2NTL}\text{-FST} \not\subseteq DT^{2UTL}\text{-FST}$;

(v) $D_sT^{2DTL}\text{-FST} \not\subseteq DT^{DTR}\text{-FST}$.

To begin with we prove (i). In [17] it was shown that $D_sT^{DTR}\text{-FST} = (\text{DT-FST})^2$ and in [21] the example after Theorem I.2 shows that $\text{DT-FST} \subset (\text{DT-FST})^2$. Thus, $\text{DT-FST} \subset D_sT^{DTR}\text{-FST}$.

We proceed by proving (ii). According to Fig. 2, $D_sT^{DTR}\text{-FST} \subset DT^{DTR}\text{-FST}$ and $D_sT^{R}\text{-FST} \subset \text{FTA} \circ \text{DT-FST} \subset DT^{R}\text{-FST}$. By Theorem 3.10, for any modifier $Y \in \{2\text{DTL}, 2\text{NTL}, 2\text{UTL}\}$,

$$\text{DT-FST} \circ D_sT^{Y}\text{-FST} = \text{FTA} \circ \text{DT-FST} \subset DT^{R}\text{-FST} = \text{DT-FST} \circ DT^{Y}\text{-FST}.$$

Thus, $D_sT^Y\text{-FST} \subset DT^Y\text{-FST}$.

Here we prove (iii). We adopt the two-way tree walking languages introduced in Example 3.2 of [19]. Let $\Sigma = \Sigma_0 \cup \Sigma_2$, $\Sigma_0 = \{a, b\}$ and $\Sigma_2 = \{\sigma\}$. Thus, T_Σ is the set of all binary trees in which every interior node is labeled by σ and every leaf is labeled by a or b. Consider the following 2-fta $\mathscr{C} = (\{q_0, q_F\}, \Sigma, \delta, q_0, \{q_F\})$, where $\delta(q_0, \sigma) = \{(q_0, 1), (q_0, 2)\}$, $\delta(q_0, a) = (q_F, 0)$, $\delta(q_F, \sigma) = (q_F, 0)$ and all other transitions are undefined. Then

$$L_N(\mathscr{C}) = \{t \in T_\Sigma \mid t \text{ has a leaf labeled by } a\}$$

and

$$L_U(\mathscr{C}) = \{t \in T_\Sigma \mid \text{every leaf of } t \text{ is labeled by } a\}.$$

The set of balanced binary trees over Σ is the least set BBT that satisfies the following two conditions:

(i) $a, b \in \text{BBT}$.

(ii) For any $p, q \in \text{BBT}$ such that $\text{hg}(p) = \text{hg}(q)$, $\sigma(p, q) \in \text{BBT}$.

We prove the following result by slightly modifying the proof of Theorem 5.6 in [20].

Proposition 4.1. *For every 2-fta* $\mathscr{B} = (Q, \Sigma, \delta, q_0, F)$ *such that* $L_N(\mathscr{B}) \subseteq L_U(\mathscr{C})$ *and for every balanced binary tree* t *in* $L_N(\mathscr{B})$, $\text{hg}(t) \leqslant |Q| + 1$.

Proof. Consider any balanced binary tree $t \in L_N(\mathscr{B})$ of height greater than or equal to $|Q| + 2$. Let P be an arbitrary accepting computation path of $\mathscr{B}$ on t. Along P, $\mathscr{B}$ must visit all leaves of t since otherwise $\mathscr{B}$ would also accept the tree obtained from t by changing the label of an unvisited node from a to b. Let $a_1, a_2, \ldots, a_k$ be the sequence of leaves of t visited by $\mathscr{B}$ along P in this order, some leaves may be repeated in the sequence. For any two leaves, a_i and a_j, the distance between a_i and a_j is defined to be the length of the path from a_i (or a_j) to their nearest common ancestor. If $a_i = a_j$, then the distance between them is 0. It should be clear that there are two leaves a_i and a_{i+1} with distance greater than or equal to $|Q| + 2$ and that $\mathscr{B}$ goes from a_i to a_{i+1} through their nearest common ancestor n. In the transition process from a_i to n, $\mathscr{B}$ visits at least $|Q| + 1$ different internal nodes, all of which are labeled by σ. Therefore, $\mathscr{B}$ must visit two different internal nodes, n_1 and n_2 say, n_2 being above n_1, in the same state. Since $\mathscr{B}$ does not visit any leaf between a_i and a_{i+1}, it must repeat the computation between n_1 and n_2 until it arrives to the root of t and "falls off" t. But this is impossible since having "fallen off" t, $\mathscr{B}$ cannot visit a_{i+1}. This contradiction proves the proposition. □

The following result is immediate by Proposition 4.1.

Consequence 4.2. *Let $n \geqslant 1$ and let, for each $1 \leqslant i \leqslant n$, $\mathscr{A}_i = (Q_i, \Sigma_i, \delta_i, q_i, F_i)$ be a 2-fta with $L_N(\mathscr{A}_i) \subseteq L_U(\mathscr{C})$. Then, $\bigcup (L_N(\mathscr{A}_i) \mid 1 \leqslant i \leqslant n) \subset L_U(\mathscr{C})$.*

Using Consequence 4.2 now we can prove (iii).

Theorem 4.3. $D_sT^{DTR}\text{-FST} \not\subseteq DT^{2NTL}\text{-FST}$.

Proof. Consider the relation $\rho = \{(t, a) \mid t \in L_U(\mathscr{C})\}$, which can be induced by a d_st^{dtr}-fst. Suppose that $\rho \in DT^{2NTL}\text{-FST}$, i.e. for some dt^{2ntl}-fst $\mathscr{T} = \langle \Sigma, \Delta, Q, q_0, R \rangle$, $\rho = \tau(\mathscr{T})$. Let $L_1, \ldots, L_n$ be all tree languages that appear as look-ahead sets in some rule of $\mathscr{T}$ such that $L_i \subseteq L_U(\mathscr{C})$ for each $i \in [1, n]$. According to Consequence 4.2, there exists a tree t such that $t \in L_U(\mathscr{C}) - \bigcup (L_i \mid i \in [1, n])$. Since $\sigma(t, t) \in \mathrm{dom}(\tau(\mathscr{T}))$, there exists a derivation $q_0(\sigma(t, t)) \Rightarrow_{\mathscr{T}}^* a$. In the first step of this derivation $\mathscr{T}$ deletes at least one occurrence of t, i.e. the rewriting rule applied at the root of t has the form $\langle q_0(\sigma(x_1, x_2)) \to s, L_{i_1}, L_{i_2} \rangle$ with $s \in \{q(x_1), q(x_2) \mid q \in Q\} \cup \{a\}$, $i_1, i_2 \in [1, n]$. Since $t \in L_{i_1}$ and $t \in L_{i_2}$, $L_{i_1} \not\subseteq L_U(\mathscr{C})$ and $L_{i_2} \not\subseteq L_U(\mathscr{C})$. Thus, there exist trees t_1 and t_2 with $t_1 \in L_{i_1} - L_U(\mathscr{C})$ and $t_2 \in L_{i_2} - L_U(\mathscr{C})$. If $s = q(x_1)$ for some $q \in Q$, then $q_0(\sigma(t, t_2)) \Rightarrow_{\mathscr{T}}^* a$ holds. If $s = q(x_2)$ for some $q \in Q$, then we have $q_0(\sigma(t_1, t)) \Rightarrow_{\mathscr{T}}^* a$. If $s = a$, then we obtain $q_0(\sigma(t_1, t_2)) \Rightarrow_{\mathscr{T}}^* a$. All three cases contradict the assumption $\mathrm{dom}(\tau(\mathscr{T})) = L_U(\mathscr{C})$. □

Now we shall prove (iv). The proof of the following proposition is based on that of Lemma 3.4 in [19].

Proposition 4.4. *Let $\mathscr{A} = \langle Q, \Sigma, \delta, q_0, F \rangle$ be a 2-fta such that $L_U(\mathscr{A}) \subseteq L_N(\mathscr{C})$ and let t be a binary balanced tree with $\mathrm{hg}(t) > \log_2(2 \cdot |Q| + 2)$ in which every leaf is labeled by b. Then there is a nonaccepting maximal computation path P of $\mathscr{A}$ on t such that $\mathscr{A}$ visits at most $2 \cdot |Q| + 2$ leaves during P.*

Proof. Note that t has more than $2 \cdot k + 2$ leaves. Since $t \notin L_U(\mathscr{A})$, there is a nonaccepting maximal computation path $P' = \{I_n \mid n < k \leqslant \omega\}$ of $\mathscr{A}$ on t. Suppose that $\mathscr{A}$ visits at least $2 \cdot |Q| + 3$ leaves during P' and let $\langle a_i \mid i < k \leqslant \omega \rangle$ be the sequence of leaves visited in this order by $\mathscr{A}$ during P'. Since there are more than $|Q|$ leaves, $\mathscr{A}$ must visit two leaves during P', say a_i and a_j, in the same state. Let a_i and a_j be the first such nodes in the sequence. Hence, there are $0 < l_i < l_j < k$ and $q \in Q$ such that $I_{l_i} = (q, a_i, t)$, $I_{l_j} = (q, a_j, t)$ and $I_{l_i} \vdash_{\mathscr{A}}^* I_{l_j}$. Furthermore, $\mathscr{A}$ visits at most $|Q| + 1$ leaves during the sequence $I_0, \ldots, I_{l_i}, \ldots, I_{l_j}$ in P. Here we construct the sequence of ID's such that $I_{l_j} \vdash_{\mathscr{A}}^* I_{l_j}$ by applying the same transition at each step as in $I_{l_i} \vdash_{\mathscr{A}}^* I_{l_j}$, but by starting at n_j instead of n_i. Note that this is possible because t is a balanced tree in which all the internal nodes together with the root and all leaves are labeled by the same symbols (σ and b, respectively). Then, we can obtain a maximal computation path P by infinitely repeating this sequence after $I_0, \ldots, I_{l_j}$. Clearly, P is not accepting; moreover, $\mathscr{A}$ visits at most $2 \cdot |Q| + 2$ leaves during P since it visits the same leaves, the number of which is at most $|Q| + 1$, in the repeated portion $I_{l_j} \vdash_{\mathscr{A}}^* I_{l_j}$ of P. □

From Proposition 4.4 we directly get the following result.

Corollary 4.5. *Let $n \geqslant 1$, and for each $i \in [1, n]$, let $\mathscr{A}_i = \langle Q_i, \Sigma, \delta_i, q_i, F_i \rangle$ be a 2-fta such that $L_U(\mathscr{A}_i) \subseteq L_N(\mathscr{C})$. Then $\bigcup (L_U(\mathscr{A}_i) \mid i \in [1, n]) \subset L_N(\mathscr{C})$.*

Proof. Let t be a binary balanced tree with $\mathrm{hg}(t) > \log_2(n \cdot \max(2 \cdot |Q_i| \mid i \in [1, n]) + 2 \cdot n)$ in which every leaf is labeled by b. For $i \in [1, n]$, let P_i be a nonaccepting maximal computation path of $\mathscr{A}_i$ on t such that $\mathscr{A}_i$ visits at most $2 \cdot |Q_i| + 2$ leaves during P_i (cf. Proposition 4.4). Since t has more than $n \cdot \max(2 \cdot |Q_i| \mid i \in [1, n]) + 2 \cdot n$ leaves, there is a leaf which is not visited by $\mathscr{A}_i$ during P_i for each $i \in [1, n]$. Define r from t by changing the label of this leaf from b to a. Then $r \in L_N(\mathscr{C}) - \bigcup (L_U(\mathscr{A}_i) \mid i \in [1, n])$. □

Now we are able to prove (iv).

Theorem 4.6. $D_sT^{2NTL}\text{-FST} \not\subseteq DT^{2UTL}\text{-FST}$.

Proof. Let us consider the tree transformation $\rho = \{(\sigma(t_1, t_2), a) \mid t_1, t_2 \in L_N(\mathscr{C})\}$, which can be induced by a d_st^{2ntl}-fst and assume that $\rho \in DT^{2UTL}\text{-FST}$, i.e. that there exists a dt^{2utl}-fst $\mathscr{T} = \langle \Sigma, \Sigma, Q, q_0, R \rangle$ such that $\rho = \tau(\mathscr{T})$. Let $L_1, \ldots, L_n$ be all tree languages that appear as look-ahead sets in some rule of $\mathscr{T}$ such that $L_i \subseteq L_N(\mathscr{C})$ for each $i \in [1, n]$. According to Corollary 4.5, there exists a tree t such that $t \in L_N(\mathscr{C}) - \bigcup (L_i \mid i \in [1, n])$. Now $\sigma(t, t)$ is in $L_N(\mathscr{C})$ as well. Thus, $q_0(\sigma(t, t)) \Rightarrow_{\mathscr{T}}^* a$ holds. From now on the proof can be finished in the same way as that of Theorem 4.3 was finished. □

Finally, we prove (v). Let $\Sigma = \Sigma_0 \cup \Sigma_2$ where $\Sigma_0 = \{a\}$ and $\Sigma_2 = \{\sigma\}$. The tree language $K_0 \subseteq T_\Sigma$ is the set of all trees over Σ that contain an even number of leaves while K_1 is the complement of K_0, i.e. $K_1 = T_\Sigma - K_0$. Let $\tilde{K}_0 = \{\mathrm{ind}(p) \mid p \in K_0\}$ and $\tilde{K}_1 = \{\mathrm{ind}(p) \mid p \in K_1\}$. Using an argument similar to the proof of Lemma 5.9 in [12] one can prove the following result.

Lemma 4.7. *Let $n \geqslant 1$, and for each $i \in [1, n]$, let $\mathscr{A}_i = \langle \Sigma, \Sigma, Q, q_0, R \rangle$ be a dfta such that $L(\mathscr{A}_i) \subseteq \tilde{K}_0$. Then $\bigcup (L(\mathscr{A}_i) \mid i \in [1, n]) \subset \tilde{K}_0$.*

On the other hand, $\tilde{K}_0$ is in 2DTL.

Lemma 4.8. $\tilde{K}_0 \in 2\mathrm{DTL}$.

Proof. We construct a 2-dfta $\mathscr{A}$ with $L(\mathscr{A}) = \tilde{K}_0$; $\mathscr{A}$ traverses a subtree $p = \langle \sigma, i \rangle (p_1, p_2)$ with $i \in [1, 2]$ of the input tree in the following order:

(1) $\mathscr{A}$ visits the root of p,

(2) $\mathscr{A}$ traverses the subtree p_1,

(3) $\mathscr{A}$ visits the root of p,

(4) $\mathscr{A}$ traverses the subtree p_2,

(5) $\mathscr{A}$ "falls off" the subtree p.

While $\mathscr{A}$ is wandering up and down the input tree t, it examines if t is an indexed copy of some tree in T_Σ and counts the parity of the number of the already visited leaves.

The formal construction of $\mathscr{A}$ is as follows.

$$\mathscr{A} = \langle \Delta, Q, \delta, \text{start}, \{\text{yes}\}\rangle, \text{ where}$$

(1) $\Delta = \Delta_0 \cup \Delta_2$, $\Delta_0 = \{\langle a, i\rangle \mid i \in [0, 2]\}$ and $\Delta_2 = \{\langle \sigma. i\rangle \mid i \in [0, 2]\}$;

(2) $Q = \{\langle i, \text{even}, 0\rangle, \langle i, \text{odd}, 0\rangle, \langle \text{even}, i\rangle, \langle \text{odd}, i\rangle \mid i \in [1, 2]\} \cup \{\text{start}, \text{yes}\}$.

Intuitively the state $\langle i, \text{even}, 0\rangle$ means that

(a) $\mathscr{A}$ checks whether the current node is the ith son of its father (in this way $\mathscr{A}$ examines whether the input tree is an indexed copy of some tree in T_Σ), that

(b) the parity of the number of the already visited leaves is even, and that

(c) $\mathscr{A}$ came to the current node from its father. Similarly, the state $\langle \text{even}, j\rangle$ means that

(a) the number of the already visited leaves is even, and that

(b) $\mathscr{A}$ came to the current node from its jth son.

(3) δ is defined as follows:

$$\delta(\text{start}, \langle \sigma, 0\rangle) = (\langle 1, \text{even}, 0\rangle, 1),$$

$$\delta(\langle 1, \text{even}, 0\rangle, \langle \sigma, 1\rangle) = (\langle 1, \text{even}, 0\rangle, 1),$$

$$\delta(\langle 1, \text{even}, 0\rangle, \langle a, 1\rangle) = (\langle \text{odd}, 1\rangle, 0),$$

$$\delta(\langle 2, \text{even}, 0\rangle, \langle \sigma, 2\rangle) = (\langle 1, \text{even}, 0\rangle, 1),$$

$$\delta(\langle 2, \text{even}, 0\rangle, \langle a, 2\rangle) = (\langle \text{odd}, 2\rangle, 0),$$

$$\delta(\langle \text{even}, 1\rangle, \langle \sigma, 1\rangle) = (\langle 2, \text{even}, 0\rangle, 2),$$

$$\delta(\langle \text{even}, 1\rangle, \langle \sigma, 2\rangle) = (\langle 2, \text{even}, 0\rangle, 2),$$

$$\delta(\langle \text{even}, 1\rangle, \langle \sigma, 0\rangle) = (\langle 2, \text{even}, 0\rangle, 2),$$

$$\delta(\langle \text{even}, 2\rangle, \langle \sigma, 1\rangle) = (\langle \text{even}, 1\rangle, 0),$$

$$\delta(\langle \text{even}, 2\rangle, \langle \sigma, 2\rangle) = (\langle \text{even}, 2\rangle, 0),$$

$$\delta(\langle \text{even}, 2\rangle, \langle \sigma, 0\rangle) = (\text{yes}, 0),$$

$$\delta(\langle 1, \text{odd}, 0\rangle, \langle \sigma, 1\rangle) = (\langle 1, \text{odd}, 0\rangle, 1),$$

$$\delta(\langle 1, \text{odd}, 0\rangle, \langle a, 1\rangle) = (\langle \text{even}, 1\rangle, 0),$$

$$\delta(\langle 2, \text{odd}, 0\rangle, \langle \sigma, 2\rangle) = (\langle 1, \text{odd}, 0\rangle, 1),$$

$\delta(\langle 2, \text{odd}, 0\rangle, \langle a, 2\rangle) = (\langle \text{even}, 2\rangle, 0)$,

$\delta(\langle \text{odd}, 1\rangle, \langle \sigma, 1\rangle) = (\langle 2, \text{odd}, 0\rangle, 2)$,

$\delta(\langle \text{odd}, 1\rangle, \langle \sigma, 2\rangle) = (\langle 2, \text{odd}, 0\rangle, 2)$,

$\delta(\langle \text{odd}, 1\rangle, \langle \sigma, 0\rangle) = (\langle 2, \text{odd}, 0\rangle, 2)$,

$\delta(\langle \text{odd}, 2\rangle, \langle \sigma, 1\rangle) = (\langle \text{odd}, 2\rangle, 0)$,

$\delta(\langle \text{odd}, 2\rangle, \langle \sigma, 2\rangle) = (\langle \text{odd}, 2\rangle, 0)$,

and all other transitions are undefined.

It can be seen that, using the states $\langle i, \text{even}, 0\rangle$ and $\langle i, \text{odd}, 0\rangle$ with $i \in [1, 2]$, $\mathscr{A}$ checks whether the input tree is an indexed copy of a tree over Σ. Having traversed a proper subtree, $\mathscr{A}$ goes up to the father of its root in state $\langle \text{even}, i\rangle$ or $\langle \text{odd}, i\rangle$ with $i \in [1, 2]$ depending on the number of leaves in it and on the position of its root with respect to its brothers. The 2-dfta $\mathscr{A}$ "falls off" the input tree in state yes if there are an even number of leaves in it.

It is not hard to see that $L(\mathscr{A}) = \tilde{K}_0$. □

Now we are able to prove (v).

Theorem 4.9. $D_sT^{2DTL}\text{-FST} \not\subseteq DT^{DTR}\text{-FST}$.

Proof. Consider the tree transformation $\rho = \{(\sigma(r, t), a) \mid r, t \in \tilde{K}_0\}$, which can be induced by a d_st^{2dtl}-fst. Similarly to the proof of Theorem 4.3, using Lemma 4.7, one can show that $\rho \notin DT^{DTR}\text{-FST}$. □

The correctness of the diagram of Fig. 3 immediately follows from (i), (ii), (iii), (iv) and (v).

Theorem 4.10. *The diagram of Fig.* 3 *is the inclusion diagram of the poset* $\langle \{DT^R\text{-FST}, DT^{2NTL}\text{-FST}, DT^{2UTL}\text{-FST}, DT^{2DTL}\text{-FST}, DT^{DTR}\text{-FST}, \text{FTA} \circ \text{DT-FST}, D_sT^R\text{-FST}, D_sT^{2NTL}\text{-FST}, D_sT^{2UTL}\text{-FST}, D_sT^{2DTL}\text{-FST}, D_sT^{DTR}\text{-FST}, \text{DT-FST}\}, \subseteq \rangle$.

Proof. The formulae (i), (ii), (iii), (iv) and (v) imply that all inclusions shown are proper and that all unrelated classes are incomparable. □

5. Composition results

In this section we investigate the composition powers of top-down tree transformation classes with some type of two-way tree walking look-ahead. In the strongly deterministic and deterministic cases the second powers properly contain the first

powers and equal the third powers while in the nondeterministic case we obtain proper hierarchies.

First we study the strongly deterministic transformation classes.

Theorem 5.1. *For any* $Y \in \{2DTL, 2NTL, 2UTL\}$, $D_sT^Y\text{-FST} \subset (D_sT^Y\text{-FST})^2 = (D_sT^Y\text{-FST})^3 = FTA \circ DT\text{-FST}$.

Proof. $D_sT^Y\text{-FST} \subset FTA \circ DT\text{-FST}$ by Theorem 4.10. Thus,

$$\begin{aligned}(D_sT^Y\text{-FST})^3 &\subseteq (FTA \circ DT\text{-FST})^3\\ &= FTA \circ DT\text{-FST} \quad \text{(by Lemma 4.3 and Theorem 6.2 of [12])}\\ &= DT\text{-FST} \circ D_sT^Y\text{-FST} \quad \text{(by Theorem 3.10)}\\ &\subseteq (D_sT^Y\text{-FST})^2. \quad \square\end{aligned}$$

Now we turn to the deterministic case.

Theorem 5.2. *For each* $Y \in \{2DTL, 2NTL, 2UTL\}$, $DT^Y\text{-FST} \subset (DT^Y\text{-FST})^2 = (DT^Y\text{-FST})^3 = DT^R\text{-FST}$.

Proof. By Theorem 4.10, $DT^Y\text{-FST} \subset DT^R\text{-FST}$. Moreover,

$$\begin{aligned}DT^R\text{-FST} &= DT\text{-FST} \circ DT^Y\text{-FST} \text{ by Theorem 3.10,}\\ &\subseteq (DT^Y\text{-FST})^k \quad \text{for } k \geqslant 2,\\ &\subseteq (DT^R\text{-FST})^k\\ &= DT^R\text{-FST} \quad \text{(by Theorem 2.11 of [2]).} \quad \square\end{aligned}$$

Finally, we study the composition powers of nondeterministic tree transformation classes.

Theorem 5.3. *For every* $k \geqslant 1$ *and* $Y \in \{2DTL, 2NTL, 2UTL\}$, $(T^Y\text{-FST})^k \subset (T^Y\text{-FST})^{k+1}$.

Proof. Assume that $(T^Y\text{-FST})^k = (T^Y\text{-FST})^{k+1}$ for some $Y \in \{2DTL, 2NTL, 2UTL\}$ and $k \geqslant 1$. Then for every $n \geqslant k$, $(T^Y\text{-FST})^n = (T^Y\text{-FST})^k$. Using the fact that $T^R\text{-FST} \subseteq (T\text{-FST})^2$ (see Corollary 2.13 of [2]), we have $T\text{-FST} \subseteq T^Y\text{-FST} \subseteq T^R\text{-FST} \subseteq (T\text{-FST})^2$. Thus, for $n \geqslant k$, $(T^Y\text{-FST})^n \subseteq (T\text{-FST})^{2 \cdot n} \subseteq (T^Y\text{-FST})^{2 \cdot n} \subseteq (T\text{-FST})^{4 \cdot n} \subseteq (T^Y\text{-FST})^{4 \cdot n} = (T^Y\text{-FST})^n$. Thus, we have $(T\text{-FST})^{2 \cdot n} = (T\text{-FST})^{4 \cdot n}$, which contradicts Theorem 3.14 of [4]. $\square$

6. Conclusion

We have investigated top-down tree transducers with two-way tree walking look-ahead. These devices are natural restrictions of top-down tree transducers with regular look-ahead. We compared the transformational power of deterministic and strongly deterministic versions of top-down tree transducers with some type of two-way tree walking look-ahead by presenting the inclusion diagram for the tree transformation classes induced by them. We also studied composition powers of the above classes. In the strongly deterministic and deterministic cases the second powers properly contain the first powers and equal the third powers while in the nondeterministic case we obtain proper hierarchies.

Along this line of research we obtained new decompositions of the transformational system of Rounds and of the top-down tree transformation with regular look-ahead.

For further investigations many areas remain open:

(1) Show that $\text{DT-FST} \circ \text{DT}^{\text{DTR}}\text{-FST} = \text{D}_s\text{T}^{\text{DTR}}\text{-FST} \circ \text{DT}^{\text{DTR}}\text{-FST} = \text{DT}^{\text{DTR}}\text{-FST}$. Give another proof for (v) in Section 4 using this result, Theorem 3.10 and the noninclusion $\text{FTA} \circ \text{DT-FST} \not\subseteq \text{DT}^{\text{DTR}}\text{-FST}$.

(2) Give a finite presentation or a finite complete rewriting system for the compositions of those tree transformation classes that appear in Fig. 3. Similar research work was carried out in [11, 15, 16].

(3) For any modifier C taken from the set $\{2\text{DTL}, 2\text{NTL}, 2\text{UTL}\}$, iterate the look-ahead tree languages as follows. Let $C_0 = C$ and let, for $n \geq 1$, C_n be the class of tree languages recognizable by deterministic top-down tree automata with C_{n-1} look-ahead. (In [13] the iteration was started with $C_0 = \text{DTREC}$.) Do we obtain an infinite hierarchy in this way?

(4) The concept of look-ahead on arbitrary storage type was formalized as an operator on storage types, i.e. if S is a storage type, then S with look-ahead, denoted by S_{LA}, is one too (see [5, 8]). The storage type S_{LA} is obtained from S by adding special predicates, the so called look-ahead tests. They have the form $\langle q, \mathscr{A} \rangle$ where q is a nonterminal of a CF(S) transducer $\mathscr{A}$. The look-ahead test $\langle q, \mathscr{A} \rangle$ is true on a configuration c if and only if the transducer $\mathscr{A}$ can derive a terminal string from $q(c)$. Since the domain of a top-down tree-to-string transducer is a regular tree language (see [21]), it follows that the top-down tree transducer with regular look-ahead is the $\text{RT}(\text{TR}_{\text{LA}})$ transducer, where RT abbreviates the modifier regular tree and TR denotes the storage type tree (see [5, 8]).

Engelfriet presented the decomposition

$$\text{T}^{\text{R}}\text{-FST} = \text{T-FST} \circ \text{LH}$$

in Corollary 2.13 of [2], where LH denotes the class of tree transformations induced by linear homomorphisms. This result may be written in the form

$$\text{RT}(\text{TR}_{\text{LA}}) = \text{RT}(\text{TR}) \circ \text{LH}$$

as well.

Generalize the above result for arbitrary storage type S, i.e. show the equality

$$\mathrm{RT}(S_{\mathrm{LA}}) = \mathrm{RT}(S) \circ \mathrm{LH}.$$

(5) The notion of a top-down tree transducer with regular check was introduced in [12]. Introduce and study top-down tree transducers with some type of two-way tree walking check.

References

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

[2] J. Engelfriet, Top-down tree transducers with regular look-ahead, *Math. Systems Theory* **10** (1977) 289–303.

[3] J. Engelfriet, On tree transducers for partial functions, *Inform. Process. Lett.* **7** (1978) 170–172.

[4] J. Engelfriet, Three hierarchies of transducers, *Math. Systems Theory* **15** (1982) 95–125.

[5] J. Engelfriet, Context-free grammars with storage, Report 86–11, University of Leiden, 1986.

[6] J. Engelfriet, G. Rozenberg, and G. Slutzki, Tree transducers, L-systems, and two-way machines, *J. Comput. System Sci.* **20** (1980) 150–202.

[7] J. Engelfriet and H. Vogler, Macro tree transducers, *J. Comput. System Sci.* **31** (1985) 71–146.

[8] J. Engelfriet and H. Vogler, Pushdown machines for the macro tree transducer, *Theoret. Comput. Sci.* **42** (1986) 251–369.

[9] J. Engelfriet and H. Vogler, Look-ahead on pushdowns, *Inform. and Comput.* **73** (1987) 245–279.

[10] J. Engelfriet and H. Vogler, High level tree transducers and iterated pushdown tree transducers, *Acta Inform.* **26** (1988) 131–192.

[11] Z. Fülöp, A complete description for a monoid of deterministic bottom-up tree transformation classes, *Theoret. Comput. Sci.* **88** (1991) 253–268.

[12] Z. Fülöp and S. Vágvölgyi, Variants of top-down tree transducers with look-ahead, *Math. Systems Theory* **21** (1989) 125–145.

[13] Z. Fülöp and S. Vágvölgyi, Iterated deterministic top-down look-ahead, in: J. Csirik, J. Demetrovics and F. Gécseg, eds., *Proc. FCT '89*, Lecture Notes in Computer Science Vol. 380 (Springer, Berlin, 1989) 175–184.

[14] Z. Fülöp and S. Vágvölgyi, A characterization of irreducible sets modulo left-linear term rewriting systems by tree automata, *Fund. Inform.* **XIII** (1990) 211–226.

[15] Z. Fülöp and S. Vágvölgyi, A finite presentation for a monoid of tree transformation classes, in: *Proc. 2nd Conf. on Automata, Languages and Programming systems* (Karl Marx University of Economics, Budapest, 1988) 115–124.

[16] Z. Fülöp and S. Vágvölgyi, A complete rewriting system for a monoid of tree transformation classes, *Inform. and Comput.* **86** (1990) 195–212.

[17] Z. Fülöp and S. Vágvölgyi, Top-down tree transducers with deterministic top-down look-ahead, *Inform. Process. Lett.* **33** (1989/90) 3–5.

[18] F. Gécseg and M. Steinby, *Tree Automata* (Akadémiai Kiadó, Budapest, 1984).

[19] T. Kamimura, Tree automata and attribute grammars, *Inform. and Control* **57** (1983) 1–20.

[20] T. Kamimura and G. Slutzki, Parallel and two-way automata on directed ordered acyclic graphs, *Inform. and Control* **49** (1981) 10–51.

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

[22] G. Slutzki, Alternating tree automata, *Theoret. Comput. Sci.* **41** (1985) 305–318.

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