Query Rewriting for Nondeterministic Tree Transducers

Kazuki MIYAHARA†(a), Student Member, Kenji HASHIMOTO††, Member, and Hiroyuki SEKI††, Fellow

SUMMARY We consider the problem of deciding whether a query can be rewritten by a nondeterministic view. It is known that rewriting is decidable if views are given by single-valued non-copying devices such as compositions of single-valued extended linear top-down tree transducers with regular look-ahead, and queries are given by deterministic MSO tree transducers. In this paper, we extend the result to the case that views are given by nondeterministic devices that are not always single-valued. We define two variants of rewriting: universal preservation and existential preservation, and discuss the decidability of them.

key words: tree transducers, query rewriting, query preservation

1. Introduction

Almost every computer program runs along the structure of input data. The structure of input data would be sometimes modified due to changes or updates of its specifications. In the case, the program is needed to be rewritten for being adapted to the new structure of input data, one then may naturally ask the fundamental question—is it possible to rewrite the program automatically without changing its operation results? Some solutions to the automatic program rewriting have been presented in database theory, in which the problem has been formalized as query rewriting or query answering, which are motivated by data integration and query optimization (see, e.g., surveys [1], [2]). Let Q, Q’ be classes of queries and V’ be a class of views. Query rewriting for Q to Q’ under V’ is the problem deciding whether, given a query q ∈ Q and a view v ∈ V’, a mapping q’ ∈ Q’ can be constructed from q and v such that q = v ◦ q’, where v ◦ q’ denotes the composition of v and q’ defined as (v ◦ q’)(t) = q’(v(t)) for every input t. If such a mapping q’ exists, we say that q can be rewritten in terms of v. Especially, the case when Q’ = Q is called query preservation. This case is important because it is not required to use a more expressive class than Q to rewrite the query. If q can be rewritten in terms of v and Q’ = Q, we say that v preserves q.

Various techniques of query rewriting (and query preservation) have been developed in relational database theory. Query rewriting for semi-structured data, especially, tree-structured data such as XML documents has received attention recently due to the enormous success of the model on the Web (see, e.g., [3]–[10]). In this paper, we focus on query rewriting for tree-structured data.

The case when both views and queries are single-valued (or deterministic) tree transducers was studied in [7], [10]. Single-valued tree transducers output just one tree for each input tree (see [7], [8], [10]). Our main contribution is to extend their results to nondeterministic tree transducers as views, which transform each input tree to a set of output trees. To our knowledge, no previous work has considered the query rewriting for nondeterministic tree transducers (that are not always single-valued) as views.

Nondeterminism of tree transducers is required in some applications, e.g., probabilistic database (see a survey [11]) and natural language processing. In machine translation, each sentence in a source natural language can possibly be translated into more than one sentence in a target language (see, e.g., [12]–[14]). Thus we need nondeterminism in tree transducers that model syntax-based machine translations (see, e.g., [15], [16]). Bilingual documents are essentially required to construct statistical syntax-based translators. The translation accuracy of statistical syntax-based translators depends heavily on quality and quantity of the documents that are used to construct the translators (see, e.g., [17]). However, preparing huge and high-quality bilingual documents requires many efforts and costs in general. Query rewriting for nondeterministic tree transducers suggests a solution to this problem, which we call rewriting-based construction of machine translators. For instance, let q, v be English-to-Japanese and English-to-French machine translators, respectively, realized by nondeterministic tree transducers. One can construct by the rewriting-based construction French-to-Japanese translator q’ from q and v, if q can be rewritten in terms of v (see Fig. 1). Advantages of the rewriting-based construction of translators are: (1) the translation accuracy of q’ is guaranteed to be almost the same as q, because q(t) = q’(v(t)) holds for every input t by the definition of query rewriting, (2) any bilingual documents are not required. In the above example, French-to-Japanese bilingual documents are not needed to construct the French-to-Japanese translator q’. Our work is partially motivated to establish the rewriting-based construction of machine translators.

Our results in this paper contribute to the rewriting-based construction by extending the previous work [7], [10]...
on the query rewriting from single-valued models to nondeterministic ones. More specifically, Hashimoto et al. showed in [7] that the query rewriting problem (deciding whether \( q \) can be rewritten in terms of \( v \)) is decidable when views are realized by single-valued extended linear bottom-up tree transducers and queries by single-valued bottom-up tree transducers. Benedikt et al. generalized in [10] the results of [7]. They showed that the problem is decidable when views are realized by compositions of single-valued extended linear top-down tree transducers with regular look-ahead and queries by deterministic MSO tree transducers. Note that the problem is undecidable even if the views can copy only once at each root of input trees [10]. Thus [7], [10] and we treat views that cannot copy\(^1\). Also, many applications of the query rewriting require the classes of \( q \) and \( q' \) coincide, so we focus on the query preservation.

As mentioned above, we generalize the results of [7] and [10] to that for nondeterministic views that are not always single-valued. We first define two variants of rewriting, which are natural extensions of query preservation for nondeterministic views: universal preservation and, its relaxed version, existential preservation.

Let \( \mathcal{V} \), \( Q \) be classes of queries and views, respectively. Given a view \( v \in \mathcal{V} \) and a query \( q \in Q \), \( v \) universally preserves \( q \) if there is a query \( q' \in Q \) such that for every input \( t \) to \( q \) and for every output \( t' \in v(t) \), \( q(t) = q'(t') \) holds; \( v \) existentially preserves \( q \) if there is a query \( q' \) such that for every input \( t \) to \( q \) there is an output \( t' \in v(t) \) satisfying \( q(t) = q'(t') \). Obviously, if \( v \) universally preserves \( q \), then \( v \) existentially preserves \( q \). Intuitively, if \( v \) universally preserves \( q \) then the result of \( q \) can be computed from any output of \( v \). Whereas if \( v \) existentially preserves \( q \) then there exists at least one output of \( v \) from which the result of \( q \) can be computed. Existential preservation is useful in the case that the result of query is more important than that of view. For example, in the case of Fig. 1, \( v \) does not need to universally preserve \( q \) to construct \( q' \). The French-to-Japanese translator \( q' \) can be constructed if at least one result of \( v \) preserves the result of \( q \), that is, if \( v \) existentially preserves \( q \) then \( q' \) can be constructed.

To obtain the decidability of universal preservation for nondeterministic views, we first extend slightly the results in [20] on the equivalence problem for deterministic MSO tree transducers (see Theorem 9). Namely, we show that for a deterministic MSO tree transducer \( q_1 \) and a nondeterministic one \( q_2 \), the equivalence of \( q_1 \) and \( q_2 \) is decidable.

\(^1\)Usually the non-copying property is called \textit{linearity}.

### Table 1

Decidability results on query preservation. Our result is indicated in bold. Incomparability is denoted by \( \rangle \).

<table>
<thead>
<tr>
<th>Query ( \setminus ) View</th>
<th>single-valued</th>
<th>nondeterministic</th>
</tr>
</thead>
<tbody>
<tr>
<td>single-valued ( \setminus )</td>
<td>decidable ( [10] )</td>
<td>decidable ( [10] )</td>
</tr>
<tr>
<td>( \overline{\text{dmsot}} )</td>
<td>decidable ( [7] )</td>
<td>( \overline{\text{dmsot}} )</td>
</tr>
<tr>
<td>( \overline{\text{dt}} )</td>
<td>( \overline{\text{DT}} )</td>
<td>( \overline{\text{DT}} )</td>
</tr>
<tr>
<td>( \text{finite-valued LB} )</td>
<td>( (\text{ELT}^{R})^{'} )</td>
<td>( (\text{ELT}^{R})^{'} )</td>
</tr>
<tr>
<td>( \text{finite-valued LB} )</td>
<td>( (\text{ELT}^{R})^{'} )</td>
<td>( (\text{ELT}^{R})^{'} )</td>
</tr>
</tbody>
</table>

### Table 2

Summary of decidability results, where \( \forall \) and \( \exists \) stand for universal and existential preservation, respectively. "part" stands for the preservation for nondeterministic queries (see Sect. 4), and "sound" means that we give a sound algorithm of the problem for the classes in the line. Nondeterministic classes are indicated in bold.

<table>
<thead>
<tr>
<th>Query</th>
<th>View</th>
<th>( \forall ) or ( \exists )</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \overline{\text{dmsot}} )</td>
<td>( (\text{ELT}^{R})^{'} )</td>
<td>( \forall )</td>
<td>decidable (Thm. 10)</td>
</tr>
<tr>
<td>( \overline{\text{dmsot}} / \overline{\text{DT}} )</td>
<td>( (\text{ELT}^{R})^{'} )</td>
<td>( \exists )</td>
<td>decidable (Thm. 13)</td>
</tr>
<tr>
<td>( \text{finite-valued LB} )</td>
<td>( (\text{ELT}^{R})^{'} )</td>
<td>( \forall \text{ part} )</td>
<td>sound (Thm. 18)</td>
</tr>
<tr>
<td>( \text{finite-valued LB} )</td>
<td>( (\text{ELT}^{R})^{'} )</td>
<td>( \exists \text{ part} )</td>
<td>sound (Cor. 19)</td>
</tr>
</tbody>
</table>

**Related work.** There have been interesting results on query rewriting, though not directly related to this paper. In [3], [18], it was shown that the query rewriting is unde-
able for the class of views and queries that are able to simu-
late first-order logic (FO) queries and projection queries, re-
spectively, and even for views and queries expressed as 
unions of conjunctive queries (that are much weaker than 
FO queries), in the relational case. The problem was also 
shown to be undecidable in [3] for XSLT (or XQuery) as 
vists and simple selection queries, in the XML context. In 
[4], views are defined as transformations that retrieve 
nodes selected by queries, such as Regular XPath and MSO 
queries in the context of unranked trees. Similarly, in [5] 
both queries and views are n-ary node-selecting queries rep-
resented by tree automata. In [8], a hybrid approach was 
adopted, that is, for views tree transducers were used and 
for queries n-ary node-selecting queries were adopted. In 
[8], query preservation was shown to be decidable if a given 
transducer is a single-valued extended linear top-down tree 
transducer with regular look-ahead and a given query is a 
run-based n-ary node-selecting query. The result in [19] is 
for deterministic tree transducers that require “origin” in-
formation.

**Organization.** This paper is organized as follows: Section 2 
provides terminology and definitions of graphs, trees, tree 
automata, various tree transducers with their class hierar-
chies, and uniformizers. In Sect. 3, we show additional class 
hierarchies of tree transducers, which are needed to prove 
our main results. Section 4 defines universal and existential 
preservation for nondeterministic views. Furthermore, for 
nondeterministic queries two variants of query preservation 
are provided, which are analogue of existential preservation. 
Our main result on the decidability of universal preservation 
is presented in Sect. 5. The result is proved using a deci-
dability result on the equivalence for deterministic and non-
deterministic MSO tree transducers, which is included in 
the same section. A sound algorithm for existential preser-
vation with single-valued queries is presented in Sect. 6, and 
others for nondeterministic queries are provided in Sect. 7. 
These sound algorithms employ a method that decomposes 
finite-valued tree transducers into a finite union of single-
valued ones, and then the algorithms apply decision pro-
cedures of query preservation for single-valued tree trans-
ducers presented in the previous work [7], [10] to decomposed 
transducers. Section 6 also includes a solution for the prob-
lem of how to extend the domain of a view, which may be 
useful if a view does not preserve a query because of the 
difference between the domains of the view and the query. 
Section 8 concludes the paper and outlines future work.

2. Preliminaries

We denote the set of all nonnegative integers by \( \mathbb{N} \). For \( n \in \mathbb{N} \), the set \( \{1, \ldots, n\} \) is denoted by \([n]\). A (ranked) alphabet 
is a finite set \( \Sigma \) of symbols with a mapping \( \text{rk} : \Sigma \rightarrow \mathbb{N} \), and 
let \( \Sigma^{(n)} \) be the set \( \{ \sigma \in \Sigma \mid \text{rk}(\sigma) = n \} \). For a binary relation 
\( R \subseteq A \times B \), let \( \text{dom}(R) = \{ a \in A \mid (a, b) \in R \} \), \( \text{ran}(R) = \{ b \in B \mid (a, b) \in R \} \), \( R^{-1} = \{ (b, a) \in B \times A \mid (a, b) \in R \} \), and 
\( R|_{A'} = \{ (a, b) \in R \mid a \in A' \} \) for \( A' \subseteq A \). The composition of 
relations \( R_1 : A \rightarrow B \) and \( R_2 : B \rightarrow C \), denoted by \( R_1 \circ R_2 \), is the relation 
\( A \rightarrow C \) defined by \( R_1 \circ R_2 = \{ (a, c) \mid (a, b) \in R_1 \) and \((b, c) \in R_2 \) for some \( b \in B \)\). For classes of binary 
relations \( \mathcal{R}, \mathcal{S} \), we write \( \mathcal{R} \circ \mathcal{S} = \{ R \circ S \mid R \in \mathcal{R}, S \in \mathcal{S} \} \), 
\( \mathcal{R}^N = \{ R_1 \circ \cdots \circ R_n \mid n \geq 0, R_i \in \mathcal{R} \ (1 \leq i \leq n) \} \), and 
\( \mathcal{R}^{-1} = \{ R^{-1} \mid R \in \mathcal{R} \} \).

2.1 Graphs, Trees, Strings and MSO Graph Transducers

We basically follow the definitions of [20] and omit here formal 
definitions of monadic second-order (MSO) logic and 
MSO graph transducers [20], [22], [23] for the sake of sim-
plicity.

A graph alphabet is a pair \((\Sigma, \Gamma)\) where \( \Sigma \) and \( \Gamma \) are 
ranked alphabets of node labels and edge labels, respect-
ively. A graph over \((\Sigma, \Gamma)\) is a tuple \((V,E,\ell)\), with \( V \) a finite 
set of nodes, \( E \subseteq V \times V \times V \) the set of labeled edges, and 
\( \ell : V \rightarrow \Sigma \) the node-labeling function. The set of graphs over 
\((\Sigma, \Gamma)\) is denoted by \( G(\Sigma, \Gamma) \).

For an alphabet \( \Delta \) and \( a_1, \ldots, a_n \in \Delta \) \((n \geq 0)\), 
we identify the string \( w = a_1, \ldots, a_n \) over \( \Delta \) with the graph 
in \( G([\#], \Delta) \) that has \#-labeled nodes \( v_1, \ldots, v_{n+1} \), and an \( a_i \)-
labeled edge from \( v_i \) to \( v_{i+1} \) for \( 1 \leq i \leq n \).

Let \( \Sigma \) be a ranked alphabet and \( m \) be the maximal rank 
of symbols in \( \Sigma \). The set of all trees over \( \Sigma \) is denoted by \( T_\Sigma \).
A tree whose root is labeled with \( \sigma \in \Sigma^{(k)} \) and has subtrees 
\( t_1, \ldots, t_k \) from left to right is denoted by \( \sigma(t_1, \ldots, t_k) \). For 
\( \sigma \in \Sigma^{(0)} \) we write \( \sigma() \) as \( \sigma \) for simplicity.

For a deterministic MSO graph transducer \( M \), we 
write \([M]\) to denote the graph transduction \( G(\Sigma_1, \Gamma_1) \rightarrow 
G(\Sigma_2, \Gamma_2) \) realized by \( M \). Instead of \([M](g)\) we write \( M(g) \) 
by identifying a transducer \( M \) with its transduction \([M]\) 
(and similarly for other transducers).

A (nondeterministic) MSO graph transducer \( M' \) is ob-
tained from a deterministic one by allowing all formulas 
to have fixed free node-set variables called parameters. The 
transducer binds each parameter to a set of nodes of the 
input graph \( g \) satisfying the domain formula, then for each set 
of nodes, the node formulas and the edge formulas define the 
output graph as the deterministic one does. Thus, the 
graph transduction realized by \( M' \) is (not always a function 
but) a relation \( [M'] \subseteq G(\Sigma_1, \Gamma_1) \times G(\Sigma_2, \Gamma_2) \).

An MSO graph transducer \( M \) is called an MSO graph-
to-tree transducer if the range of \( M \) is a set of trees. For sets of 
graphs \( A \) and \( B \), \( M \) is called an MSO A-to-B transducer 
if \( \text{dom}(M) \subseteq A \) and \( \text{ran}(M) \subseteq B \). In the case of \( A = B \), 
\( M \) is called an MSO A transducer, e.g., an MSO tree trans-
ducer (abbreviated by \text{MSOT} transducer, and by \text{DMSOT} for 
deterministic one). The class of all transductions realized 
by \text{MSOT} transducers is denoted by \text{MSOT}, and similarly for 
other transducers.

The closure properties of MSO transducers under com-
position are shown, e.g., in Proposition 3.2(2) of [22] and 
Proposition 2 of [24].

**Proposition 1.** MSO graph transductions and (determinis-
tic) MSO tree transductions are closed under composition.
2.2 Tree Automata and Tree Transducers

We omit here formal definitions of tree automata and several tree transducers (see, e.g., [13], [25]) for the sake of simplicity and instead provide their intuitive descriptions with their class hierarchies.

Deterministic bottom-up tree automata (TA) are a natural extension of deterministic (finite) word automata, which deal with ranked trees. A TA starts its computation at the leaves of an input tree and moves upward. The tree language accepted by a TA $A$ is denoted by $L(A)$. A tree language $L$ is regular if there exists a TA $A$ such that $L = L(A)$.

Extended bottom-up tree transducers (EB transducers) are an extension of word transducers. An EB transducer $M$ starts its transduction at the leaves of an input tree and moves upward. It can read a subtree of an input tree in one step. An EB transducer is linear if it cannot copy any subtrees of outputted trees during its computation. A bottom-up tree transducer (B transducer) is an EB transducer that can read only one symbol of an input tree in one step. We write $(\text{EB})_{\text{lin}}$ transducers to denote linear $(\text{EB})$ ones. In this paper, we use EB transducers that do not allow a transduction rule that does not read any subtree but only changes a state, which is called an input-$\epsilon$ rule (see, e.g., [7] and [25], in which the class of EB transducers with no input-$\epsilon$ rules is denoted by $\text{XBOT}^\epsilon$).

Extended top-down tree transducers (ET transducers) are also an extension of word transducers. An ET transducer $M$ starts its transduction at the root of an input tree and moves downward. $M$ can read a subtree of an input tree in one step as with EB transducers. An ET transducer is linear if it cannot copy any subtrees of an input tree during its computation. A top-down tree transducer (T transducer) is an ET transducer that can read only one symbol of an input tree in one step. We write $(\text{ET})_{\text{lin}}$ transducers to denote linear $(\text{ET})$ ones. An extended top-down tree transducer with regular look-ahead $(\text{ET})_{\text{R}}$ transducer $M$ is an $(\text{ET})_{\text{R}}$ transducer that has a TA $A$. For a subtree $t_s$ of an input tree, $M$ first checks if $t_s \in L(A)$, and if so, transduces $t_s$. We write $(\text{ET})_{\text{R}}$ transducers to denote linear $(\text{ET})_{\text{R}}$ ones. As with EB transducers, we use ET transducers that do not allow an input-$\epsilon$ rule (see, e.g., [13], [26]–[28], in which ET is denoted by $\text{XTOP}^\epsilon$, or e-XTOP$^\epsilon$).

For a tree transducer $M$ and an input tree $t$, $\text{val}_M(t) = |M(t)|$ denotes the number of different outputs of $M$ for $t$, and let $\text{val}(M) = \sup\{|\text{val}_M(t)| \mid t \in T_2\}$, which is called valuedness of $M$. $M$ is finite-valued if $\text{val}(M) < \infty$. For any tree transducer $M$, we say that $M$ is single-valued (or functional) if $\text{val}(M) \leq 1$, i.e., for each input $t$, there exists at most one output $M(t)$. We use the prefix ‘s’ to denote the transducers, e.g., a single-valued EB transducer is denoted by an s-EB transducer. We also say that $M$ is finite-copying (denoted with the subscript ‘fc’) if each subtree of the input tree is transduced by a bounded number of times (see [29], [30] for more formal definition of finite-copying). We denote by $\langle M \rangle$ the transduction realized by $M$ that is the binary relation on trees. For transducers $M_1$ and $M_2$, we say that $M_1$ and $M_2$ are equivalent if $\langle M_1 \rangle = \langle M_2 \rangle$.

We will use the abbreviation such as EB, B for a class of tree transducers to denote the class of transductions realized by that class of tree transducers. By the definitions above the relations $B \subseteq \text{EB}$, $L \subseteq \text{ELB}$, and $T \subseteq T_\text{R} \subseteq \text{ET}_{\text{R}}$ hold. Additionally, as noted in Sect. 1, there are well-known class hierarchies of tree transducers such that $\text{EB} \subseteq \text{ELB} \subseteq \text{ET}_{\text{R}}$, $\text{B} \subseteq \text{TA} \Rightarrow \text{MSOT}$, and $\text{MSOT} \Rightarrow \text{finite-valued LB} \subseteq \text{ET}_{\text{R}}$, where incomparability is denoted by $\not\Rightarrow$.

2.3 Uniformizers

We define uniformizers following [10]: Let $R$ be a binary relation. A function $u$ is a uniformizer of $R$ if $u \subseteq R$ and $\text{dom}(u) = \text{dom}(R)$. For classes $\mathcal{T}$, $\mathcal{U}$ of transductions, we say that $\mathcal{T}$ has uniformizers in $\mathcal{U}$ if for every $t \in \mathcal{T}$ we can construct a uniformizer $u$ of $t$ such that $u \in \mathcal{U}$. Benedikt et al. [10] showed that it is decidable whether $v$ preserves $q$ by reducing the problem to the equivalence of the query $q$ and the composed mapping $v\circ u\circ q$, where $u$ is a uniformizer of $v^{-1}$.

The following result is used later to prove our main result.

Theorem 2. [Theorem 11 of [10]] $((\text{ELB}_{\text{R}}^*)^{-1})$ has uniformizers in $\text{DT}_{\text{fc}}^\epsilon$.

3. Class Hierarchies of Tree Transducers

In the section, we show some class hierarchies of tree transducers that are needed in Sect. 5 and Sect. 7.

Theorem 3.

$((\text{ELB}_{\text{fc}}^*) \circ \text{DT}_{\text{fc}}^\epsilon \circ \text{MSOT}) \subseteq ((\text{ELB}_{\text{R}}^*) \circ \text{MSOT})$.

Proof. $\text{DT}_{\text{fc}}^\epsilon \subseteq \text{MSOT}$ by Theorem 7.4 of [32], and $\text{ELR} \subseteq \text{TA}_{\text{fc}}$ by the construction in the proof of Theorem 4.8 of [13], which states $\text{ET}_{\text{R}} = \text{R}$ (see also the paragraph under Corollary 18 of [26]), moreover.

$\text{TA}_{\text{fc}} \subseteq \text{DBQREL} \circ \text{TA}_{\text{fc}}$ \hspace{1cm} (1)

$\subseteq \text{DBQREL} \circ \text{HOM}_{\text{fc}} \circ \text{LT}$ \hspace{1cm} (2)

$\subseteq \text{MSOT}$ \hspace{1cm} (3)

where DBQREL is the class of deterministic B transducers that can only relabel, and HOM$_{\text{fc}}$ is the class of finite-copying tree homomorphisms. (1) follows by the fact that $\text{TA}_{\text{fc}} \subseteq \text{DBQREL} \circ \text{TA}$ (Theorem 2.6 [33]), and (2) follows by the construction in the proof of Lemma 3.6 of [34], which states $\text{TA} \subseteq \text{HOM} \circ \text{LT}$. It is not difficult to see that MSOT can simulate DBQREL, HOM$_{\text{fc}}$, and LT, then (3) follows by Proposition 1.

We note that finite-valued $\text{LB} \subseteq (\text{ELB}_{\text{R}}^*) \subseteq \text{MSOT}$.

Proposition 4. $\text{s-LB} \subseteq \text{MSOT}$.
Proof. LB ⊆ ELB by the definition of ELB, ELB = ELR by Theorem 1 of [10], s-ELR ⊆ DTc by Corollary 13 of [10], and DTc ⊆ DMSOT by Theorem 7.4 of [32]. □

4. Query Preservations

Let $\mathcal{V}$ and $\mathcal{Q}$ be a class of single-valued views and a class of single-valued queries for some tree-structured data. For a query $q \in \mathcal{Q}$ and a view $v \in \mathcal{V}$, we say that $v$ preserves $q$ if there exists a query $q' \in \mathcal{Q}$ such that $q(t) = (v \circ q')(t)$ for all tree $t$.

As mentioned in Sect. 1, previous studies on query preservation for tree transducers focus on single-valued (or deterministic) views. In contrast, nondeterministic views output a set of trees for each input tree. We consider two definitions of query preservation for nondeterministic views that are not always single-valued: Let $\mathcal{V}$ be a class of nondeterministic views and $\mathcal{Q}$ be a class of single-valued queries. Given a view $v \in \mathcal{V}$ and a query $q \in \mathcal{Q}$ such that $\text{dom}(q) \subseteq \text{dom}(v)$, we say that $v$ universally preserves $q$ ($\forall$-preserves $q$ for short) if

$$\exists q' \in \mathcal{Q} : (\forall t \in \text{dom}(q)) : q(t) = (v \circ q')(t).$$

The above definition coincides with the definition of the query preservation if $v$ is single-valued. We also say that $v$ existentially preserves $q$ ($\exists$-preserves $q$ for short) if

$$\exists q' \in \mathcal{Q} \forall t \in \text{dom}(q) \exists t' \in v(t) : q(t) = q'(t').$$

This condition is equivalent to the following one: There exists a uniformizer $u$ of $v$ such that

$$\exists q' \in \mathcal{Q} \forall t \in \text{dom}(q) : q(t) = (u \circ q')(t).$$

By definition, if $q$ is universally preserved by $v$, then $q$ is also existentially preserved by $v$.

Furthermore, for a class $\mathcal{Q}$ of nondeterministic queries we define additional two variants of query preservation, which are analogue of existential preservation. We say that $v$ $\forall$-preserves a part of $q$ if

$$\exists q' \in \mathcal{Q} : (\forall t \in \text{dom}(q)) \exists t_q \in q(t) : t_q = (v \circ q')(t_q).$$

Similarly, we say that $v$ $\exists$-preserves a part of $q$ if

$$\exists q' \in \mathcal{Q} \forall t \in \text{dom}(q) \exists t_q \in q(t) : t_q = (u \circ q')(t_q).$$

This condition is equivalent to the following one: There exists a uniformizer $u$ of $v$ such that

$$\exists q' \in \mathcal{Q} \forall t \in \text{dom}(q) : t_q = (u \circ q')(t_q).$$

These two variants of query preservation restrict implicitly the class of $q'$ to a single-valued one. Algorithms that are sound for the variants are presented in Sect. 7.

5. Universal Preservation

The main result of this section is Theorem 10: For $v \in (\text{ELR})^*$ as a view and $q \in \text{DMSOT}$ as a query, it is decidable whether $v$ $\forall$-preserves $q$. To obtain the result, we adopt the proof strategy taken by Benedikt et al. [10]. In [10], in order to show the decidability of the query preservation for views realized by (s-ELR)$^*$ transducers and queries by DMSOT transducers, Benedikt et al. reduced this problem to the equivalence problem of two DMSOT transducers, which is known to be decidable [20]. According to the strategy we reduce the problem to the equivalence of MSOT transducers $M_1$ and $M_2$, where $M_1$ is deterministic and $M_2$ is nonde-terministic. The previous result [20] is for the case that $M_1$ and $M_2$ are both deterministic MSOT transducers (see Theorem 8 below). We slightly extend the previous result (see Theorem 9). Note that if $M_1$ and $M_2$ are both nondeterministic MSOT transducers, the equivalence for them is known to be undecidable due to the negative result for nondeterministic word transducers [31], which are strictly less expressive than MSOT transducers.

To explain our result, let us summarize the decision procedure of [20] for the equivalence of DMSOT transducers. It can be shown that DMSOT transducers $M_1$ and $M_2$ are not equivalent if and only if there exist an input $t$ and a position $n$ such that the symbol at position $n$ of $M_1(t)$ is different from the symbol at position $n$ of $M_2(t)$. Hence, roughly speaking, the procedure tests whether a position $n$ and distinct symbols $a, b$ exist such that the pair $(n, n)$ is contained in the set $S^{ab}$ of all pairs $(i, j)$ where $a$ is the symbol at position $i$ of $M_1(t)$ and $b$ is the symbol at position $j$ of $M_2(t)$, for some input $t$. In [20] the set $S^{ab}$ is shown to be semi-linear (defined below) and then the existence of a pair $(n, n)$ in $S^{ab}$ is decidable (as stated in Lemma 6). The set $S^{ab}$ is constructed using Parikh mapping (also defined below).

Additional definitions are needed to describe the decision procedure precisely. For a string $w$, we denote by $w[i]$ the $i$-th letter of $w$. The Parikh mapping for graphs is the function $\text{Par} : G(\Sigma, \Gamma) \rightarrow \mathbb{N}^k$ defined as $\text{Par}(g) = (n_1, \ldots, n_k)$ where $g$ is a graph over $(\Sigma, \Gamma)$ with $G = (\Sigma, \Gamma)$ and $ni$ is the number of $\sigma_i$-labeled nodes in $g$ for $i \in [k]$. Similarly, the Parikh mapping for strings over $\Sigma = \{\sigma_1, \ldots, \sigma_k\}$ is the function $\text{Par} : \Sigma^* \rightarrow \mathbb{N}^k$ defined as $\text{Par}(w) = (n_1, \ldots, n_k)$ where $ni$ is the number of occurrences of $\sigma_i$ in $w$ for $i \in [k]$. A discrete graph (abbreviated as dgraph) is a graph that has no edges. Let $\text{dgr}$ be a function $\Sigma^* \rightarrow G(\Sigma, \emptyset)$ such that $\text{Par}(w) = \text{Par}(\text{dgr}(w))$ for any string $w \in \Sigma^*$. For a set $G$ of graphs, we denote by $\text{Par}(G)$ the set $\{\text{Par}(g) : g \in G\}$. Similarly, for a string language $L$, let $\text{Par}(L) = \{\text{Par}(w) : w \in L\}$. A set $S \subseteq \mathbb{N}^k$ is semilinear if there exists a regular string language $\mathcal{R}$ such that $S = \text{Par}(\mathcal{R})$. The set $G$ is Parikh if $\text{Par}(G)$ is semilinear. A set of graphs is $\mathcal{V}R$ if it is generated by a context-free vertex replacement graph grammar (or a C-edNCe or an S-HH grammar, see, e.g., [22], [23]). It should be noted that the set of all trees and the set of all strings are VR. The following two lemmas state useful properties of semilinear sets.

**Lemma 5.** [Theorem 7.1 of [22], Lemma 3 of [20]] The images of VR sets of graphs under MSO graph-to-dgraph
transductions are Parikh.

**Lemma 6.** [Lemma 4 of [20]] It is decidable for a semi-linear set $S \subseteq \mathbb{N}^2$ whether there exists an $n, n \in \mathbb{N}$ such that $(n, n) \in S$.

Let us refer to the important lemma of [20].

**Lemma 7.** [Lemma 7 of [20]] Let $a, b$ be distinct symbols and let $M_1, M_2$ be MSO graph-to-string transducers. There exists an MSO graph-to-digraph transducer $M^{a, b}$ such that for every graph $g$,

$$M^{a, b}(g) = \{ \text{dgr}(a^m b^n) \ | \ \exists w_1 \in M_1(g), w_2 \in M_2(g) : w_1/m = a \text{ and } w_2/n = b \}.$$

The decision procedure of [20] consists of three steps, in which the above three lemmas are used:

**Step 1.** Let $M_1, M_2$ be deterministic MSO tree transducers. Construct a deterministic MSO tree-to-string transducer $W$ that “flattens” input trees to strings, then compose them with $M_1, M_2$, i.e., construct $M'_1$ and $M'_2$ with $[M'_1] = [M_1] \circ [W]$ and $[M'_2] = [M_2] \circ [W]$, respectively. Since DMSOT transductions are closed under composition (see Proposition 1), $M'_1$ and $M'_2$ are obtained as DMSOT transducers.

**Step 2.** Let $N^S$ be a deterministic MSO string transducer that reads a given input string $w \in A_1 \cup A_2$ and outputs the string $wS$, where $S$ is a symbol not in $A_1 \cup A_2$. For $i \in \{2\}$, let $M_i$ be a deterministic MSO tree-to-string transducer with $[M_i] = [M_i] \circ [N]$. Now we can say that $[M_1] \neq [M_2]$ if and only if $[M'_1] \neq [M'_2]$.

**Step 3.** Let $D_1$ and $D_2$ be the domains of $M_1$ and $M_2$, respectively. Clearly, if $D_1 \neq D_2$ then $[M_1] \neq [M_2]$. It is decidable whether $D_1$ coincides with $D_2$ because the domain of every (D)MSOT transducer is regular (see, e.g., Theorem 5.82 of [23]). Then, assume $D_1 = D_2$ here and denote the domains of them by $D_1$. $[M_1] \neq [M_2]$ if and only if there exist $a \in A_1 \cup \{\}$, $b \in A_2 \cup \{\}$, $n \in \mathbb{N}$, and $t \in D$ such that $a \neq b$, $M_1(a)n/ = a$, and $M_2(b)n = b$. Let $M^{a, b}$ be the transducer of Lemma 7 for $a, b$, $M_1$, and $M_2$, $[M_1] \neq [M_2]$ iff $\text{dgr}(a^m b^n)$ is in $M^{a, b}(D)$ for some $n \in \mathbb{N}$. Note that $\text{dgr}(a^m b^n)$ is in $M^{a, b}(D)$ iff $(n, n)$ is in Par($M^{a, b}(D)$). By Lemma 5 the set of vectors Par($M^{a, b}(D)$) is Parikh because $D$ is regular and thus VR. By Lemma 6 it can be decided whether $(n, n)$ is in Par($M^{a, b}(D)$) for some $n \in \mathbb{N}$. This proves the main result of [20].

**Theorem 9.** It is decidable whether a deterministic MSOT transducer $M_1$ and a nondeterministic MSOT transducer $M_2$ are equivalent.

**Proof.** Recall that the domain of every (D)MSOT is regular, so we can decide whether the domain of $M_1$ coincides with that of $M_2$. Let us assume dom($M_1$) = dom($M_2$) = $D$. Since Lemmas 5–7 described above are not restricted to deterministic MSOT transducers, the same idea for DMSOT can be applied to our case. Thus we can state that $M_1 \in \text{DMSOT}$ and $M_2 \in \text{MSOT}$ are not equivalent if and only if there exist $a \in A_1 \cup \{\}$, $b \in A_2 \cup \{\}$, $n \in \mathbb{N}$, and $t \in D$ such that $a \neq b$, $M_1(a)n = a$, and $M_2(b)n = b$. The nondeterminism of $M_2$ makes differences in Steps 1–3, in which nondeterministic MSOT transducers $M_1$ and $M_2$ such that $[M_1'] = [M_1] \circ [W]$ and $[M_2'] = [M_2] \circ [W]$ are constructed instead of deterministic ones. Recall that MSOT transductions are closed under composition (Proposition 1), i.e., MSOT $\circ$ MSOT $\subseteq$ MSOT, but MSOT $\circ$ DMSOT $\not\subseteq$ DMSOT. For an input tree $t, a \in A_1 \cup \{\}$, and $b \in A_2 \cup \{\}$, assume $M_1(t) = w_1$ and $M_2(t) = w_2$ with $i \in \mathbb{N}, w_1/i = a$, $w_2/i = b$, $a \neq b$. In the case, obviously $M_1$ and $M_2$ are not equivalent. Let $M^{a, b}$ be an MSOT transducer in Step 3, which can be effectively constructed because Lemma 7 is not restricted to DMSOT transducers. By the assumption, $\text{dgr}(a^m b^n)$ is in $M^{a, b}(D)$, and so Par($M^{a, b}(D)$) contains $(i, i) \in \mathbb{N}^2$. Conversely, for distinct symbols $c \in A_1 \cup \{\}$ and $d \in A_2 \cup \{\}$, assume Par($M^{c, d}(D)$) includes $(j, j) \in \mathbb{N}^2$ with $j \in \mathbb{N}$. By the assumption, there exist a tree $t \in D$, $w_1 \in M_1$, $w_2 \in M_2'$ such that $M_1(t) = w_1S$, $w_2S \in M_2'(t)$, $w_1S /j = c$, and $w_2S /j = d$. Hence $M_1$ and $M_2$ are not equivalent. It is decidable whether there exists $(i, i) \in \mathbb{N}^2$ in Par($M^{a, b}(D)$) because Lemmas 5 and 6 are not restricted to DMSOT transducers.

We are now ready to describe our main result.

**Theorem 10.** For $v \in \text{ELR}_v^*$ as a view and $q \in \text{DMSOT}$ as a query, it is decidable whether $v \text{-preserves } q$.

**Proof.** By Theorem 2 we can construct a DT$_{ic}^R$ transducer that realizes a uniformizer $u$ of $v^{-1}$. Let $v \in \text{ELR}_v^*$, $q \in \text{DMSOT}$. We show that $v \text{-preserves } q$ if and only if $q = v \circ u \circ q$. The right-to-left direction is obvious. For the other direction, assume that $v \text{-preserves } q$ and let $\tilde{q}$ be a query in DMSOT such that $q = v \circ \tilde{q}$. Then, $v \circ u \circ q = v \circ u \circ \tilde{q}$ if $q$ holds. Precisely, let $t \in \text{dom}(q)$ and then,

$$q(\text{u}(t)) = \begin{cases} q(t') & t' \in \text{v}(t), \text{t''} \in \text{v}(t') \end{cases} = \begin{cases} q(t') & t' \in \text{v}(t), \text{t''} \in \text{v}(t') \end{cases} = q(t'), \text{t'} \in \text{v}(t'), \text{t''} \in \text{v}(t'') \begin{cases} q(t) & \text{v} \circ \tilde{q} \text{is single-valued}, \text{t'} \in \text{v}(t') \end{cases} = q(t).$$

It follows that $u \circ q \in \text{DMSOT}$ by Proposition 1 and Theorem 7.4 of [32] that states DT$_{ic}^R$ $\circ$ DMSOT $\subseteq$ DMSOT, and
6. Existential Preservation

Even when \( v \) does not universally preserve \( q \), it is still too early to give up on preserving information of \( q \). In the case, we would like to know whether \( v \) existentially preserves \( q \). There is a simple relation between the existential preservation and the universal preservation.

**Proposition 11.** Let \( v \), \( v' \) be (nondeterministic) views and \( q \) a single-valued query. If \( v' \subseteq v \) and \( v' \forall \)-preserves \( q \), then \( v \exists \)-preserves \( q \).

Based on Proposition 11, we give a decidable sufficient condition of the existential preservation. We only focus on finite-valued views, because we use a decomposition theorem for finite-valued B transducers (recall that the results in the previous section focus not only on finite-valued views but also on “infinite”-valued ones). Note that it is decidable in deterministic polynomial time whether a B transducer is finite-valued or not (Theorem 6.9 of [35]).

The following theorem states that every finite-valued B transducer can be effectively decomposed into a finite union of single-valued ones.

**Theorem 12.** [Theorem 6.2 of [21]] For every finite-valued B transducer \( M \) such that \( \|M\| = \|M_1\| \cup \cdots \cup \|M_k\| \), where \( K \leq 2^{\mathsf{p}(\mathsf{m})} \), \( \|M_j\| \leq 2^{\mathsf{p}(\mathsf{m})} \), \( j \in [K] \), for some polynomial \( P \) independent of \( M \).

By Proposition 11 and Theorem 12, we obtain a decidable sufficient condition of the existential preservation for views realized by finite-valued LB transducers and queries realized by DMSOT (or DT) transducers as follows. Notice that, in the following theorem, each \( v_j \) is single-valued, hence we just say “\( v_j \) preserves” instead of “\( v_j \exists \)-preserves.”

**Theorem 13.** Let \( v \) be a finite-valued LB transduction and \( q \) a DMSOT (or DT) transduction. Let \( v_1, \ldots, v_k \) be s-LB transductions such that \( v = v_1 \cup \cdots \cup v_k \). Then, \( v \exists \)-preserves \( q \) if \( v_j \) preserves \( q \) for some \( j \in [K] \).

Let \( \exists \)-Pres be the algorithm that decides the above sufficient condition by using Theorem 12. We are not sure whether \( \exists \)-Pres (and the other algorithms that are sound in the rest of this paper) is complete for existential preservation, because even when \( \exists \)-Pres answers “no,” another s-LB transduction \( v' \) may exist such that \( v' \subseteq v \) and \( v' \) preserves \( q \). The problem of deciding whether such \( v' \) exists seems to be hard, because one is required to prove a given query \( q \) is not preserved by \( v_i \) (\( i \in [K] \)) for every possible way of decomposing a finite-valued tree transduction \( v \) into single-valued ones \( v_1, \ldots, v_k \).

**Extending the domain of a view.** Let \( v \) be a view given by a finite-valued LB transducer and \( q \) a query given by a DMSOT transducer. When \( \exists \)-Pres answers “no,” there remains a possibility that \( v \) existentially preserves \( q \). When decomposing a view \( v \) into \( v_1, \ldots, v_k \) by Theorem 12, the domain of the resulting transduction \( v_i \) may be a proper subset of the domain of the original transduction \( v \). Consider the case when \( \text{dom}(q) \supseteq \text{dom}(v_i) \) for each \( i \in [K] \). In the case, by the definition of query preservation, every \( v_i \) does not preserve \( q \). Still, there may be \( S \subseteq [K] \) such that the union \( v' \) of transductions \( \forall \)-preserves \( q \) where \( v' = \bigcup_{j \in S} v_j \). If so, we can conclude that \( \bigcup_{i \in [K]} v_j \exists \)-preserves \( q \). Hence, the following result holds as a corollary of Theorem 10. Note that every union of s-LB transductions is an LB transduction.

**Corollary 14.** Let \( v_1, \ldots, v_k \in s\text{-LB} \) and \( q \in \text{DMSOT} \). It is decidable whether there is \( S \subseteq [K] \) such that \( v' \forall \)-preserves \( q \) where \( v' = \bigcup_{j \in S} v_j \).

Let \( q \) be a query given by a s-B transducer. Since s-B is incomparable with DMSOT, we cannot apply Theorem 10 directly to obtain a decidable sufficient condition similar to the one stated in Corollary 14 for \( q \in s\text{-LB} \). For the case, in order to construct an appropriate view \( v' (\subseteq v) \) from \( v_1, \ldots, v_k \) obtained by decomposing \( v \) by Theorem 12, each component \( v_j \) of \( v' \) is required not to be joinable with another. For views \( v_1, v_2 \) and a query \( q \), we say that \( v_1 \) and \( v_2 \) are joinable against \( q \) if there exists a pair of trees \( (t_1, t_2) \) such that \( q(t_1) \neq q(t_2) \) and \( v_1(t_1) = v_2(t_2) \). To show why transductions must not be joinable, let us suppose that \( v_1 \) and \( v_2 \) are joinable against \( q \) with a pair \((t_1, t_2)\) and \( v' = v_1 \cup v_2 \) (at least) existentially preserves \( q \), then there exists \( q' \) in the same class as \( q \)’s (s-B) such that for all \( t \in \text{dom}(q) \) there exists \( t' \in v'(t) \) satisfying \( q(t) = q'(t') \). However, \( q' \)’s \((t_1', t_2')\) for \( t_1' = v_1(t_1) = v_2(t_2) \) has to be \((q(t_1), q(t_2)) \) but \( q' \) is single-valued, which is a contradiction. Thus, each component \( v_j \) of \( v' \) is required not to be joinable against \( q \) with another. We show that joinability is decidable by using the single-valuedness test for EB transducers with “grafting” presented in [7].

**Lemma 15.** Let \( v_1, v_2 \in s\text{-LB} \) and \( q \in s\text{-B} \). It is decidable whether \( v_1 \) and \( v_2 \) are joinable against \( q \).

**Proof.** Construct ELB transducers with grafting \( (\text{ELB}^\exists)^\forall \) for short) that realize \( v_1^{-1} \) and \( v_2^{-1} \), respectively. The detailed construction is given in [7] (see Lemma 6 in it). Next, construct a TA A that accepts the intersection of the ranges of \( v_1 \) and \( v_2 \), i.e., \( L(A) = \text{ran}(v_1) \cap \text{ran}(v_2) \). A can be effectively constructed by the fact that the range of every LB transducer...
is effectively regular (see, e.g., Corollary 3.11 of [34]) and by the construction of product automaton. After that, construct an ELB-transducer that realizes \( w = \epsilon_1 \cup \epsilon_2^{\ast} \) to \( L(A) \), which is the union of \( \epsilon_1 \) and \( \epsilon_2^{\ast} \), and the domain of which is restricted to \( L(A) \). Finally, decide whether \( q = w \circ q \) is single-valued or not, which is decidable by Lemmas 7–8 of [7]. It is not difficult to show that \( v_1 \) and \( v_2 \) are joinable against \( q \) if and only if \( q' \) is not single-valued. The left-to-right direction holds obviously by the definition of joinability. Conversely, if \( q' \) is not single-valued, there exists a tree \( t \) such that \( t' \) is \( q'(t) \) with \( t' \neq t' \). Since \( q' \) is single-valued, there exist \( t' \) such that \( q(t_1) = t' \neq t'(t_2) = t_2(t_2) \). Thus \( v_1 \) and \( v_2 \) are joinable against \( q \).

**Lemma 16.** Let \( v_1, \ldots, v_k \in s-LB \) and \( q \in s-B \). The union \( q' \) preserves \( q \) if and only if (1) each component \( v_i \) of \( q' \) is not joinable against \( q \) with another, (2) \( v_j \) preserves \( q(\text{dom}(v_j)) \), and (3) \( \text{dom}(q) \subseteq \text{dom}(q') \).

**Proof.** As described in the beginning of this subsection, if (1) does not hold, \( q' \) does not preserve \( q \), and hence \( q' \) also does not preserve \( q \). On the other side, suppose (1), (2), and (3) hold. For simplicity, suppose \( K = 2 \). By (2), let \( q_1', q_2' \) be s-LB transductions such that \( q(\text{dom}(v_i)) = v_i \circ q_i' \) (\( i \in [2] \)). For any \( t \in \text{dom}(v_i) \) \( \cap \text{dom}(v_j) \), \( q(t) = v_i \circ q_i'(t) = v_i \circ q_2'(t) \). Hence for any \( t \in \text{dom}(v_i) \cup \text{dom}(v_j) \), \( q(t) = v_i \circ q_i'(t) \) holds. As \( q_i' \) is single-valued by (1). Therefore, \( v_i \circ q_i' \subseteq q \). Hence \( v_i \cup v_j \subseteq q \). Generally, \( v_1 \cup 
vdots \cup v_k \subseteq q \). Thus \( v_i \circ q_i' \subseteq q \).

By Lemmas 15, 16, we obtain the following result for queries realized by s-B transducers.

**Theorem 17.** Let \( v_1, \ldots, v_k \in s-LB \) and \( q \in s-B \). It is decidable whether there is \( S \subseteq [K] \) such that \( q \) preserves \( q \) where \( v' = \cup_{j \in S} v_j \). If such an \( S \) exists, \( \cup_{j \in S} v_j \) \( \not\exists \)-preserves \( q \) (by Proposition 11).

**Proof.** It suffices to show that the three conditions (1)–(3) in Lemma 16 are decidable. (1) is decidable by Lemma 15 and (2) is also decidable [7] (see Table 1). (3) is decidable due to the regularity of \( \text{dom}(q) \) and \( \text{dom}(q') \).

7. **Nondeterministic Queries**

Let \( v \) be a view and \( q \) be a nondeterministic query. In this section, we show two algorithms that are sound for the problem of deciding whether \( v \) universally or existentially preserves a part of \( q \) (see Sect. 4 for definition).

We adopt the idea of Theorem 13 to obtain an algorithm called \( V-PresPart \) that is sound for query preservation for finite-valued queries. \( V-PresPart \) is almost the same as \( \exists-Pres \) (see the previous section), but it decomposes a given query \( q \) into \( q_1, \ldots, q_k \) instead of a given view \( v \), after that for each \( i \in [K] \) it tests whether \( v \) preserves \( q_i \).

**Theorem 18.** For \( v \in (ELB)^{\ast} \) as a view and a finite-valued LB transduction \( q \) as a query, the algorithm \( V-PresPart \) is sound for the problem of deciding whether \( v \) preserves a part of \( q \).

**Proof.** Suppose \( q_1, \ldots, q_K \in s-LB \) that are obtained by decomposing \( q \). By Proposition 4 and by Theorem 10, it is decidable whether \( v \) preserves \( q_i \) for each \( i \in [K] \). Our algorithm \( V-PresPart \) answers “yes” (that means \( v \) preserves a part of \( q \)) if and only if there exists \( j \in [K] \) such that \( v \) preserves \( q_j \).

By Theorems 13, 18, and by Theorem 21 of [7] that states query preservation is decidable for a view realized by a s-LB transducer and a query realized by an s-B transducer (see Table 1), we also obtain an algorithm (called \( \exists-PresPart \)) for the case when views and queries are both finite-valued. We do not describe explicitly the algorithm \( \exists-PresPart \) here because it can easily be obtained by modifying slightly the algorithm \( V-PresPart \).

**Corollary 19.** For a view \( v \) (resp. query \( q \)) realized by a finite-valued LB (resp. B) transducer, the algorithm \( \exists-PresPart \) is sound for the problem of deciding whether \( v \) preserves a part of \( q \).

By the way, in [10] the “weak condition” is considered. For a view \( v \) and a query \( q \), \( v \) preserves \( q \) if there exists a mapping \( q' \in Q \) such that \( (1) (d \circ q') = \text{dom}(q) \) and \( (2) (d \circ q')(t) = q(t) \) for every \( t \in \text{dom}(q) \). For some situation in practice, the condition (1) is weakened to \( \text{dom}(v \circ q') \subseteq \text{dom}(q) \). The above results (Theorems, Lemmas, and Corollaries 10–19) can be applied to the weak condition by restricting \( \text{dom}(d) \) to \( \text{dom}(q) \cap \text{dom}(v)^{\ast} \).

8. **Conclusions and Open Problems**

We have defined two kinds of query preservation problem for nondeterministic views and queries on ranked trees: universal preservation and existential preservation. We have proved that the universal preservation problem is decidable for compositions of extended linear top-down tree transducers with regular look-ahead as views and deterministic MSO tree transducers as queries (see Theorem 10). To obtain the result we have slightly generalized the result [20] of the equivalence problem for deterministic MSO tree transducers (see Theorem 9). Moreover, we have shown an algorithm that is sound for the existential preservation for finite-valued linear bottom-up tree transducers as views and deterministic MSO tree transducers as queries (see Theorem 13), and also shown additional algorithms that are sound for the problem for nondeterministic queries realized by finite-valued (linear) bottom-up tree transducers (see Theorem 18 and Corollary 19). We would like to know whether (1) a sound
and complete algorithm exists for the existential preservation, and (2) our results can be extended to more expressive classes of tree transducers such as macro tree transducers (see, e.g., [32], [36], [37]). Obtaining a positive solution for the question (1) seems difficult, because one is required to prove a given query $q$ is not preserved by $v_i$ ($i \in [K]$) for every possible way of decomposing a finite-valued tree transduction $v$ into single-valued ones $v_1, \ldots, v_K$.

As mentioned in Theorem 12, finite-valued bottom-up tree transducers can be effectively decomposed into a finite number of single-valued ones of double-exponential order of the size of the original transducers. Whereas, in the word case, $k$-valued (word) transducers can be effectively decomposed into $k$ single-valued (unambiguous) ones [38], [39] of single-exponential size [40]. Can $k$-valued tree transducers decomposed into $k$ single-valued ones of single-exponential size? It is an important problem that remains open for twenty years.

Acknowledgements

This work was partially supported by JSPS KAKENHI Grant Numbers 26870270 and 23300008.

References

[32] J. Engelfriet and S. Maneth, "Macro tree transducers, attribute grammars, and MSO definable tree transductions," Information and Com-
Kazuki Miyahara received the B.E. degree from Nagaoka University of Technology and obtained his M.E. in Information Science from Nara Institute of Science and Technology in 2011 and 2013, respectively. He is currently a doctoral student of Graduate School of Information Science, Nara Institute of Science and Technology. His research interests include signal processing and formal language theory.

Kenji Hashimoto received the Ph.D. degree in Information and Computer Sciences from Osaka University in 2009. From 2009 to 2013, he was an Assistant Professor of Nara Institute of Science and Technology. Since Oct. 2013, he has been an Assistant Professor of Graduate School of Information Science, Nagoya University. His research interests include formal language, database theory, and information security.

Hiroyuki Seki received his Ph.D. degree from Osaka University in 1987. He was an Assistant Professor, and later, an Associate Professor from 1987 to 1994. In 1994, he joined Nara Institute of Science and Technology, where he was a Professor during 1996 to 2013. Currently, he is a Professor in Nagoya University. His current research interests include formal language theory and formal approach to software development.