

Determinacy and Rewriting of Top-Down and MSO Tree Transformations

Michael Benedikt¹, Joost Engelfriet², and Sebastian Maneth¹

¹ University of Oxford `first.last@cs.ox.ac.uk`

² LIACS, Leiden University `engelfr@liacs.nl`

Abstract. A query is determined by a view, if the result to the query can be reconstructed from the result of the view. We consider the problem of deciding for two given tree transformations, whether one is determined by the other. If the view transformation is induced by a tree transducer that may copy, then determinacy is undecidable, even for identity queries. For a large class of non-copying views, namely compositions of functional extended linear top-down tree transducers with regular look-ahead, we show that determinacy is decidable, where queries are given by deterministic top-down tree transducers with regular look-ahead or by MSO tree transducers. We also show that if a query is determined, then it can be rewritten into a query that works directly over the view and is in the same class as the given query. The proof relies on the decidability of equivalence for the two considered classes of queries, and on their closure under composition.

1 Introduction

Given a transformation between data structures, a basic question is what sort of information it preserves. In some contexts, one desires a transformation that is “fully information-preserving” – one can recover the input from the output. In other cases it may be acceptable, or even important, to hide certain pieces of information in the input; but necessarily there is *some* important information in the input that must be recoverable from the output. This notion has been studied in the database community [27, 24]: a query q is determined by another query v if there exists a function f such that $f(v(s)) = q(s)$ for every input s . The query v is referred to as “view”. Note that nothing is said about how efficiently f can be computed (or if it can be computed at all). We can then strengthen determinacy by requiring the function f to lie within a certain class \mathcal{C} ; then f is a “rewriting in \mathcal{C} ”. These notions have received considerable attention in the database setting [27, 24, 25, 1].

In this paper we study determinacy and rewriting for classes of tree transformations (or, tree translations). Injectivity is undecidable for deterministic top-down tree transducers [15, 17]; hence, one cannot decide if the identity query is determined by such a transducer. This holds for transducers that only copy once. We therefore restrict our attention to views induced by *linear* tree transducers. For the same reason we restrict to a single view (while in database research, normally multiple views are considered). Our main result is that determinacy is decidable for views that are compositions of functional extended linear top-down tree transducers (with regular look-ahead) and for queries that are either deterministic top-down tree transducers (with regular

look-ahead) or deterministic MSO definable tree transducers (where MSO stands for Monadic Second-Order logic). Extended transducers generalize the left-hand sides of conventional finite-state tree transducers (from one input symbol to an arbitrary “pattern tree”). They were invented by Arnold and Dauchet [3] and have recently been studied in [22, 23, 9]. Extended linear transducers are convenient because (1) they are more powerful than ordinary linear top-down or bottom-up transducers and (2) they allow to elegantly capture the inverses of translations.

As an example, consider the transformation v taking binary trees as input, with internal nodes labeled a, b, c , and leaves labeled l . It relabels the b nodes as a nodes, and otherwise copies the tree as is. A linear top-down transducer implementing this translation v has a single state p and these translation rules:

$$\begin{array}{ll} p(a(x, y)) \rightarrow a(p(x), p(y)) & p(b(x, y)) \rightarrow a(p(x), p(y)) \\ p(c(x, y)) \rightarrow c(p(x), p(y)) & p(l()) \rightarrow l() \end{array}$$

Information about the (labels of) b nodes and a nodes is lost in the translation – e.g., from the output of v we cannot determine the answer to the identity query q_0 . In contrast, information about the l nodes and their relationship to c nodes is maintained. For example, the query q_1 that removes a and b nodes but keeps c and l nodes is determined by v . Our algorithm can decide that q_0 is not determined and q_1 is.

Our decision procedure for determinacy establishes several results that are interesting on their own. For a view v realized by an extended linear top-down tree transducer, its inverse v^{-1} is a binary relation on trees. Our approach converts v^{-1} into a composition of two nondeterministic translations, a translation τ_1 of a very simple form and a translation τ_2 in the same class as v . We then construct uniformizers u_1, u_2 of τ_1, τ_2 and compose them to form a uniformizer u of v^{-1} . A *uniformizer* of a binary relation R is a function u such that $u \subseteq R$ and u has the same domain as R ; thus u “selects” one of the possibly several elements that R associates with an element of its domain. It is easy to see that a query q is determined by v if and only if $v \circ u \circ q = q$ (where \circ denotes sequential composition, see the Preliminaries). We show that if q is a deterministic top-down or MSO definable tree translation, then so is $v \circ u \circ q$. This is achieved by proving that u_1, u_2 , and v are deterministic top-down *and* MSO definable tree translations. Since our two query classes are closed under composition and $u = u_1 \circ u_2$, this shows that $v \circ u \circ q$ is in the same class as q . We then decide $v \circ u \circ q = q$, and hence determinacy, making use of the decidability of equivalence for deterministic top-down or MSO definable tree translations [13, 12]. The same proof also shows that if q is determined by v , then $u \circ q$ is a rewriting belonging to the same class as q .

Related Work. The notion of view-query determinacy was introduced by Segoufin and Vianu in [27]. They focus on relational queries definable in first-order logic and show that if such queries are determined over arbitrary structures, then they can be rewritten in first-order, but that if they are determined over finite structures, they may require a much more powerful relational query to be rewritten. Nash, Segoufin, and Vianu’s [24] summarizes a number of other results on the relational case. Due to the differing data models and notions of equality used in relational queries and tree structures, results on determinacy for queries in the relational case do not (directly) apply

to transducers, and vice versa. In the context of unranked trees, determinacy is considered in Groz’s thesis [19] for XML views and queries, see also [20]. Two notions of determinacy are considered, depending on whether or not the output trees preserve provenance information (i.e., node identities) from the input document. It is shown that both notions of determinacy are undecidable for views and queries defined using a transformation language that can select subtrees using regular XPath filters. On the positive side, it is shown that if the views are “interval-bounded” – there is a bound on the number of consecutive nodes skipped along a path – then determinacy can be tested effectively. The most related work is [21], which considers the determinacy problem (and rewriting) explicitly for tree transducers, and solves it for functional extended linear bottom-up views and deterministic bottom-up queries. Their approach is to decide determinacy by testing functionality of the inverse of the view composed with the query. To this end they generalize the functionality test for bottom-up transducers in [28] to extended bottom-up transducers with “grafting” (needed for the inverse of the view). Our main result generalizes the one of [21], and provides an alternative proof of it.

2 Preliminaries

For $k \in \mathbb{N} = \{0, 1, \dots\}$ let $[k]$ denote the set $\{1, \dots, k\}$. For a binary relation R and a set A we denote by $R(A)$ the set $\{y \mid \exists x \in A : (x, y) \in R\}$, and by $R(x)$ the set $R(\{x\})$. If $R \subseteq B \times C$ for sets B and C , then $\text{ran}(R) = R(B)$ and $\text{dom}(R) = R^{-1}(C)$. For two relations R and S we denote the sequential composition “ R followed by S ” by $R \circ S$, i.e., for an element x , $(R \circ S)(x) = S(R(x))$. Note this is in contrast to the conventional use of \circ . If \mathcal{R}, \mathcal{S} are classes of binary relations, then $\mathcal{R} \circ \mathcal{S} = \{R \circ S \mid R \in \mathcal{R}, S \in \mathcal{S}\}$ and $\mathcal{R}^* = \{R_1 \circ \dots \circ R_n \mid n \geq 1, R_i \in \mathcal{R}\}$. We define determinacy and rewritability, following [24]. Let \mathcal{Q}, \mathcal{V} be classes of partial functions and let $q \in \mathcal{Q}$ and $v \in \mathcal{V}$. We say that q is *determined* by v , if there exists a function f such that $v \circ f = q$. Note that the latter means that the domains of $v \circ f$ and q coincide, and that $f(v(s)) = q(s)$ for each s in that domain. *Determinacy for \mathcal{Q} under \mathcal{V}* is the problem that takes as input $q \in \mathcal{Q}$ and $v \in \mathcal{V}$ and outputs “yes” if q is determined by v , and “no” otherwise. Determinacy says that there is a functional dependency of q on v , with no limit on how complex it is to reconstruct the answer to q from the answer to v . A finer notion requires that the reconstruction be in a given class. Let \mathcal{Q}' be a class of partial functions. We say that \mathcal{Q}' is *complete for \mathcal{V} -to- \mathcal{Q} rewritings*, if for every $q \in \mathcal{Q}$ and $v \in \mathcal{V}$ such that q is determined by v , there is an $f \in \mathcal{Q}'$ with $v \circ f = q$.

Trees and Tree Automata. A ranked alphabet consists of a finite set Σ together with a mapping $\text{rank}_\Sigma : \Sigma \rightarrow \mathbb{N}$. We write $a^{(k)}$ to denote that $\text{rank}_\Sigma(a) = k$ and define $\Sigma^{(k)}$ as the set $\{a \in \Sigma \mid \text{rank}_\Sigma(a) = k\}$. The set of (ranked, ordered, node-labeled, finite) trees over Σ , denoted T_Σ , is the set of words defined recursively as the smallest set T such that $a(s_1, \dots, s_k) \in T$ if $a \in \Sigma^{(k)}$, $k \geq 0$, and $s_1, \dots, s_k \in T$. For a tree $a()$ we simply write a . For a set of trees T , we denote by $T_\Sigma(T)$ the set of trees obtained from trees in T_Σ by replacing arbitrary leaves by trees in T . We fix a countable infinite set $X = \{x_1, x_2, \dots\}$ of variables. For $k \in \mathbb{N}$, let X_k be the ranked alphabet $\{x_1^{(0)}, \dots, x_k^{(0)}\}$. For $k \in \mathbb{N}$, an X_k -context (over Σ) is a tree C in $T_\Sigma(X_k)$ such that each variable in X_k occurs exactly once in C . For such a context C and trees s_1, \dots, s_k ,

$C[s_1, \dots, s_k]$ denotes the tree obtained from C by replacing each $x_i \in X_k$ by s_i . Let T_1, \dots, T_n be sets of trees. For trees s_1, \dots, s_n that are not subtrees of another, we denote by $s[s_i \leftarrow T_i \mid i \in [n]]$ the set of trees obtained from s by replacing each occurrence of a subtree s_i of s by a tree from T_i (where different occurrences of s_i need not be replaced by the same tree). For a ranked alphabet Q with $Q^{(1)} = Q$ we denote by $Q(X_k)$ the set of trees $\{q(x_i) \mid q \in Q, i \in [k]\}$. A *deterministic bottom-up tree automaton* (dbta) over Σ is a tuple $A = (P, \Sigma, F, \delta)$ where P is a finite set of states, Σ is a ranked alphabet, $F \subseteq P$ is the set of final states, and δ is the transition function. For every $a \in \Sigma^{(k)}$, $k \geq 0$, and $p_1, \dots, p_k \in P$, $\delta(a, p_1, \dots, p_k)$ is an element of P . The function δ is extended to trees s in T_Σ in the usual way; the resulting function from T_Σ to P is denoted δ as well. Thus $\delta(s)$ is the state reached by A at the root of s . The language accepted by A is $L(A) = \{s \in T_\Sigma \mid \delta(s) \in F\}$.

Convention: All lemmas, theorems, etc., stated in this paper (except in Section 4) are *effective*.

3 Extended Top-Down and Bottom-Up Tree Transducers

Extended top-down transducers are studied in, e.g., [3, 22, 23, 9]. The class we denote ELT is denoted $\text{L-XTOP}_{\text{ef}}$ in [22], where the “L” denotes linearity, and “ef” epsilon-freeness, meaning the left-hand sides of rules are not of the form $q(x_i)$. An *extended top-down tree transducer with regular look-ahead* (ET^R transducer) is a tuple $M = (Q, \Sigma, \Delta, I, R, A)$ where Q is a ranked alphabet of states all of rank 1, Σ and Δ are ranked alphabets of input and output symbols, respectively, $I \subseteq Q$ is a set of initial states, A is a dbta over Σ called the look-ahead automaton, and R is a finite set of rules of the form $q(C) \rightarrow \zeta \langle p_1, \dots, p_k \rangle$, where $q \in Q$, $C \neq x_1$ is an X_k -context over Σ , $k \geq 0$, $\zeta \in T_\Delta(Q(X_k))$, and $p_1, \dots, p_k \in P$. For an input tree $s \in T_\Sigma$, the q -translation $\llbracket q \rrbracket_M(s)$ is the smallest set of trees $T \subseteq T_\Delta$ such that for every rule $q(C) \rightarrow \zeta \langle p_1, \dots, p_k \rangle$ and all $s_1, \dots, s_k \in T_\Sigma$, if $s = C[s_1, \dots, s_k]$ and $\delta(s_i) = p_i$ for every $i \in [k]$, then T contains the set of trees $\zeta[q'(x_i) \leftarrow \llbracket q' \rrbracket_M(s_i) \mid q' \in Q, i \in [k]]$. The translation $\llbracket M \rrbracket$ realized by M is the binary relation $\{(s, t) \in T_\Sigma \times T_\Delta \mid s \in L(A), t \in \cup_{q \in I} \llbracket q \rrbracket_M(s)\}$. The class of all translations realized by ET^R transducers is denoted ET^R (and similarly for other transducers). The transducer M is *linear*, if the right-hand side ζ of each rule is linear in the set of variables X , i.e., each variable x_i occurs at most once in ζ . We use “L” to abbreviate “linear”, i.e., ELT^R is the class of $\llbracket M \rrbracket$ where M is a linear ET^R transducer. Transducers *without* look-ahead are defined by transducers with a trivial one-state look-ahead automaton (accepting T_Σ); this is indicated by omitting the superscript “R” for transducers and classes.

An *extended linear bottom-up tree transducer* (ELB transducer) is a tuple $B = (Q, \Sigma, \Delta, F, R)$ where Q is a ranked alphabet of states all of rank 1, Σ and Δ are ranked alphabets of input and output symbols, respectively, $F \subseteq Q$ is a set of final states, and R is a finite set of rules of the form $C[q_1(x_1), \dots, q_k(x_k)] \rightarrow q(\zeta)$, where $k \geq 0$, $C \neq x_1$ is an X_k -context over Σ , $q_1, \dots, q_k, q \in Q$, and $\zeta \in T_\Delta(X_k)$ is linear in X_k . If $t_i \in \llbracket q_i \rrbracket_B(s_i)$ then $\llbracket q \rrbracket_B(C[s_1, \dots, s_k])$ contains the tree $\zeta[x_i \leftarrow \{t_i\} \mid i \in [k]]$. The translation realized by B is $\llbracket B \rrbracket = \cup_{q \in F} \llbracket q \rrbracket_B$. Extended linear bottom-up transducers are studied in, e.g., [3, 9, 21]. In [9, 21], the left-hand side of a rule is allowed to be of

the form $q(x_i)$, and the corresponding (larger) class of translations is denoted l-XBOT. It is easy to show that such rules can effectively be removed from a transducer B when it is known that $\llbracket B \rrbracket$ is a function.

As we show in Section 4, determinacy is undecidable if the view transducers copy. We therefore define views using linear transducers. We first show that for linear extended transducers, top-down (with look-ahead) gives the same translations as bottom-up, just as for non-extended transducers (see Theorem 2.8 of [7]). The following result was already pointed out below Proposition 5 in [9].

Theorem 1. $\text{ELT}^R = \text{ELB}$.

Proof. \subseteq : Let $M = (Q, \Sigma, \Delta, I, R, (P, \Sigma, F, \delta))$ be an ELT^R transducer. We construct the ELB transducer B with set of states $P \cup (Q \times P)$ and set of final states $I \times F$. Its rules are defined as follows. For the first set of rules let d_0 be a fixed element of $\Delta^{(0)}$. Let $p_1, \dots, p_k, p \in P$ and $a \in \Sigma^{(k)}$ such that $\delta(a, p_1, \dots, p_k) = p$. (1) We add $a(p_1(x_1), \dots, p_k(x_k)) \rightarrow p(d_0)$ as a rule of B . This rule outputs d_0 and changes the state to p , but recursive calls to it will only make use of the computed state, not the output. (2) If M has the rule $q(C) \rightarrow \zeta \langle p_1, \dots, p_k \rangle$, then we add $C[\text{state}(x_1), \dots, \text{state}(x_k)] \rightarrow \langle q, p \rangle(\text{erase}(\zeta))$ as a rule of B , where $\text{erase}(\zeta)$ is obtained from ζ by replacing every $q(x_i)$ by x_i , while $\text{state}(x_i) = \langle q, p_i \rangle(x_i)$ if q is the unique state such that $q(x_i)$ occurs in ζ and $\text{state}(x_i) = p_i(x_i)$ if no such q exists. The correctness of the construction follows from the following claim (for $q \in I$ and $p \in F$), which can be proved by a straightforward induction on the structure of s . Let $s \in T_\Sigma$, $q \in Q$, and $p \in P$.

Claim. (1) $\delta(s) = p$ if and only if $\llbracket p \rrbracket_B(s) \neq \emptyset$. (2) If $s \in \text{dom}(\llbracket \langle q, p \rangle \rrbracket_B)$, then $p = \delta(s)$. (3) $\llbracket \langle q, \delta(s) \rangle \rrbracket_B(s) = \llbracket q \rrbracket_M(s)$.

\supseteq : By the proof of Lemma 6 of [9], ELB is included in the class BIM of all tree translations $\{(f(s), g(s)) \mid s \in L\}$ where f is a linear non-deleting non-erasing tree homomorphism, g is a linear tree homomorphism, and L is a regular tree language. By Theorem 17 of [22], $\text{BIM} = \text{ELT}^R$. \square

We now give a useful property of the composition closure of ELT^R .

Lemma 2. If $\tau \in (\text{ELT}^R)^*$ and R is a regular tree language, then $\text{dom}(\tau)$, $\text{ran}(\tau)$, $\tau(R)$, and $\tau^{-1}(R)$ are regular tree languages.

Proof. We consider ELB translations, which suffices by Theorem 1. By Lemma 6 of [9], every ELB translation is of the form $\{(f(s), g(s)) \mid s \in L\}$ where f, g are linear tree homomorphisms and L is a regular tree language. From this it follows (as stated in Corollary 7 of [9]) that ELB translations preserve regularity. This implies that $\text{ran}(\tau)$ is regular. The above form means that inverse ELB translations are also of that form and hence preserve regularity. This implies that $\text{dom}(\tau)$ is regular. \square

Functionality Test. Later when we prove determinacy results, we restrict our views to classes of transducers that realize *functions*. In particular, we use the class $(\text{fu-ELT}^R)^*$ of compositions of functional translations in ELT^R , which properly contains fu-ELT^R by the proof of Theorem 5.2 of [23]. It is therefore important to know the next proposition.

Proposition 3. For an ELT^R transducer M it is decidable whether $\llbracket M \rrbracket$ is functional.

Proof. By Theorem 4.8 of [23], $\text{ET}^R = \text{T}^R$. The result follows because functionality is decidable for T^R transducers by [16] (see the sentence after Theorem 8 of [16]). \square

We note that it can be shown, using a variation of the Lemma of [8], that our class $(\text{fu-ELT}^R)^*$ is equal to the class $\text{fu-}(\text{ELT}^R)^*$ of functional compositions of ELT^R translations. However, we do not know whether functionality is decidable for such compositions. Note also that it was recently shown in [18] that $(\text{ELT}^R)^3 = (\text{ELT}^R)^*$.

Ordinary Top-Down Tree Transducers. The ET^R transducer M is an (ordinary, not extended) *top-down tree transducer with regular look-ahead* (T^R transducer) if the left-hand side C of each of its rules contains exactly one symbol in Σ , i.e., each rule is of the form $q(a(x_1, \dots, x_k)) \rightarrow \zeta \langle p_1, \dots, p_k \rangle$ with $a \in \Sigma^{(k)}$ and $k \geq 0$. A T^R transducer is *deterministic* if it has exactly one initial state and for every state q and input symbol a it has at most one rule with left-hand side $q(a(x_1, \dots, x_k))$. Determinism is denoted by the letter “D”, thus we have DT^R and DLT^R transducers. A T^R transducer M is *finite-copying* (a T_{fc}^R transducer) if each input node is translated only a bounded number of times. Formally this means there exists a number K such that for every $p \in P$, $s \in T_\Sigma(\{\square\})$, and $t \in \llbracket M_p \rrbracket(s)$, if \square occurs exactly once in s , then \square occurs $\leq K$ times in t ; here we assume that \square is a new input and output symbol of rank 0, and that M_p is M extended with the look-ahead transition $\delta(\square) = p$ and the rules $q(\square) \rightarrow \square$ for every state q . A DT_{fc}^R transducer is a deterministic T_{fc}^R transducer. Note that $\text{LT}^R \subseteq \text{T}_{\text{fc}}^R$ and that translations τ in T_{fc}^R are of linear size increase [11], i.e., there is a number N such that the size of t is at most N times the size of s for every $(s, t) \in \tau$.

We later need the following four results. Let DMSOTT be the class of deterministic (or, parameterless) MSO definable tree translations (see, e.g., Chapter 8 of [5]).

Proposition 4. (1) $\text{DT}_{\text{fc}}^R \circ \text{DT}_{\text{fc}}^R \subseteq \text{DT}_{\text{fc}}^R$, (2) $\text{DT}_{\text{fc}}^R \subseteq \text{DMSOTT}$, (3) $\text{DT}^R \circ \text{DT}^R \subseteq \text{DT}^R$, and (4) $\text{DMSOTT} \circ \text{DMSOTT} \subseteq \text{DMSOTT}$.

For result (4) see, e.g., [5]. Results (1) and (2) follow from Proposition 2 of [4] and Theorem 7.4 of [10]. Result (1) is already mentioned in Theorem 5.4 of [14]. Result (3) is proved in Theorem 2.11 of [7]. It is not difficult to prove (1) and (3) directly via straightforward product constructions.

4 Undecidability Results

Let HOM denote the class of translations realized by tree homomorphisms. The latter are total deterministic one-state tree transducers (not extended), defined either top-down or bottom-up (both coincide, cf. Lemma 3.2 of [6]). As observed in [21], a function v is injective if and only if q is determined by v , where q is the identity on $\text{dom}(v)$. Since the injectivity problem for HOM is undecidable by [17], one obtains (as stated in Theorem 3 of [21]) undecidability of the determinacy problem for ID under HOM , where ID is the class of identity translations on T_Σ , for any ranked alphabet Σ .

We show that determinacy is undecidable for ID under total deterministic copy-once top-down tree transducers ($\text{tot-DT}_{\text{co}}$ transducers). A DT transducer is *total* if for

each state q and input symbol a , it has a rule with left-hand side $q(a(x_1, \dots, x_k))$. It is *copy-once* if for every rule $q(a(x_1, \dots, x_k)) \rightarrow \zeta$ the initial state q_0 does not occur in ζ , and ζ is linear in X if $q \neq q_0$. Thus the transducer copies at most once, at the root node of the input tree. The undecidability of injectivity for non-total DT_{co} transducers was proved by Ésik in [15], and in his PhD thesis (in Hungarian). Our proof for total DT_{co} transducers (given in the Appendix) is a slight variation of Ésik’s proof.

Theorem 5. Determinacy for ID under $\text{tot-DT}_{\text{co}}$ is undecidable.

Since, obviously, every DT_{co} transducer is a DT_{fc} transducer, and DT_{fc} is (effectively) included in DMSOTT by Proposition 4(2), this immediately gives undecidability of determinacy for ID under DMSOTT (which slightly strengthens Theorem 4 of [21]).

One often considers determinacy for a query q under a *set of views* \mathcal{V} . The extended definition states that if two inputs give the same output for each view in \mathcal{V} , then they give the same output for q . In this case one has undecidability even when the views are deterministic finite-state word transformations. Thus, in what follows we consider only a single non-copying view.

5 Inverses and Uniformizers of Linear Extended Transducers

As Theorem 5 shows, determinacy cannot be decided under view transducers that copy, not even for a single initial copy at the input root node. Let us therefore restrict our attention to classes induced by linear view transducers. The results in this section hold for arbitrary linear extended transducers. When we want to decide determinacy in Section 6, we restrict the views to *functional* linear translations.

5.1 Inverses of Extended Linear Bottom-Up Transducers

Given an ELB transducer B , we would like to construct a transducer realizing its *inverse* $\llbracket B \rrbracket^{-1}$. In general this is not a function, but a relation over $T_\Delta \times T_\Sigma$. We prove a “decomposition theorem” showing that this inverse can be constructed as a composition of two translations. Since B can map the set of all input trees in T_Σ to a single output tree, a transducer realizing $\llbracket B \rrbracket^{-1}$ may need to map a tree back to any tree in T_Σ . This is not possible by conventional top-down or bottom-up tree transducers because the height of an output tree is linearly bounded by the height of the input tree. The next lemma “factors out” this problem by decomposing an ELB transducer into a part that can be inverted as a top-down transducer, and a component of a very simple form – a “projection mapping”. Let n-ELB denote the class of *non-deleting non-erasing* ELB transducers: those in which every rule is of the form $C[q_1(x_1), \dots, q_k(x_k)] \rightarrow q(\zeta)$, such that each variable in X_k occurs in ζ and $\zeta \neq x_1$. The phrase “non-deleting” indicates that we do not drop an input x_i , thus removing an entire subtree from the input. Non-erasing indicates that we do not have a rule such as $q(a(x_1, b)) \rightarrow q'(x_1)$, which “erases” the symbols a and b . Let Δ be a ranked alphabet and H a set of symbols disjoint from Δ each of rank at least 1. The *projection mapping from $\Delta \cup H$ to Δ* is the tree homomorphism $\pi_{\Delta, H} = \pi : T_{\Delta \cup H} \rightarrow T_\Delta$ defined as: $\pi(h(s_1, \dots, s_k)) = \pi(s_1)$ for $h \in H^{(k)}$ and $\pi(d(s_1, \dots, s_k)) = d(\pi(s_1), \dots, \pi(s_k))$ for $d \in \Delta^{(k)}$, for all $s_1, \dots, s_k \in T_{\Delta \cup H}$. We denote by PROJ the class of all projection mappings.

Lemma 6. $\text{ELB} \subseteq \text{n-ELB} \circ \text{PROJ}$.

Proof. Let $B = (Q, \Sigma, \Delta, F, R)$ be an ELB transducer and let m be the maximal number of variables that occur in the left-hand side of any rule of B . We define the ranked alphabet $H = \{\#_n^{(n+1)} \mid 0 \leq n \leq m\}$ and the n-ELB transducer $B' = (Q, \Sigma, \Delta \cup H, F, R')$. For every rule $C[q_1(x_1), \dots, q_k(x_k)] \rightarrow q(\zeta)$ in R we let the rule $C[q_1(x_1), \dots, q_k(x_k)] \rightarrow q(\#_n(\zeta, x_{i_1}, \dots, x_{i_n}))$ be in R' , where x_{i_1}, \dots, x_{i_n} are all the variables from X_k that do not occur in ζ . Clearly, $\llbracket B' \rrbracket \circ \pi_{\Delta, H} = \llbracket B \rrbracket$. \square

As shown in [3], the inverse of an n-ELB can be converted to an ELT by just “inverting the rules”.

Lemma 7. $\text{n-ELB}^{-1} \subseteq \text{ELT}$.

Proof. Let $B = (Q, \Sigma, \Delta, F, R)$ be an n-ELB transducer. We construct the ELT transducer $M = (Q, \Delta, \Sigma, I, R')$ realizing B 's inverse. Let $I = F$ and for every rule $C[q_1(x_1), \dots, q_k(x_k)] \rightarrow q(\zeta)$ in R let the rule $q(\zeta) \rightarrow C[q_1(x_1), \dots, q_k(x_k)]$ be in R' . It should be clear that $\llbracket M \rrbracket = \llbracket B \rrbracket^{-1}$. \square

5.2 Uniformizers

Let $\tau \subseteq A \times B$ be a translation and u a function from A to B . We say that u is a *uniformizer* of τ if $u \subseteq \tau$ and $\text{dom}(u) = \text{dom}(\tau)$. For classes of translations \mathcal{T}, \mathcal{U} we say that \mathcal{T} has *uniformizers in* \mathcal{U} if for every $\tau \in \mathcal{T}$ we can construct a uniformizer u of τ such that $u \in \mathcal{U}$. We say that the sequence τ_1, \dots, τ_n of translations is *compatible*, if for $i \in [n-1]$, $\text{ran}(\tau_i) \subseteq \text{dom}(\tau_{i+1} \circ \dots \circ \tau_n)$. It is easy to see that if u_1, \dots, u_n are uniformizers of τ_1, \dots, τ_n , respectively, and τ_1, \dots, τ_n is compatible, then $u_1 \circ \dots \circ u_n$ is a uniformizer of $\tau_1 \circ \dots \circ \tau_n$. Our goal is to show that $((\text{ELT}^R)^*)^{-1}$ and $(\text{ELT}^R)^*$ have uniformizers in DT_{fc}^R . We do this by decomposing into compatible translations, constructing uniformizers in DT_{fc}^R for them, and then obtaining a uniformizer in DT_{fc}^R through Proposition 4(1). A similar idea was used in [8] to obtain uniformizers for compositions of top-down and bottom-up tree translations.

Lemma 8. ELT^R has uniformizers in DT_{fc}^R .

Proof. By Theorem 4.8 of [23], $\text{ET}^R = \text{T}^R$. For a T^R transducer M with $\llbracket M \rrbracket \in \text{ELT}^R$, we construct a dbta A recognizing its domain (cf. Corollary 2.7 of [7]). We now change M so that the look-ahead automaton checks M 's domain (by building a product automaton with A and changing the rules of the transducer appropriately). The resulting transducer can be decomposed (by an obvious variant of Theorem 2.6 of [7]) into a finite state relabeling B with the same domain as M , followed by a top-down tree transducer T . Note that $\llbracket B \rrbracket$ is a function; for the notion of finite-state relabeling see Definition 3.14 in [6]. It follows that $\text{ran}(\llbracket B \rrbracket) \subseteq \text{dom}(\llbracket T \rrbracket)$. Thus, $\llbracket B \rrbracket, \llbracket T \rrbracket$ are compatible. A finite state relabeling can be seen as a top-down tree transducer, so by the Lemma in [8] we obtain uniformizers for $\llbracket B \rrbracket$ and $\llbracket T \rrbracket$, both in DT^R . Since DT^R is closed under composition by Proposition 4(3), the composition of these uniformizers is a uniformizer of $\llbracket M \rrbracket$, in DT^R . Obviously, ELT^R translations are of linear size increase, and so this uniformizer

is of linear size increase. We obtain the desired result because DT^R translations of linear size increase are in DT_{fc}^R by Section 7.1 of [11] (in fact, by the obvious generalization of the latter result to partial transducers: introduce output dummies for undefined rules and remove them later). \square

Note that there is an alternative proof to Lemma 8 which avoids the last step (of applying linear size increase): (1) From the construction in the proof of Theorem 4.8 of [23] it follows that $\text{ELT}^R \subseteq \text{T}_{\text{fc}}^R$. (2) The proof of the Lemma in [8] can easily be modified into a proof that every T^R transducer has a uniformizer in DT^R , and the proof preserves the finite-copying property.

An FTA *transducer* is a dbta A , seen as a tree transducer realizing the translation $\llbracket A \rrbracket$, which is the identity function on $L(A)$; composing a tree translation τ with $\llbracket A \rrbracket$ amounts to restricting the range of τ to $L(A)$: $\tau \circ \llbracket A \rrbracket = \{(s, t) \in \tau \mid t \in L(A)\}$.

Lemma 9. $\text{PROJ}^{-1} \circ \text{FTA}$ has uniformizers in DLT^R .

Proof. Let $\tau = \pi^{-1} \circ \llbracket A \rrbracket$ where $\pi \in \text{PROJ}$ and A is an FTA transducer. Thus, $\pi = \pi_{\Sigma, H}$ for disjoint ranked alphabets Σ and H such that $H^{(0)} = \emptyset$, and A is a dbta $(Q, \Sigma \cup H, F, \delta)$. Let \mathcal{C} be the set of all X_1 -contexts C over $\Sigma \cup H$ such that the left-most leaf of C has label x_1 and all the ancestors of this leaf have labels in H . For every $a \in \Sigma^{(k)}$ and $q, q_1, \dots, q_k \in Q$, let $\mathcal{C}(a, q, q_1, \dots, q_k)$ be the set of $C \in \mathcal{C}$ such that $\delta(C[a(t_1, \dots, t_k)]) = q$ for all $t_1, \dots, t_k \in T_{\Sigma \cup H}$ with $\delta(t_i) = q_i$ for every $i \in [k]$. Let $C_0(a, q, q_1, \dots, q_k)$ be one (fixed) such C – since the set $\mathcal{C}(a, q, q_1, \dots, q_k)$ is effectively regular, one can always compute such an element C if the set is nonempty. If there does not exist such a C then $C_0(a, q, q_1, \dots, q_k)$ is undefined.

Since the construction in the proof of the Lemma of [8] preserves linearity, LT has uniformizers in DLT^R . Hence, it suffices to construct an LT transducer M that is a “specializer” of τ , i.e., $\llbracket M \rrbracket \subseteq \tau$ and $\text{dom}(\llbracket M \rrbracket) = \text{dom}(\tau)$. We define $M = (Q, \Sigma, \Sigma \cup H, I, R')$ with $I = F$ and R' consists of all rules $q(a(x_1, \dots, x_k)) \rightarrow C_0(a, q, q_1, \dots, q_k)[a(q_1(x_1), \dots, q_k(x_k))]$ such that $C_0(a, q, q_1, \dots, q_k)$ is defined. Intuitively, for $s \in T_\Sigma$, M simulates top-down the state behavior of A on some tree t in $\pi^{-1}(s)$ and, at each node of s , outputs a context in \mathcal{C} on which A has the same state behavior as on the context in \mathcal{C} that is “above” the corresponding node in t . Formally, the correctness of the construction follows from the following claim (for $q \in F$), which can easily be proved by structural induction on s and induction on the size of t , respectively. Let $q \in Q$, $s \in T_\Sigma$, and $t \in T_{\Sigma \cup H}$.

Claim. (1) If $t \in \llbracket q \rrbracket_M(s)$, then $\pi(t) = s$ and $\delta(t) = q$. (2) If $\delta(t) = q$, then $\pi(t) \in \text{dom}(\llbracket q \rrbracket_M)$.

In both proofs one uses that $\pi(C[t]) = \pi(t)$ for every $C \in \mathcal{C}$. In the proof of (2) one uses that t is of the form $C[a(t_1, \dots, t_k)]$ with $C \in \mathcal{C}$, $k \geq 0$, $a \in \Sigma^{(k)}$ and $t_1, \dots, t_k \in T_{\Sigma \cup H}$, and one applies the induction hypothesis to t_1, \dots, t_k . \square

Lemma 10. ELB^{-1} has uniformizers in DT_{fc}^R .

Proof. Let $\tau \in \text{ELB}^{-1}$. By Lemmas 6 and 7, $\tau \in \text{PROJ}^{-1} \circ \text{ELT}$. The domains of translations in ELT are effectively regular by Lemma 2, thus we obtain $\tau = \tau_1 \circ \tau_2$ such that the translations $\tau_1 \in \text{PROJ}^{-1} \circ \text{FTA}$, $\tau_2 \in \text{ELT}$ are compatible (by definition of the

FTA transducer). For τ_1, τ_2 we obtain, by Lemmas 9 and 8, uniformizers $u_1, u_2 \in \text{DT}_{\text{fc}}^{\text{R}}$. Then $u_1 \circ u_2$ is a uniformizer for τ ; it is in $\text{DT}_{\text{fc}}^{\text{R}}$ by Proposition 4(1). \square

Theorem 11. $((\text{ELT}^{\text{R}})^*)^{-1}$ has uniformizers in $\text{DT}_{\text{fc}}^{\text{R}}$.

Proof. Let T_1, \dots, T_n be ELT^{R} transducers. We change the T_i so that the sequence of translations $\llbracket T_1 \rrbracket^{-1}, \dots, \llbracket T_n \rrbracket^{-1}$ is compatible, i.e., $\text{ran}(\llbracket T_i \rrbracket^{-1}) \subseteq \text{dom}(\llbracket T_{i+1} \rrbracket^{-1} \circ \dots \circ \llbracket T_n \rrbracket^{-1})$. We change the domain of T_i to be included in the range of $\llbracket T_n \rrbracket \circ \dots \circ \llbracket T_{i+1} \rrbracket$. This range is regular by Lemma 2. The domain of T_i is changed using look-ahead, as in the proof of Lemma 8. Using Theorem 1 and Lemma 10 we obtain uniformizers in $\text{DT}_{\text{fc}}^{\text{R}}$ for the $\llbracket T_i \rrbracket^{-1}$. This proves the theorem, by Proposition 4(1). \square

Theorem 12. $(\text{ELT}^{\text{R}})^*$ has uniformizers in $\text{DT}_{\text{fc}}^{\text{R}}$.

Proof. Let T_1, \dots, T_n be ELT^{R} transducers. We change the T_i so that $\llbracket T_1 \rrbracket, \dots, \llbracket T_n \rrbracket$ is compatible, i.e., restrict T_i 's range to $D = \text{dom}(\llbracket T_{i+1} \rrbracket \circ \dots \circ \llbracket T_n \rrbracket)$, which is regular by Lemma 2. The range of T_i can be restricted to D as follows. As mentioned in the proof of Theorem 1, ELT^{R} is the class of all translations of the form $\tau = \{(f(s), g(s)) \mid s \in L\}$ where f is a linear non-deleting non-erasing tree homomorphism, g is a linear tree homomorphism, and L is a regular tree language. The restriction of the range of τ to D is $\{(f(s), g(s)) \mid s \in L, g(s) \in D\} = \{(f(s), g(s)) \mid s \in L \cap g^{-1}(D)\}$. Since $L \cap g^{-1}(D)$ is regular, this translation is again of the above form and hence in ELT^{R} . We obtain uniformizers in $\text{DT}_{\text{fc}}^{\text{R}}$ for the $\llbracket T_i \rrbracket$ by Lemma 8, and a uniformizer for $\llbracket T_1 \rrbracket \circ \dots \circ \llbracket T_n \rrbracket$ by Proposition 4(1). \square

6 Decidability of Determinacy and Rewriting

Consider a query q , a view v , and a uniformizer u of v^{-1} , each of them a partial function. Clearly, q is determined by v if and only if $v \circ u \circ q = q$. For queries in DT^{R} or DMSOTT , equivalence is decidable [13, 12], and they are closed under left composition with $\text{DT}_{\text{fc}}^{\text{R}}$ by Proposition 4. Thus, if v and u are in $\text{DT}_{\text{fc}}^{\text{R}}$, then we can decide determinacy. We will show that this holds for the views in the class $(\text{fu-ELT}^{\text{R}})^*$ of compositions of functions in ELT^{R} . If v is in this class, then v^{-1} has a uniformizer u in $\text{DT}_{\text{fc}}^{\text{R}}$ by Theorem 11. As the next corollary states, v itself is also in $\text{DT}_{\text{fc}}^{\text{R}}$. The inclusion is a direct consequence of Theorem 12; it is proper by Lemma 2 (because $\text{DT}_{\text{fc}}^{\text{R}}$ contains translations that do not preserve regularity).

Corollary 13. $(\text{fu-ELT}^{\text{R}})^* \subsetneq \text{DT}_{\text{fc}}^{\text{R}}$.

The main results of this paper are presented in the next two theorems.

Theorem 14. Determinacy is decidable for DT^{R} and DMSOTT under $(\text{fu-ELT}^{\text{R}})^*$.

Proof. Let $v \in (\text{fu-ELT}^{\text{R}})^*$. According to Corollary 13 and Theorem 11 we construct $\text{DT}_{\text{fc}}^{\text{R}}$ transducers M_1, M_2 such that $\llbracket M_1 \rrbracket = v$ and $\llbracket M_2 \rrbracket = u$ is a uniformizer of v^{-1} . If a query is given as a DT^{R} (DMSOTT) transducer N , then DT^{R} (DMSOTT) transducers N' and N'' can be constructed with $\llbracket N' \rrbracket = u \circ \llbracket N \rrbracket$ and $\llbracket N'' \rrbracket = v \circ \llbracket N' \rrbracket = v \circ u \circ \llbracket N \rrbracket$. This follows from Proposition 4. We can decide if $\llbracket N'' \rrbracket = \llbracket N \rrbracket$ because equivalence is decidable for DT^{R} and DMSOTT transducers [13, 12]. \square

The proof of Theorem 14 also proves Theorem 15.

Theorem 15. Let $\mathcal{V} = (\text{fu-ELT}^R)^*$, $v \in \mathcal{V}$, and let N be a DT^R (DMSOTT) transducer such that $\llbracket N \rrbracket$ is determined by v . A DT^R (DMSOTT) transducer N' can be constructed such that $v \circ \llbracket N' \rrbracket = \llbracket N \rrbracket$. That is, DT^R (DMSOTT) is complete for \mathcal{V} -to- DT^R (\mathcal{V} -to-DMSOTT) rewritings.

Since the class fu-B of functional bottom-up translations is included in DT^R by [8], it is immediate from Theorems 14 and 1 that determinacy is decidable for fu-B under fu-ELB, as proved in Theorem 2 of [21]. In Theorem 5 of [21] it is shown to be decidable for $q \in \text{fu-B}$ and $v \in \text{fu-ELB}$ whether there exists $f \in \text{fu-B}$ such that $q = v \circ f$. Theorem 15 shows that such an f can always be found in DT^R .

Weakly Determined Queries. A query q is determined by a view v if there exists a function f such that (1) $\text{dom}(v \circ f) = \text{dom}(q)$ and (2) $f(v(s)) = q(s)$ for every $s \in \text{dom}(q)$. For practical purposes, condition (1) could be weakened to $\text{dom}(v \circ f) \supseteq \text{dom}(q)$. For a given element s , one first checks if $s \in \text{dom}(q)$, and if so, obtains $q(s)$ as $f(v(s))$. We say that q is *weakly determined by v* if there exists f with $f(v(s)) = q(s)$ for every $s \in \text{dom}(q)$. As an example consider $q = \{(1, 1)\}$ and $v = \{(1, 1), (2, 1)\}$. Then q is *not* determined by v , but is weakly determined. Let $\mathcal{Q}, \mathcal{V}, \mathcal{Q}'$ be classes of partial functions. We say that \mathcal{Q}' is *complete for weak \mathcal{V} -to- \mathcal{Q} rewritings*, if for every $q \in \mathcal{Q}$ and $v \in \mathcal{V}$ such that q is weakly determined by v , there is an $f \in \mathcal{Q}'$ with $f(v(s)) = q(s)$ for every $s \in \text{dom}(q)$. We now show that our main results also hold for weak determinacy and weak rewritings. For a function $\tau : A \rightarrow B$ and a set L let $\tau \upharpoonright L$ denote the restriction of τ to inputs in L . Then q is weakly determined by v if and only if q is determined by $v \upharpoonright \text{dom}(q)$, with the same functions f . For $q \in \text{DT}^R$ or $q \in \text{DMSOTT}$, $\text{dom}(q)$ is effectively regular. And if $v \in (\text{fu-ELT}^R)^*$ then (effectively) $v \upharpoonright L$ is in $(\text{fu-ELT}^R)^*$ for every regular tree language L (simply by adding it to the look-ahead of the first transducer). This shows that Theorems 14 and 15 also hold for weak determinacy.

Corollary 16. Let $\mathcal{V} = (\text{fu-ELT}^R)^*$. Weak determinacy is decidable for DT^R and for DMSOTT under \mathcal{V} . The classes DT^R and DMSOTT are complete for weak \mathcal{V} -to- DT^R and weak \mathcal{V} -to-DMSOTT rewritings, respectively.

Future Work. We would like to know the complexity of deciding determinacy. The complexity of our algorithm is dominated by that of the equivalence tests in [13, 12]: double exponential time for DT^R , non-elementary for DMSOTT (and nondeterministic exponential time for streaming tree transducers [2]). Can we find subclasses of tree transducers for which determinacy is polynomial-time testable? Can our results be extended to larger classes of tree transducers, such as deterministic macro tree transducers (see, e.g., [10, 11])? For those transducers, decidability of equivalence is a long-standing open problem. It is interesting and practically important (for XML) to study determinacy for unranked tree transducers [26].

References

1. F. N. Afrati. Determinacy and query rewriting for conjunctive queries and views. *Theor. Comput. Sci.*, 412(11):1005–1021, 2011.

2. R. Alur and D. L. Streaming tree transducers. In *ICALP* (2), pages 42–53, 2012.
3. A. Arnold and M. Dauchet. Bi-transductions de forêts. In *ICALP*, 1976.
4. R. Bloem and J. Engelfriet. A comparison of tree transductions defined by monadic second order logic and by attribute grammars. *J. Comput. Syst. Sci.*, 61(1):1–50, 2000.
5. B. Courcelle and J. Engelfriet. *Graph Structure and Monadic Second-Order Logic, a Language-Theoretic Approach*. Cambridge University Press, 2012.
6. J. Engelfriet. Bottom-up and top-down tree transformations - a comparison. *Math. Systems Theory*, 9(3):198–231, 1975.
7. J. Engelfriet. Top-down tree transducers with regular look-ahead. *Math. Systems Theory*, 10:289–303, 1977.
8. J. Engelfriet. On tree transducers for partial functions. *Inform. Proc. Letters*, 7(4):170–172, 1978.
9. J. Engelfriet, E. Lilin, and A. Maletti. Extended multi bottom-up tree transducers. *Acta Inf.*, 46(8):561–590, 2009.
10. J. Engelfriet and S. Maneth. Macro tree transducers, attribute grammars, and MSO definable tree translations. *Inf. Comput.*, 154(1):34–91, 1999.
11. J. Engelfriet and S. Maneth. Macro tree translations of linear size increase are mso definable. *SIAM J. Comput.*, 32(4):950–1006, 2003.
12. J. Engelfriet and S. Maneth. The equivalence problem for deterministic MSO tree transducers is decidable. *Inform. Proc. Letters*, 100(5):206–212, 2006.
13. J. Engelfriet, S. Maneth, and H. Seidl. Deciding equivalence of top-down XML transformations in polynomial time. *J. Comput. Syst. Sci.*, 75(5):271–286, 2009.
14. J. Engelfriet, G. Rozenberg, and G. Slutzki. Tree transducers, L systems, and two-way machines. *J. Comput. Syst. Sci.*, 20(2):150–202, 1980.
15. Z. Ésik. On decidability of injectivity of tree transformations. In *Les arbres en algèbre et en programmation, Lille*, pages 107–133, 1978.
16. Z. Ésik. Decidability results concerning tree transducers I. *Acta Cybern.*, 5:1–20, 1981.
17. Z. Fülöp and P. Gyenezse. On injectivity of deterministic top-down tree transducers. *Inform. Proc. Letters*, 48(4):183–188, 1993.
18. Z. Fülöp and A. Maletti. Composition closure of linear extended top-down tree transducers. *CoRR*, abs/1301.1514, 2013.
19. B. Groz. *XML Security Views: Queries, Updates, and Schemas*. PhD thesis, Université Lille 1, 2012.
20. B. Groz, S. Staworko, A.-C. Caron, Y. Roos, and S. Tison. Static analysis of XML security views and query rewriting. *Inf. Comput.*, 2013. To appear.
21. K. Hashimoto, R. Sawada, Y. Ishihara, H. Seki, and T. Fujiwara. Determinacy and subsumption for single-valued bottom-up tree transducers. Technical Report NAIST-IS-TR2012002, Nara Institute of Science and Technology.
22. A. Maletti. Compositions of extended top-down tree transducers. *Inf. Comput.*, 206(9-10):1187–1196, 2008.
23. A. Maletti, J. Graehl, M. Hopkins, and K. Knight. The power of extended top-down tree transducers. *SIAM J. Comput.*, 39(2):410–430, 2009.
24. A. Nash, L. Segoufin, and V. Vianu. Views and queries: Determinacy and rewriting. *ACM Trans. Database Syst.*, 35(3), 2010.
25. D. Pasailă. Conjunctive queries determinacy and rewriting. In *ICDT*, 2011.
26. T. Perst and H. Seidl. Macro forest transducers. *Inform. Proc. Letters*, 89(3):141–149, 2004.
27. L. Segoufin and V. Vianu. Views and queries: determinacy and rewriting. In *PODS*, 2005.
28. H. Seidl. Equivalence of finite-valued tree transducers is decidable. *Math. Systems Theory*, 27(4):285–346, 1994.

Appendix

Proof of Undecidability (from Section 4)

Here is the proof of Theorem 5, which states that determinacy for ID under tot-DT_{co} is undecidable.

Proof. We use an encoding of the Modified Post Correspondence Problem (MPCP). An instance of this problem consists of two sequences of words $(\alpha_1, \dots, \alpha_n)$ and $(\beta_1, \dots, \beta_n)$, and has a solution if there exists a sequence of numbers i_1, \dots, i_k such that (i) $i_1 = 1$ and (ii) the words $\alpha_{i_1} \dots \alpha_{i_k}$ and $\beta_{i_1} \dots \beta_{i_k}$ are equal. It is known to be undecidable whether an MPCP instance has a solution. Let Ω be the ranked alphabet consisting of all symbols that appear in the words α_i and β_i , where each symbol has rank one. For a word $\alpha = a_1 \dots a_m$ and a tree t , we write $\alpha(t)$ to mean the tree $a_1(a_2(\dots a_m(t) \dots))$. We define the tot-DT_{co} transducer $M = (Q, \Sigma, \Delta, q_0, R)$ where $Q = \{q_0, q_a, q_b, q_{id}\}$, $\Sigma = \{i^{(1)} \mid i \in [n]\} \cup \{a^{(1)}, b^{(1)}, e^{(0)}\}$, and $\Delta = \Sigma \cup \Omega \cup \{f^{(2)}\}$. The state q_{id} realizes the identity on trees in T_Σ . For every $\sigma \in \Sigma^{(k)}$, $k \geq 0$, it has the rule $q_{id}(\sigma(x_1, \dots, x_k)) \rightarrow \sigma(q_{id}(x_1), \dots, q_{id}(x_k))$ in R . For $i \in [n]$ we define the following rules in R .

$$\begin{array}{ll} q_0(i(x_1)) \rightarrow i(q_{id}(x_1)) & q_a(i(x_1)) \rightarrow \alpha_i(q_a(x_1)) \\ q_0(e) \rightarrow e & q_b(i(x_1)) \rightarrow \beta_i(q_b(x_1)) \\ q_0(a(x_1)) \rightarrow f(q_{id}(x_1), \alpha_1(q_a(x_1))) & q_u(v(x_1)) \rightarrow e \text{ for all } u, v \in \{a, b\} \\ q_0(b(x_1)) \rightarrow f(q_{id}(x_1), \beta_1(q_b(x_1))) & q_u(e) \rightarrow e \text{ for all } u \in \{a, b\} \end{array}$$

The transducer M translates every input tree of which the root is not labeled by a or by b into itself. It translates every input tree $ai_2 \dots i_k(s)$ where the root of $s \in T_\Sigma$ is not a number, into $f(i_1 \dots i_k(s), \alpha_1 \alpha_{i_2} \dots \alpha_{i_k}(e))$, and similarly for b and β_{i_j} . Let d_Σ be the identity on T_Σ . Then d_Σ is determined by $\llbracket M \rrbracket$ if and only if $\llbracket M \rrbracket$ is injective if and only if the given MPCP instance has no solution. \square

For a set of views \mathcal{V} consisting of two deterministic finite-state word transducers, the proof is the same as for Theorem 5, but producing each branch of the output tree by a separate view.