Decision Problems of Tree Transducers with Origin [⋆]

Emmanuel Filiot[1], Sebastian Maneth[2], Pierre-Alain Reynier[3], and Jean-Marc Talbot[3]

[1] Université Libre de Bruxelles
[2] University of Edinburgh
[3] Aix-Marseille Université & CNRS

Abstract. A tree transducer with origin translates an input tree into a pair of output tree and origin info. The origin info maps each node in the output tree to the unique input node that created it. In this way, the implementation of the transducer becomes part of its semantics. We show that the landscape of decidable properties changes drastically when origin info is added. For instance, equivalence of nondeterministic top-down and MSO transducers with origin is decidable. Both problems are undecidable without origin. The equivalence of deterministic top-down tree-to-string transducers is decidable with origin, while without origin it is a long standing open problem. With origin, we can decide if a deterministic macro tree transducer can be realized by a deterministic top-down tree transducer; without origin this is an open problem.

Tree transducers were invented in the early 1970's as a formal model for compilers and linguistics [24, 23]. They are being applied in many fields of computer science, such as syntax-directed translation [13], databases [22, 15], linguistics [19, 4], programming languages [27, 21], and security analysis [16]. The most essential feature of tree transducers is their good balance between expressive power and decidability.

Bojańczyk [3] introduces (string) transducers with origin. For "regular" string-to-string transducers with origin he presents a machine independent characterization which admits Angluin-style learning and the decidability of natural subclasses. These results indicate that classes of translations with origin are mathematically better behaved than their origin-less counter parts.

We initiate a rigorous study of tree transducers with origin by investigating the decidability of *equivalence*, *injectivity* and *query determinacy* on the following models: top-down tree-to-tree transducers [24, 23], top-down tree-to-string transducers [11], and MSO definable tree-to-string transducers (see, e.g., [10]).

[⋆] *The authors are grateful to Joost Engelfriet for his remarks for improvements and corrections on a preliminary of this work.* This work has been carried out thanks to the support of the ARCHIMEDE Labex (ANR-11-LABX-0033) and the A*MIDEX project (ANR-11-IDEX-0001-02) funded by the "Investissements d'Avenir" French Government program, managed by the French National Research Agency (ANR) and by the PEPS project "Synthesis of Stream Processors" funded by CNRS.

	top-down tree-to-tree		top-down tree-to-string		MSO tree-to-string	
	det	nd	det	nd	det	nd
	+ [12]	− [14]	?	− [14]	+ [10]	−
with origin	+	+	+	−	+	+

Table 1. Decidability of equivalence

Unlike the string transducers of Bojańczyk [3], we will see that equivalent models of tree-to-string transducers do *not* remain equivalent in the presence of origin. This motivates the study of *subclass definability* problems (definability of a transduction from a class in a subclass) when considering the origin semantics.

Table 1 summarizes our results on equivalence; non-/deterministic are abbreviated by nd/det and decidable/undecidable by +/−. The "?" marks a long-standing open problem, already mentioned by Engelfriet [7]. The first change from − to + is the equivalence of nondeterministic top-down tree transducers. In the non-origin case this problem is already undecidable for restricted string-to-string transducers [14]. In the presence of origin it becomes decidable for tree transducers, because origin implies that any connected region of output nodes with the same origin is generated by one single rule. Hence, the problem reduces to letter-to-letter transducers [1]. What about nondeterministic top-down *tree-to-string* transducers (column four in Table 1)? Here output patterns cannot be treated as letters. By deferring output generation to a leaf they can simulate non-origin translations with undecidable equivalence [14]. Finally, we discuss column three. Here the origin information induces a structure on the output strings: recursive calls of origin-equivalent transducers must occur in similar "blocks", so that the same children of the current input node are visited in the same order (but possibly with differing numbers of recursive calls). This block structure allows to reason over single input paths, and to reduce the problem to deterministic tree-to-string transducers with monadic input. The latter can be reduced [20] to the famous HDT0L sequence equivalence problem.

Injectivity for deterministic transducers is undecidable for all origin-free models of Table 1. With origin, we prove undecidability in the tree-to-string case and decidability in the MSO and top-down tree cases. The latter is again due to the rigid structure implied by origins. We can track if two different inputs, over the *same* input nodes, produce the same output tree. We use the convenient framework of recognizable relations to show that the set of trees for which a transducer with origin produces the same output can be recognized by a tree automaton.

Motivation. Clearly, the more information we include in a transformation, the more properties become decidable. Consider invertability: on the one extreme, if all reads and writes are recorded (under ACID), then any computation becomes invertible. The question then arises, how much information needs to be included in order to be invertible. This problem has recently deserved much attention in the programming language community (see, e.g., [26]). Our work here was inspired by the very similar view/query determinacy problem. This problem asks for a given view and query, whether the query can be answered on the output of the view. It was shown decidable in [2] for views that are linear extended tree transducers, and queries that are deterministic MSO or top-down transducers. For views that include copying, the problem quickly becomes undecidable [2]. Our results show that such views *can* be supported, if origin is included. Consider for instance a view that regroups a list of publications into sublists of books, articles, etc. A tree transducer realizing this view *needs copying* (i.e., needs to process the original list multiple times). Without origin, we do not know a procedure that decides determinacy for such a view. With origin, we prove that determinacy is decidable. As expected: the world becomes safer with origin, but more restrictive (e.g., the query "is book X before article Y in the original list?" becomes determined when origin is added to the above view).

The tracking of origin information was studied in the programming language community, see [25]. As a technical tool it was used in [9] to characterize the MSO definable macro tree translations, and, in [17] to give a Myhill-Nerode theorem for deterministic top-down tree transducers. From a linguistic point of view, origin mappings on their own are subject of interest and are called "dependencies" or "links". Maletti [18] shows that dependencies (i.e., origins) give "surprising insights" into the structure of tree transformations: many separation results concerning expressive power can be obtained on the level of dependencies.

1 Preliminaries

For a nonnegative integer k we denote by $[k]$ the set $\{1, \ldots, k\}$. For an alphabet A, we denote by A^* the set of strings over A, and by ε the empty string. Let $w \in A^*$ be a string of length k. Its length is denoted by $|w| = k$, and its set of positions by $V(w) = [k]$. For $j \in [k]$, $w[j]$ denotes the j-th symbol of the string w. A *ranked alphabet* Σ is a finite set of symbols σ each with an associated natural k called its rank. We write $\sigma^{(k)}$ to denote that σ has rank k, and denote by $\Sigma^{(k)}$ the set of all symbols in Σ of rank k. The *set T_Σ of trees over Σ* is the smallest set T so that if $k \geq 0$, $t_1, \ldots, t_k \in T$, and $\sigma \in \Sigma^{(k)}$, then $\sigma(t_1, \ldots, t_k) \in T$. For the tree $\sigma()$ we simply write σ. The set $V(t)$ of nodes of tree $t \in T_\Sigma$ is the subset of $\mathbb{N}^*$ defined as $\{\varepsilon\} \cup \{iu \mid i \in [k], u \in V(t_i)\}$ if $t = \sigma(t_1, \ldots, t_k)$. Thus ε denotes the root node, and ui denotes the i-th child of a node u. For a tree t and $u \in V(t)$ we denote by $t[u]$ the label of node u in t, and by t/u the subtree of t rooted at u. For a tree t', we denote by $t[u \leftarrow t']$ the tree obtained from t by replacing the subtree rooted in position u by the tree t'. Given a tree $t \in T_\Sigma$ and $\Delta \subseteq \Sigma$, $V_\Delta(t)$ denotes the set of nodes $u \in V(t)$ such that $t[u] \in \Delta$.

Translations Let Σ, Δ be two ranked alphabets. A *tree translation* (*from T_Σ to T_Δ*) is a relation $R \subseteq T_\Sigma \times T_\Delta$. Let A be an alphabet. A *tree-to-string translation* is a relation $R \subseteq T_\Sigma \times A^*$. The *domain* of a translation R, denoted $\mathsf{dom}(R)$, is defined as the projection of R on its first component. A translation R is *functional* if R is a function.

Origin Translations Let s_1, s_2 be two structures (strings or trees). An *origin mapping* of s_2 in s_1 is a mapping $o : V(s_2) \to V(s_1)$. An *origin translation* is a set of pairs $(s_1, (s_2, o))$ such that o is an origin mapping of s_2 in s_1. Given $v \in V(s_2)$ and $u \in V(s_1)$, if $o(v) = u$ then we say that "v has origin u" or that "the origin of v is u".

2 Tree Translations with Origin

Top-down Transducers A top-down tree transducer (TOP for short) is a rule-based finite-state machine that translates trees over a ranked alphabet Σ to trees over a ranked alphabet Δ. Rules are of the form $q(\sigma(x_1, \ldots, x_k)) \to \zeta$, where q is a state, $\sigma \in \Sigma^{(k)}$ a symbol of rank k, and ζ is a tree over Δ of which the leaves may also be labeled with symbols of the form $q'(x_i)$, for some state q' and $i \in [k]$. Applying this rule to a tree $s = \sigma(s_1, \ldots, s_k)$ produces, on the output, a tree t obtained by replacing in ζ all symbols $q'(x_i)$ by a tree over Δ, itself obtained as a result of evaluating s_i in state q'. The origin of all the nodes of ζ labeled in Δ is precisely the root node of s.

As an example, consider a TOP M_1 over $\Sigma = \{h^{(1)}, a^{(0)}\}$ and $\Delta = \{f^{(2)}, a^{(0)}\}$ with single state q and rules $q(h(x_1)) \to f(q(x_1), q(x_1))$ and $q(a) \to a$. It translates a monadic input tree of height n into a full binary tree of height n. Thus, the origin of any output node u is the input node $1^{|u|}$.As another example, consider M_2 with states q_0 and q and the rules $q_0(h(x_1)) \to f(q_0(x_1), q(x_1))$, $q(h(x_1)) \to h(q(x_1))$ and $q_0(a) \to a$, $q(a) \to a$. This transducer translates a monadic input tree of height n into a left-comb of monadic subtrees of decreasing height. Thus, $h(h(h(a)))$ is translated into $f(f(f(a, a), h(a)), h(h(a)))$. Again, the origin of node u is $1^{|u|}$.

Formally, a *top-down tree transducer* M is a tuple $(Q, \Sigma, \Delta, q_0, R)$ where Q is a finite set of states, $q_0 \in Q$ is the initial state, and R is a set of rules of the form $q(\sigma(x_1, \ldots, x_k)) \to \zeta$, where ζ is a tree over $\Delta \cup \{q'(x_i) \mid q' \in Q, i \in [k]\}$, where each symbol $q'(x_i)$ has rank 0. Every state $q \in Q$ realizes an origin translation $[\![q]\!]_\mathrm{o}$ defined recursively as follows. Let $s = \sigma(s_1, \ldots, s_k)$ and ζ such that $q(\sigma(x_1, \ldots, x_k)) \to \zeta$ is a rule of M. Let $V = V(\zeta) \setminus V_\Delta(\zeta)$. For every $v \in V$, let $(s_{i_v}, (t_v, o_v)) \in [\![q_v]\!]_\mathrm{o}$ where $q_v \in Q$ and $i_v \in [k]$ such that $\zeta[v] = q_v(x_{i_v})$. Then $(s, (t, o)) \in [\![q]\!]_\mathrm{o}$ with

- $t = \zeta[v \leftarrow t_v \mid v \in V]$,
- $o(v') = \varepsilon$ for $v' \in V_\Delta(\zeta)$ and $o(vv') = i_v o_v(v')$ for $v \in V$ and $v' \in V(t_v)$.

The translation realized by q is defined as $[\![q]\!] = \{(s, t) \mid \exists o : (s, (t, o)) \in [\![q]\!]_\mathrm{o}\}$. The *origin* (*tree*) *translation realized by* M is $[\![M]\!]_\mathrm{o} = [\![q_0]\!]_\mathrm{o}$, and the (*tree*) *translation realized by* M is $[\![M]\!] = [\![q_0]\!]$.

Note that TOPs are forced to produce at least one symbol when they read the leaf of an input tree. To inspect parts of the input tree without producing output, a TOP can be equipped with *regular look-ahead*, leading to the class of top-down tree transducers with regular look-ahead (TOP$^\mathrm{R}$). This is done by changing the rules so that each left-hand side is of the form $q(\sigma(x_1, \ldots, x_k) : L)$ where L is a regular tree language. Such a rule can be applied only if the input tree $\sigma(s_1, \ldots, s_k)$ is in L. Alternatively, instead of L a state p of some given finite tree automaton can be used. A TOP$^{\mathrm{R}}$ M is *deterministic* if for any two rules with left-hand sides $q(\sigma(x_1,\ldots,x_k) : L_1)$ and $q(\sigma(x_1,\ldots,x_k) : L_2)$, we have $L_1 \cap L_2 = \emptyset$. Note that any TOP M can be transformed into a TOP$^{\mathrm{R}}$ M^R by adding universal look-ahead languages. M is deterministic if M^R is. The classes of deterministic top-down tree transducers without and with regular look-ahead are respectively denoted by DTOP and DTOP$^{\mathrm{R}}$. Note that DTOP and DTOP$^{\mathrm{R}}$ realize only functional translations (and functional origin translations).

MSO Transducers Deterministic MSO tree transducers (DMSOT for short) are logic-based tree transducers defined over monadic second-order logic (MSO). Any tree s over a ranked alphabet Σ of maximal rank k is seen as a logical structure of domain $V(s)$ over the node label predicates $\sigma(x)$, $\sigma \in \Sigma$, and successor predicates $i(x,y)$, $1 \leq i \leq k$, that relate a node x to its i-th child y. By MSO[Σ] we denote all monadic second-order formulas over this signature, and write $s \models \phi$ whenever a tree s satisfies a formula $\phi \in$ MSO[Σ]. Let Δ be a ranked alphabet of maximal rank ℓ. To define the output tree $t \in T_\Delta$ of an input tree $s \in T_\Sigma$, a DMSOT uses MSO[Σ] formulas with one or two free variables, interpreted over a fixed number of copies of s, to define the predicates of t over Δ. Formally, a DMSOT from T_Σ to T_Δ is a tuple $M = (C, \phi_{dom}, (\phi^c_{\mathsf{n}}(x))_{c\in C}, (\phi^c_\delta(x))_{\delta\in\Delta,c\in C}, (\phi^{c,c'}_i(x,y))_{i\in[\ell],c,c'\in C})$ such that C is a finite set of copy indices, ϕ_{dom} is an MSO[Σ]-sentence which defines whether the input tree s is in the domain, ϕ^c_{n}, ϕ^c_δ are MSO[Σ]-formulas with one free variable x which respectively define the nodes $V(t)$ in the output tree t and their labels, and $\phi^{c,c'}_i$ are MSO[Σ]-formulas with two free variables x and y which define the edge relations between the nodes in $V(t)$.

Given a tree $s \in T_\Sigma$, $[\![M]\!]_{\mathsf{o}}(s)$ is defined if $s \models \phi_{dom}$, and it is then equal to (t,o) where t is the structure whose domain is $D = \{(u,c) \mid s \models \phi^c_{\mathsf{n}}(u)\}$ (each node (u,c) is denoted hereafter by u^c), and for all $u^c \in D$, u^c is labeled by $\delta \in \Delta$ if $s \models \phi^c_\delta(u)$, and a node $u_2^{c_2} \in D$ is the i-th child of a node $u_1^{c_1} \in D$ if $s \models \phi^{c_1,c_2}_i(u_1,u_2)$. The origin mapping o is defined by $o(u^c) = u$ (hence, for any input node u, there are at most $|C|$ nodes in the output with u as origin). Additionally, for M to be an MSO tree transducer, it is required that $t \in T_\Delta$.

For example, let $\Sigma = \{f^{(2)}, a^{(0)}, b^{(0)}\}$ and $\Delta = \{a^{(1)}, b^{(1)}, e^{(0)}\}$. The yield of a tree $s \in T_\Sigma$ is the monadic tree in T_Δ obtained from its leaves, in preorder. E.g., the yield of $f(f(a,b),b)$ is $a(b(b(e)))$. The yield translation is not in DTOP$^{\mathrm{R}}$ but in DMSOT. The preorder relation $\preceq$ on tree nodes obtained from the preorder traversal of the tree is known to be MSO[Σ]-definable. To realize the yield in DMSOT, we only need one copy ($C = 1$). The domain formula is true and all internal nodes are filtered out by $\phi^1_{\mathsf{n}}(x) = \mathrm{leaf}(x)$, where leaf$(x)$ holds true if x is a leaf. Labels are unchanged: $\phi^1_\sigma(x) = \sigma(x)$ for all $\sigma \in \Sigma$, and the first-child relation is defined by $\phi^{1,1}_1(x,y)$, which expresses that x and y are leaves, and that $x \preceq y \wedge \neg(\exists z. x \preceq z \wedge z \preceq y)$.

DMSOT can be extended with non-determinism, leading to the class of MSO tree transducers (MSOT). All formulas ϕ_{dom}, ϕ^c_δ and $\phi^{c,c'}_i$ can use a fixed additional finite set of free second-order variables $\overline{X}$. Once an assignment ν of each variable of $\overline{X}$ by a set of nodes of an input tree s is fixed, the previous formulas can be interpreted as before with respect to this assignment, thus defining an output pair (t_ν, o_ν) (if the domain formula holds true). The set of outputs associated with s is the set of all such pairs (t_ν, o_ν), for all assignments ν of $\overline{X}$. In the previous example, using a free variable X, one could also associate all scattered substrings (seen as monadic trees) of the yield. Only leaves in X are kept, by letting $\phi^1_{\mathsf{n}}(x) = x \in X \wedge \text{leaf}(x)$, and $\phi^{1,1}_1(x,y)$ also requires that $x, y \in X$.

Origin-Equivalence Problem Given two tree transducers M_1, M_2 which are either both in TOP^{R} or both in MSOT, decide whether $[\![M_1]\!]_{\mathsf{o}} = [\![M_2]\!]_{\mathsf{o}}$.

Theorem 1. *Origin-equivalence is decidable for* MSO *tree transducers and top-down tree transducers with regular look-ahead.*

Sketch of Proof. Given a tree transducer M, we simply write $\mathsf{dom}(M)$ for $\mathsf{dom}([\![M]\!])$. Inequality of the domains implies inequivalence. As a first step, for both classes of transducers, an equality test of the domains is performed. This is obviously decidable due to the effective regularity of these sets. Then, for TOP^{R}, by modifying the output alphabet of M_1 and M_2 and their rules, we show how to turn them into non-deleting [4] non-erasing [5] TOP without look-ahead, while preserving origin-equivalence. Let us notice then that the constraint that origins should be the same is strong: for all input trees $s \in \mathsf{dom}(M_1)$, for all $(t, o) \in [\![M_1]\!]_{\mathsf{o}}(s)$, M_2 must produce, in a successful execution, the symbols of t exactly at the same moment as M_1, and conversely for all $(t, o) \in [\![M_2]\!]_{\mathsf{o}}(s)$. When considering non-erasing TOP, this property has a nice consequence: both M_1 and M_2 can be seen as symbol-to-symbol transducers, which means that each right-hand side of a rule contains exactly one node with a label from Δ. To be precise, the Δ-part of any right-hand side ζ of a rule of M_1 or M_2, with n leaves labeled by some $q(x_i)$, can be seen as a single symbol of rank n. For instance, if $\zeta = f(q_1(x_1), h(q_2(x_1)), q_3(x_2))$, then $f(., h(.), .)$ is seen as a single symbol of arity 3. M'_1 and M'_2, two non-deleting non-erasing TOP without look-ahead, can be built from M_1 and M_2 respectively such that $[\![M_1]\!]_{\mathsf{o}} = [\![M_2]\!]_{\mathsf{o}}$ iff $[\![M'_1]\!] = [\![M'_2]\!]$. Finally, it leads to solve an equivalence problem for (nondeterministic) non-deleting symbol-to-symbol top-down tree transducers which is known to be decidable [1].

For MSOT, we show that the equality set $E([\![M_1]\!]_{\mathsf{o}}, [\![M_2]\!]_{\mathsf{o}})$ of $[\![M_1]\!]_{\mathsf{o}}$ and $[\![M_2]\!]_{\mathsf{o}}$, defined as the set of trees $s \in \mathsf{dom}(M_1)$ such that $[\![M_1]\!]_{\mathsf{o}}(s) = [\![M_2]\!]_{\mathsf{o}}(s)$, is effectively regular. Without origins, even simple MSOT translations yield a non-regular set: *e.g.*, the translations $R_1 : f(s_1, s_2) \mapsto s_1$ and $R_2 : f(s_1, s_2) \mapsto s_2$ are both MSOT definable but their (origin-free) equality set $\{f(s,s) \mid s \in T_\Sigma\}$, is not regular. To show that the set $E([\![M_1]\!]_{\mathsf{o}}, [\![M_2]\!]_{\mathsf{o}})$ is regular, we construct an MSO formula that defines it. This formula expresses for instance that any sequence of input nodes $u_1, \dots, u_n$ that are connected with successor formulas $\phi^{c,c'}_i(x,y)$ by M_1 are also connected with successor formulas of M_2 with the same sequence of indices i, and conversely. It also expresses that the sequence of label

[4] Non-deleting means that every x_i occurring in the left-hand side of a rule also occurs in its right-hand side.

[5] Non-erasing means that the right-hand side of each rule contains at least one symbol from Δ.

formulas $\phi^{\mathsf{c}}_{\delta}(x)$ of M_1 and M_2 that hold on $u_1, \ldots, u_n$ respectively, carry the same respective output symbols δ. As a consequence, to check origin-equivalence of M_1 and M_2, it suffices to check that $\mathsf{dom}(M_1) = \mathsf{dom}(M_2) = E(\llbracket M_1 \rrbracket_{\mathsf{o}}, \llbracket M_2 \rrbracket_{\mathsf{o}})$, which is decidable since all these sets are effectively regular. □

Origin-Injectivity Problem Given a tree transducer M from T_Σ to T_Δ either in DMSOT or in DTOP$^{\mathrm{R}}$, decide whether the function $\llbracket M \rrbracket_{\mathsf{o}}$ is injective.

Theorem 2. *Origin-injectivity is decidable for deterministic* MSO *tree transducers and deterministic top-down tree transducers with regular look-ahead.*

Sketch of Proof. Let us denote by $R(\llbracket M \rrbracket_{\mathsf{o}})$ the set of pairs of trees $(s_1, s_2) \in \mathsf{dom}(M)^2$ such that $\llbracket M \rrbracket_{\mathsf{o}}(s_1) = \llbracket M \rrbracket_{\mathsf{o}}(s_2)$. Clearly, $\llbracket M \rrbracket_{\mathsf{o}}$ is injective iff $R(\llbracket M \rrbracket_{\mathsf{o}}) \cap (\neq_{T_\Sigma}) = \varnothing$, where $\neq_{T_\Sigma}$ is the difference relation over T_Σ. Take a pair $(s_1, s_2) \in R(\llbracket M \rrbracket_{\mathsf{o}})$. We can define its top- and left-most overlap $s_1 \otimes s_2$ that aligns the same nodes of s_1 and s_2 (recall that a node is a string over $\mathbb{N}$). Nodes in $V(s_1) \cap V(s_2)$ are labeled, in $s_1 \otimes s_2$, by the pair of their respective labels in s_1 and s_2. Nodes in $V(s_1) \setminus V(s_2)$ or $V(s_2) \setminus V(s_1)$ are labeled by pairs $(\sigma, \bot)$ or $(\bot, \sigma)$, for a special padding symbol $\bot$. Interestingly, $\llbracket M \rrbracket_{\mathsf{o}}(s_1) = \llbracket M \rrbracket_{\mathsf{o}}(s_2)$ if M produces the same output symbols when processing a node in $V(s_1) \cap V(s_2)$, and does not produce anything when processing a node in $V(s_2) \setminus V(s_1) \cup V(s_1) \setminus V(s_2)$.

When $M \in$ DTOP$^{\mathrm{R}}$, this last observation allows us to construct a DTOP$^{\mathrm{R}}$ M' reading trees $s_1 \otimes s_2 \in T_\Sigma \otimes T_\Sigma$, that simulates in parallel two executions of M on s_1 and s_2 respectively, and checks that M produces the same symbols at the same moment, for the common nodes of s_1 and s_2, and nothing elsewhere. Then, $\mathsf{dom}(M')$ equals the set of trees $s_1 \otimes s_2$ such that $(s_1, s_2) \in R(\llbracket M \rrbracket_{\mathsf{o}})$ and is regular, as DTOP$^{\mathrm{R}}$ have regular domains. In other words, the relation $R(\llbracket M \rrbracket_{\mathsf{o}})$ is recognizable [5]. It is easily shown that $\neq_{T_\Sigma}$ is, as well, recognizable. Since recognizable relations are closed under intersection, one gets decidability of injectivity for origin-translations of DTOP$^{\mathrm{R}}$.

When $M \in$ DMSOT, $R(\llbracket M \rrbracket_{\mathsf{o}})$ is also a recognizable relation. To prove that, we first transform M into two transducers $M_1, M_2 \in$ DMSOT that run on trees in $T_\Sigma \otimes T_\Sigma$. While processing trees $s_1 \otimes s_2$, M_i simulates M on s_i, so that $\llbracket M_i \rrbracket_{\mathsf{o}}(s_1 \otimes s_2) = \llbracket M \rrbracket_{\mathsf{o}}(s_i)$. Then, $\{s_1 \otimes s_2 \mid (s_1, s_2) \in R(\llbracket M \rrbracket_{\mathsf{o}})\} = E(\llbracket M_1 \rrbracket_{\mathsf{o}}, \llbracket M_2 \rrbracket_{\mathsf{o}})$, and the result follows since $E(\llbracket M_1 \rrbracket_{\mathsf{o}}, \llbracket M_2 \rrbracket_{\mathsf{o}})$ is regular for $M_1, M_2 \in$ DMSOT. □

Application to Query Determinacy Let Q (resp. V) be a functional tree translation (resp. origin tree translation) from T_Σ to T_Δ, called the query (resp. the view). We say that Q is *determined* by V if for all trees $s_1, s_2 \in \mathsf{dom}(V)$, if $V(s_1) = V(s_2)$ then $Q(s_1) = Q(s_2)$. This generalizes the injectivity problem: the identity tree translation is determined by a view V iff V is injective.

Corollary 3. *Let Q (resp. V) be a tree translation (resp. an origin tree translation) defined by either a* DTOP^{R} *or a* DMSOT. *It is decidable whether Q is determined by V.*

Proof. Let Q_1, Q_2 be tree translations from $T_\Sigma \otimes T_\Sigma$ to T_Δ defined by $Q_i(s_1 \otimes s_2) = Q(s_i)$, for all $s_1, s_2 \in T_\Sigma$. Note that the Q_i are definable by DTOP$^{\mathrm{R}}$ (resp.

DMSOT) if Q is. Let $r(V)$ be the set of trees $s_1 \otimes s_2$ such that $s_1, s_2 \in \mathsf{dom}(V)$ and $V(s_1) = V(s_2)$. Q is determined by V iff $Q_1(s) = Q_2(s)$ for all s in $r(V)$, iff Q_1 and Q_2 are equivalent on $r(V)$. As seen before to solve the injectivity problem, we show that the pairs (s_1, s_2) such that $V(s_1) = V(s_2)$ is a recognizable relation, for transducers in DTOP$^{\mathrm{R}}$ or DMSOT. So, $r(V)$ is regular. The result follows as equivalence of DTOP$^{\mathrm{R}}$ or DMSOT is decidable on regular languages [20]. □

3 Tree-to-String Translations with Origin

A *top-down tree-to-string transducer* (yTOP for short) M is a tuple $(Q, \Sigma, \Delta, q_0, R)$ where Q, q_0, Σ are defined as for top-down tree transducers, Δ is an alphabet, and every rule is of the form $q(\sigma(x_1, \ldots, x_k)) \to \zeta$, where ζ is a *string* over Δ and the symbols $q'(x_i)$, with $q' \in Q$ and $i \in [k]$. The definition of $[\![q]\!]_{\mathsf{o}}$ is as for top-down tree transducers, only that t and t_v are strings over Δ. In this way we obtain $[\![M]\!]_{\mathsf{o}}$ and $[\![M]\!]$. yTOP generalize TOP, as right-hand sides of rules of TOP can be encoded as well-bracketed strings. The converse is false: even considering strings as monadic trees, a yTOP can for instance easily implement string reversal, which is impossible using TOP. As for TOP, we can equip this model with regular look-ahead, and consider deterministic machines. This defines the classes of deterministic top-down tree-to-string transducers (with regular look-ahead): yDTOP (yDTOP$^{\mathrm{R}}$).

We give a yDTOP M implementing string reversal. It takes as input monadic trees over the alphabet $\Sigma = \{a^{(1)}, b^{(1)}, e^{(0)}\}$ and produces output strings over the alphabet $\Delta = \{a, b\}$. It has states q, q_a, q_b and rules $q(\sigma(x_1)) \to q(x_1)q_\sigma(x_1)$ and $q_\sigma(\sigma'(x_1)) \to q_\sigma(x_1)$ for $\sigma, \sigma' \in \Sigma^{(1)}$, and leaf rules $q(e) \to \varepsilon$ and $q_\sigma(e) \to \sigma$ for $\sigma \in \Sigma^{(1)}$. Clearly, for $s = a(a(b(e)))$ $[\![M]\!](s) = baa = w$. Note that the origin of each letter of w is the leaf of s (here, 111). Hence, the origin translation $[\![M]\!]_{\mathsf{o}}$ is *not* MSO definable: there may be unboundedly many letters having such a leaf as origin (by contrast, the translation $[\![M]\!]$ is MSO definable, as tree-to-string MSO transducers are equivalent to yDTOP$^{\mathrm{R}}$ of linear size increase [9, 8]). In Sec. 4 we show how to decide whether the origin translation of a yDTOP$^{\mathrm{R}}$ is MSO definable.

Undecidability Results By a construction similar to the above string reversal example, any string-to-string rational relation can be shown to be implementable as a yTOP such that the origin of every output symbol is the unique leaf of the input monadic tree. This "erasing" of origin info shows that origin-equivalence of yTOP is harder than equivalence of string-to-string rational relations known to be undecidable [14]. A similar technique can be used to show that origin-injectivity of yDTOP is also undecidable, by an encoding of the Post Correspondence Problem. This contrasts with the positive results presented in Sec. 2. There, origin-equivalence of MSOT relied on the regularity of the set $E([\![M_1]\!]_{\mathsf{o}}, [\![M_2]\!]_{\mathsf{o}})$, and one can easily come up with two yDTOP M_1, M_2 such that this set is not regular. E.g., take the transducer $M_1 = M$ for string reversal of before, and M_2 the identity with rules $q(\sigma(x_1)) \to q_\sigma(x_1)q(x_1)$ and $q_\sigma(\sigma'(x_1)) \to q_\sigma(x_1)$ for $\sigma, \sigma' \in \Sigma^{(1)}$, and leaf rules as for M. Now $E([\![M_1]\!]_{\mathsf{o}}, [\![M_2]\!]_{\mathsf{o}})$ is the set of palindromes on Δ^* (seen as monadic trees), which is not regular.

Equivalence of ydtop$^{\mathbf{R}}$ Though it is not possible to obtain decidability results by regular sets (or recognizable relations), we manage to prove, using more involved techniques, that origin-equivalence of yDTOP$^{\mathrm{R}}$ is decidable.

Theorem 4. *Origin-equivalence is decidable for deterministic top-down tree-to-string transducers with regular look-ahead.*

Sketch of Proof. Let M_1, M_2 be two yDTOP$^{\mathrm{R}}$. We first check whether M_1 and M_2 have the same domain D. If not we output "not equivalent". Otherwise, we build yDTOP transducers *without* look-ahead M_1', M_2', and a regular tree language D' such that $[\![M_1]\!]_o = [\![M_2]\!]_o$ iff M_1' and M_2' are origin-equivalent on D'.

Secondly, we transform the two yDTOP into end-marked leaf-producing yDTOP such that origin-equivalence is preserved: the leaf-producing property requires that transducers produce only at the leaves. The end-marked property means that every output string has a final end-marker. Both properties are obtained by modifying the input alphabet and the rules. For the last one, only the initial rules of the transducers are concerned. We still denote them $(M_i')_{i=1,2}$.

Last, we reduce the origin-equivalence problem of these yDTOP to the equivalence problem of monadic yDTOP, where 'monadic' means that every input symbol has rank 0 or 1. This is done by only considering partial output strings produced on root-to-leaf paths of the input tree. We give now some details on this last part. Let s be a tree, $w = a_1 \dots a_n \in \Delta^*$ a string with $a_i \in \Delta$ for all i, and $o : V(w) \to V(s)$ an origin mapping. Let $U = \{u_1, \dots, u_k\} \subseteq V(s)$ be a set of (distinct) nodes (in this order). We define $\Pi_{u_1,\dots,u_k}(w, o)$ as the string in $(\Delta \times [k])^*$ obtained from w by erasing a_i if $o(i) \notin \{u_1, \dots, u_k\}$, and changing a_i into (a_i, j) if $o(i) = u_j$. We give a key result which allows one to reduce our origin-equivalence problem of yDTOP to two instances of the equivalence problem of monadic yDTOP: for $s \in \mathsf{dom}(M_1') \cap \mathsf{dom}(M_2')$, $M_1'(s) = M_2'(s)$ iff the following two conditions are satisfied: (1) $\Pi_u(M_1'(s)) = \Pi_u(M_2'(s))$ for every $u \in V_{\Sigma^{(0)}}(s)$, (2) $\Pi_{u_1,u_2}(M_1'(s)) = \Pi_{u_1,u_2}(M_2'(s))$ for every $u_1 \neq u_2 \in V_{\Sigma^{(0)}}(s)$.

4 Subclass Definability Problems

Deterministic MSO tree-to-string transducers (DMSOTS for short) can be defined as a particular case of DMSOT transducers (their origin-equivalence is decidable by Theorem 1). While DMSOTS are equivalent to yDTOP$^{\mathrm{R}}$ of linear size increase [9, 8], this is not true in the presence of origin; there are such yDTOP$^{\mathrm{R}}$ for which no origin-equivalent DMSOTS exists (*e.g.* the string reversal example of Sec. 3). However, every DMSOTS effectively has an origin-equivalent yDTOP$^{\mathrm{R}}$ (obtained by following the respective constructions for origin-less transducers). Can we decide for a given yDTOP$^{\mathrm{R}}$ whether its origin translation is DMSOTS definable?

Theorem 5. *For a given y*DTOP^{R} *M, it is decidable whether or not there exists an origin-equivalent* DMSOTS*. If so, then such a* DMSOTS *can be constructed.*

Sketch of Proof. It relies on the notion of *bounded origin*: an origin translation τ is of bounded origin if there exists a number k such that for every $(s, (w, o)) \in \tau$

and $u \in V(s)$: $|\{v \in V(w) \mid o(v) = u\}| \leq k$, *ie* every input node can be the origin of only a bounded number of output positions. By their definition, origin translations of MSO transducers have bounded origin. The bounded origin property can be decided for a yDTOP$^{\mathrm{R}}$ M: we transform M into a yDTOP$^{\mathrm{R}}$ transducer M' that takes input trees of M, but with one node u marked. The transducer M' produces output only on the marked node. Thus, the length of its output equals the number of positions that have u as origin. Decidability follows from that of finiteness of ranges [6]. The proof then builds an origin-equivalent DMSOTS following the constructions in the literature. □

Macro Tree Transducers At last we consider a more powerful type of transducer: the macro tree transducer (MAC). For simplicity, we only look at total deterministic such transducers. A MAC extends a top-down tree transducer by *nesting* of recursive state calls. Thus, a state q is now of rank $m + 1$ and takes, besides the input tree, m arguments of type output trees. In the rules, these arguments are denoted by *parameters* $y_1, \dots, y_m$. Thus, a rule is of the form $q(\sigma(x_1, \dots, x_k), y_1, \dots, y_m) \to \zeta$, where ζ is a tree over (nested) states, output symbols, and the parameters which may occur at leaves. As example, consider a MAC with initial rule $q_0(h(x_1)) \to q(x_1, a)$ and these rules: $q(h(x_1), y_1) \to q(x_1, q(x_1, y_1))$ and $q(a, y_1) \to b(y_1, y_1)$. For a monadic input tree $h(\dots h(a) \dots)$ of height $n + 1$, it produces a full binary tree of height 2^n. Thus, MACs can have *double-exponential* size increase; all models discussed so far have at most exponential size increase. A *total deterministic macro tree transducer* (MAC) is a tuple $M = (Q, \Sigma, \Delta, q_0, R)$ where Σ, Δ are as before, Q is a ranked alphabet with $Q^{(0)} = \emptyset$, $q_0 \in Q^{(1)}$, and R contains for every $q \in Q^{(m+1)}$, $m \geq 0$, $\sigma \in \Sigma^{(k)}$, and $k \geq 0$, a rule $q(\sigma(x_1, \dots, x_k), y_1, \dots, y_m) \to \zeta$, where ζ is a tree over $Q \cup \Delta \cup \{x_1, \dots, x_k, y_1, \dots, y_m\}$ such that x_i occurs in ζ at a node u if and only if u is the first child of a Q-labeled node (and symbols y_j are of rank zero). We denote ζ by $\mathsf{rhs}(q, \sigma)$. Every state $q \in Q^{(m+1)}$ of M induces a function $[\![q]\!] : T_\Sigma \times T_\Delta^m \to T_\Delta$. Let $s = \sigma(s_1, \dots, s_k) \in T_\Sigma$ and $t_1, \dots, t_m \in T_\Delta$. Then $[\![q]\!](s, t_1, \dots, t_m) = [\zeta]$ where $\zeta = \mathsf{rhs}(q, \sigma)$ and $[\zeta]$ is defined recursively as follows. If $\zeta = y_j$ then $[\zeta] = t_j$. If $\zeta = d(\zeta_1, \dots, \zeta_\ell)$ with $d \in \Delta^{(\ell)}$, then $[\zeta] = d([\zeta_1], \dots, [\zeta_\ell])$. If $\zeta = q'(x_i, \zeta_1, \dots, \zeta_\ell)$ with $q' \in Q^{(\ell+1)}$ and $i \in [k]$, then $[\zeta] = [\![q']\!](s_i, [\zeta_1], \dots, [\zeta_\ell])$.

Origin Semantics We define the origin semantics of M using the MAC M^s and the *decorated version* $\mathsf{dec}(s)$ of an input tree s (see Definition 4.15 of [9]). Let s be an input tree of M. Then $\mathsf{dec}(s)$ is obtained from s by relabeling every node u by $\langle s[u], u \rangle$. For a state q and input symbol $\langle \sigma, u \rangle$ the MAC M^s applies the (q, σ)-rule of M, but with every output symbol d replaced by $\langle d, u \rangle$. The origin of an output node then simply is the second component of the label of that node. Intuitively, when a MAC applies a rule at input node u and generates output inside of parameter positions, then all these outputs have origin u. Note that such nodes may be duplicated later and appear unboundedly often (at arbitrary positions of the output tree). Let us see an example of an origin translation that cannot be defined by the previous models (but for which the non-origin translation can be defined): in fact we consider the identity on trees s over $\{f^{(2)}, a^{(0)}\}$. The MAC M has these rules for q_0: $q_0(a) \to a$ and $q_0(f(x_1, x_2)) \to f(q(x_1, a), q(x_2, a))$, and these rules for q: $q(f(x_1, x_2), y_1) \to f(q(x_1, y_1), q(x_2, y_1))$ and $q(a, y_1) \to y_1$. Thus, $[\![M]\!]_\mathsf{o}(s) = (s, o)$ where $o(u) = u$ if u is an internal node, and $o(u) = \varepsilon$ if u is a leaf. Thus, all leaves have the root node as origin. Clearly, none of our previous models can realize such an origin translation.

Deciding whether the translation of a MAC can be defined by a DTOP$^\mathrm{R}$ is a difficult open problem: a MAC can use its parameters in complex ways, but still be definable by a DTOP$^\mathrm{R}$. However, with origin, we are able to prove decidability.

Theorem 6. *For a given* MAC *M, it is decidable whether or not there exists an origin-equivalent* DTOP^R*. If so, then such a* DTOP^R *can be constructed.*

Sketch of Proof. First, if τ is the origin translation of a DTOP$^\mathrm{R}$, and v, v' are nodes in $\tau(s)$ for some tree s such that v' is a descendant of v, then the origin of v' must be a descendant of the origin of v (see [17]). We call this property of an origin translation *order-preserving.* Consider now a MAC with order-preserving origin translation. Is it definable by a DTOP$^\mathrm{R}$? To see this is not true, consider this MAC for the identity: $q_0(a) \to a$ and $q_0(f(x_1, x_2)) \to q(x_1, q_0(x_1), q_0(x_2))$, plus the rules $q(f(x_1, x_2), y_1, y_2) \to q(x_1, y_1, y_2)$ and $q(a, y_1, y_2) \to f(y_1, y_2)$. For the input tree $f(f(\cdots f(a, a) \cdots), a)$ that is a left-comb, the origin of each f-node is its left-most descendant leaf. Not the order is the problem, but, there are too many *connected* output nodes with the same origin. Intuitively, the connected output nodes of a DTOP$^\mathrm{R}$ can only span the size of a right-hand side. We say that an origin translation τ is *path-wise bounded-origin* if there exists a number k such that are at most k output nodes with the same origin on each path of the output tree. Both the order-preserving and path-wise bounded-origin properties can decided. For a MAC with these two properties, and nondeleting and nonerasing in its parameters (which can both be obtained by regular look-ahead), the depth of nested state calls on the same input node is bounded. An origin-equivalent DTOP$^\mathrm{R}$ can be constructed by introducing one state for each of these finitely many different nestings of state calls. □

Conclusions and Future Work. We have shown that several important decision problems for tree transducers become decidable in the presence of origin information. Some problems remain open, such as the decidability of equivalence of MACs with origin. In the future we would like to study other notions of origin such as unique identifiers instead of Dewey nodes, or, sets of nodes (one of which is guaranteed to be origin) instead of one node. A lot of work remains to be done on determinacy; for instance, we would like to show that a query determined by a view with origin can be rewritten into a tractable class of queries.

References

1. Y. Andre and F. Bossut. On the equivalence problem for letter-to-letter top-down tree transducers. *Theor. Comput. Sci.*, 205(1-2):207–229, 1998.
2. M. Benedikt, J. Engelfriet, and S. Maneth. Determinacy and rewriting of top-down and MSO tree transformations. In *MFCS*, pages 146–158, 2013.
3. M. Bojanczyk. Transducers with origin information. In *ICALP*, pages 26–37, 2014.

4. F. Braune, N. Seemann, D. Quernheim, and A. Maletti. Shallow local multi-bottom-up tree transducers in statistical machine translation. In *ACL*, pages 811–821, 2013.
5. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, C. Löding, S. Tison, and M. Tommasi. Tree automata techniques and applications, 2007.
6. F. Drewes and J. Engelfriet. Decidability of the finiteness of ranges of tree transductions. *Inf. Comput.*, 145(1):1–50, 1998.
7. J. Engelfriet. Some open questions and recent results on tree transducers and tree languages. In R. Book, editor, *Formal language theory; perspectives and open problems.* Academic Press, New York, 1980.
8. J. Engelfriet and S. Maneth. Macro tree transducers, attribute grammars, and MSO definable tree translations. *Inf. Comput.*, 154(1):34–91, 1999.
9. J. Engelfriet and S. Maneth. Macro tree translations of linear size increase are MSO definable. *SIAM J. Comput.*, 32(4):950–1006, 2003.
10. J. Engelfriet and S. Maneth. The equivalence problem for deterministic MSO tree transducers is decidable. *Inf. Process. Lett.*, 100(5):206–212, 2006.
11. J. Engelfriet, G. Rozenberg, and G. Slutzki. Tree transducers, L systems, and two-way machines. *J. Comput. Syst. Sci.*, 20(2):150–202, 1980.
12. Z. Ésik. Decidability results concerning tree transducers I. *Acta Cybern.*, 5(1):1–20, 1980.
13. Z. Fülöp and H. Vogler. *Syntax-Directed Semantics - Formal Models Based on Tree Transducers.* Monographs in Theoretical Computer Science. An EATCS Series. Springer, 1998.
14. T. V. Griffiths. The unsolvability of the equivalence problem for lambda-free nondeterministic generalized machines. *J. ACM*, 15(3):409–413, 1968.
15. S. Hakuta, S. Maneth, K. Nakano, and H. Iwasaki. Xquery streaming by forest transducers. In *ICDE*, pages 952–963, 2014.
16. R. Küsters and T. Wilke. Transducer-based analysis of cryptographic protocols. *Inf. Comput.*, 205(12):1741–1776, 2007.
17. A. Lemay, S. Maneth, and J. Niehren. A learning algorithm for top-down XML transformations. In *PODS*, pages 285–296, 2010.
18. A. Maletti. Tree transformations and dependencies. In *MOL*, pages 1–20, 2011.
19. 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.
20. S. Maneth. Equivalence problems for tree transducers (survey). In *AFL*, pages 74–93, 2014.
21. K. Matsuda, K. Inaba, and K. Nakano. Polynomial-time inverse computation for accumulative functions with multiple data traversals. In *PEPM*, pages 5–14, 2012.
22. T. Milo, D. Suciu, and V. Vianu. Typechecking for XML transformers. *J. Comput. Syst. Sci.*, 66(1):66–97, 2003.
23. W. C. Rounds. Mappings and grammars on trees. *Math. Syst. Th.*, 4(3):257–287, 1970.
24. J. W. Thatcher. Generalized sequential machine maps. *JCSS*, 4(4):339–367, 1970.
25. A. van Deursen, P. Klint, and F. Tip. Origin tracking. *J. Symb. Comput.*, 15(5/6):523–545, 1993.
26. J. Voigtländer, Z. Hu, K. Matsuda, and M. Wang. Enhancing semantic bidirectionalization via shape bidirectionalizer plug-ins. *J. Funct. Program.*, 23(5):515–551, 2013.
27. J. Voigtländer and A. Kühnemann. Composition of functions with accumulating parameters. *J. Funct. Program.*, 14(3):317–363, 2004.