When Is a Bottom-Up Deterministic Tree Translation Top-Down Deterministic?

Sebastian Maneth
Universität Bremen, Germany
maneth@uni-bremen.de

Helmut Seidl
TU München, Germany
seidl@in.tum.de

Abstract

We consider two natural subclasses of deterministic top-down tree-to-tree transducers, namely, linear and uniform-copying transducers. For both classes we show that it is decidable whether the translation of a transducer *with* look-ahead can be realized by a transducer *without* look-ahead. The transducers constructed in this way, may still make use of *inspection*, i.e., have an additional tree automaton restricting the domain. We provide a second procedure which decides whether inspection can be removed and if so, constructs an equivalent transducer *without* inspection. The construction relies on a fixpoint algorithm that determines inspection requirements and on dedicated earliest normal forms for linear as well as uniform-copying transducers which can be constructed in polynomial time. As a consequence, equivalence of these transducers can be decided in polynomial time. Applying these results to deterministic bottom-up transducers, we obtain that it is decidable whether or not their translations can be realized by deterministic uniform-copying top-down transducers without look-ahead (but with inspection) – or without both look-ahead and inspection.

2012 ACM Subject Classification Theory of computation → Models of computation; Theory of computation → Formal languages and automata theory

Keywords and phrases Top-Down Tree Transducers, Earliest Transformation, Linear Transducers, Uniform-copying Transucers, Removal of Look-ahead, Removal of Inspection

Digital Object Identifier 10.4230/LIPIcs.ICALP.2020.134

Category Track B: Automata, Logic, Semantics, and Theory of Programming

1 Introduction

Even though top-down and bottom-up tree transducers are well-studied formalisms that were introduced already in the 1970's (by Rounds [13] and Thatcher [14] independently, and by Thatcher [15], respectively), some fundamental questions have remained open until today. A prominent example of such a question is: can we decide for a given deterministic bottom-up tree transducer whether or not its translation can be realized by a top-down tree transducer? We answer this question affirmatively, however, for a slight restriction on the considered top-down tree transducers: they must be *uniform-copying* (*uc*). This means that all copies of the same input subtree must be processed by the same state.

It is well-known that for every deterministic bottom-up tree transducer an equivalent deterministic top-down tree transducer can be constructed which, however, makes use of regular look-ahead [7]. That transducer indeed is *uc*. The question which we ask therefore is: can regular look-ahead in *uc* transducers be eliminated? In order to answer this question, we provide a canonical earliest normal form for *uc* (as well as for linear) deterministic top-down transducers with and without look-ahead. We prove that if an earliest such transducer A can be realized by such a transducer A' *without* look-ahead (but with input inspection),

47th International Colloquium on Automata, Languages, and Programming (ICALP 2020).
Editors: Artur Czumaj, Anuj Dawar, and Emanuela Merelli; Article No. 134; pp. 134:1–134:18

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany


then A must be synchronizing, twinning, and erasing. To understand the *synchronizing* property, consider a transducer with look-ahead that translates an input tree of the form $a(f(t_1, t_2)))$ into the tree $\langle a, f\rangle(t_1, t_2)$, where $\langle a, f\rangle$ is a binary output symbol. Clearly this translation *can* be done by a transducer without look-ahead: it outputs nothing at the root node, keeps the node label in its state, and at its child outputs the corresponding binary symbol. Now consider that $a(f(g(t_1), t_2)))$ is translated to $\langle a, g\rangle(f(t_1, t_2))$. Such a translation can*not* be realized by a transducer without look-ahead. The information about the g-node cannot be "synchronized" at the f-node because it comes after the output must be produced (contradicting the order of origins of output nodes [12], see also [10, 11]).

The *twinning* property is similar to the string case [5] (see also [2, 3]), but now for paths. To understand the *erasing* property, consider a transducer with the following rules.

$$\begin{array}{lcllcl} q_0(a(x_1 : h_e)) & \to & a(a(e)) & q_{\mathrm{id}}(a(x_1 : h_f)) & \to & a(q_{\mathrm{id}}(x_1)) \\ q_0(a(x_1 : h_f)) & \to & a(q_{\mathrm{id}}(x_1)) & q_{\mathrm{id}}(f) & \to & f \\ q_0(f) & \to & f & & & \end{array}$$

Here, input trees are of the form $a(\cdots a(e)\cdots)$ or $a(\cdots a(f)\cdots)$. The look-ahead automaton has two states h_e and h_f, indicating that the input tree is of the first form (e-leaf) or the second form (f-leaf). The transducer translates input trees of the first form to the fixed tree $a(a(e))$ and realizes the identity on trees of the second form. This translation can*not* be realized by a transducer without look-ahead. The *erasing* property demands that if an input path depends on two different look-ahead states h_1, h_2, where for h_1 a constant output tree is produced (viz. the tree $a(a(e))$), then for h_2 no output may be produced in any loop.

Given a *uc* transducer A with look-ahead that is synchronizing, twinning, and erasing we construct an equivalent *uc* transducer with *inspection* (if it exists), i.e., a *uc* transducer where the domain is given separately via some top-down deterministic tree automaton.

The *third highlight* of our contribution is a procedure that removes inspection (if possible). The idea here is quite different from what we have discussed until now. Let us consider an example. The domain automaton accepts trees of the form $f(t_1, t_2)$ where t_2 is an arbitrary binary tree (with internal nodes labeled f and leaves labeled a or b) and t_1 is a tree which has a left-most leaf labeled a and a right-most leaf labeled b. The transducer has this rule:

$$q_0(f(x_1, x_2)) \to f(f(b, b), q_{\mathrm{id}}(x_2)),$$

where state q_{id} realizes the identity. Does there exist an equivalent top-down tree transducer *without* inspection? As it turns out, the answer is "yes". The idea is that the output subtree $f(b, b)$ can be used to simulate inspection! These are the rules of an equivalent transducer without inspection:

$$\begin{array}{ccc} q_0(f(x_1, x_2)) \to f(q(x_1), q_{\mathrm{id}}(x_2)) & q_a(f(x_1, x_2)) \to q_a(x_1) & q_b(f(x_1, x_2)) \to q_b(x_2) \\ q(f(x_1, x_2)) \to f(q_a(x_1), q_b(x_2)) & q_a(a) \to b & q_b(b) \to b \end{array}$$

We show that it is decidable for a given top-down deterministic tree language, whether or not it can be simulated on a given output tree. The challenge now is that it may be necessary to *delay* outputting certain output subtrees, until rules are encountered which require these output trees for simulating their inspection needs. Similar as before, such delay is only possible along input paths and must stop when two input subtrees of an input node are processed. Using a fixpoint algorithm we are able to determine whether or not sufficiently large output subtrees can be made available in order to satisfy all inspection needs. The approach we have sketched here is quite different from earlier methods for look-ahead removal [9] that rely on *difference bounds*, i.e., the differences in the translation with respect

to different look-ahead states. In order to obtain an effective construction, however, so far a variety of technical restrictions had to be introduced – implying that even for *linear* deterministic top-down tree transducers, look-ahead removal remained an open problem. To the best of our knowledge, this paper is the first to present look-ahead removal for natural and known subclasses of top-down tree transducers.

2 Basics

Let Σ denote a ranked alphabet. Then Σ_k is the set of all symbols in Σ of rank k. As usual, we define the set $\mathcal{T}_\Sigma$ of all (finite) *trees* over Σ as the set of all strings $t = f(t_1, \ldots, t_k)$ where $f \in \Sigma_k$ for some $k \geq 0$ and $t_1, \ldots, t_k \in \mathcal{T}_\Sigma$. For convenience, we also write f for $f()$ if f is of rank 0. A subtree of that form is also called *leaf*. Let $X = \{x_i \mid i \in \mathbb{N}\}$ denote an infinite set of distinct *variables* which is disjoint from any other occurring ranked alphabet (be it the input or the output alphabet of a transducer). All elements of the set X are assumed to have rank 0. For a finite set $J \subseteq \mathbb{N}$ we denote by X_J the set of variables $\{x_j \mid j \in J\}$, and we write $\mathcal{T}_\Sigma(X_J)$ for the set of all trees t over $\Sigma \cup X_J$. E.g., $f(h(x_2), a) \in \mathcal{T}_\Sigma(\{x_2\})$ where $f \in \Sigma_2$, $h \in \Sigma_1$, and $a \in \Sigma_0$. Trees in $\mathcal{T}_\Sigma(X_J)$ are also called *patterns*. Of particular importance is the set of *unary* patterns $\mathcal{T}_\Sigma(\{x_1\}) = \mathcal{T}_\Sigma(x_1)$. This set forms a *free monoid* where the monoid operation "$\cdot$" is substitution into the variable x_1. The tree $t = f(h(x_1), g(a, h(x_1)))$, e.g., can be uniquely factored into $f(x_1, g(a, x_1))$ and $h(x_1)$. We thus write $f(h(x_1), g(a, h(x_1))) = f(x_1, g(a, x_1)) \cdot h(x_1)$. We also consider the set $\mathcal{C}_\Sigma \subseteq \mathcal{T}_\Sigma(x_1)$ of *contexts* over Σ which is the subset of unary patterns which contain exactly one occurrence of x_1. Technically, this means that each context t either is equal to x_1, or is of the form $t = f(t_1, \ldots, t_k)$ for some $f \in \Sigma_k$ for some $k \geq 1$ and $1 \leq j, j' \leq k$ so that t_j is a context and $t_{j'} \in \mathcal{T}_\Sigma$ for all $j' \neq j$.

In the following, Σ and Δ denote fixed non-empty ranked alphabets of input and output symbols, respectively. In this paper, we consider deterministic top-down tree transducers with *uniform copying*, or *uc*-transducers for short. Intuitively, *uniform copying* means that each subtree of the input is processed at most once – while the produced output may be copied arbitrarily often. This restriction is trivially met by *linear* deterministic top-down transducers – but also by those that arise from the top-down simulation of deterministic *bottom-up* transducers by means of regular look-ahead. Here, we refrain from introducing transducers with *look-ahead* and *inspection* separately, as this would result in awkward duplication of almost identical definitions. Instead, we find it convenient to introduce yet another model, namely, deterministic transducers with (unambiguous) *advice* – which later can be instantiated either with top-down deterministic inspection (no interference with the computation of the transducer, only restriction to relevant input) or bottom-up deterministic look-ahead (interference with the computation as well as restriction to relevant input).

A *finite tree automaton over* Σ, (for short, *TA*) B consists of

1. a finite set H of states,
2. a subset $F \subseteq H$ of accepting states, and a transition relation $\delta \subseteq \bigcup_{k \geq 0} H \times \Sigma_k \times H^k$.

The computation of B on some input tree t can be represented by a tree in $\mathcal{T}_T$ where the ranked alphabet T consists of all transitions $\tau = \langle h, f, h_1 \ldots h_k \rangle \in \delta$ where the rank of τ equals the rank of the input symbol f. For $h \in H$, an h-computation ϕ for some $t = f(t_1, \ldots, t_k) \in \mathcal{T}_\Sigma$ is a tree $\phi = \tau(\phi_1, \ldots, \phi_k)$ where ϕ_i is a h_i-computation for t_i for all $i = 1, \ldots, k$. We write $h : t$ to indicate that there is an h-computation for t. We write $\mathsf{dom}_B(h) = \{t \in \mathcal{T}_\Sigma \mid h : t\}$ and define the set of trees accepted by B as $\mathcal{L}(B) = \{t \in \mathcal{T}_\Sigma \mid \exists h_0 \in F \text{ such that } h_0 : t\}$. An (h, h')-computation ϕ of B on some context $t \in \mathcal{C}_\Sigma$ is analogously defined as a context in $\mathcal{C}_T$ where h is the state at the root and h' is assumed at the variable leaf. We write $(h, h') : t$ to

indicate that a (h, h')-computation for t exists. In particular, $(h, h) : x_1$ for every state h of B. In general, we assume that every occurring TA B is *trim*, i.e., every transition of B occurs in some *accepting* computation, i.e., some h-computation with $h \in F$. We call B

- *bottom-up deterministic*, when for every tuple $(f, h_1 \dots h_k) \in \Sigma_k \times H^k$, there is at most one $h \in H$ so that $\langle h, f, h_1 \dots h_k \rangle \in \delta$;
- *top-down deterministic* when F consists of a single state only, and for every $k \geq 0$ and pair $(h, f) \in H \times \Sigma_k$, there is at most one tuple $(h_1 \dots h_k) \in H^k$ such that $\langle h, f, h_1 \dots h_k \rangle \in \delta$;
- *unambiguous*, if for each $t \in \mathcal{T}_\Sigma$, there is at most one $h \in F$ and at most one h-computation ϕ of B for t.

It is well-known that B is unambiguous whenever B is bottom-up deterministic, or top-down deterministic. In the following definition, we assume that the TA B is trim, unambiguous, and has a single accepting state h_0.

A *deterministic uniform-copying top-down tree transducer with advice over* Σ *and* Δ (for short, a $\mathsf{DT}^{\mathrm{A}}_{\mathrm{uc}}$ *transducer*, or *uc-transducer*, or a $\mathsf{DT}^{\mathrm{A}}_{\mathrm{uc}}$) A is a tuple (B, Q, ι, T_0, R) where

1. B is an unambiguous *advice* TA with a single final state h_0;
2. Q is a finite set of states together with a mapping $\iota : Q \to H$
3. T_0 is an axiom which is either a tree from $\mathcal{T}_\Delta$, or of the form $T_0 = p \cdot q_0(x_1)$ where $p \in \mathcal{T}_\Delta(x_1)$ with $q_0 \in Q$ and $\iota(q_0) = h_0$;
4. R is the set of rules such that for every transition $\langle h, f, h_1 \dots h_k \rangle$ of B and every state $q \in Q$ with $\iota(q) = h$, R contains one rule

$$q(f(x_1 : h_1, \dots, x_k : h_k)) \to T \tag{1}$$

where $f \in \Sigma_k$ for some $k \geq 0$, and $T = p\{x_j \mapsto q_j(x_j) \mid j \in J\}$ where $p \in T_\Delta(X_J)$ for some subset $J \subseteq \{1, \dots, k\}$ and for all $j \in J$, $q_j \in Q$ with $\iota(q_j) = h_j$ (thus $\{x_j \mapsto \dots\}$ is our notation of substituting leaves labeled x_j by the correspoding trees).

A *uc*-transducer is *linear*, if each input variable x_i occurs at most once in the right-hand side of every rule (we also say "$\mathsf{DT}^{\mathrm{A}}_{\mathrm{lin}}$ transducer").

We remark that we view the set Q of states of A as symbols of rank 1 distinct from all symbols in Δ. Given an h-computation ϕ of B on some input tree $t \in \mathcal{T}_\Sigma$, the rule (in R) at each node of t is uniquely determined by the state $q \in Q$ (with $\iota(q) = h$) at the root of t, and the transitions chosen in ϕ. Assume that $t = f(t_1, \dots, t_k)$, and $\phi = \tau(\phi_1, \dots, \phi_k)$ for $\tau = \langle h, f, h_1 \dots h_k \rangle$. A q-computation ψ of A on t with output s is given by $\rho(\sigma_1, \dots, \sigma_k)$ provided the following holds:

1. ρ is a rule of the form (1);
2. if $j \in J$, then σ_j is a q_j-computation for t_j with some output s_j and otherwise, $\sigma_j = \phi_j$;
3. $s = T\{x_j \mapsto s_j \mid j \in J\}$.

If such a q-computation for t exists with output s, we write $q : t \to s$.

As for TAs, we not only require the notion of a q-computation of A for input trees $t \in \mathcal{T}_\Sigma$ with output s, but also the notion of a (q, h)-computation of A on a context $t \in \mathcal{C}_\Sigma$ with output s. Let ϕ denote a $(\iota(q), h)$-computation of B. If $t = x_1$, then x_1 is a (q, h)-computation for x_1 with output $s = q(x_1)$ whenever $\iota(q) = h$. Assume that $t = f(t_1, \dots, t_k)$ and $t_j \in \mathcal{C}_\Sigma$ is a context, and let $\phi = \tau(\phi_1, \dots, \phi_k)$ denote the corresponding $(\iota(q), h)$-computation of B. Assume that the rule ρ is of the form (1). Then $\rho(\psi_1, \dots, \psi_k)$ is a (q, h)-computation for t with output $s = p\{x_j \mapsto s_j \mid j \in J\}$, if the following holds:

1. If $j' \notin J$, then $\psi_{j'} = \phi_{j'}$;
2. If $j' \in J \setminus \{j\}$, then $\psi_{j'}$ is a q_j-computation for $t_{j'}$ with output $s_{j'}$;
3. If $j' = j \in J$, then $\psi_{j'}$ is a (q_j, h)-computation for t_j with output s_j.

We remark that if s is non-ground, i.e., is not contained in $\mathcal{T}_\Delta$, then $s = s' \cdot q'(x_1)$ with $\iota(q') = h$. If such a (q, h)-computation exists, we write $(q, h) : t \to s$.

The *translation* of A is the partial mapping $[\![.]\!]_A : \mathcal{T}_\Sigma \to \mathcal{T}_\Delta$ defined by $[\![t]\!]_A = p_0$ if the axiom of A is the ground tree p_0 and $t \in \mathcal{L}(B)$, and $[\![t]\!]_A = p_0 \cdot s$ if the axiom is of the form $p_0 \cdot q_0(x_1)$ and $q_0 : t \to s$ holds.

Let us briefly list instances of *uc*-transducers with advice that are of interest here.

Transducers with Look-ahead. If the advice automaton B is chosen as bottom-up deterministic, the transducer A is a *uc*-transducer with regular look-ahead (for short, a $\mathsf{DT}^{\mathrm{R}}_{\mathrm{uc}}$ transducer, or a $\mathsf{DT}^{\mathrm{R}}_{\mathrm{uc}}$). We remark that the single axiom does not impose a severe restriction. Consider the generalization of a look-ahead automaton B with a non-singleton set F of accepting states and, accordingly, equip A with one dedicated axiom T_h for each accepting state $h \in F$. Instead, we may introduce a fresh unary input symbol \$ and define a new look-ahead automaton B' with a single fresh accepting state h_0 and a transducer A' with a single axiom such that $\mathcal{L}(B') = \{\$(t) \mid t \in \mathcal{L}(B)\}$ and $[\![\$(t)]\!]_{A'} = s$ holds iff $[\![t]\!]_A = s$ holds. In order to achieve that, we add to the set of transitions of B, all transitions $\langle h_0, \$, h\rangle$, $h \in F$, and likewise introduce a fresh axiom $q_0(x_1)$ for A' together with a fresh state q_0 where $\iota(q_0) = h_0$ and the rules $q_0(\$(x_1 : h)) \to T_h$ whenever $h \in F$ and T_h is the axiom of A for h.

Transducers with Inspection. If the advice automaton B is chosen as top-down deterministic, the transducer A can be considered as a *uc*-transducer with inspection automaton B (for short, a $\mathsf{DT}^{\mathrm{I}}_{\mathrm{uc}}$ transducer, or a $\mathsf{DT}^{\mathrm{I}}_{\mathrm{uc}}$). In this case, the state annotations h_i in the rule (1), can be dropped since these are obtained from $\iota(q)$ (q the current state of the transducer) and the input symbol f. A deterministic *bottom-up* tree transducer in the classical sense, e.g., as in [6] is obtained in our model as a $\mathsf{DT}^{\mathrm{A}}_{\mathrm{uc}}$ transducer where $Q = H$ and ι is the identity. A classical deterministic *top-down* tree transducer with uniform copying, on the other hand, is obtained as a $\mathsf{DT}^{\mathrm{I}}_{\mathrm{uc}}$ transducer where the inspection does not restrict the domain. This can be achieved, e.g., by setting $H = Q \cup \{\top\}$ for a fresh symbol $\top$ and $\iota(q) = q$. Moreover, for each $f \in \Sigma_k$, $\langle \top, f, \top^k\rangle \in \delta$ as well as $\langle q, f, q'_1 \dots q'_k\rangle \in \delta$ whenever there is a rule $q(f(x_1, \dots, x_k)) \to T$ such that for each $i = 1, \dots, k$, $q'_i = q_i$ if $q_i(x_i)$ occurs in T and $q'_i = \top$ otherwise.

We remark that in the same way, deterministic *linear* top-down tree transducers with look-ahead as well as deterministic linear bottom-up transducers and deterministic linear top-down transducers with inspection are instances of *linear* $\mathsf{DT}^{\mathrm{A}}_{\mathrm{uc}}$ transducers.

3 A Dedicated Earliest Normal Form for *uc*-Transducers with Advice

In this section we present the construction of earliest normal-forms for *uc*-transducers with advice. We also indicate how a corresponding construction is obtained for *linear* transducers. In the following, we fix some unambiguous TA B for advice. A construction of *earliest* top-down transducers has already been provided in [8] for top-down deterministic domain automata B and as well as in [4] for bottom-up deterministic B. Here, we are slightly more liberal by allowing *unambiguous* B to generalize both cases. The constructions from [8, 4], on the other hand, neither preserve linearity nor uniform-copying.

▶ **Example 1.** Consider a linear top-down transducer with the rules:

$$\begin{array}{lcllcl} q_0(g(x_1)) & \to & q_1(x_1) & q_0(a) & \to & a \\ q_1(f(x_1,x_2)) & \to & f(q_0(x_1), q_0(x_2)) & q_0(b) & \to & b \end{array}$$

and the axiom $q_0(x_1)$. The (canonical) earliest transducer constructed according to the methods in [8] has the same axiom $q_0(x_1)$, but the rules:

$$\begin{array}{lcllcl} q_0(g(x_1)) & \to & f(q_{11}(x_1), q_{12}(x_1)) & q_0(a) & \to & a \\ q_{11}(f(x_1,x_2)) & \to & q_0(x_1) & q_0(b) & \to & b \\ q_{12}(f(x_1,x_2)) & \to & q_0(x_2) & & & \end{array}$$

where no inspection automaton is required. The non-linearity in the first rule arises inevitably, because the output f-node is already determined at this point and therefore must be output for the transducer to be *earliest*.

We introduce *dedicated* constructions which allow to construct equivalent *canonical* transducers, but retain linearity or uniform-copying. It turns out that these normal forms can be obtained in polynomial time. Our key insight is that *dedicated* notions should be provided for the notion of *maximal common prefix* of a given set S of trees. In [8], a pattern such as $p = a(x_1, g(x_1))$ was used to represent the common part of trees in S (note that they do not use the symbol x_1, but the symbol $\top$ to denote "any tree"). This meant for an element $t \in S$ such as $t = a(b, g(c))$ that different occurrences of the symbol x_1 in p could correspond to not necessarily isomorphic subtrees of t, in the example, b and c, respectively. In that point, we will now be more restrictive and only allow *substitutions*, i.e., equal replacements of the occurrences of the single variable x_1 in patterns. For a distinction, we call such patterns *uniform*. Let us denote by $\mathcal{P}_\Delta$ the set $\mathcal{T}_\Delta \cup \mathcal{T}_\Delta(x_1) \cup \{\bot\}$ of all ground trees and unary patterns, extended with one specific element $\bot$. This set forms a *partial order* where for $t_1, t_2 \in \mathcal{P}_\Delta$, $t_1 \sqsubseteq t_2$ iff $t_1 = \bot$ or $t_1 = t_2\{x_1 \mapsto s\}$ for some $s \in \mathcal{T}_\Delta \cup \mathcal{T}_\Delta(x_1)$. In fact, $\mathcal{P}_\Delta$, partially ordered in this way, forms a *complete lattice* with finite ascending chains. In particular, the top-most element is x_1, and the binary least upper bound operation $\sqcup$ for incomparable elements $t_1, t_2 \neq \bot$, is given by $t_1 \sqcup t_2 = s$ where s is the maximal prefix such that $s\{x_1 \mapsto t_i'\} = t_i$ for suitable trees t_i' $(i = 1, 2)$.

We remark that uniform patterns may contain more than one occurrence of x_1 – all representing, though, isomorphic subtrees. Let $\mathcal{P}_\Delta^{(1)} \subseteq \mathcal{P}_\Delta$ denote the subset $\mathcal{T}_\Delta \cup \mathcal{C}_\Delta \cup \{\bot\}$ of all elements which either equal $\bot$ or contain at most one occurrence of x_1. Patterns in that set are also called *1-patterns*. For the induced partial ordering on $\mathcal{P}_\Delta^{(1)}$, we again obtain a complete lattice with finite ascending chains only. For a distinction, let us denote the least upper bound operation with respect to $\mathcal{P}_\Delta^{(1)}$ with $\sqcup^{(1)}$.

▶ **Example 2.** The difference between the two least upper bound operations becomes apparent when considering trees which differ in more than one subtree:

$$\begin{array}{lcl} f(g(a,a),c) \sqcup f(g(b,b),c) & = & f(g(x_1,x_1),c) \\ f(g(a,a),c) \sqcup^{(1)} f(g(b,b),c) & = & f(x_1,c). \end{array}$$

On the other hand, $f(g(a,a),c) \sqcup f(g(b,b),d) = f(g(a,a),c) \sqcup^{(1)} f(g(b,b),d) = x_1$. We remark that the earliest construction in [8] would return $f(g(x_1,x_1),x_1)$ in the latter case – implying that the place holder x_1 no longer represents *isomorphic* subtrees.

For $q \in Q$, let

$$\mathsf{pref}_A(q) = \bigsqcup\{s \in \mathcal{T}_\Delta \mid \exists t \in \mathcal{T}_\Sigma.\, q : t \to s\} \text{ and } \mathsf{pref}_A^{(1)}(q) = \bigsqcup{}^{(1)}\{s \in \mathcal{T}_\Delta \mid \exists t \in \mathcal{T}_\Sigma.\, q : t \to s\}$$

In the following, we show that $\mathsf{pref}_A : Q \to \mathcal{P}_\Delta$ as the least solutions of the set $\mathcal{C}_A$ of constraints. The case of $\mathsf{pref}_A^{(1)}$ is analogous. The set $\mathcal{C}_A$ consists of one constraint $c(\rho)$ for each rule ρ of A. Assume that $\tau \equiv q(f(\ldots)) \to T$ of A where $T = p\{x_j \mapsto q_j(x_j) \mid j \in J\}$ for some $p \in \mathcal{T}_\Delta(X_J)$ and suitable $q_j \in Q$. Then the constraint $c(\tau)$ is given by

$$\sigma(q)^\sharp \quad \sqsupseteq \quad [\![T]\!]^\sharp \sigma^\sharp \tag{2}$$

where the value $[\![T]\!]^\sharp\sigma^\sharp$ returns $\bot$ if for any state $q_j, j \in J$, $\sigma^\sharp(q_j) = \bot$. Otherwise, assume that p' is obtained from p and $\sigma^\sharp$ by replacing $q_j(x_j)$ with $\sigma^\sharp(q_j) \cdot x_j$ where $j \in J$ and with $\sigma^\sharp(q_j)$ whenever $\sigma^\sharp(q_j)$ is ground. If p' is ground, we set $[\![T]\!]^\sharp\sigma^\sharp = p'$. If p' contains occurrences of x_j for a single $j \in J$, i.e., is of the form $p' = p'' \cdot x_j$ for some $p'' \in \mathcal{T}_\Delta(x_1)$, then $[\![T]\!]^\sharp\sigma^\sharp = p''$. Otherwise, i.e., if p' contains occurrences of more than one variable, then $[\![T]\!]^\sharp\sigma^\sharp = u$ where $u \in \mathcal{T}_\Delta(x_1)$ is the maximal prefix such that $p' = u \cdot p''$ for some p'' containing all x_j, i.e., u has maximal size with this property and thus is *least* with respect to the ordering on patterns.

▶ **Example 3.** Let $T = f(g(q_2(x_2), a), g(q_2(x_2), q_1(x_1))$, and thus $p = f(g(x_2, a), g(x_2, x_1))$. Then for $\sigma^\sharp = \{q_1 \mapsto a, q_2 \mapsto h(x_1)\}$, we have that $p' = f(g(h(x_2), a), g(h(x_2), a))$, and thus, $[\![T]\!]^\sharp\sigma^\sharp = f(x_1, x_1) \cdot g(x_1, a) \cdot h(x_1) = f(g(h(x_1), a), g(h(x_1), a))$.

We remark that each right-hand side of a constraint in $\mathcal{C}_A$ represents a function which is *distributive* in each argument, i.e., commutes with the binary operator $\sqcup$ in each accessed argument $\sigma^\sharp(q_j)$. Recall that any distributive function is also monotonic. Since the partial ordering on $\mathcal{P}_\Delta$ and likewise on $\mathcal{P}_\Delta^{(1)}$ are complete lattices with finite ascending chains, the constraint system (2) as well as the respective system for linear transducers and 1-patterns, has a least solution. We thus obtain:

▶ **Lemma 4.** *Let $\mathit{pref}_A(q), q \in Q$, denote the least solution of the set of constraints (2) over the complete lattice $\mathcal{P}_\Delta$ ($\mathcal{P}_\Delta^{(1)}$). Then for every $a \in Q$,*

$$\mathit{pref}_A(q) = \bigsqcup\{s \mid \exists t \in \mathcal{T}_\Sigma.\ q : t \to s\} \tag{3}$$

holds. Moreover, this least solution can be computed in polynomial time.

Proof. Recall that by our assumption, the advice automaton is trim. Therefore, according to our construction, there is a q-computation for every state $q \in Q$, i.e., $\mathsf{pref}_A(q) \neq \bot$ for each $q \in Q$. The equality in equation (3) then is due to the fixpoint transfer lemma [1]. More explicitly, let $X_q^{(i)}$ denote the ith iterate of the fixpoint iteration for the constraint system for $i \geq 0$. By induction on i, it can be verified that $X_q^{(i)}$ equals the maximal common prefix of all s such that $q : t \to s$ for trees $t \in \mathcal{T}_\Sigma$ of depth less than i. Thereby, the prefixes $X_q^{(i+1)}$ can be determined from the preixes $X_q^{(i)}$ in polynomial time. This is obvious for *linear* transducers A. When A is *uniform copying*, and general uniform patterns are used, polynomial time can be obtained when trees are represented as *dags* where isomorphic subtrees are represented only once. Since the number of iterations required for reaching the least fixpoint of the constraint system is bounded by the size of the transducer A, the overall complexity statement follows. ◀

Now let A denote some *uc*-transducer (*linear* transducer) A with advice and a non-empty set of states. In particular, the axiom of A contains an occurrence of some state q_0 (with $\iota(q_0) = h_0$). We call A an earliest *uc*-transducer (*linear* transducer), if $\mathsf{pref}_A(q) = x_1$ ($\mathsf{pref}_A^{(1)}(q) = x_1$) for all states q of A. If this is not yet the case, we construct a transducer A' of the same kind as A as follows where we only present the construction for *uc* transducers (the linear case is analogous). The set Q' of states of A' is obtained from the set Q of states of A by $Q' = \{q \in Q \mid \mathsf{pref}_A(q) \notin \mathcal{T}_\Delta\}$. Assume that the axiom T_0 of A equals $T_0 = p \cdot q_0(x_1)$. If $\mathsf{pref}_A(q_0) = s \in \mathcal{T}_\Delta$, then the axiom T_0' of A' is given by $T_0' = p \cdot s$. Otherwise, the new axiom T_0' is given by $T_0' = p \cdot \mathsf{pref}_A(q_0) \cdot q_0(x_1)$. Now assume that $q \in Q'$, and $\mathsf{pref}_A(q) = u$. Then for each rule $q(f(x_1 : h_1, \ldots, x_k : h_k)) \to p\{x_j \mapsto q_j(x_j) \mid j \in J\}$ of A, A' has a rule $q(f(x_1 : h_1, \ldots, x_k : h_k) \to T'$ where T' is defined as follows. For $j \in J$, let $s_j = \mathsf{pref}_A(q_j)$ if $\mathsf{pref}_A(q_j) \in \mathcal{T}_\Delta$, and $s_j = u_j \cdot q_j(x_j)$ if $\mathsf{pref}_A(q_j) = u_j \in \mathcal{T}_\Delta(x_1)$. Then u must be a prefix of $p\{x_j \mapsto s_j \mid j \in J\}$, and we choose T' such that $p\{x_j \mapsto s_j \mid j \in J\} = u \cdot T'$ holds.

▶ **Lemma 5.** *Assume that A is a DT^A_{uc} and A' the DT^A_{uc} as constructed above. Then,*

1. *For each state q of A' with $u = \mathsf{pref}_A(q)$, it holds that*
   a. *If $q : t \to s$ holds for A, then $q : t \to s'$ holds for A' for some $s' \in \mathcal{T}_\Delta$ so that $s = u \cdot s'$, and vice versa,*
   b. *If $q : t \to s'$ holds for A', then $q : t \to u \cdot s'$ holds for A.*
   c. *A and A' are equivalent where $\mathsf{pref}_{A'}(q) = x_1$ holds for all states q of A'.*
2. *A' can be constructed from A in polynomial time.*

The analogous properties hold for DT^A_{lin} transducers.

Due to this lemma, we obtain that for each *uc*-transducer (*linear* transducer) an equivalent earliest transducer can be constructed in polynomial time. In fact that transducer can further be *minimized*. For that, we define $\equiv$ as the coarsest equivalence relation on states such that $q \equiv q'$ implies that $\iota(q) = \iota(q')$, and for each input symbol $f \in \Sigma$ there is a rule $q(f(x_1 : h_1, \ldots, x_k : h_k)) \to T$ iff there is a rule $q'(f(x_1 : h_1, \ldots, x_k : h_k)) \to T'$ such that $T = p\{x_j \mapsto q_j(x_j) \mid j \in J\}$ and $T' = p\{x_j \mapsto q'_j(x_j) \mid x_j \in X_J\}$ for some common pattern $p \in \mathcal{T}_\Delta(X_J)$ and states $q_j, q'_j \in Q$ such that for all $j \in J$, $q_j \equiv q'_j$ holds.

▶ **Lemma 6.** *Let A be an earliest* uc*-transducer (*linear *transducer) and $\equiv$ the equivalence relation as defined above. Then the following holds:*

1. *$q \equiv q'$ iff for all $[\![q]\!]_A = [\![q']\!]_A$;*
2. *$\equiv$ can be constructed in polynomial time.*

The proof of Lemma 6 follows closely the corresponding proof of Theorem 13 of [8]. Putting Lemmas 5 and 6 together, we obtain:

▶ **Theorem 7.** *For each DT^A_{uc} (DT^A_{lin}) transducer A, a unique canonical earliest DT^A_{uc} (DT^A_{lin}) transducer A' can be constructed such that (1) A' has at most as many states as A, (2) A' is equivalent to A, and (3) A' can be constructed in polynomial time.*

4 How to Remove Look-ahead

In the following, we assume that we are given a deterministic top-down tree transducer A with regular look-ahead. By Theorem 7, we may assume that A is earliest. Our goal is to decide whether the translation of A can be realized by a deterministic top-down tree transducer *without* look-ahead (but with inspection). A necessary condition for the latter is that the domain of the given translation can be accepted by a top-down deterministic automaton. By assumption, the domain of the translation of A is given by the set $\mathcal{L}(B)$ of all trees accepted by B. If the translation can be realized by a *uc*-transducer with inspection only, $\mathcal{L}(B) = \mathcal{L}(B')$ for some top-down deterministic automaton B'. One such B' can be obtained by means of the *powerset* construction. The set H' of states of B' are subsets of states of B where in particular, $\{h_0\} \in H'$ is the accepting state. Moreover, if $S \subseteq H$ is a state in H', then for every input symbol $f \in \Sigma_k$ and every $j \in \{1, \ldots, k\}$,

$$S_j = \{h_j \in H \mid \exists h \in S,\ h_1, \ldots, h_{j-1}, h_{j+1}, \ldots, h_k \in H.\, \langle h, f, h_1 \ldots h_k\rangle \in \delta\} \ \in\ H'$$

and $\langle S, f, S_1 \ldots S_k\rangle$ is in the transition relation of B'. As B is assumed to be trim, the automaton B' constructed in this way, is trim as well. Checking whether or not $\mathcal{L}(B) = \mathcal{L}(B')$ is decidable. In fact, the two automata are equivalent iff for each transition $\langle S, f, S_1 \ldots S_k\rangle$ constructed for B', and every tuple of states $(h_1, \ldots, h_k) \in S_1 \times \ldots \times S_k$ there is some $h \in S$ such that $\langle h, f, h_1 \ldots h_k\rangle$ is a transition of B.

Let us thus assume that B and B' are equivalent. The following construction is given for uniform copying transducers (the construction for linear transducers is analogous). Let us assume that the earliest $\mathsf{DT}^{\mathrm{R}}_{\mathrm{uc}}$ transducer A is canonical and equivalent to a $\mathsf{DT}^{\mathrm{I}}_{\mathrm{uc}}$ transducer. As the case where the axiom T_0 of A is ground, is trivial, we now assume that the axiom is non-ground, i.e., $T_0 = s_0 \cdot q_0(x_1)$ with $\iota(q_0) = h_0$. Then we construct a $\mathsf{DT}^{\mathrm{I}}_{\mathrm{uc}}$ transducer A' as follows. The states of A' are given by $\langle\rho\rangle$ for mappings ρ which assign to the states h in some set $S \in H'$, trees $\rho(h)$ which are either in $\mathcal{T}_\Delta$ or of the form $\rho(h) = s \cdot q(x_1)$ where $s \in \mathcal{T}_\Delta(x_1)$ and $q \in Q$ is a state of A with $\iota(q) = h$. For that mapping ρ, we define $\iota'(\langle\rho\rangle) = S$. The axiom of A' is given by $T'_0 = s_0 \cdot \langle\{h_0 \mapsto q_0(x_1)\}\rangle$.

Assume now that $\langle\rho\rangle$ is a state of A' with domain S. Consider some input symbol $f \in \Sigma$ of rank $k \geq 0$ where $\langle S, f, S_1 \dots S_k\rangle$ is a transition of B'. Let p' denote a pattern in $\mathcal{T}_\Delta(X_{J'})$ for some $J' \subseteq \{1, \dots, k\}$. Let ρ_i, $i = 1, \dots, k$ be mappings with domains S_i such that for $h \in S$ and $\langle h, f, h_1 \dots h_k\rangle \in \delta$,

1. If $\rho(h) \in \mathcal{T}_\Delta$, then $h_j \in S_j$ for all $j \in \{1, \dots, k\}$, and $\rho(h) = p'\{x_j \mapsto \rho_j(h_j) \mid j \in J'\}$;
2. If $\rho(h) = s \cdot q(x_1)$, and A has a rule of the form (1), then $J \subseteq J'$ and $s \cdot p = p'\{x_j \mapsto u_j \mid j \in J'\}$ where $u_j = \rho_j(h_j)$ if $\rho_j(h_j)$ is ground, and $u_j = \rho_j(h_j) \cdot x_j$ otherwise.
3. For each $j \in J'$, the mapping ρ_j is (up to states in Q) prefix-free, i.e., the longest common prefix of $\rho_j(h), h \in S_j$, in $\mathcal{T}_\Delta(x_1)$ is x_1.

If A is equivalent to some $\mathsf{DT}_{\mathrm{uc}}$ transducer, p' and ρ_j with these properties must always exist, and then are uniquely defined.

▶ **Example 8.** Assume that $\rho = \{h_1 \mapsto f(a, g(c)), h_2 \mapsto f(b, g(c)), h_3 \mapsto f(a, b), h_4 \mapsto f(b, b), h_5 \mapsto c\}$ and for the binary input symbol f, B has the transitions $\langle h_1, f, h_a h_c\rangle, \langle h_2, f, h_b h_c\rangle, \langle h_3, f, h_a h_b\rangle, \langle h_4, f, h_b h_b\rangle$ while there is no transition for f resulting in state h_5. By comparing the outputs for h_1 and h_2, we identify the subtrees a and b whose outputs cannot be decided depending on the input symbol f alone, but require information about the first child of f. Likewise, by comparing the outputs for h_3 and h_4, we identify the corresponding subtrees $g(a)$ and b whose outputs can be discriminated only depending on the second child of f in the input. Accordingly, the pattern is given by $p' = f(x_1, x_2)$ where $\rho_1 = \{h_a \mapsto a, h_b \mapsto b\}$ and $\rho_2 = \{h_c \mapsto g(c), h_b \mapsto b\}$.

Example 8 illustrates the perhaps most complicated case, namely, when all outputs stored in ρ are ground. Given that J', p' and $\rho_j, j \in J'$, with the given properties exist, we add to A' the states $\langle\rho_j\rangle, j \in J'$, together with the rule

$$\langle\rho\rangle(f(x_1 : S_1, \dots, x_k : S_k)) \to p'\{x_j \mapsto \langle\rho_j\rangle(x_j) \mid j \in J'\}. \tag{4}$$

The resulting transducer is a $\mathsf{DT}^{\mathrm{I}}_{\mathrm{uc}}$ transducer A' which is equivalent to A. Now assume that the construction successfully terminates. The following two lemmas summarize the properties of the resulting transducer A'.

▶ **Lemma 9.** *Consider a state $\langle\rho\rangle$ of A' for some mapping ρ with domain S, $h \in S$ and $t \in \mathcal{T}_\Sigma$ with $h : t$.*

1. *If $\rho(h) = u$ is ground, then $\langle\rho\rangle : t \to u$;*
2. *If $\rho(h) = u \cdot q(x_1)$ for some $u \in \mathcal{T}_\Delta(x_1)$, and $q : t \to s$, then $\langle\rho\rangle : t \to u \cdot s$.*

▶ **Lemma 10.** *Assume that $t \in \mathcal{C}_\Sigma$ is a context where $(S_0, S) : t$ holds in B' for $S_0 = \{h_0\}$.*

1. *$S = \{h \in H \mid \exists s_h.\, (q_0, h) : t \to s_h\}$, i.e., S is the set of all h such that there is a (q_0, h)-computation of A for t;*
2. *Assume that for each $h \in S$, $(q_0, h) : t \to s_h$. Then $(\langle\rho_0\rangle, S) : t \to s$ for $\rho_0 = \{h_0 \mapsto q_0(x_1)\}$ so that the following holds:*

- *If s is ground, then s_h is ground for each $h \in S$, and $s = s_h$ holds for each $h \in S$;*
- *If $s = s' \cdot \langle\rho\rangle(x_1)$, then ρ is a mapping with domain S, and for each $h \in S$,*
  - *$\rho(h)$ is ground iff s_h is ground where $s' \cdot \rho(h) = s_h$.*
  - *If $\rho(h) = u_h \cdot q(x_1)$ for some u_h, then $s' \cdot u_h \cdot q(x_1) = s_h$.*

The proof of these two lemmas is by induction on the structure of t, following the definition of A'. We conclude that, upon successful termination, A and A' are equivalent. It thus suffices to prove successful termination whenever A is equivalent to *some* $\mathsf{DT}_{\mathrm{uc}}$ transducer.

We introduce three properties. The $\mathsf{DT}^{\mathrm{R}}_{\mathrm{uc}}$ transducer A is called *synchronizing*, if for every input tree $t_0 \in \mathcal{C}_\Sigma$, input symbol $f \in \Sigma$ of rank $k \geq 0$ and look-ahead states h'_1, h'_2 so that $(q_0, h_i) : t \to s_i \cdot q_i(x_1)$ for $i = 1, 2$, the following holds. Let $q_i(f(x_1 : h_{i,1}, \ldots, x_k : h_{i,k})) \to T_i$ be rules of A according to (1) where both T_1 and T_2 are non-ground. Then one of the following two cases occurs.

Case 1. There are patterns $p_1, p_2 \in \mathcal{T}_\Delta(X_J)$ which agree in their sets of occurring variables, and factorizations $T_i = p_i\{x_j \mapsto u_{i,j} \mid j \in J\}$ for $i = 1, 2$ such that

- $s_1 \cdot p_1 = s_2 \cdot p_2$,
- for each $j \in J$, each $u_{i,j}$ is either ground, or is of the form $s' \cdot q'(x_j)$ for suitable s', q'.

Case 2. There is some j such that both $T_1 = s'_1 \cdot q'_1(x_j)$ and $T_2 = s'_2 \cdot q'_2(x_j)$ for suitable $s'_1, s'_2 \in \mathcal{T}_\Delta(x_1)$ and states $q'_1, q'_2 \in Q$.

Secondly, the $\mathsf{DT}^{\mathrm{R}}_{\mathrm{uc}}$ transducer A is called *twinning*, if the following holds for all states q_1, q_2 and contexts $t, t' \in \mathcal{C}_\Sigma$ such that $(q_0, \iota(q_i)) : t \to s_i \cdot q_i(x_1)$ and $(q_i, \iota(q_i)) : t' \to s'_i \cdot q_i(x_1)$.

- Either $s'_1 = s'_2 = x_1$,
- or there are trees $u, v \in \mathcal{T}_\Delta(x_1)$ such that $s_1 = s_2 \cdot w$, $s'_1 = v \cdot w$ and $s'_2 = w \cdot v$ or vice versa, $s_2 = s_1 \cdot w$, $s'_2 = v \cdot w$ and $s'_1 = w \cdot v$ for suitable $w, v \in \mathcal{T}_\Delta(x_1)$.

Finally, the $\mathsf{DT}^{\mathrm{R}}_{\mathrm{uc}}$ transducer A is called *erasing*, if the following holds for all input trees $t, t' \in \mathcal{C}_\Sigma$ and states $h_1, h_2 \in H$. Assume that $(q_0, h_i) : t \to s_i$ for $i = 1, 2$ where s_1 is of the form $s'_1 \cdot q(x_1)$ (thus, $\iota(q) = h_1$) and s_2 is ground. Then $(q, h_1) : t' \to u \cdot q(x_1)$ for some $u \in \mathcal{T}_\Delta(x_1)$ and $(h_2, h_2) : t'$ implies that $u = x_1$.

The *variation* $\|t_1, t_2\|$ of $t_1, t_2 \in \mathcal{T}_\Delta(x_1)$ is the minimal depth of u_1, u_2 such that $t_i = t_0 \cdot u_i, i = 1, 2$ for a $t_0 \in \mathcal{T}_\Delta(x_1)$. The $\mathsf{DT}^{\mathrm{R}}_{\mathrm{uc}}$ transducer A has *bounded variation* if $\exists K \geq 0$ such that for every $t \in \mathcal{C}_\Sigma$ and $h_1, h_2 \in H$ with $(q_0, h_i) : t \to s_i$ for $i = 1, 2$, $\|s'_1, s'_2\| \leq K$ holds. – Assume that A is synchronizing, erasing and twinning. Then the outputs of any two computations for the same input tree, cannot not differ much. Intuitively, the variation is synchronized at branching rules, does not increase in monadic loops and may increase only marginally once one of the outputs is ground. Let us define the *size* $|A|$ of some $\mathsf{DT}^{\mathrm{A}}_{\mathrm{uc}}$ A as the sum of the sizes of all rules of A where the size of the rule (1) is $k + 1$ plus the number on nodes in the right-hand side. Altogether, we prove:

▶ **Lemma 11.** *Assume that the domain of A is top-down deterministic where the bottom-up deterministic look-ahead automaton B has $m \geq 1$ states.*

1. *If A is equivalent to some DT^{I}_{uc} A', then A is synchronizing, erasing and twinning.*
2. *If the DT^{R}_{uc} A is synchronizing, erasing and twinning, then A has bounded variation where the bound is given by $|A| \cdot (|A| + m)$.*

Proof. The proof that A then must be twinning follows along the same lines as for word transducers. Here, we only consider synchronization. Assume that A is equivalent to some $\mathsf{DT}^{\mathrm{I}}_{\mathrm{uc}}$ A', and $(\{h_0\}, S) : t$ for some $t \in \mathcal{C}_\Sigma$ and state S of B'. Then $(\bar{q}, S) : t \to s_0 \cdot \bar{q}(x_1)$ holds for the initial state $\bar{q}_0$ of A' and some state $\bar{q}$ of A' where $\iota(\bar{q}_0) = \{h_0\}$ and $\iota(\bar{q}) = S$. Assume that there is a transition $\bar{q}(f(\ldots)) \to T$ of A'. Consider any $h \in S$ so that $\langle h, f, h_1 \ldots h_k\rangle$ is

a transition of B, and assume that $(q_0, h) : t_0 \to s \cdot q(x_1)$ for some $s \in \mathcal{T}_\Delta(x_1)$. Since A is assumed to be earliest, we have that $s_0 \cdot u_0 = s$ for some $u_0 \in \mathcal{T}_\Delta(x_1)$. Moreover, there is a rule $q(f(x_1 : h_1, \ldots, x_k : h_k)) \to T'$ of A for f. We consider three cases.

1. T is ground. Then T' is ground with $s_0 \cdot T = s \cdot T' = s_0 \cdot u_0 \cdot T'$. Consequently by top-cancellation, $T = u_0 \cdot T'$ holds.
2. T contains occurrences of a single variable x_j only, i.e., is of the form $p \cdot \bar{q}'(x_j)$ for some state $\bar{q}'$ of A'. If T' is ground, then $s_0 \cdot p \cdot v = s \cdot T' = s_0 \cdot u \cdot T'$ for some $v \in \mathcal{T}_\Delta$, i.e., $p \cdot v = u_0 \cdot T'$. If T' in non-ground, it necessarily is of the form $T' = p' \cdot q'(x_j)$ where (again by top-cancellation) $p \cdot u' = u_0 \cdot p'$ holds for some $u' \in \mathcal{T}_\Delta(x_1)$.
3. T contains occurrences of variables $x_{j_1} \neq x_{j_2}$. Let J denote the set of indices j so that x_j occurs in T. Then $T = u_0 \cdot p\{x_j \mapsto \bar{q}_j(x_j) \mid j \in J\}$ holds for some $p \in \mathcal{T}_\Delta(X_J)$ so that $T' = p\{x_j \mapsto u_j \mid j \in J\}$ for some u_j which are either ground or of the form $s_j \cdot q_j(x_j)$ with $q_j \in Q$.

Now assume that we have $(q_0, h_i) : t_0 \to s_i \cdot q_i(x_1)$, and let $(\bar{q}_0, S) : t \to s_0 \cdot \bar{q}(x_1)$ denote the corresponding computation of the $\mathsf{DT}^{\mathrm{I}}_{\mathrm{uc}}$ A'. In particular, this means that $h_1, h_2 \in S$, and we can apply the observations listed above. Let $q_i(f_i(x_1 : h_{i,1}, \ldots, x_k : h_{i,k}) \to T_i, i = 1, 2$, be rules of A for q_i and f such that T_1, T_2 are both non-ground. Then there also must be a rule $\bar{q}(f(\ldots)) \to T$ of A' where T is not ground as well, i.e., the first of the three cases does not apply. Now, assume that the monadic second case of the synchronization property also does not apply. Then T contains at least two variables, and there are factorizations $T = u_i \cdot p_i$ for $i = 1, 2$ so that $T_i = p_i\tau_i$ for substitutions of τ_i mapping each x_j to some ground tree or expression $s_{i,j} \cdot q_{i,j}(x_j)$ where $s_i = s_0 \cdot u_i$. We conclude that

$$s_1 \cdot p_1 = s_0 \cdot u_1 \cdot p_1 = s_0 \cdot u_2 \cdot p_2 = s_2 \cdot p_2$$

holds. Finally, consider some j where $h_{1,j} = h_{2,j} = h'$ for some h'. Assume for a contradiction that $\tau_1(h') \neq \tau_2(h')$, and consider any input tree t' so that $h' : t'$. Since $h_1 \neq h_2$ holds, some $j' \neq j$ exists so that $h_{1,j'} \neq h_{2,j'}$ holds. In particular, this means that the right-hand side T of ρ for f contains an occurrence of $\bar{q}'(x_j)$ for some state $\bar{q}'$ of A'. Then $\bar{q}' : t' \to s'$ for some ground tree s'. If $\tau_i(h) = v_i \cdot q'_i(x_j)$ for states q'_i of A, then $q'_i : t' \to s'_i$ with $s' = v_i \cdot s'_i$. Now since A is earliest, it follows that $v_1 = v_2$ must hold while q'_1 and q'_2 are equivalent, as their outputs coincide for each input. Since A is canonical, this further means that $q'_1 = q'_2$. Likewise, if $\tau_1(h') = s'_1$ is ground, then necessarily $\tau_2(h')$ also must be ground and coincide. Thus, the synchronization property follows.

It remains to prove the second assertion of lemma, namely, that every $\mathsf{DT}^{\mathrm{R}}_{\mathrm{uc}}$ A which is synchronizing, erasing and twinning, has a variation bounded by $|A| \cdot (|A| + m)$. Let B' denote the top-down deterministic automaton accepting the domain of A, and assume for a contradiction that $t \in \mathcal{C}_\Sigma$ is a context with a minimal number of nodes violating the claim of the lemma. Let S denote the state of B' such that $(\{h_0\}, S) : t$ holds. For $h_1, h_2 \in S$, assume that $(h_f, h_\nu) : t \to s_\nu$, $\nu = 1, 2$, holds for A. Assume that $t = t_1 \cdot \ldots \cdot t_m$ where $t_i = f_i(u_{i,1}, \ldots, u_{i,j_i-1}, x_1, u_{i,j_i+1}, \ldots, u_{i,k_i})$ for some some $1 \leq j_i \leq k_i$, some $f_i \in \Sigma_{k_i}$ and ground trees $u_{i,j'}, j' \neq j_i$. Let $h_{i,0}, \ldots, h_{i,m}$ states of B so that $(h_{i-1}, h_i) : t_i$ holds for $i = 1, \ldots, m$. Clearly, if $m = 0$, $s_1 = s_2 = x_1$ and the assertion holds. First, we consider the case that there is a maximal $m' \leq m$ where $(q_0, h'_m) : t_1 \ldots t_{m'-1} \to s'_i \cdot q_i(x_0)$ holds such that $(q_i, h_{i,m'}) : p_i\tau_i$ for some $p_1, p_2 \in \mathcal{T}_\Delta(X_J)$ and substitutions τ_i, where

- $s'_1 \cdot p_1 = s'_2 \cdot p_2$;
- $\tau_1(x_{j'}) = \tau_2(x_j) \in \mathcal{T}_\Delta$ for $j' \neq j_{m'}$;
- $\tau_i(x_{j_{m'}})$ is of the form $v'_i \cdot q'_i(x_1)$ where $v'_i \cdot q'_i(x_{j_{m'}})$ is a subtree of some right-hand side of A.

If $m' = m$, then obviously, the claim of the lemma holds. Therefore, either there is no such m' or $m' < m$. Assume that u is obtained from p_1 by substituting x_j with $\tau_1(x_j)$ for all $j \neq j_{m'}$. Thus, $s' = s'_1 \cdot u$ is a common prefix of s_1 and s_2. In case that the properties above are never satisfied, we set $m' = 0$, $s' = x_1$, and let $v'_i \cdot q'_i(x_1)$ equal $q_i(x_1)$.

Assume that $m'' \geq m'$ is chosen maximal so that $(q'_i, h_{i,m''}) : t_{m'+1} \dots t_{m''} \to s''_i \cdot q''_i(x_i)$, i.e., both outputs are non-ground. The variation for the context $t_1 \cdot \ldots \cdot t_{m''}$ then is given by $\|v'_1 \cdot s''_1 \cdot q''_1(x_1), v'_2 \cdot s''_2 \cdot q''_2(x_1)\|$. We claim that this variation is bounded by $|A|^2$. Assume for a contradiction that this were not the case. Then due to the synchronization property and the choice of m', this implies that $m'' - m' > n^2$ where n is the number of states of A. Therefore, at least one pair of states occurs at least twice. But then, due to the twinning property, the same variation is attained with a smaller context – contradicting the minimality of t.

To continue with our argument, we conclude that m'' must be less than m. By the definition of m'' this means that one of the right-hand sides chosen for q''_i and $f_{m''+1}$ must be ground. W.l.o.g., assume that this is the right-hand side T_1 for q''_1. But then due to the erasing property, the depth of the output for $t_{m''+1} \cdot \ldots \cdot t_m$ is bounded by $|A| \cdot m$. Altogether therefore, the variation is bounded by $|A|^2 + |A| \cdot m = |A| \cdot (|A| + m)$ – in contradiction to our assumption. This concludes the proof. ◀

In summary, we obtain:

▶ **Theorem 12.** *Let A be a DT^R_{uc}. It is decidable whether or not the translation of A can be realized by a DT^I_{uc}, and if so an equivalent DT^I_{uc} A' can be constructed.*

A corresponding theorem also holds for $\mathsf{DT}^{\mathrm{R}}_{\mathrm{lin}}$ transducers.

5 How to Inspect Top-Down Deterministic Languages

Now consider a $\mathsf{DT}^{\mathrm{I}}_{\mathrm{uc}}$ A with underlying top-down deterministic automaton B which is assumed to be canonical earliest. For the following, we denote the unique state h of B with $\mathsf{dom}_B(h) = \mathcal{T}_\Sigma$ (given that there is such a state), by $\top$. The $\mathsf{DT}^{\mathrm{I}}_{\mathrm{uc}}$ ($\mathsf{DT}^{\mathrm{I}}_{\mathrm{lin}}$) A is *without inspection* (denoted by $\mathsf{DT}_{\mathrm{uc}}$ and $\mathsf{DT}_{\mathrm{lin}}$) if for every rule $q(f(x_1 : h_1, \dots, x_k : h_k) \to T$ of A, $h_j = \top$ whenever x_j does not occur in T. When B does not have a state $\top$, then A is without inspection only if the right-hand side of every rule of A contains all variables x_j occurring in its left-hand side, i.e., A is *non-deleting*. Note that a $\mathsf{DT}_{\mathrm{uc}}$ can easily be changed in such a way that no advice automaton is present at all (and still the same translation is realized).

Consider a fixed output tree $s \in \mathcal{T}_\Delta$. A language $L \subseteq \mathcal{T}_\Sigma$ is called $\mathsf{DT}_{\mathrm{uc}}$ ($\mathsf{DT}_{\mathrm{lin}}$) *output recognizable by s* iff there is a $\mathsf{DT}_{\mathrm{uc}}$ ($\mathsf{DT}_{\mathrm{lin}}$) A such that $[\![t]\!]_A$ is defined iff $t \in L$, where $[\![t]\!]_A = s$ for all $t \in L$. It turns out that a language is output recognizable via some $\mathsf{DT}_{\mathrm{uc}}$ iff it is output recognizable via some $\mathsf{DT}_{\mathrm{lin}}$. Hence, we drop the qualification. The language L is called *output recognizable* (without further mentioning of an output tree) if L is out recognizable by some $s \in \mathcal{T}_\Delta$. Assume that the language L is accepted by the trim top-down deterministic TA B. Then it can be decided in *polynomial time* whether or not L is output recognizable, and if so, whether or not L is output recognizable by a particular given tree s.

▶ **Lemma 13.** *$\mathcal{L}(B)$ is output recognizable iff for every strongly connected component H' of the transition relation of B, every transition $\langle h', f, h_1 \dots h_k\rangle$ of B, and i with $h', h_i \in H'$, it holds that $h_j = \top$ for all $j \neq i$.*

Let s denote any output tree in $\mathcal{T}_\Delta$ and h a state of B. Then $\mathsf{dom}_B(h)$ is output recognizable by s if for every transition $\langle h, f, h_1 \dots h_k\rangle$ with subsequence $h_{i_1} \dots h_{i_r}$ of states different from $\top$, there is a pattern $s' \in \mathcal{T}_\Delta(X_r)$ such that $s = s'\{x_j \mapsto s_j \mid j = 1, \dots, r\}$ and $\mathsf{dom}_B(h_{i_j})$ is

output recognizable by s_j for all $j = 1, \ldots, r$. In case that h and one of the h_j are contained in the same strongly connected component of B, then r must equal 1. Accordingly, then s' can be chosen as x_1. We further remark that the pattern s' can be chosen to be *linear*, i.e., each variable x_j occurs exactly once. The constructed transducer thus is in fact, a DT_{lin}. Finally we note that the set of all states h such that $\mathsf{dom}_B(h)$ is output recognizable by s, can be determined by a TA running over s.

▶ **Theorem 14.** *For a given state h of a top-down deterministic TA B,*

1. *it can be decided in polynomial time whether or not $\mathsf{dom}_B(h)$ is output recognizable;*
2. *it can be decided in polynomial time whether or not $\mathsf{dom}_B(h)$ is output recognizable by a particular tree s and if so, a DT_{lin} $A_{B,h,s}$ without inspection can be constructed in polynomial time with domain $\mathsf{dom}_B(h)$ such that $[\![t]\!]_{A_{B,h,s}} = s$ for all $t \in \mathsf{dom}_B(h)$.*

6 How to Satisfy Inspection Needs

In the following, we consider an arbitrary $\mathsf{DT}^{\text{I}}_{\text{uc}}$ transducer A with underlying top-down deterministic TA B as *inspection* automaton. Consider a rule τ of the form $q(f(x_1 : h_1, \ldots, x_k : h_k)) \to T$ of A. For every x_i not occurring in T, it must be verified that the corresponding subtree of the input is contained in $\mathsf{dom}_B(h_i)$. This verification is trivial if $\mathsf{dom}_B(h_i) = \mathcal{T}_\Sigma$. Such a state h_i (if present) has been denoted by $\top$. Accordingly, let J_τ denote the set of indices j such that x_j does *not* occur in the right-hand side of T while at the same time, $h_j \neq \top$. Let us thus call the *multiset* $\eta_\tau = \{h_j \mid j \in J_\tau\}$ the *inspection need* of the rule τ. Assume that T has disjoint ground subtrees $s_j, j \in J_\tau$, such that $\mathsf{dom}_B(h_j)$ is output recognizable by s_j for $j \in J_\tau$. Then the rule τ can equivalently be replaced by a rule without inspection need. In this case, we say that τ *satisfies* its inspection need.

▶ **Example 15.** Consider the rule $q(f(x_1 : h_1, x_2 : h_2)) \to g(x_1, r(b))$ where $\mathsf{dom}_B(h_2)$ equals the set $L = \{g(a, t) \mid t \in \mathcal{T}_\Sigma\}$. Then $J_\tau = \{2\}$ where the language L is output realizable with respect to $r(b)$. The latter can be seen by means of the rules $q_1(g(x_1, x_2)) \to r(q_2(x_1))$ and $q_2(a) \to b$. Accordingly, the given rule for q and f satisfies its inspection need.

Our goal is to construct for a given $\mathsf{DT}^{\text{I}}_{\text{uc}}$ transducer A an equivalent DT_{uc} transducer A' such that each rule of A' satisfies its inspection need. If each rule of A satisfies its inspection need, this need no longer be the case for the *earliest* transducer equivalent to A. The reason is that some ground subtrees of prefixes of right-hand sides may have been moved to the right-hand sides of other rules. Satisfying inspection needs of rules therefore requires to partly *revert* the *earliest* transformation. In the following, we call a state q of the $\mathsf{DT}^{\text{I}}_{\text{uc}}$ A *constant*, if there is a single output tree s such that $s = s'$ whenever $q : t \to s'$ holds.

▶ **Lemma 16.** *For a partial mapping $\mu : \mathcal{T}_\Sigma \to \mathcal{T}_\Delta$, the following are equivalent: (1) μ is realized by a DT_{uc} without inspection; (2) μ is realized by a DT^I_{uc} without constant states, but where all inspection needs are satisfied.*

Assume that A' is a DT_{uc} and A the corresponding $\mathsf{DT}^{\text{I}}_{\text{uc}}$ in canonical earliest normal form. This means that for each state q of A and each state $q' \in q$ of A' with $\mathsf{pref}_{A'}(q') = p$, $q' : t \to s'$ holds for A' iff $q : t \to s$ with $s' = p \cdot s$ holds. In particular, the constant outputs for some states of A' may occur as subtrees in p and thus are already produced before A processes t. In order to recover the (yet unknown) DT_{uc} A' without inspection from A, we determine the minimal suffix p' of p so that all inspections possibly encountered when q processes its input, can be satisfied. Such a *generalized inspection need* of a q-computation is represented by a sequence $(M_1, \emptyset) \ldots (M_{r-1}, \emptyset)(M_r, \phi)$, $r \geq 0$, where $M_1, \ldots, M_r$ are

multisets of inspection states and ϕ is a downward-closed subset of sub-multisets of M_r with $M_r \notin \psi$. Intuitively, a generalized inspection need is the *sequence* of future inspections yet to be simulated. The pair (M_r, ϕ) to the right is meant to occur farest in the future. Thereby, the set ϕ describes which sub-multisets of individual inspections of M_r can already be accomplished (the available ground terms might be used in more than one way). If $M_r \in \phi$, then *all* languages $\mathsf{dom}_B(h), h \in M_r$, are simultaneously realizable – implying that the whole pair (M_r, ϕ) can be dropped.

For a finite multiset G of trees in $\mathcal{T}_\Delta$ and a finite multiset M of states in H, define $\langle\langle G, M\rangle\rangle$ as the set of all sub-multisets M' of M so that the multiset of languages $\{\mathsf{dom}_B(h) \mid h \in M'\}$ are simultaneously output recognizable by means of disjoint subtrees of trees in G.

Now assume that $p \in \mathcal{T}_\Delta(x_1)$ where $p = s_1 \cdot \ldots \cdot s_m$ where each of the $s_j \in \mathcal{T}_\Delta(x_1)$ is *irreducible*, i.e., cannot be written as a product of patterns different from x_1. For p, we define the auxiliary transformation $[\![p]\!]^{\sharp\sharp}$ as the composition $[\![s_1]\!]^{\sharp\sharp} \circ \ldots \circ [\![s_m]\!]^{\sharp\sharp}$ where for an irreducible pattern $s \in \mathcal{T}_\Delta(x_1)$, the transformation is defined as follows. First, $[\![s]\!]^{\sharp\sharp}\epsilon = \epsilon$. For $\alpha = \alpha'\,(M, \phi)$, the pattern s can only be used to satisfy the inspection need of the *last* pair in α. Let G denote the set of maximal distinct ground subtrees of s. Let ϕ' denote the set of multisets of inspection needs which become satisfiable when the ground terms from G are additionally available, i.e., $\phi' = \{R_1 \cup R_2 \mid R_1 \in \phi, R_2 \in \langle\langle G, M \setminus R_1\rangle\rangle\}$. Then $[\![s]\!]^{\sharp\sharp}\alpha = \alpha'$ if $M \in \phi'$ and $[\![s]\!]^{\sharp\sharp}\alpha = \alpha'\,(M, \phi')$ otherwise. Let I the set of all possible generalized inspection needs. Let us introduce some notation. For a particular state q and input tree t, let us define the inspection need $\eta_q(t)$ of q for t as follows. Assume that $t = f(t_1, \ldots, t_k)$ and τ is the rule of A of the form $q(f(x_1 : h_1, \ldots, x_k : h_k)) \to p\{x_j \mapsto q_j(x_j) \mid j \in J\}$ such that $p \in \mathcal{T}_\Delta(X_J)$ holds with $h_i : t_i$ for $i = 1, \ldots, k$. For the rule τ, we define the transformation

$$[\![\tau]\!]^\sharp : (X_J \to I) \to I \quad \text{such that} \quad \eta_q(t) = [\![\tau]\!]^\sharp\{x_j \mapsto \eta_{q_j}(t_j) \mid j \in J\} \quad \text{holds.}$$

The transformation $[\![\tau]\!]^\sharp$ is defined by case distinction. If $J = \emptyset$, i.e., p is ground, we check in how far p itself is sufficient for η_τ to be output realizable. Let $\phi = \langle\langle\{p\}, \eta_\tau\rangle\rangle$. Then

$$[\![\tau]\!]^\sharp\emptyset = \begin{cases} \epsilon & \text{if } \eta_\tau \in \phi \\ (\eta_\tau, \phi) & \text{otherwise} \end{cases}$$

$$[\![\tau]\!]^\sharp\{x_j \mapsto \alpha\} = [\![p]\!]^{\sharp\sharp}((\eta_\tau, \emptyset)\alpha)$$

Finally, assume that J contains more than one index. Let $p = p'\{x_j \mapsto p_j \cdot x_j \mid j \in J\}$ for *maximal* patterns $p_j \in \mathcal{T}_\Delta(x_1)$. For $\alpha_j, j \in J$, assume that $p_j = p'_j \cdot u_j$ for some *minimal* suffix $u_j \in \mathcal{T}_\Delta(x_1)$ with $[\![u_j]\!]^{\sharp\sharp}\alpha_j = \epsilon$. These suffixes must exist, whenever A is equivalent to some $\mathsf{DT}_{\mathrm{uc}}$ without inspection. Let G denote the set of distinct maximal ground subtrees of $p'\{x_j \mapsto p'_j \cdot x_j \mid j \in J\}$, and $\phi = \langle\langle G, \eta_\tau\rangle\rangle$. Then we define

$$[\![\tau]\!]^\sharp\{x_j \mapsto \alpha_j \mid x_j \in X_J\} = \begin{cases} \epsilon & \text{if } \eta_\tau \in \phi \\ (\eta_\tau, \phi) & \text{otherwise} \end{cases}$$

We have:

▶ **Lemma 17.** *Assume that A is the canonical earliest normal form of some DT_{uc} A' without inspection. Let q denote some state of A, i.e., an equivalence class of states of A'. Let $q' \in q$ be state of A', let $p = \mathsf{pref}_{A'}(q')$ the maximal common prefix of outputs of A' for q', and $t \in \mathsf{dom}_B(\iota(q))$. Then (1) $\eta_q(t)$ is defined and (2) $[\![p]\!]^{\sharp\sharp}(\eta_q(t)) = \epsilon$.*

The proof is by induction on the structure of t where we use that for $q' : t \to s'$ and $q : t \to s$ we have that $s' = p \cdot s$ for $p = \mathsf{pref}_{A'}(q')$.

By computing $\eta_q(t)$, we partly recover information on the (unknown) common prefix p of the (yet to be constructed) DT_{uc} A' for q'. By computing the set of *all* these inspection needs for q, we determine the maximal requirement on a suffix of output already produced by A when reaching q, whose delay then is sufficient to satisfy all possible future inspection needs. Therefore, let $S[q] = \{\eta_q(t) \mid t \in \mathsf{dom}_B(\iota(q))\}$. In order to calculate this set, we construct a constraint system $\mathcal{C}_I$ with unknowns X_q, where q is state of A. The system consists of one constraint per rule of A. Assume that τ is the rule $q(f(\ldots)) \to p\{x_j \mapsto q_j(x_j) \mid j \in J\}$ where $p \in \mathcal{T}_\Delta(X_J)$ and $q_j, j \in J$ are states of A. Then the constraint system $\mathcal{C}_I$ has the constraint

$$X_q \supseteq \{[\![\tau]\!]^\sharp\{x_j \mapsto \alpha_j \mid j \in J\} \mid \forall j \in J.\, \alpha_j \in X_{q_j}\}$$

The given constraint system is over subsets of I where all right-hand sides are monotonic w.r.t. subset inclusion. Therefore, it has a *least* solution. Moreover, we have:

▶ **Lemma 18.** *Assume that A is a canonical DT^I_{uc} transducer which is equivalent to some DT_{uc} A' without inspection. Let X_q, q a state of A, denote the least solution of the system of constraints $\mathcal{C}_I$. Then for each state q of A: (1) $X_q = S[q]$ and (2) the length of each $\alpha \in S[q]$ is bounded by $|A|$.*

Proof. In order to verify the first statement, we prove by induction that the ith iterate $X_q^{(i)}$ of the constraint system exactly equals the set of all $\eta_q(t)$ for trees t of depth less than i. For a proof of the second statement, we note that each $\alpha \in S[q]$ is the inspection need of some execution starting in q which must be accomplished by every pattern available at q. More precisely, for every state q of A there is a context $t \in \mathcal{C}_\Sigma$ such that $(q_0, h) : t \to u \cdot q(x_1)$ holds where $\iota(q) = h$. Here, we rely on the minimality of A, implying that each state q can be reached in this way. Let $T'_0 \cdot q_0(x_1)$ denote the axiom of A. Then t can be chosen in such a way that $T_0 \cdot u$ consists of at most $|A|$ factors. Since at least one factor of the pattern is required to realize the inspection at one rule, the upper bound to the lengthes of inspection needs $\alpha \in S[q]$ follows. ◀

As a consequence, the sets $S[q]$, q a state of A, are effectively computable.

For a finite set $S \subseteq I$, let t^S denote the minimal suffix v of t such that $[\![v]\!]^{\sharp\sharp}\alpha = \epsilon$ for all $\alpha \in S$. The states of the new DT_{uc} A' are pairs $\langle q, s\rangle$, q a state of A and $s \in \mathcal{T}_\Delta(x_1)$ an output pattern for A, which will also be called the *buffer*. Assume that we are given for each state q of A, the (finite) set $S[q]$ of inspection needs which are to be satisfied by q-computations. Assume that $T'_0 = u \cdot v$ where $v = (T'_0)^{S[q_0]}$. Then the axiom of A' is given by $u \cdot \langle q_0, v\rangle(x_1)$. Assume that state $\langle q, u\rangle$ of A' has already been constructed, and τ is a rule of A of the form $q(f(\ldots)) \to p\{x_j \mapsto q_j(x_j) \mid j \in J\}$ where $p \in \mathcal{T}_\Delta(X_J)$.

If $J = \emptyset$, A' has a rule $\langle q, u\rangle(f(\ldots)) \to u \cdot p$. In case that $\eta_\tau \neq \epsilon$, $u \cdot p$ must be sufficient to satisfy the inspection needs incurred by the rule τ, i.e., $\langle\langle\{u \cdot t\}, \eta_\tau\rangle\rangle$ must contain η_τ.

Next assume that $J = \{j\}$, i.e., $p = p' \cdot x_j$. Then $[\![u \cdot p']\!]^{\sharp\sharp}((\eta_\tau, \emptyset)\,\alpha) = \epsilon$ must hold for all $\alpha \in S[q_j]$. Therefore, there is a factorization such that $u \cdot p' = u' \cdot v$ where $v = (u \cdot p')^{S[q_j]}$; we add the rule $\langle q, u\rangle(f(\ldots)) \to u' \cdot \langle q', v\rangle(x_i)$ to A'. Finally, assume that J contains at least two elements. Then p is of the form $p = p'\{x_j \mapsto p_j \cdot x_j \mid j \in J\}$ for $p' \in \mathcal{T}_\Delta(X_J)$ and maximal patterns $p_j \in \mathcal{T}_\Delta(x_1)$. For each $j \in J$, there must be a factorization $p_j = u_j \cdot v_j$ where $v_j = (p_j)^{S[q_j]}$. Moreover, the subset G of ground subtrees of u, p' and $u_j, j \in J$, must be sufficient to satisfy the inspection need of τ itself, i.e., $\langle\langle G, \eta_\tau\rangle\rangle$ contains η_τ. Then add the rule $\langle q, u\rangle(f(\ldots)) \to u \cdot p'\{x_j \mapsto u_j \cdot \langle q_j, v_j\rangle(x_j) \mid j \in J\}$ to A'. Correctness and termination of the construction follows from the following lemma.

▶ **Lemma 19.** *Assume that A is the canonical earliest DT^I_{uc} for a DT_{uc} A'' without inspection. Then the construction in Section 6 terminates with some A' so that the following properties are satisfied:*

1. *A is equivalent to A';*
2. *Each inspection need η_τ of A' is satisfiable;*
3. *For each constructed state $\langle q, u\rangle$ of A', $u \in \mathcal{T}_\Delta(x_1)$ has length at most $|A|^2 \cdot (a-1)$ if a is the maximal rank of an input symbol.*

Proof. Assume that A is equivalent to some A'' without inspection, i.e., is the canonical earliest normal form of A''. This means that for states q_1, q_2 of A, $t_1 \in \mathcal{C}_\Sigma$ and $t_2 \in \mathcal{T}_\Sigma$ with $(q_1, \iota(q_2)) : t_1 \to s_1 \cdot q_2(x_1)$, and $q_2 : t_2 \to s_2$ and all states q_1', q_2' of A'' so that $q_i' \in q_i$, $(q_1', \iota(q_2')) : t \to s_1' \cdot q_2'(x_1)$ and $q_2' : t \to s_2'$, for some s_1', s_2', it holds that $v_1 \cdot s_1 = s_1' \cdot v_2$ and $v_2 \cdot s_2 = s_2'$ for $v_i = \mathsf{pref}_{A'}(q_i')$.

Consider a pair $\langle q, u\rangle$ returned by the construction and let $q' \in q$ denote a state of A'' contained in q with $v_{q'} = \mathsf{pref}_{A''}(q')$. Then u is a suffix of $v_{q'}$. Consequently, the set of all constructed states $\langle q, u\rangle$ is finite, and all pre-conditions at the construction of rules are met. This means that all inspection needs of the resulting transducer are satisfiable. Moreover, we have that $q : t \to s$ holds for A iff $\langle q, u\rangle : t \to s'$ holds for A' where $u \cdot s = s'$ – implying that A and A' are equivalent.

It remains to prove item (3), i.e., to provide an upper bound for the lengthes of the patterns u occurring in the construction as second components of states $\langle q, u\rangle$ – not in terms of the transducer A'', but in terms of the transducer A, which serves as the input to the construction. According to thr construction, we know that u is a minimal pattern to satisfy all generalized inspection needs in $S[q]$.

First, assume that $(q, \iota(q)) : t \to s \cdot q(x_1)$ holds for A some context $t \in \mathcal{C}_\Sigma$ so that only rules τ are applied whose right-hand sides have occurrences of single variables only. In that case, $t = t_1 \cdot \ldots \cdot t_m$ for irreducible contexts $t_i \in \mathcal{C}_\Sigma$ where each computation for t_i consists in the application of a single rule τ_i. *Case 1*: $s = x_1$. Then η_{τ_i} must be $\emptyset$ for all $i = 1, \ldots, m$. *Case 2*: $s \neq x_1$. Then $u \cdot s = u_1 \cdot v$ so that all inspections to be satisfied by the τ_i together can be satisfied by u_1 while v is sufficient to satisfy all generalized inspection needs in $S[q]$. Accordingly, $[\![s^{|A|\cdot(a-1)}]\!]^{\sharp\sharp}\alpha = \epsilon$ for each $\alpha \in S[q]$. Therefore, s (independently of u) is at least able to satisfy each (M, ϕ) occurring inside some $\alpha \in S[q]$.

Now assume for a contradiction, that a state $\langle q, u\rangle$ is constructed where u is of length exceeding $|A|^2 \cdot (a-1)$. Then there is a minimal context $t \in \mathcal{C}_\Sigma$ of the form $t = t' \cdot t_1 \cdot \ldots \cdot t_N \cdot t''$ for $N = |A|^2 \cdot (a-1) + 1$ such that $(q_1, \iota(q')) : t' \to s' \cdot q'(x_1)$, $(q', \iota(q')) : t_i \to s_i \cdot q'(x_1)$ for $i = 1, \ldots, N$ with $s_i \neq x_1$, and $(q', \iota(q)) : t' \to s'' \cdot q(x_1)$ so that for some $v_0 \in \mathcal{T}_\Delta(x_1)$, one of the following two conditions holds:

- The axiom of A is of the form $s_0 \cdot v_1 \cdot q_1(x_1)$; or
- there is a right-hand side of a rule of A which is of the form $p\{x_j \mapsto v_j' \cdot q_j'(x_j) \mid j \in J\}$ for some J of cardinality exceeding 1, where $v_1 = v_{j'}'$ and $q_1 = q_{j'}'$ for some $j' \in J$.

By construction, all inspection needs along the way, are satisfied by $v_1 \cdot s'$. According to our observation above, though, u must be a suffix of $s_2 \cdot \ldots \cdot s_N \cdot s''$ – contradicting the minimality of the context t. We conclude that the maximal length of the buffer is bounded by $|A|^2 \cdot (a-1)$. ◀

In summary, we have shown:

▶ **Theorem 20.** *(1) For a DT^I_{uc} (DT^I_{lin}) A it is decidable if is equivalent to a DT_{uc} (DT_{lin}) A', and if so, such A' can be constructed. (2) For a DT^R_{uc} (DT^R_{lin}) A is decidable if A is equivalent to a DT_{uc} (DT_{lin}) A', and if so, such A' can be constructed.*

In Section 2 we have seen that a bottom-up deterministic transducer (DB) can be seen as a particularly simple $\mathsf{DT}^{\mathrm{R}}_{\mathrm{uc}}$ transducer. Accordingly, we obtain as a corollary:

▶ **Corollary 21.** *Let A be a (linear) DB. It is decidable if an equivalent DT^I_{uc} (DT^I_{lin}) or an equivalent DT_{uc} (DT_{lin}) exists, and if so, such a transducer can be constructed.*

7 Conclusion

We showed for two natural subclasses of deterministic top-down tree transducers how to remove (bottom-up deterministic) look-ahead and replace it whenever possible, with (top-down deterministic) inspection. We then also showed for the given classes how to remove inspection (if possible). The constructions are technically intricate, but crucially rely on canonical earliest normal forms for the transducers in question. As a corollary we obtain that for a given deterministic bottom-up transducer it is decidable whether or not it can be realized by a deterministic top-down tree transducer that is either *uc* or linear.

One may wonder if our results imply that for a given deterministic bottom-up tree transducer U it is decidable whether or not it can be realized by an *arbitrary* deterministic top-down tree transducer. I.e., if U *can* be realized by top-down transducer can be realized by a *uc* such transducer? Interestingly, this is *not* the case: let h_a, h_b be look-ahead states that indicate that the left-most leaf of the input tree is labeled a and b, respectively and consider a transducer which has these rules (for every $h \in \{h_a, h_b\}$):

$$q_0(f(x_1 : h_a, x_2 : h)) \to g(a, b, q_{\mathrm{id}}(x_2)) \qquad q_0(f(x_1 : h_b, x_2 : h)) \to g(c, d, q_{\mathrm{id}}(x_2))$$

The corresponding translation *can* be realized by a deterministic top-down tree transducer! However, the transducer is *not uc* (viz. the output leaves a and b must be produced by different states, but both on the input variable x_1).

References

1 Krzysztof R. Apt and Gordon D. Plotkin. Countable nondeterminism and random assignment. *J. ACM*, 33(4):724–767, 1986. doi:10.1145/6490.6494.

2 Marie-Pierre Béal and Olivier Carton. Determinization of transducers over finite and infinite words. *Theor. Comput. Sci.*, 289(1):225–251, 2002. doi:10.1016/S0304-3975(01)00271-7.

3 Jean Berstel. *Transductions and context-free languages*, volume 38 of *Teubner Studienbücher : Informatik*. Teubner, 1979. URL: http://www.worldcat.org/oclc/06364613.

4 Adrien Boiret, Aurélien Lemay, and Joachim Niehren. Learning top-down tree transducers with regular domain inspection. In *Proceedings of the 13th International Conference on Grammatical Inference, ICGI 2016, Delft, The Netherlands, October 5-7, 2016*, pages 54–65, 2016. URL: http://proceedings.mlr.press/v57/boiret16.html.

5 Christian Choffrut. Une caracterisation des fonctions sequentielles et des fonctions sous-sequentielles en tant que relations rationnelles. *Theor. Comput. Sci.*, 5(3):325–337, 1977. doi:10.1016/0304-3975(77)90049-4.

6 Joost Engelfriet. Bottom-up and top-down tree transformations - A comparison. *Mathematical Systems Theory*, 9(3):198–231, 1975. doi:10.1007/BF01704020.

7 Joost Engelfriet. Top-down tree transducers with regular look-ahead. *Mathematical Systems Theory*, 10:289–303, 1977. doi:10.1007/BF01683280.

8 Joost Engelfriet, Sebastian Maneth, and Helmut Seidl. Deciding equivalence of top-down XML transformations in polynomial time. *J. Comput. Syst. Sci.*, 75(5):271–286, 2009. doi:10.1016/j.jcss.2009.01.001.

9 Joost Engelfriet, Sebastian Maneth, and Helmut Seidl. Look-ahead removal for total deterministic top-down tree transducers. *Theor. Comput. Sci.*, 616:18–58, 2016. doi:10.1016/j.tcs.2015.12.005.

10 Emmanuel Filiot, Sebastian Maneth, Pierre-Alain Reynier, and Jean-Marc Talbot. Decision problems of tree transducers with origin. *Inf. Comput.*, 261(Part):311–335, 2018. doi:10.1016/j.ic.2018.02.011.

11 Zoltán Fülöp and Andreas Maletti. Linking theorems for tree transducers. *J. Comput. Syst. Sci.*, 82(7):1201–1222, 2016. doi:10.1016/j.jcss.2016.05.003.

12 Aurélien Lemay, Sebastian Maneth, and Joachim Niehren. A learning algorithm for top-down XML transformations. In *Proceedings of the Twenty-Ninth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2010, June 6-11, 2010, Indianapolis, Indiana, USA*, pages 285–296, 2010. doi:10.1145/1807085.1807122.

13 William C. Rounds. Mappings and grammars on trees. *Mathematical Systems Theory*, 4(3):257–287, 1970. doi:10.1007/BF01695769.

14 James W. Thatcher. Generalized sequential machine maps. *J. Comput. Syst. Sci.*, 4(4):339–367, 1970. doi:10.1016/S0022-0000(70)80017-4.

15 James W. Thatcher. There's a lot more to finite automata theory than you would have thought. In *Proc. 4th Ann. Princeton Conf. on Informations Sciences and Systems*, pages 263–276, 1970. Also published in revised form under the title "Tree automata: an informal survey" in Currents in the Theory of Computing (ed. A. V. Aho), Prentice-Hall, 1973, 143-172.