INFORMATION

SCIENCE

TECHNICAL

REPORT

NAIST-IS-TR2012002

ISSN　0919-9527

Determinacy and Subsumption for Single-valued Bottom-up Tree Transducers

Kenji Hashimoto, Ryuta Sawada, Yasunori Ishihara, Hiroyuki Seki and Toru Fujiwara

November 2012

NAIST

〒630-0192

奈良県生駒市高山町 8916-5

奈良先端科学技術大学院大学

情報科学研究科

Graduate School of Information Science

Nara Institute of Science and Technology

8916-5 Takayama, Ikoma, Nara 630-0192, Japan

Determinacy and Subsumption for Single-valued Bottom-up Tree Transducers

Kenji Hashimoto[1], Ryuta Sawada[2], Yasunori Ishihara[2], Hiroyuki Seki[1], and Toru Fujiwara[2]

[1] Nara Institute of Science and Technology, Japan
{k-hasimt,seki}@is.naist.jp
[2] Osaka University, Japan
{ishihara,fujiwara}@ist.osaka-u.ac.jp

Abstract. This paper discusses the decidability of determinacy and subsumption for tree transducers. For two tree transducers T_1 and T_2, T_1 determines T_2 if the output of T_2 is identified by the output of T_1, that is, there is a partial function f such that $[\![T_2]\!] = f \circ [\![T_1]\!]$ where $[\![T_1]\!]$ and $[\![T_2]\!]$ are tree transformation relations induced by T_1 and T_2, respectively. Also, T_1 subsumes T_2 if T_1 determines T_2 and the partial function f such that $[\![T_2]\!] = f \circ [\![T_1]\!]$ can be defined by a transducer in a designated class that T_2 belongs to. In this paper, we show that determinacy is decidable for single-valued linear extended bottom-up tree transducers as the determiner class and single-valued bottom-up tree transducers as the determinee class. We also show that subsumption is decidable for these classes.

1 Introduction

In data transformation, it is desirable that certain information in source data be preserved through transformation. As a formalization for information preservation in data transformation, the notions of *deteminacy* and *subsumption* (or query rewriting) are known [1–3]. Let Q be a query to a database and V be a data transformation (or a view definition) of the database. Determinacy of Q by V means that the answer to Q is identified by the answer to V. When information to be preserved is specified by a query Q, determinacy guarantees that for any database instance D, $V(D)$ gives enough information to uniquely determine the specified information $Q(D)$ for D. Subsumption means that the answer to Q can always be computed from the answer to V by some query in a designated class that Q belongs to. Compared with determinacy, subsumption guarantees that the necessary information $Q(D)$ can be extracted from the transformed data $V(D)$ by the same query language expressing Q.

We study the decidability of determinacy and subsumption when both a query and a data transformation are given by tree transducers. Tree transducers are machines that model relations between labeled ordered trees. A tree transducer is said to be *single-valued* if the tree transformation induced by the transducer is a partial function. Since an XML document has a tree structure, tree transducers are often used as a model of XML document transformations. Formally, for two single-valued tree transducers T_1 and T_2 in classes Π_1 and Π_2 of transducers, respectively, we say T_1 *determines* T_2 if there is a partial function f such that $[\![T_2]\!] = f \circ [\![T_1]\!]$ (see Fig. 1(a)), where $[\![T_1]\!]$ and

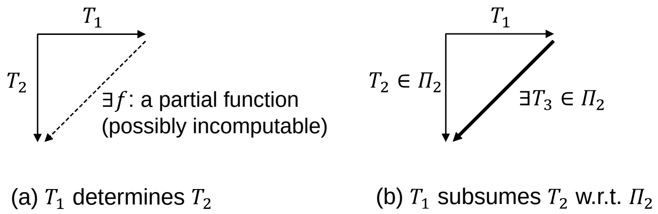

Fig. 1. Determinacy and subsumption

$[\![T_2]\!]$ are the tree transformation relations induced by T_1 and T_2, respectively. Π_1 and Π_2 are called the determiner class and the determinee class, respectively. We also say T_1 *subsumes* T_2 with respect to Π_2, if T_1 determines T_2 and the partial function f such that $[\![T_2]\!] = f \circ [\![T_1]\!]$ can be defined by a transducer in the class Π_2 (see Fig. 1(b)). Our goal is to find practical subclasses of tree transducers for which determinacy and subsumption are decidable, and to consider the problem of constructing a tree transducer T_3 in the determinee class such that $[\![T_2]\!] = [\![T_3]\!] \circ [\![T_1]\!]$ if T_1 subsumes T_2.

In this paper, we first show that determinacy is decidable for single-valued linear extended bottom-up tree transducers (sl-xbots) as the determiner class and single-valued bottom-up tree transducers (s-bots) as the determinee class running over a ranked-tree encoding of the given XML document. Transformations induced by transducers in the classes include simple filterings, relabelings, insertions, and deletions of elements. Especially, sl-xbots do not allow duplications of elements. Given an sl-xbot T_1 and an s-bot T_2, the decision procedure works as follows: (1) construct a transducer T_1^{inv} that induces the inverse of T_1, (2) construct a transducer T_3 that induces the composition of T_1^{inv} followed by T_2, then (3) decide whether T_3 is single-valued. We introduce a class of transducers with *grafting*, which allows to insert any tree in a specified tree language, in order to capture inverses of transformations induced by sl-xbots and the composition of the inverses and s-bots. Next, we prove that single-valuedness for the class is decidable. For some other classes, we show that determinacy is undecidable even for homomorphism tree transducers as the determiner class, which is a proper subclass of s-bots and single-valued top-down tree transducers (s-tops). Moreover, determinacy is undecidable for deterministic monadic second-order logic defined tree transducers (dmsott) [4, 5] as the determiner class, which form a class incompatible with s-bots and s-tops but is a proper superclass of sl-xbots. Lastly, we show that subsumption is decidable for sl-xbots as the determiner class and s-bots as the determinee class. The proof gives a construction method of an s-bot T_3 satisfying $[\![T_2]\!] = [\![T_3]\!] \circ [\![T_1]\!]$ if T_1 subsumes T_2.

Related Work. Determinacy and subsumption (or query rewriting) have been well studied mainly for relational queries such as first order logic and conjunctive queries [1–3]. In XML context, query preservation [6] has been studied as a notion of information preservation of XML mappings. Let $\mathcal{L}$ be an XML query language. For queries $Q, Q' \in \mathcal{L}$ and a view V, V *preserves* Q with Q' if $Q = Q' \circ V$. This definition is essentially the same as the definition of subsumption. A view V is *query preserving* with respect to $\mathcal{L}$ if there exists a computable function R_w such that for any query $Q \in \mathcal{L}$, V preserves Q with $R_w(Q)$. Unfortunately, it is known to be undecidable to decide whether V is query preserving with respect to projection queries, given a view V in any query class $\mathcal{L}_f$ which can simulate first order logic queries, such as XQuery and XSLT. It is also undecidable whether V preserves Q with some projection query, given V in the class $\mathcal{L}_f$ and a projection query Q. As far as we know, the decidability of query preservation for other subclasses of XQuery and XSLT has been little investigated.

2 Preliminaries

2.1 Trees and Tree Automata

We treat only ranked labeled ordered trees and tree transducers which work on such trees. Though an XML document is often modeled by an unranked labeled ordered tree, we assume that an unranked tree is encoded to a ranked tree by some encoding such as First-Child-Next-Sibling encoding [7] and DTD-based encoding [8].

We denote the set of nonnegative integers by $\mathbb{N}$. Let $[i, j] = \{d \in \mathbb{N} \mid i \leq d \leq j\}$. In particular, we denote $[1, k]$ by $[k]$. A (ranked) alphabet is a finite set Σ of symbols with a mapping rk from Σ to $\mathbb{N}$. We denote the set of k-ary symbols of Σ by $\Sigma^{(k)} = \{\sigma \in \Sigma \mid \mathsf{rk}(\sigma) = k\}$. The set $\mathcal{T}_\Sigma$ of *ranked trees* over an alphabet Σ is the smallest set T such that $\sigma(t_1, \ldots, t_k) \in T$ for every $k \in \mathbb{N}$, $\sigma \in \Sigma^{(k)}$, and $t_1, \ldots, t_k \in T$. If $\sigma \in \Sigma^{(0)}$, we write σ instead of $\sigma()$. The set of *positions* of $t = \sigma(t_1, \ldots, t_k) \in \mathcal{T}_\Sigma$, denoted by $pos(t)$, is defined by $pos(t) = \{\epsilon\} \cup \{ip \mid i \in [k], p \in pos(t_i)\}$ where $\sigma \in \Sigma^{(k)}$ and $t_1, \ldots, t_k \in \mathcal{T}_\Sigma$. The empty string ϵ is the position of the root of t, and the ith child's position of $p \in pos(t)$ is pi. We write $p \preceq p'$ when p is a prefix of p', that is, p is an ancestor position of p', and $p \prec p'$ when p is a proper prefix of p'. For $p, p' \in pos(t)$, let $\mathrm{nca}(p, p')$ be the nearest common ancestor position of p and p', that is, the longest common prefix of p and p'. For $p \in pos(t)$, $t|_p$ denotes the subtree of t at p, and $t[t']_p$ denotes the tree obtained from t by replacing the subtree at p with t'. Let $\lambda_t(p)$ be the symbol of tree t at p.

Let $X = \{x_*\} \cup \{x_i \mid i \geq 1\}$ be a set of variables of rank 0, and for every $k \geq 1$, $X_k = \{x_i \mid i \in [k]\}$. For $V \subseteq X$, we often write $\mathcal{T}_\Sigma(V)$ to mean $\mathcal{T}_{\Sigma \cup V}$. A tree $t \in \mathcal{T}_\Sigma(V)$ is *linear* if each variable in V occurs at most once in t. Let $C_\Sigma(V)$ denote the set of linear trees in $T_\Sigma(V)$. Let $\bar{\mathcal{T}}_\Sigma(V)$ (resp. $\bar{\mathcal{C}}_\Sigma(V)$) be the set of trees in $\mathcal{T}_\Sigma(V)$ (resp. $\mathcal{C}_\Sigma(V)$) such that each variable in V occurs at least once. Note that $\bar{\mathcal{T}}_{\Sigma \cup V}(V')$ denotes the set of trees in $\mathcal{T}_\Sigma(V \cup V')$ such that every variable in V' must occur at least once. For $t \in \mathcal{T}_\Sigma(X)$ and $\sigma \in \Sigma \cup X$, let $pos_\sigma(t)$ be the set of the positions of t at which σ occurs, and $pos_Y(t) = \bigcup_{\sigma \in Y} pos_\sigma(t)$ for $Y \subseteq \Sigma \cup X$. Let $\mathrm{var}(t)$ be the set of variables of t, and $\mathrm{yield}_X : \mathcal{T}_\Sigma(X) \to X^*$ be the function such that $\mathrm{yield}_X(x) = x$ for every $x \in X$ and $\mathrm{yield}_X(\sigma(t_1, \ldots, t_k)) = \mathrm{yield}_X(t_1) \cdots \mathrm{yield}_X(t_k)$ for every $\sigma \in \Sigma^{(k)}$ and $t_1, \ldots, t_k \in \mathcal{T}_\Sigma(X)$. A tree $t \in \mathcal{T}_\Sigma(X)$ is *normalized* if $\mathrm{yield}_X(t) = x_1 \cdots x_k$ for some $k \in \mathbb{N}$. Every mapping $\theta : V \to \mathcal{T}_\Sigma(X)$ with $V \subseteq X$ is called a substitution. It can be extended to $\theta : \mathcal{T}_\Sigma(V) \to \mathcal{T}_\Sigma(X)$ defined inductively as follows: $x\theta = \theta(x)$ for every $x \in V$ and $t\theta = \sigma(t_1\theta, \ldots, t_k\theta)$ for every $t = \sigma(t_1, \ldots, t_k) \in \mathcal{T}_\Sigma(V)$ where $\sigma \in \Sigma^{(k)}$. If $V = X_k$ and $x_i\theta = t_i$ for each $i \in [k]$, we also denote $t\theta$ by $t[t_1, \ldots, t_k]$, and if $V = \{x\}$ and $\theta(x) = t'$, we denote $t\theta$ by $t[x \leftarrow t']$. In particular, if $V = \{x_*\}$ and $\theta(x_*) = t'$, we denote $t\theta$ by $t[t']$ or often tt' without brackets.

A *finite tree automaton* (TA for short) is a 4-tuple $A = (Q, \Sigma, Q_a, \gamma)$, where Q is a finite set of states, Σ is an alphabet, $Q_a \subseteq Q$ is a set of accepting states, and γ is a finite set of transition rules, each of which is of the form $(q, C[q_1, \dots, q_k])$ where $q, q_1, \dots, q_k \in Q$ and $C \in \bar{\mathcal{C}}_\Sigma(X_k)$. The move relation $\Rightarrow_A$ of a TA $A = (Q, \Sigma, Q_a, \gamma)$ is defined as follows: if $(q, C[q_1, \dots, q_k]) \in \gamma$ and $t|_p = C[q_1, \dots, q_k]$ where $p \in pos(t)$, then $t \Rightarrow_A t[q]_p$. The tree language *recognized* by A, denoted as $L(A)$, is $\{t \mid t \Rightarrow_A^* q_a, q_a \in Q_a\}$ where $\Rightarrow_A^*$ is the reflexive transitive closure of $\Rightarrow_A$. For a state q of A, let $A(q)$ be a TA obtained from A by replacing the set Q_a of accepting states with the singleton $\{q\}$. A set L of trees recognized by some TA is called a regular tree language, or we say L is regular.

2.2 Tree Transducers

An *extended bottom-up tree transducer* (xbot) [9] is a 5-tuple $(Q, \Sigma, \Delta, Q_f, \delta)$, where Q is a finite set of states, Σ is an input alphabet, Δ is an output alphabet, $Q_f \subseteq Q$ is a set of final states, and δ is a set of transduction rules of the form $C_l[q_1(x_1), \dots, q_k(x_k)] \to q(t_r)$ where $k \in \mathbb{N}$, $C_l \in \bar{\mathcal{C}}_\Sigma(X_k)$, $t_r \in \mathcal{T}_\Delta(X_k)$, $q, q_1, \dots, q_k \in Q$. A rule is normalized if its left-hand side is normalized. Without loss of generality, we can assume that every rule is normalized. A rule $\rho \in \delta$ is an *ϵ-rule* if the left-hand side of ρ is the form $q(x)$ where $q \in Q$ and $x \in X$, and it is *input-consuming* otherwise. Let $T = (Q, \Sigma, \Delta, Q_f, \delta)$ be an xbot. T is a *bottom-up tree transducer* (bot) if the left-hand side of every rule in δ contains exactly one symbol in Σ. Also, we denote by an xbot^{-e} an xbot without ϵ-rules. T is a *linear extended bottom-up tree transducer* (l-xbot) if the tree t_r in the right-hand side of each rule in δ is linear.

The move relation $\Rightarrow_T$ of an xbot $T = (Q, \Sigma, \Delta, Q_f, \delta)$ is defined as follows: $t \Rightarrow_T^\rho t'$ for a rule $\rho = (C_l[q_1(x_1), \dots, q_k(x_k)] \to q(t_r)) \in \delta$ if there exists a position $p \in pos(t)$ such that $t|_p = C_l[q_1(t_1), \dots, q_k(t_k)]$ where $t_1, \dots, t_k \in \mathcal{T}_\Delta(X)$ and $t' = t[q(t_r[t_1, \dots, t_k])]_p$, and $t \Rightarrow_T t'$ if there exists $\rho \in \delta$ such that $t \Rightarrow_T^\rho t'$. The transformation *induced* by T, denoted as $[\![T]\!]$, is the relation defined as $\{(t, t') \mid t \Rightarrow_T^* q_f(t'), t \in \mathcal{T}_\Sigma, t' \in \mathcal{T}_\Delta, q_f \in Q_f\}$ where $\Rightarrow_T^*$ is the reflexive transitive closure of $\Rightarrow_T$. The domain of T, denoted by $\mathrm{dom}(T)$, is $\{t \mid (t, t') \in [\![T]\!]\}$, and the range of T, denoted by $\mathrm{rng}(T)$, is $\{t' \mid (t, t') \in [\![T]\!]\}$. For a tree t, $[\![T]\!](t) = \{t' \mid (t, t') \in [\![T]\!]\}$. For a TA A, the image $T(A)$ of $L(A)$ by T is $\{t' \mid (t, t') \in [\![T]\!], t \in L(A)\}$. For a state q of T, let $T(q)$ be an xbot obtained from T by replacing the set Q_f of final states with the singleton $\{q\}$.

The tree transducers T and T' are *equivalent* if $[\![T]\!] = [\![T']\!]$. For tree transducers T_1 and T_2, $[\![T_2]\!] \circ [\![T_1]\!] = \{(t, t') \mid (t, t'') \in [\![T_1]\!], (t'', t') \in [\![T_2]\!]\}$. A transducer T is said to be *single-valued* (or *functional*) if any two pairs of (t, t') and (t, t'') in $[\![T]\!]$ satisfy $t' = t''$. We denote the unique output tree of T on a tree t by $T(t)$. It is known that the single-valuedness of bots is decidable in polynomial time [10]. We use the prefix 's' to represent that a transducer is single-valued, e.g., we write for short an s-xbot to denote a single-valued xbot.

Without loss of generality, we assume that any alphabet contains a special symbol $\perp$, which means "no output" and does not occur in any final output tree. We recall the notion of reducedness [10], which is defined for bots but can be naturally applied to xbots. An xbot $T = (Q, \Sigma, \Delta, Q_f, \delta)$ is called *reduced* if and only if the following two conditions hold:

1. T has no useless states, that is, for every state $q \in Q$, there exists a tree $t = Ct_s \in \mathrm{dom}(T)$ where $C \in \bar{\mathcal{C}}_\Sigma(\{x_*\})$ such that $t \Rightarrow_T^* C[q(t'_s)] \Rightarrow_T^* q_f(t')$ for some $q_f \in Q_f$ and $t'_s, t' \in \mathcal{T}_\Delta$.
2. There exists a subset $U(T)$ of Q such that for every rule $C_l[q_1(x_1), \ldots, q_k(x_k)] \to q(t_r) \in \delta$,
   - if $q \in U(T)$ then $t_r = \bot$ and $q_i \in U(T)$ for each $i \in [k]$, and
   - if $q \notin U(T)$ then (1) $t_r \neq \bot$ and (2) for each $i \in [k]$, $q_i \in U(T)$ if and only if $x_i \notin \mathrm{var}(t_r)$.
3. If $q \in Q_f$ then q does not occur in the left-hand side of any rule in δ.

Note that for any $q \in U(T)$ and $t = Ct_s \in \mathrm{dom}(T)$ where $C \in \bar{\mathcal{C}}_\Sigma(\{x_*\})$, if $t \Rightarrow_T^* C[q(t'_2)]$ then $t'_2 = \bot$ and the final output for t does not contain $\bot$. That is, the intermediate output at q is always $\bot$ and it is eventually abandoned. Conversely, for $q \in Q - U(T)$, the intermediate output at q is in $\mathcal{T}_{\Delta-\{\bot\}}$ and it is contained in the final output. For every xbot T, a reduced xbot equivalent with T can be constructed in linear time in the same way as the construction for bots [10] (see also Appendix A).

2.3 Determinacy and Subsumption of Tree Transducers

Let Π_1 and Π_2 be arbitrary classes of tree transducers.

Definition 1 (Determinacy). *Let T_1 and T_2 be tree transducers in Π_1 and Π_2, respectively, such that $\mathrm{dom}(T_2) \subseteq \mathrm{dom}(T_1)$. T_1* determines *T_2 iff there exists a partial function f such that $[\![T_2]\!] = f \circ [\![T_1]\!]$. Π_1 is called the* determiner class *and Π_2 is called the* determinee class.

Definition 2 (Subsumption). *Let T_1 and T_2 be tree transducers in Π_1 and Π_2, respectively, such that $\mathrm{dom}(T_2) \subseteq \mathrm{dom}(T_1)$. T_1* subsumes *T_2 with respect to Π_2 iff there exists a single-valued transducer $T_3 \in \Pi_2$ such that $[\![T_2]\!] = [\![T_3]\!] \circ [\![T_1]\!]$.*

From the definition, if T_1 subsumes T_2 then T_1 determines T_2. Conversely, even if there exists some function f such that $[\![T_2]\!] = f \circ [\![T_1]\!]$, f cannot always be induced by some transducer in Π_2 in general.

If determinacy is decidable for a determiner class Π_1 and a determinee class Π_2, we simply say determinacy is decidable for (Π_1, Π_2). We will use a similar notation for subsumption.

3 Determinacy

3.1 Decidability for (sl-xbots, s-bots)

We consider the problem of deciding whether, given single-valued linear xbot (sl-xbot) T_1 and single-valued bot (s-bot) T_2 such that $\mathrm{dom}(T_2) \subseteq \mathrm{dom}(T_1)$, T_1 determines T_2 or not. Our approach is based on the following proposition.

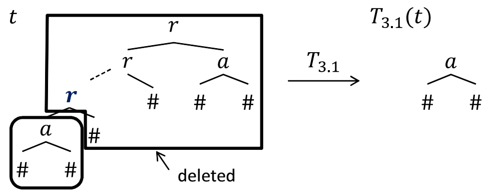

Fig. 2. A transducer $T_{3.1}$

Proposition 1. *For any single-valued transducers* T_1 *and* T_2 *such that* $\mathrm{dom}(T_2) \subseteq \mathrm{dom}(T_1)$*,* T_1 *determines* T_2 *if and only if* $[\![T_2]\!] \circ [\![T_1]\!]^{-1}$ *is a partial function, where* $[\![T_1]\!]^{-1} = \{(t', t) \mid (t, t') \in [\![T_1]\!]\}$.

According to Proposition 1, given sl-xbot T_1 and s-bot T_2, our decision algorithm works as follows:

Step 1: Construct a transducer T_1^{inv} such that $[\![T_1^{inv}]\!] = [\![T_1]\!]^{-1}$;
Step 2: Construct a transducer T_3 such that $[\![T_3]\!] = [\![T_2]\!] \circ [\![T_1^{inv}]\!]$;
Step 3: Decide whether T_3 is single-valued.

In Step 1, the inverse transducer T_1^{inv} of T_1 is computed. T_1^{inv} is not necessarily an l-xbot. Due to this, we introduce a slightly larger class, linear extended bottom-up tree transducers with *grafting* (l-xbot$^{+\mathrm{g}}$ for short), that can represent not only inverses of l-xbots but also the composition of the inverses with s-bots. In Step 2, an xbot$^{+\mathrm{g}}$ T_3 which represents the composition of T_1^{inv} followed by T_2 is constructed. Lastly, it is determined whether the composition transducer T_3 is single-valued.

Before we explain the detail of each step, we give an example, which shows that even the inverse of an sl-bot cannot always be expressed by any l-xbot.

Example 1. Let $\Sigma = \{r, a, \#\}$ and $\Delta = \{a, \#\}$. Consider an sl-bot $T_{3.1} = (\{q_r, q\}, \Sigma, \Delta, \{q_r\}, \delta)$ where

$$\delta = \{\ \# \to q(\#), \quad a(q(x_1), q(x_2)) \to q(a(x_1, x_2)),$$
$$r(q(x_1), q(x_2)) \to q_r(x_1), \quad r(q_r(x_1), q(x_2)) \to q_r(x_1)\}.$$

In Fig. 2, t is transformed by $T_{3.1}$, which leaves only the subtree at the left child of the bottom-most r-node. There is an infinite number of trees t' such that $T_{3.1}(t') = T_{3.1}(t)$ because the inverse of $T_{3.1}$ allows to insert any number of r-labeled ancestor nodes having arbitrary trees in $\mathcal{T}_{\Sigma-\{r\}}$ as their right subtrees. For any l-xbot T without ϵ-rules, the image of a tree t by T is finite. Even if ϵ-rules are allowed, no l-xbot allows to insert a node having an arbitrary tree in $\mathcal{T}_{\Sigma-\{r\}}$ as its right subtree. Therefore, there is no l-xbot T such that $[\![T]\!] = [\![T_{3.1}]\!]^{-1}$.

To express the inverse of $T_{3.1}$ in Example 1, a transducer has to, for an input tree, insert any number of internal nodes and subtrees non-deterministically. To capture the inverse of sl-xbots, we extend xbots by *grafting*. We denote a tree transducer in the class by an xbot$^{+\mathrm{g}}$ for short. A grafting is represented by a special variable $\langle L\rangle$, called a g-variable, where $L \subseteq \mathcal{T}_\Delta$. When $L = L(A)$ where A is a TA over Δ, we often write $\langle A\rangle$ instead of $\langle L(A)\rangle$. A g-variable can occur as a symbol of rank 0 in the right-hand side of a rule. Let $G(\Delta)$ be the set of all the g-variables $\langle L\rangle$ where $L \subseteq \mathcal{T}_\Delta$. Let $\tilde{\mathcal{T}}_\Delta(X_i)$ denote the set of trees over Δ with X_i and $G(\Delta)$. Note that for $\tilde{t} \in \tilde{\mathcal{T}}_\Delta(X_i)$, $\mathrm{var}(\tilde{t})$ does not contain any g-variable. For $\tilde{t} \in \tilde{\mathcal{T}}_\Delta(X_i)$, let $S(\tilde{t})$ be the set of trees in $\mathcal{T}_\Delta(X_i)$ obtained from $\tilde{t}$ by replacing each g-variable $\langle L\rangle$ with a tree in L.

Formally, a transduction rule of an xbot$^{+\mathrm{g}}$ is the form $C_l[q_1(x_1),\dots,q_k(x_k)] \to q(\tilde{t}_r)$ where $k \in \mathbb{N}$, $C_l \in \bar{\mathcal{C}}_\Sigma(X_k)$, $\tilde{t}_r \in \tilde{\mathcal{T}}_\Delta(X_k)$, and $q, q_1, \dots, q_k$ are states. The move relation by a rule $C_l[q_1(x_1),\dots,q_k(x_k)] \to q(\tilde{t}_r)$ is as follows: if $t|_p = C_l[q_1(t_1),\dots,q_k(t_k)]$ where $t_1,\dots,t_k \in \mathcal{T}_\Delta$, then $t \Rightarrow t[q(t_r[t_1,\dots,t_k])]_p$ where $t_r \in S(\tilde{t}_r)$.

For an xbot$^{+\mathrm{g}}$, we write an xbot$^{+\mathrm{g(R)}}$ when L is regular for each g-variable $\langle L\rangle$. Also, we write an xbot$^{+\mathrm{g(B(R))}}$ when each g-variable is in the form of $\langle T(A)\rangle$ for some bot T and TA A.

Example 2. Consider an l-xbot$^{+\mathrm{g(R)}}$ $T_{3.2} = (\{q, q_r\}, \Delta, \Sigma, \{q_r\}, \delta')$ where

$$\delta' = \{\ \# \to q(\#),\quad a(q(x_1), q(x_2)) \to q(a(x_1, x_2)),\\ q(x_1) \to q_r(r(x_1, \langle A\rangle)),\quad q_r(x_1) \to q_r(r(x_1, \langle A\rangle))\}$$

and A is a TA such that $L(A) = \mathcal{T}_{\Sigma-\{r\}}$. Then, $T_{3.2}$ induces the inverse of $T_{3.1}$.

Steps 1 to 3 of the decision algorithm can be refined as follows.

Step 1: Inversion of sl-xbots. We provide a way to construct an l-xbot$^{+\mathrm{g}}$ representing the inverse of an sl-xbot. Intuitively, we just swap the input and output of each rules. However, we must take care of variables occurring only in the left-hand side, which mean deletions of subtrees. In swapping, g-variables are added instead of the variables.

Let $T = (Q, \Sigma, \Delta, Q_f, \delta)$ be an l-xbot. The swapping procedure is as follows.

1. Construct a TA $A_T = (Q, \Sigma, Q_f, \gamma)$ where $\gamma = \{(q, C_l[q_1,\dots,q_k]) \mid C_l[q_1(x_1),\dots,q_k(x_k)] \to q(C_r) \in \delta\}$. Note that A_T recognizes $\mathrm{dom}(T)$.
2. Construct an l-xbot$^{+\mathrm{g(R)}}$ $T' = (Q, \Delta, \Sigma, Q_f, \delta')$ such that δ' is the smallest set satisfying the following condition: Let $C_l[q_1(x_1),\dots,q_k(x_k)] \to q(C_r)$ be an arbitrary rule in δ. Let θ_l be the substitution such that $\theta_l(x_i) = q_i(x_i)$ for each $i \in [k]$, θ_r be the substitution such that $\theta_r(x_i) = x_i$ if $x_i \in \mathrm{var}(C_r)$ and $\theta_r(x_i) = \langle A_T(q_i)\rangle$ otherwise. Moreover, let θ_n be the substitution for normalization, which is the bijective function from $\mathrm{var}(C_r)$ to $X_{k'}$ ($k' = |\mathrm{var}(C_r)|$) making $(C_r\theta_l)\theta_n$ normalized. Then, $(C_r\theta_l)\theta_n \to (C_l\theta_r)\theta_n \in \delta'$.

Lemma 1. *For any l-xbot T, an l-xbot$^{+\mathrm{g(R)}}$ T^{inv} such that $[\![T^{inv}]\!] = [\![T]\!]^{-1}$ can be constructed.*

Proof. It can be shown by induction on move relations of the transducers that the inverse transducer T^{inv} of T is correctly constructed by the swapping. □

Step 2: Composition of l-xbot$^{+\mathrm{g(R)}}$ and s-bot. This step constructs an xbot$^{+\mathrm{g}}$ equivalent with the composition of the l-xbot$^{+\mathrm{g(R)}}$ T_1^{inv} followed by an s-bot T_2.

Lemma 2. *For any l-xbot$^{+\mathrm{g(R)}}$ T and bot T', an xbot$^{+\mathrm{g(B(R))}}$ T'' such that $[\![T'']\!] = [\![T']\!] \circ [\![T]\!]$ can be constructed.*

Proof. The lemma can be shown in a similar way to the proof of the closure property of l-bots under the composition [7, 11]. The difference is the existence of g-variables. Recall that a tree t in $L(A)$ is inserted at g-variable $\langle A\rangle$. On the composition transducer, we just insert the image of t by $T'(q)$ where q is the state at which T' processes t in the tree output by T. That is, we replace $\langle A\rangle$ with $\langle T'(q)(A)\rangle$. □

Step 3: Deciding single-valuedness of xbot$^{+\mathrm{g(B(R))}}$. This step decides whether the xbot$^{+\mathrm{g(B(R))}}$ obtained in Step 2 is single-valued. It is known that single-valuedness of bots is decidable in polynomial time [10]. However, the class of transformations induced by xbot$^{+\mathrm{g}}$s is a proper superclass of the class induced by bots.

Let T_3 be the xbot$^{+\mathrm{g(B(R))}}$ obtained in Step 2. The overview of Step 3 is as follows:

Step 3-1 Construct a reduced xbot $T_{3.1}$ equivalent with T_3 by eliminating g-variables. If there is no xbot equivalent with T_3, answer that T_3 is not single-valued and halt. Otherwise, go to 3-2.

Step 3-2 Construct a reduced xbot^{-e} $T_{3.2}$ equivalent with $T_{3.1}$. If there is no xbot^{-e} equivalent with $T_{3.1}$, answer that T_3 is not single-valued and halt. Otherwise, go to 3-3.

Step 3-3 Decide whether $T_{3.2}$ is single-valued or not.

We further refine the above sub-steps as follows.

Step 3-1*: Eliminating g-variables.* We show the following lemma for Step 3-1.

Lemma 3. *Let $T = (Q, \Sigma, \Delta, Q_f, \delta)$ be a reduced xbot$^{+\mathrm{g}}$. If T has a rule whose right-hand side has a state $q \in Q - U(T)$ and a g-variable $\langle L\rangle$ such that $|L| \geq 2$, then T is not single-valued.*

Proof. Assume that T has a rule $C_l[q_1(x_1), \ldots, q_k(x_k)] \to q(\tilde{t}_r)$ where $q \in Q - U(T)$ and $\tilde{t}_r$ has a g-variable $\langle L\rangle$ such that $|L| \geq 2$. Since T is reduced, there exist $t = CC_l[t_1, \ldots, t_k] \in \mathrm{dom}(T)$ where $C \in \bar{\mathcal{C}}_\Sigma(\{x_*\})$, $t' \in \bar{\mathcal{T}}_\Delta(\{x_*\})$, $t'_1, \ldots, t'_k \in \mathcal{T}_\Delta$, and $q_f \in Q_f$ such that $t \Rightarrow_T^* CC_l[q_1(t'_1), \ldots, q_k(t'_k)] \Rightarrow_T C[q(t_r[t'_1, \ldots, t'_k])] \Rightarrow_T^* q_f(t't_r[t'_1, \ldots, t'_k])$ for any $t_r \in S(\tilde{t}_r)$. Since $|L| \geq 2$, $S(\tilde{t}_r)$ has at least two distinct trees t_r^1 and t_r^2. Also, the positions of each variable of t_r^1 and t_r^2 are identical. Hence, $t_r^1[t'_1, \ldots, t'_k] \neq t_r^2[t'_1, \ldots, t'_k]$ and thus the final outputs $t't_r^1[t'_1, \ldots, t'_k]$ and $t't_r^2[t'_1, \ldots, t'_k]$ are different. Therefore, $|[\![T]\!](t)| \geq 2$. □

For a bot T, a TA A and any $k \in \mathbb{N}$, it can be checked whether $|T(A)| \geq k$. From the above lemma, Step 3-1 can be done as follows:

(i) For each rule with grafting $\langle T(A)\rangle$ of T_3,
 - if $T(A) = \emptyset$ then delete the rule, and

– if $T(A) = \{t\}$ for some tree t then replace $\langle T(A) \rangle$ with t.

(ii) Construct an equivalent reduced xbot$^{+\mathrm{g(B(R))}}$ $T_{3.1}$.

(iii) If $T_{3.1}$ has a rule containing $\langle T(A) \rangle$ with $|T(A)| \geq 2$, answer that T_3 is not single-valued and halt.

If the condition at (iii) does not hold, $T_{3.1}$ has no g-variable. Thus, $T_{3.1}$ is an xbot.

Step 3-2*: Eliminating ϵ-rules.* We show two lemmas before giving the procedure of Step 3-2. We will use an idea similar to the proof of Proposition 10 of [9].

We say that a nonempty subset δ_e of ϵ-rules is *repeatedly-producing at state* q if $q(x_*) \Rightarrow^*_{\delta_e} q(t)$ for some tree $t \in \bar{\mathcal{T}}_\Delta(\{x_*\}) - \{x_*\}$, where $\Rightarrow^*_{\delta_e}$ means zero ore more applications of rules in δ_e.

Lemma 4. *Let $T = (Q, \Sigma, \Delta, Q_f, \delta)$ be a reduced xbot. If there is a subset δ_e of ϵ-rules in δ repeatedly-producing at some $q \in Q - U(T)$, then T is not single-valued.*

Proof. Assume that there is a subset δ_e of ϵ-rules in δ repeatedly-producing at some $q \in Q - U(T)$. Then, there are trees $t \in \mathcal{T}_\Sigma$, $t' \in \mathcal{T}_\Delta$, $D \in \bar{\mathcal{C}}_\Sigma(\{x_*\})$, $t_c, t_D \in \bar{\mathcal{T}}_\Delta(\{x_*\})$, and $q_f \in Q_f$ such that t_c has at least one symbol in Δ and $D[t] \Rightarrow^*_T D[q(t')] \Rightarrow^*_T D[q(t_c t')] \Rightarrow^*_T D[q(t_c^n t')] \Rightarrow^*_T q_f(t_D t_c^n t')$ for any positive integer n. □

After the fashion of the reference [9], we call a state $q \in Q$ an *end state* if there exists an input-consuming rule whose left-hand side has q. The set of all end states of Q is denoted by $E(T)$. For each input-consuming rule $\rho = (C_l[q_1(x_1), \ldots, q_k(x_k)] \to q(t_r)) \in \delta$, let $\mathrm{rhs}(\rho) = \{q'(t) \mid q(t_r) \Rightarrow^*_T q'(t), q' \in E(T) \cup Q_f\}$. Note that only ϵ-rules can be used in the derivation $q(t_r) \Rightarrow^*_T q'(t)$.

Lemma 5. *Let $T = (Q, \Sigma, \Delta, Q_f, \delta)$ be a reduced xbot. If there is no subset δ_e of ϵ-rules in δ repeatedly-producing at any $q \in Q - U(T)$, then $\mathrm{rhs}(\rho)$ is finite for every rule ρ of T and an xbot^{-e} equivalent with T can be constructed.*

Proof. Assume that there is no subset δ_e of ϵ-rules in δ repeatedly-producing at any $q \in Q - U(T)$. By the assumption, for any tree t, $q(t) \Rightarrow^+_T q(t')$ and $t \neq t'$ imply that t' does not contain any variable. That is, $q(t) \Rightarrow^+_T q(t') \Rightarrow^*_T q(t'')$ and $t \neq t'$ imply $t' = t''$. Thus, suppose that n_r is the number of rules of T, and then for each rule $\rho = (C_l[q_1(x_1), \ldots, q_k(x_k)] \to q(t_r)) \in \delta$, $\mathrm{rhs}(\rho) = \{q'(t) \mid q(t_r) \Rightarrow^i_T q'(t), q' \in E(T) \cup Q_f, i \in [n_r]\}$ where $\Rightarrow^i_T$ means the move relation by i times applications of rules. Therefore, $\mathrm{rhs}(\rho)$ is finite. Then, we can construct an equivalent xbot^{-e} $T_e = (Q, \Sigma, \Delta, Q_f, \delta')$ where $\delta' = \bigcup_{l \to r \in \delta^\Sigma} \{l \to r' \mid r' \in \mathrm{rhs}(l \to r)\}$ and δ^Σ is the subset of input-consuming rules of δ. □

According to Lemmas 4 and 5, Step 3-2 consists of the following two substeps:

(i) Construct the weighted graph $G_{rp} = (Q - U(T_{3.1}), E_e)$ from $T_{3.1} = (Q, \Sigma, \Delta, Q_f, \delta)$ where $E_e = \{(q, q') \mid q(x_1) \to q'(t) \in \delta, t \in \bar{\mathcal{T}}_\Delta(\{x_1\})\}$, and the weight of each (q, q') is 1 if there is a rule $q(x_1) \to q'(t)$ such that t includes at least one output symbol, and otherwise 0. Find a cycle whose weight is at least one. If such a cycle exists, answer that T_3 is not single-valued and halt.

(ii) Construct an equivalent reduced xbot^{-e} $T_{3.2}$.

Step 3-3: *Deciding single-valuedness of xbot*$^{-e}$. In this substep, it is decided whether $T_{3.2}$ is single-valued or not. The idea of the proof is the same as that of the proof of the decidability of k-valuedness of bottom-up tree transducers [12]. While the proof in [12] uses the Engelfriet's property, we use a variant of the property (Lemma 6) to prove the decidability of single-valuedness of xbot^{-e}s.

We give some notations for the property. Let $\mathcal{T}_\Sigma[X_n] = \bar{\mathcal{T}}_\Sigma(X_n) \cup \mathcal{T}_\Sigma$, that is, every $t \in \mathcal{T}_\Sigma[X_n]$ has all the variables in X_n or has no variable. For $t, s \in \mathcal{T}_\Sigma[X_n]$, ts is the tree obtained from t by replacing each variable with s. Note that $ts = t$ if t has no variable. For $m \in [n]$, let $\mathcal{T}_\Sigma^{n,m}[X_n] = \mathcal{T}_\Sigma^{m-1} \times \mathcal{T}_\Sigma[X_n] \times \mathcal{T}_\Sigma^{n-m}$. For $\mathbf{t} \in \mathcal{T}_\Sigma^{n,m}[X_n]$, we denote by $t^{(i)}$ the ith element of $\mathbf{t}$, i.e., $\mathbf{t} = (t^{(1)}, \ldots, t^{(n)})$. For $s \in \mathcal{T}_\Sigma[X_n]$ and $\mathbf{t} \in \mathcal{T}_\Sigma^{n,m}[X_n]$, $s\mathbf{t}$ is the tree obtained from s by replacing x_i with $t^{(i)}$ for all $i \in [n]$. Let $\mathbf{tu} = (t^{(1)}\mathbf{u}, \ldots, t^{(n)}\mathbf{u})$ for $\mathbf{u} \in \mathcal{T}_\Sigma^{n,m}[X_n]$. Notice that since $\mathbf{t} \in \mathcal{T}_\Sigma^{n,m}[X_n]$, so is $\mathbf{tu}$. For $\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3, \mathbf{t}_4, \mathbf{t}_5 \in \mathcal{T}_\Sigma^{n,m}[X_n]$ and $S = \{i_1, \ldots, i_{|S|}\} \subseteq [1,5]$, let $\mathbf{t}_S = \mathbf{t}_{i_1} \cdots \mathbf{t}_{i_{|S|}}$ where $i_j < i_{j+1}$ for $j \in [|S| - 1]$.

Now, we give a variant of Engelfriet's Property (see Appendix B.1).

Lemma 6. *Let n, n' be arbitrary positive integers, and $m \in [n], m' \in [n']$. Suppose that $t_0 \in \mathcal{T}_\Sigma[X_n]$, $\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3, \mathbf{t}_4, \mathbf{t}_5 \in \mathcal{T}_\Sigma^{n,m}[X_n]$, $t'_0 \in \mathcal{T}_\Sigma[X_{n'}]$, $\mathbf{t}'_1, \mathbf{t}'_2, \mathbf{t}'_3, \mathbf{t}'_4, \mathbf{t}'_5 \in \mathcal{T}_\Sigma^{n',m'}[X_{n'}]$. If $t_0\mathbf{t}_S = t'_0\mathbf{t}'_S$ for every S such that $\{5\} \subseteq S \subset [1,5]$, then $t_0\mathbf{t}_{[1,5]} = t'_0\mathbf{t}'_{[1,5]}$.*

Next, in order to argue in a similar way to the proof of Theorem 2.2(i) in the reference [12], we decompose the left-hand side of each rule into several rules each of which has only one input symbol. Actually, we construct a *multi bottom-up tree transducer* (mbot) [9] equivalent with a given xbot^{-e}. An mbot is a bot whose states might have ranks different from one. Intuitively, we decompose each rule ρ of the xbot^{-e} by adding a state for each intermediate position of the left-hand side tree l of ρ. The added states might have rank different from one to maintain two or more intermediate output trees until the obtained mbot reaches the state corresponding to the root of l (see Appendix B.2).

Example 3. Assume that an xbot^{-e} T has the transduction rule $\rho = a(b(q_1(x_1), q_2(x_2), q_3(x_3)), q_4(x_4)) \rightarrow q(c(x_1, x_2, x_4))$. Then, the mbot T_a obtained by decomposing T contains the rules $b(q_1(x_1), q_2(x_2), q_3(x_3)) \rightarrow q_1^\rho(x_1, x_2)$ and $a(q_1^\rho(x_1, x_2), q_4(x_4)) \rightarrow q(c(x_1, x_2, x_4))$ (See Fig. 3). Note that q_1^ρ maintains x_1 and x_2 but not x_3 because x_3 does not occur in the right-hand side of ρ.

Lemma 7. *Let $T_a = (Q_a, \Sigma, \Delta, Q_f, \delta_a)$ be the mbot obtained from an xbot^{-e} $T = (Q, \Sigma, \Delta, Q_f, \delta)$ by the above decomposition. Then, for every $q \in Q_a$ and $C \in \bar{\mathcal{C}}_\Sigma(\{x_*\})$, if $C[q(x_1, \ldots, x_k)] \Rightarrow_{T_a}^+ q(t_1, \ldots, t_k)$, then $(t_1, \ldots, t_k) \in \mathcal{T}_\Delta^{k,m}[X_k]$ for some $m \in [k]$.*

Henceforth, we denote $q(t_1, \ldots, t_k)$ by $q(\mathbf{t})$ where $\mathbf{t} = (t_1, \ldots, t_k)$.

Lemma 8. *Let $T_a = (Q_a, \Sigma, \Delta, Q_f, \delta_a)$ be the mbot obtained from an xbot^{-e} T by the above decomposition. Assume that T_a has n states and the maximum arity of states is k_m. T_a is not single-valued if and only if there is a tree t of depth less than $5 \cdot (n \cdot k_m)^2$ such that $|[\![T_a]\!](t)| > 1$.*

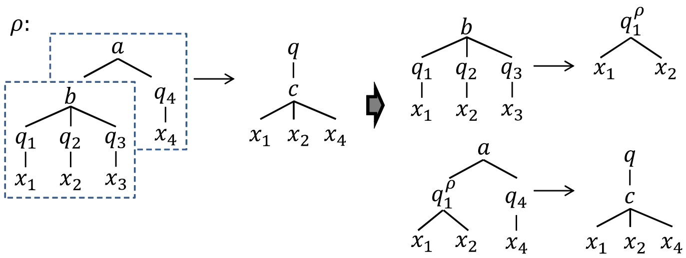

Fig. 3. An example of decomposing an xbot^{-e} to an mbot

Proof. The if part is trivial and so we prove the only if part. Assume that $t \in \mathcal{T}_\Sigma$ is a tree of minimal size such that there are two distinct derivations $t \Rightarrow^*_{T_a} q_{f1}(t_{o1})$ and $t \Rightarrow^*_{T_a} q_{f2}(t_{o2})$ where $q_{f1}, q_{f2} \in Q_f$, and $t_{o1} \neq t_{o2}$. For a contradiction, assume that the depth of t is greater than or equal to $5 \cdot (n \cdot k_m)^2$. Then, by Lemma 7, there are two states $q_1, q_2 \in Q_a$ with ranks n_1 and n_2 respectively, $C_j \in \bar{\mathcal{C}}_\Sigma(\{x_*\})$ $(0 \leq j \leq 4)$, $C_5 \in \mathcal{T}_\Sigma$, and for $i \in \{1,2\}$, $m_i \in [n_i]$, $t_0^i \in \mathcal{T}_\Delta[X_{n_i}]$, and $\mathbf{t}_j^i \in \mathcal{T}_\Delta^{n_i,m_i}[X_{n_i}]$ $(j \in [5])$ such that

$$\begin{aligned} t = C_0C_1C_2C_3C_4C_5 &\Rightarrow^*_{T_a} C_0C_{[1,4]}[q_i(\mathbf{t}_5^i)] \Rightarrow^*_{T_a} C_0C_{[1,3]}[q_i(\mathbf{t}_{[4,5]}^i)] \\ &\Rightarrow^*_{T_a} \cdots \\ &\Rightarrow^*_{T_a} C_0[q_i(\mathbf{t}_{[1,5]}^i)] \\ &\Rightarrow^*_{T_a} q_{fi}(t_0^i\mathbf{t}_{[1,5]}^i). \end{aligned}$$

By the minimality of t, we have $t_0^1\mathbf{t}_S^1 = t_0^2\mathbf{t}_S^2 \in [\![T_a]\!](C_0C_S)$ for every S such that $\{5\} \subseteq S \subset [1,5]$. From Lemma 6, $t_{o1} = t_0^1\mathbf{t}_{[1,5]}^1 = t_0^2\mathbf{t}_{[1,5]}^2 = t_{o2}$. This is a contradiction. □

From Lemma 8, single-valuedness of xbot^{-e}s is decidable.

Theorem 1. *It is decidable whether a given xbot^{-e} is single-valued. It is also decidable whether a given xbot$^{+\mathrm{g(B(R))}}$ is single-valued.*

Theorem 2. *Determinacy is decidable for (sl-xbots, s-bots).*

3.2 Undecidability for Other Classes

We show that determinacy is undecidable for (non-linear) s-bots as the determiner class. We prove the undecidability of determinacy for homomorphism tree transducers (homs) [11], which is a proper subclass of not only s-bots but also single-valued top-down tree transducers (s-top). Let id be the class of the identity transductions on $\mathcal{T}_\Sigma$ for any alphabet Σ.

Theorem 3. *Determinacy is undecidable for (homs, id).*

Proof. It can be shown by reduction from injectivity of homs, which is known to be undecidable [13]. Consider an arbitrary hom T. Then, it holds that $[\![T]\!]$ is injective if and only if T determines the identity transducer T_{id}. $\square$

Corollary 1. *Determinacy is undecidable for (s-bots, id) and (s-tops, id).*

Moreover, we have the undecidability result for deterministic monadic second-order logic defined tee transducers (dmsotts) [4, 5], which is a proper superclass of sl-xbots.

Theorem 4. *Determinacy is undecidable for (dmsotts, id_R) where id_R is the class of the identities whose domains are regular tree languages.*

Proof. It can be shown by reduction from ambiguity of context-free grammars. Consider an arbitrary context-free grammar G. Then, there is a dmsott T_G which transforms any derivation tree of each string $s \in L(G)$ to s. Thus, G is ambiguous if and only if T_G determines the identity transducer T_{id} such that $\mathrm{dom}(T_{id}) = \mathrm{dom}(T_G)$. $\square$

4 Subsumption

We show that subsumption is decidable for (sl-xbots, s-bots). As shown in Section 3, given an sl-xbot T_1 and an s-bot T_2, if T_1 determines T_2, we can construct a reduced s-xbot^{-e} T_3 such that $[\![T_3]\!] = [\![T_2]\!] \circ [\![T_1]\!]^{-1}$. So, in order to decide subsumption, we should decide whether there is a bot equivalent with T_3. The next lemma provides a necessary and sufficient condition for an s-xbot^{-e} to have an equivalent bot (see Appendix C).

Lemma 9. *Let $T = (Q, \Sigma, \Delta, Q_f, \delta)$ be a reduced s-xbot^{-e}. An s-bot equivalent with T can be constructed if and only if* (X) *for every rule $C_l[q_1(x_1), \ldots, q_k(x_k)] \to q(t_r) \in \delta$ and any three variables x_{i_1}, x_{i_2}, $x_{i_3} \in \mathrm{var}(t_r)$, if*

(X1) $\mathrm{rng}(T(q_{i_j}))$ *is infinite for all $j \in [3]$, and*
(X2) $\mathrm{nca}(p_1, p_2) \succ \mathrm{nca}(p_1, p_3)$ *where $\{p_j\} = pos_{x_{i_j}}(C_l)$ for $j \in [3]$, then*
(X3) *the minimal suffix $t_s \in \mathcal{T}_\Sigma(X_k)$ such that $t_r = t_p t_s$ for some $t_p \in \bar{\mathcal{T}}_{\Sigma \cup X_k - \{x_{i_1}, x_{i_2}\}}(\{x_*\})$ does not contain x_{i_3}.*

Proof Sketch. Assume (X) does not hold and we can construct an s-bot T' equivalent with a given s-xbot^{-e} T. Since (X) does not hold, there is a rule $C_l[q_1(x_1), \ldots, q_k(x_k)] \to q(t_r) \in \delta$ and x_{i_1}, x_{i_2}, $x_{i_3} \in \mathrm{var}(t_r)$ such that (X1) and (X2) hold but (X3) does not. Let $p_{12} = \mathrm{nca}(p_1, p_2)$ in (X2), and t_s be the minimal suffix of t_r in (X3). Since T' is an s-bot equivalent with T, T' must have rules of which left-hand sides 'cover' the subtree $C_l|_{p_{12}}$, which contains x_{i_1} and x_{i_2} and does not contain x_{i_3}. Also, since $C_l[x_*]_{p_{12}}$ does not contain x_{i_1} and x_{i_2}, some suffix t'_s of t_r in the right-hand side such that $t_r = t'_p t'_s$ for some $t'_p \in \bar{\mathcal{T}}_{\Sigma \cup X_k - \{x_{i_1}, x_{i_2}\}}(\{x_*\})$ should be generated by T' corresponding to $C_l|_{p_{12}}$. However, the minimal suffix t_s contains x_{i_3}, and thus so does t'_s. That is, t'_s including x_{i_3} should be generated from $C_l|_{p_{12}}$ without x_{i_3}, which leads a contradiction. Conversely, if (X) holds, we can divide each rule of T into non-extended rules, each of which has exactly one symbol in the left-hand side. $\square$

For any xbot T, it can be decided whether $\mathrm{rng}(T)$ is infinite. Thus, it is decidable whether there is an s-bot equivalent with a given s-xbot^{-e}.

Theorem 5. *Subsumption is decidable for (sl-xbots, s-bots).*

5 Conclusion

We have shown that determinacy and subsumption are decidable for single-valued linear extended bottom-up tree transducers as the determiner class and single-valued bottom-up tree transducers as the determinee class. As for more powerful classes, we have shown that determinacy is undecidable for single-valued top-down/bottom-up tree transducers (s-tops/bots) and deterministic MSO tree transducers (dmsotts) as the determiner class.

As future work, we will investigate whether subsumption for more powerful classes, such as s-tops/bots and dmsotts, is decidable or not. Though determinacy is undecidable for tops and dmsotts, decidability of subsumption for the classes is still open. We also consider whether, given two transducers T_1 and T_2 in the classes such that T_1 subsumes T_2, a transducer T_3 such that $[\![T_2]\!] = [\![T_3]\!] \circ [\![T_1]\!]$ can be effectively constructed or not.

References

1. Zheng, L., Chen, H.: Determinacy and rewriting of conjunctive queries over unary database schemas. In: Proceedings of the 2011 ACM Symposium on Applied Computing. (2011) 1039–1044
2. Afrati, F.N.: Determinacy and query rewriting for conjunctive queries and views. Theoretical Computer Science **412**(11) (2011) 1005–1021
3. Fan, W., Geerts, F., Zheng, L.: View determinacy for preserving selected information in data transformations. Information Systems **37**(1) (2012) 1–12
4. Courcelle, B.: Monadic second-order definable graph transductions: A survey. Theoretical Computer Science **126**(1) (1994) 53–75
5. Engelfriet, J., Maneth, S.: The equivalence problem for deterministic MSO tree transducers is decidable. Information Processing Letters **100**(5) (2006) 206–212
6. Fan, W., Bohannon, P.: Information preserving XML schema embedding. ACM Transactions on Database Systems **33**(1) (2008) 4:1–4:44
7. Comon, H., Dauchet, M., Gilleron, R., Jacquemard, F., Lugiez, D., Löding, C., Tison, S., Tommasi, M.: Tree automata techniques and applications. http://www.grappa.univ-lille3.fr/tata (2007)
8. Lemay, A., Maneth, S., Niehren, J.: A learning algorithm for top-down XML transformations. In: Proceedings of the 29th ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems. (2010) 285–296
9. Engelfriet, J., Lilin, E., Maletti, A.: Extended multi bottom-up tree transducers. Acta Informatica **46** (2009) 561–590
10. Seidl, H.: Single-valuedness of tree transducers is decidable in polynomial time. Theoretical Computer Science **106**(1) (1992) 135–181
11. Engelfriet, J.: Bottom-up and top-down tree transformations - a comparison. Mathematical Systems Theory **9**(3) (1975) 198–231
12. Seidl, H.: Equivalence of finite-valued tree transducers is decidable. Theory of Computing Systems **27** (1994) 285–346
13. Fulop, Z., Gyenizse, P.: On injectivity of deterministic top-down tree transducers. Information Processing Letters **48**(4) (1993) 183–188

A Construction of a Reduced xbot

It is known that for every bot T an equivalent reduced bot can be constructed from T in linear time [10]. For xbots and xbot$^{+\mathrm{g}}$s defined in Section 3, equivalent reduced ones can also be constructed in the same way. Here, we recall the construction: Given xbot$^{+\mathrm{g}}$ $T = (Q, \Sigma, \Delta, Q_f, \delta)$, to satisfy condition 2 for the reducedness in Section 2.2, first construct $T' = (Q \times \{0,1\}, \Sigma, \Delta, Q_f \times \{1\}, \delta')$ such that for each rule $C_l[q_1(x_1), \ldots, q_k(x_k)] \to q(\tilde{t}_r) \in \delta$,

- $C_l[\hat{q}_1(x_1), \ldots, \hat{q}_k(x_k)] \to (q,1)(\tilde{t}_r) \in \delta'$ where $\hat{q}_i = (q_i, 1)$ if x_i appears in $\tilde{t}_r$, and $\hat{q}_i = (q_i, 0)$ otherwise; and
- $C_l[(q_1,0)(x_1), \ldots, (q_k,0)(x_k)] \to (q,0)(\perp) \in \delta'$.

Next, for condition 3 for the reducedness, add a new state q^f and replace the set of final states with the singleton $\{q^f\}$. Then for any rule $\rho \in \delta'$ whose right-hand side has a state $\hat{q} \in Q_f \times \{1\}$, add the new rule obtained from ρ by replacing $\hat{q}$ with q^f in the right-hand side. Now, the reduced xbot$^{+\mathrm{g}}$ T'' equivalent with T is obtained by removing useless states for condition 1 of the reducedness. Removing useless states can be done in the same way as for tree automata [7]. Notice that $U(T'') = Q'' \cap (Q \times \{0\})$ where Q'' is the set of states of T''.

B Proofs of Lemmas for Theorem 1

B.1 Proof of Lemma 6

We first recall the notations. Let $\mathcal{T}_\Sigma[X_n] = \bar{\mathcal{T}}_\Sigma(X_n) \cup \mathcal{T}_\Sigma$, that is, every $t \in \mathcal{T}_\Sigma[X_n]$ has all the variables in X_n or has no variable. For $t, s \in \mathcal{T}_\Sigma[X_n]$, ts is the tree obtained from t by replacing each variable with s. Note that $ts = t$ if t has no variable. For $m \in [n]$, let $\mathcal{T}_\Sigma^{n,m}[X_n] = \mathcal{T}_\Sigma^{m-1} \times \mathcal{T}_\Sigma[X_n] \times \mathcal{T}_\Sigma^{n-m}$. For $\mathbf{t} \in \mathcal{T}_\Sigma^{n,m}[X_n]$, we denote by $t^{(i)}$ the ith element of $\mathbf{t}$, i.e., $\mathbf{t} = (t^{(1)}, \ldots, t^{(n)})$. For $s \in \mathcal{T}_\Sigma[X_n]$ and $\mathbf{t} \in \mathcal{T}_\Sigma^{n,m}[X_n]$, $s\mathbf{t}$ is the tree obtained from s by replacing x_i with $t^{(i)}$ for all $i \in [n]$. Let $\mathbf{t}s = (t^{(1)}s, \ldots, t^{(n)}s)$, and $\mathbf{tu} = (t^{(1)}\mathbf{u}, \ldots, t^{(n)}\mathbf{u})$ for $\mathbf{u} \in \mathcal{T}_\Sigma^{n,m}[X_n]$. Notice that since $\mathbf{t} \in \mathcal{T}_\Sigma^{n,m}[X_n]$, so are $\mathbf{t}s$ and $\mathbf{tu}$. Lastly, the above substitution operation is associative. That is, for $s \in \mathcal{T}_\Sigma[X_n]$ and $\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3 \in \mathcal{T}_\Sigma^{n,m}[X_n]$, $(s\mathbf{t}_1)\mathbf{t}_2 = s(\mathbf{t}_1\mathbf{t}_2)$ and $(\mathbf{t}_1\mathbf{t}_2)\mathbf{t}_3 = \mathbf{t}_1(\mathbf{t}_2\mathbf{t}_3)$ hold. For $\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3, \mathbf{t}_4, \mathbf{t}_5 \in \mathcal{T}_\Sigma^{n,m}[X_n]$ and $S = \{i_1, \ldots, i_{|S|}\} \subseteq [1,5]$, let $\mathbf{t}_S = \mathbf{t}_{i_1} \cdots \mathbf{t}_{i_{|S|}}$ where $i_j < i_{j+1}$ for $j \in [|S| - 1]$.

We use some propositions to prove Lemma 6. The following is the same as the top cancellation in [12] except that we use tuples of trees in $\mathcal{T}_\Sigma^{n,m}[X_n]$.

Proposition 2 (Top Cancellation). *Let $s \in \mathcal{T}_\Sigma[X_n]$ and $\mathbf{t}_1, \mathbf{t}_2 \in \mathcal{T}_\Sigma^{n,m}[X_n]$. If $s\mathbf{t}_1 = s\mathbf{t}_2$, then $s \in \mathcal{T}_\Sigma$ or $\mathbf{t}_1 = \mathbf{t}_2$.*

Proof. Assume that $s\mathbf{t}_1 = s\mathbf{t}_2$, $s \notin \mathcal{T}_\Sigma$ and $\mathbf{t}_1 \neq \mathbf{t}_2$. Then, s contains all the variables in X_n, and $t_1^{(i)} \neq t_2^{(i)}$ for some $i \in [n]$. Since distinct trees are substituted to x_i, $s\mathbf{t}_1 \neq s\mathbf{t}_2$. This is a contradiction. □

The decidability of single-valuedness (or more generally, k-valuedness) of bots was proved in [12] by using Engelfriet's Property T1(i), which was proved by Engelfriet's Property T2(i).

Proposition 3 (Engelfriet's Property T1(i) [12]). *Assume* $t_i, t'_i \in \mathcal{T}_\Sigma(\{x_*\})$, $i = 0, 1, 2, 3, 4$. *Then,*

$$t_0 \cdots t_{i-1} t_j \cdots t_4 = t'_0 \cdots t'_{i-1} t'_j \cdots t'_4 \quad \textit{for all } 0 < i < j \leq 4$$

implies $t_0 t_1 t_2 t_3 t_4 = t'_0 t'_1 t'_2 t'_3 t'_4$.

Proposition 4 (Engelfriet's Property T2(i) [12]). *Assume* $s_i, u_i, v_i, y_i, z_i \in \mathcal{T}_\Sigma(\{x_*\})$ $(i = 1, 2)$, s_1 *or* s_2 *contains* x_*, $y_1 \neq z_1$, *and* $y_2 \neq z_2$. *Then,*

$$\begin{array}{l} s_1 y_1 = s_2 y_2 \\ s_1 z_1 = s_2 z_2 \\ s_1 v_1 = s_2 v_2 \\ u_1 y_1 = u_2 y_2 \\ u_1 z_1 = u_2 z_2 \end{array} \quad \textit{implies} \quad u_1 v_1 = u_2 v_2.$$

Now, we prove Lemma 6, a variant of Engelfriet's Property T1(i). For $\mathbf{t} = (t^{(1)}, \ldots, t^{(n)}) \in \mathcal{T}_\Sigma^{n,m}[X_n]$, $\mathbf{t}^c$ denotes the n-tuple of trees obtained from $\mathbf{t}$ by replacing the mth element with x_*, that is, $\mathbf{t}^c = (t^{(1)}, \ldots, t^{(m-1)}, x_*, t^{(m+1)}, \ldots, t^{(n)})$. Hence, we have $\mathbf{t} = \mathbf{t}^c t^{(m)}$.

Lemma 6. *Let* n, n' *be arbitrary positive integers, and* $m \in [n], m' \in [n']$. *Suppose that* $t_0 \in \mathcal{T}_\Sigma[X_n]$, $\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3, \mathbf{t}_4, \mathbf{t}_5 \in \mathcal{T}_\Sigma^{n,m}[X_n]$, $t'_0 \in \mathcal{T}_\Sigma[X_{n'}]$, $\mathbf{t}'_1, \mathbf{t}'_2, \mathbf{t}'_3, \mathbf{t}'_4, \mathbf{t}'_5 \in \mathcal{T}_\Sigma^{n',m'}[X_{n'}]$. *Then, if* $t_0 \mathbf{t}_S = t'_0 \mathbf{t}'_S$ *for every* S *such that* $\{5\} \subseteq S \subset [1,5]$, *then* $t_0 \mathbf{t}_{[1,5]} = t'_0 \mathbf{t}'_{[1,5]}$.

Proof. Assume that $t_0 \mathbf{t}_S = t'_0 \mathbf{t}'_S$ for every S such that $\{5\} \subseteq S \subset [1,5]$. Let

$$\begin{array}{ll} s_1 = t_0 \mathbf{t}_2^c, & s_2 = t'_0 \mathbf{t}'^c_2, \\ u_1 = t_0 \mathbf{t}_1 \mathbf{t}_2^c, & u_2 = t'_0 \mathbf{t}'_1 \mathbf{t}'^c_2, \\ y_1 = t_2^{(m)} \mathbf{t}_5, & y_2 = t'^{(m')}_2 \mathbf{t}'_5, \\ z_1 = t_2^{(m)} \mathbf{t}_{[4,5]}, & z_2 = t'^{(m')}_2 \mathbf{t}'_{[4,5]}, \\ v_1 = t_2^{(m)} \mathbf{t}_{[3,5]}, & v_2 = t'^{(m')}_2 \mathbf{t}'_{[3,5]}. \end{array}$$

By the assumption, we have

$$\begin{aligned} s_1 y_1 &= (t_0 \mathbf{t}_2^c)(t_2^{(m)} \mathbf{t}_5) = t_0 \mathbf{t}_2 \mathbf{t}_5 = t'_0 \mathbf{t}'_2 \mathbf{t}'_5 = (t'_0 \mathbf{t}'^c_2)(t'^{(m')}_2 \mathbf{t}'_5) = s_2 y_2, \\ s_1 z_1 &= (t_0 \mathbf{t}_2^c)(t_2^{(m)} \mathbf{t}_{[4,5]}) = t_0 \mathbf{t}_2 \mathbf{t}_{[4,5]} = t'_0 \mathbf{t}'_2 \mathbf{t}'_{[4,5]} = (t'_0 \mathbf{t}'^c_2)(t'^{(m')}_2 \mathbf{t}'_{[4,5]}) = s_2 z_2, \\ s_1 v_1 &= (t_0 \mathbf{t}_2^c)(t_2^{(m)} \mathbf{t}_{[3,5]}) = t_0 \mathbf{t}_{[2,5]} = t'_0 \mathbf{t}'_{[2,5]} = (t'_0 \mathbf{t}'^c_2)(t'^{(m')}_2 \mathbf{t}'_{[3,5]}) = s_2 v_2, \\ u_1 y_1 &= (t_0 \mathbf{t}_1 \mathbf{t}_2^c)(t_2^{(m)} \mathbf{t}_5) = t_0 \mathbf{t}_1 \mathbf{t}_2 \mathbf{t}_5 = t'_0 \mathbf{t}'_1 \mathbf{t}'_2 \mathbf{t}'_5 = (t'_0 \mathbf{t}'_1 \mathbf{t}'^c_2)(t'^{(m')}_2 \mathbf{t}'_5) = u_2 y_2, \\ u_1 z_1 &= (t_0 \mathbf{t}_1 \mathbf{t}_2^c)(t_2^{(m)} \mathbf{t}_{[4,5]}) = t_0 \mathbf{t}_{[1,2]} \mathbf{t}_{[4,5]} = t'_0 \mathbf{t}'_{[1,2]} \mathbf{t}'_{[4,5]} = (t'_0 \mathbf{t}'_1 \mathbf{t}'^c_2)(t'^{(m')}_2 \mathbf{t}'_{[4,5]}) \\ &= u_2 z_2. \\ u_1 v_1 &= (t_0 \mathbf{t}_1 \mathbf{t}_2^c)(t_2^{(m)} \mathbf{t}_{[3,5]}) = t_0 \mathbf{t}_{[1,5]} \\ u_2 v_2 &= (t'_0 \mathbf{t}'_1 \mathbf{t}'^c_2)(t'^{(m')}_2 \mathbf{t}'_{[3,5]}) = t'_0 \mathbf{t}'_{[1,5]} \end{aligned}$$

There are four cases.
Case (1) Both of s_1 and s_2 contain no variable.
Then, t_0 and t'_0 do not contain x_m and $x_{m'}$, respectively. Moreover, by definition of $\mathcal{T}_\Sigma[X]$, t_0 and t'_0 do not contain any variable. Thus, $t_0\mathbf{t}_{[1,5]} = t_0 = t_0\mathbf{t}_5$ and $t'_0\mathbf{t}'_{[1,5]} = t'_0 = t'_0\mathbf{t}'_5$. Since $t_0\mathbf{t}_5 = t'_0\mathbf{t}'_5$, we have $t_0\mathbf{t}_{[1,5]} = t'_0\mathbf{t}_{[1,5]}$.
Case (2) s_1 or s_2 contains variable x_*, $y_1 \neq z_1$ and $y_2 \neq z_2$.
Proposition 4 implies $t_0\mathbf{t}_{[1,5]} = u_1v_1 = u_2v_2 = t'_0\mathbf{t}_{[1,5]}$.
Case (3) s_1 or s_2 contains variable x_* and $y_1 = z_1$.
Because $y_1 = z_1$, by top cancellation (Proposition 2), $\mathbf{t}_5 = \mathbf{t}_{[4,5]}$ or $t_2^{(m)}$ has no variable. If $\mathbf{t}_5 = \mathbf{t}_{[4,5]}$ then $t_0\mathbf{t}_{[1,3]}\mathbf{t}_5 = t_0\mathbf{t}_{[1,5]}$. If $t_2^{(m)}$ has no variable then we also have $t_0\mathbf{t}_{[1,3]}\mathbf{t}_5 = t_0\mathbf{t}_{[1,2]} = t_0\mathbf{t}_{[1,5]}$. Moreover, we have $s_1y_1 = s_1z_1$ and thus $s_2y_2 = s_2z_2$. By top cancellation, $y_2 = z_2$ or s_2 has no variable. Assume that $y_2 = z_2$. By the same argument as the above, we have $t'_0\mathbf{t}'_{[1,3]}\mathbf{t}'_5 = t'_0\mathbf{t}'_{[1,5]}$. On the other hand, assume that s_2 has no variables. Then, t'_0 has no variable and thus we also have $t'_0\mathbf{t}'_{[1,3]}\mathbf{t}'_5 = t'_0 = t'_0\mathbf{t}'_{[1,5]}$. Therefore, $t_0\mathbf{t}_{[1,5]} = t'_0\mathbf{t}'_{[1,5]}$ because $t_0\mathbf{t}_{[1,3]}\mathbf{t}_5 = t'_0\mathbf{t}'_{[1,3]}\mathbf{t}'_5$.
Case (4) s_1 or s_2 contains variable x_* and $y_2 = z_2$.
This is analogous to Case (3). □

B.2 Proof of Lemma 7

To decompose the left-hand side of each rule of a given xbot^{-e} into several rules each of which has only one input symbol, we construct a *multi bottom-up tree transducer* equivalent with the xbot^{-e}. A multi bottom-up tree transducer [9] is a bottom-up tree transducer whose states might have ranks different from one. A multi bottom-up tree transducer (mbot) is a 5-tuple $(Q, \Sigma, \Delta, Q_f, \delta)$ where $Q_f \subseteq Q^{(1)}$ and δ is a set of rules of the form $l \to r$ where $l \in \Sigma(Q(X))$ is linear in X and $r \in Q(\mathcal{T}_\Delta(\mathrm{var}(l)))$. For $q \in Q^{(k)}$ and a finite subset $X_q = \{x_{i_1}, \ldots, x_{i_k}\}$ of $X - \{x_*\}$, let $q(X_q)$ denote $q(x_{i_1}, \ldots, x_{i_k})$ where $i_j < i_{j+1}$ for every $j \in [k-1]$. The move relation $\Rightarrow_T$ of a mbot $T = (Q, \Sigma, \Delta, Q_f, \delta)$ is defined as follows: $t \Rightarrow_T t'$ if there is a rule $l \to r \in \delta$, $p \in pos(t)$, and a substitution $\theta : X \to \mathcal{T}_{\Sigma\cup\Delta}$ such that $t|_p = l\theta$ and $t' = t[r\theta]_p$.

The decomposition is as follows: Given an xbot^{-e} $T = (Q, \Sigma, \Delta, Q_f, \delta)$, construct an mbot $T_\mathrm{m} = (Q \cup Q_\mathrm{m}, \Sigma, \Delta, Q_f, \delta_\mathrm{m})$ where

- $Q_\mathrm{m} = \{q_p^\rho \mid \rho = (C_l[q_1(x_1), \ldots, q_k(x_k)] \to q(t_r)) \in \delta,\ p \in pos_\Sigma(C_l) - \{\epsilon\}\}$,
- δ_m is the smallest set such that for each $\rho = (C_l[q_1(x_1), \ldots, q_k(x_k)] \to q(t_r)) \in \delta$,
  - $\sigma_\epsilon(q_1^\rho(V_1), \ldots, q_{k_\epsilon}^\rho(V_{k_\epsilon})) \to q(t_r) \in \delta_\mathrm{m}$ where $\sigma_\epsilon = \lambda_{C_l}(\epsilon)$, $k_\epsilon = \mathsf{rk}(\sigma_\epsilon)$, and $V_i = \mathrm{var}(C_l|_i) \cap \mathrm{var}(t_r)$ for each $i \in [k_\epsilon]$,
  - for each position $p \in pos_\Sigma(C_l) - \{\epsilon\}$, let $V_p = \mathrm{var}(C_l|_p)$ and $\theta_n : V_p \to X_{|V_p|}$ such that $C_l|_p\theta_n$ is normalized, then the rule obtained by normalizing the both sides of $\sigma_p(q_{p1}^\rho(V_{p1}), \ldots, q_{pk_p}^\rho(V_{pk_p})) \to q_p^\rho(V_p)$ by θ_n belongs to δ_m, where $\sigma_p = \lambda_{C_l}(p)$, $k_p = \mathsf{rk}(\sigma_p)$, and $V_{pi} = \mathrm{var}(C_l|_{pi}) \cap \mathrm{var}(t_r)$ for each $i \in [k_p]$,

  where $q_p^\rho = q_i$ if $p \in pos_{x_i}(C_l)$ for some $x_i \in X_k$.

Lemma 7. *Let $T_\mathrm{m} = (Q \cup Q_\mathrm{m}, \Sigma, \Delta, Q_f, \delta_\mathrm{m})$ be the mbot obtained from an xbot^{-e} $T = (Q, \Sigma, \Delta, Q_f, \delta)$ by the above decomposition. Then, for every $q \in Q \cup Q_\mathrm{m}$ and $C \in \bar{\mathcal{C}}_\Sigma(\{x_*\})$, if $C[q(x_1, \ldots, x_k)] \Rightarrow_{T_\mathrm{m}}^+ q(t_1, \ldots, t_k)$, then $(t_1, \ldots, t_k) \in \mathcal{T}_\Delta^{k,m}[X_k]$ for some $m \in [k]$.*

Proof. Assume that $q \in Q$. Since every state in Q has rank one, for any $C \in \bar{\mathcal{C}}_\Sigma(\{x_*\})$, if $C[q(x_*)] \Rightarrow^*_{T_\mathrm{m}} q(t)$, then $t \in \mathcal{T}_\Delta(\{x_*\}) = \mathcal{T}^{1,1}_\Delta$.

Assume that $q \in Q_\mathrm{m}$ and C is an arbitrary tree in $\bar{\mathcal{C}}_\Sigma(\{x_*\})$ such that $C[q(x_1,\ldots,x_k)] \Rightarrow^+_{T_\mathrm{m}} q(t_1,\ldots,t_k)$. From the decomposition procedure, there exists a rule $\rho = (C_l[q_1(x_1),\ldots,q_k(x_k)] \to q_\rho(t_r)) \in \delta$ of T such that $q = q^\rho_p$ for some position p in $pos_\Sigma(C_l) - \{\epsilon\}$. Thus, $C[q(x_1,\ldots,x_k)] \Rightarrow^*_\rho C'[q_\rho(t'')] \Rightarrow^*_{T_\mathrm{m}} q(t_1,\ldots,t_k)$ for some $C' \in \bar{\mathcal{C}}_\Sigma(\{x_*\})$ and $t'' \in \bar{\mathcal{T}}_\Delta(X_k)$, where $\Rightarrow^*_\rho$ means zero or more applications of only the rules obtained by decomposing ρ. Note that t'' must contain all the variables $x_1,\ldots,x_k$. In the above derivation $C'[q_\rho(t'')] \Rightarrow^*_{T_\mathrm{m}} q(t_1,\ldots,t_k)$, t'' is either abandoned or contained as a subtree in t_m for some $m \in [k]$. Thus, $(t_1,\ldots,t_k) \in \mathcal{T}^{k,m}_\Delta$ for some $m \in [k]$. □

C Proof of Lemma 9

To prove Lemma 9, we use the following property, which is a slight extension of Proposition 1.5 in [10].

Proposition 5. *Assume $t \in \mathcal{T}_\Sigma(X_k)$ contains at least one occurrence of each variable in $V \subseteq X_k$ where $|V| \geq 2$. Then, $t = t_p t_s$ for some $t_p \in \bar{\mathcal{T}}_{\Sigma\cup X_k - V}(\{x_*\})$ and $t_s \in \mathcal{T}_\Sigma(X_k)$ satisfying the following conditions:*

1. *If $t = u_p u_s$ for some $u_p \in \bar{\mathcal{T}}_{\Sigma\cup X_k-V}(\{x_*\})$ and $u_s \in \mathcal{T}_\Sigma(X_k)$ then $t_p = u_p r$ and $u_s = r t_s$ for some $r \in \bar{\mathcal{T}}_{\Sigma\cup X_k-V}(\{x_*\})$. That is, t_p and t_s are the unique maximal prefix and minimal suffix of t, respectively.*
2. *Assume $t_i \in \bar{\mathcal{T}}_\Sigma(\{x_i\})$ for each $x_i \in X_k$. Then, $t_p[t_1,\ldots,t_k] \in \bar{\mathcal{T}}_{\Sigma\cup X_k-V}(\{x_*\})$ and $t_s[t_1,\ldots,t_k] \in \mathcal{T}_\Sigma(X_k)$ are the maximal prefix and the minimal suffix of $t[t_1,\ldots,t_k]$, respectively.*

Lemma 9. *Let $T = (Q,\Sigma,\Delta,Q_f,\delta)$ be a reduced s-xbot^{-e}. An s-bot equivalent with T can be constructed if and only if* (X) *for every rule $C_l[q_1(x_1),\ldots,q_k(x_k)] \to q(t_r) \in \delta$ and any three variables $x_{i_1}, x_{i_2}, x_{i_3} \in \mathrm{var}(t_r)$ if*

(X1) $\mathrm{rng}(T(q_{i_j}))$ *is infinite for all $j \in [3]$, and*
(X2) $\mathrm{nca}(p_1,p_2) \succ \mathrm{nca}(p_1,p_3)$ *where $\{p_j\} = pos_{x_{i_j}}(C_l)$ for $j \in [3]$, then*
(X3) *the minimal suffix $t_s \in \mathcal{T}_\Sigma(X_k)$ such that $t_r = t_p t_s$ for some $t_p \in \bar{\mathcal{T}}_{\Sigma\cup X_k-\{x_{i_1},x_{i_2}\}}(\{x_*\})$ does not contain x_{i_3}.*

Proof. For the only if part, assume that (X) does not hold, i.e., there is a rule $\rho = (C_l[q_1(x_1),\ldots,q_k(x_k)] \to q(t_r)) \in \delta$ that satisfies (X1) and (X2) but does not satisfy (X3). That is, the rule ρ has three variables $x_{i_1}, x_{i_2}, x_{i_3} \in \mathrm{var}(t_r)$ such that

- $\mathrm{rng}(T(q_{i_j}))$ is infinite for all $j \in [3]$,
- $\mathrm{nca}(p_1,p_2) \succ \mathrm{nca}(p_1,p_3)$ where $\{p_j\} = pos_{x_{i_j}}(C_l)$ for $j \in [3]$, and
- the minimal suffix $t_{s12} \in \mathcal{T}_\Sigma(X_k)$ such that $t_r = t_{p12} t_{s12}$ for some $t_{p12} \in \bar{\mathcal{T}}_{\Sigma\cup X_k-\{x_{i_1},x_{i_2}\}}(\{x_*\})$ contains x_{i_3}.

For a contradiction, assume that there is a reduced s-bot $T^s = (Q^s, \Sigma, \Delta, Q_f^s, \delta^s)$ equivalent with T. Since T is reduced and the three variables $x_{i_1}, x_{i_2}, x_{i_3}$ belong to $\mathrm{var}(t_r)$ of the rule ρ, we have that $q, q_{i_1}, q_{i_2}, q_{i_3} \in Q - U(T)$. Let $C_l' = C_l\theta \in \bar{\mathcal{C}}_\Sigma(X_3)$ such that $\theta(x_{i_j}) = x_j$ for $j \in [3]$ and $\theta(x_i)$ is a tree in $\mathrm{dom}(T(q_i))$ for $x_i \in X_k - \{x_{i_1}, x_{i_2}, x_{i_3}\}$. Then,

$$CC_l'[q_{i_1}(x_1), q_{i_2}(x_2), q_{i_3}(x_3)] \Rightarrow_T^* C[q(t_r')] \Rightarrow_T^* q_f(t't_r')$$

for some $C \in \bar{\mathcal{C}}_\Sigma(\{x_*\})$, $t' \in \bar{\mathcal{T}}_\Delta(\{x_*\})$, $q_f \in Q_f$ and $t_r' = t_r\theta' \in \bar{\mathcal{T}}_\Delta(X_3)$ such that $\theta'(x_{i_j}) = x_j$ for $j \in [3]$ and $\theta'(x_i) = T(q_i)(\theta(x_i))$ for $x_i \in X_k - \{x_{i_1}, x_{i_2}, x_{i_3}\}$. Since T is single-valued, for each $j \in [3]$, there is an injective mapping $\xi_j : \mathrm{rng}(T(q_{i_j})) \to \mathrm{dom}(T(q_{i_j}))$ such that $T(q_{i_j})(\xi_j(t')) = t'$ for any $t' \in \mathrm{rng}(T(q_{i_j}))$. Since T^s is equivalent with T, for every $t_1' \in \mathrm{rng}(T(q_{i_1}))$, $t_2' \in \mathrm{rng}(T(q_{i_2}))$, and $t_3' \in \mathrm{rng}(T(q_{i_3}))$, we have that $CC_l'[\xi_1(t_1'), \xi_2(t_2'), \xi_3(t_3')] \Rightarrow_{T^s}^* q_f^s(t't_r'[t_1', t_2', t_3'])$ for some $q_f^s \in Q_s$. By the fact that $\mathrm{rng}(T(q_{i_j}))$ ($j \in [3]$) is infinite but Q_s is finite, and $\mathrm{nca}(p_1, p_2) \succ \mathrm{nca}(p_1, p_3)$, we can say that there are infinite sets $L_1 \subseteq \mathrm{rng}(T(q_{i_1}))$, $L_2 \subseteq \mathrm{rng}(T(q_{i_2}))$, and $L_3 \subseteq \mathrm{rng}(T(q_{i_3}))$, states $q_1^s, q_2^s, q_3^s, q_{12}^s, q^s, q_f^s \in Q_s$, and trees $u_{12}^s \in \mathcal{T}_\Delta(\{x_1, x_2\}), u_3^s \in \mathcal{T}_\Delta(\{x_*, x_3\}), u_o^s \in \mathcal{T}_\Delta(\{x_*\})$ such that for all $t_1' \in L_1$, $t_2' \in L_2$, and $t_3' \in L_3$,

$$\begin{aligned}
&CC_l'[\xi_1(t_1'), \xi_2(t_2'), \xi_3(t_3')] \\
&\Rightarrow_{T^s}^* CC_l'[q_1^s(t_1^s), q_2^s(t_2^s), q_3^s(t_3^s)] \\
&\Rightarrow_{T^s}^* C(C_l'[q_{12}^s(u_{12}^s[t_1^s, t_2^s])]_{\mathrm{nca}(p_1,p_2)})[x_3 \leftarrow q_3^s(t_3^s)] \\
&\Rightarrow_{T^s}^* C[q^s((u_3^s u_{12}^s)[t_1^s, t_2^s, t_3^s])] \\
&\Rightarrow_{T^s}^* q_f^s(u_o^s(u_3^s u_{12}^s)[t_1^s, t_2^s, t_3^s])
\end{aligned}$$

and $u_o^s(u_3^s u_{12}^s)[t_1^s, t_2^s, t_3^s] = t't_r'[t_1', t_2', t_3']$ where $\xi_j(t_j') \Rightarrow_{T^s}^* q_j^s(t_j^s)$ ($j \in [3]$) (See Fig. 4). Now, fix $t_2' \in L_2$ and $t_3' \in L_3$. Consider $D_1 = t't_r'[x_1, t_2', t_3'] \in \mathcal{T}_\Delta(\{x_1\})$ and $D_1^s = u_o^s(u_3^s u_{12}^s)[x_1, t_2^s, t_3^s] \in \mathcal{T}_\Delta(\{x_1\})$. Then, L_1 has two distinct trees t_1' and t_1'', and $D_1[t_1'] \neq D_1[t_1'']$ because $q_{i_1} \notin U(T)$ and thus D_1 must have x_1. Since T is single-valued and T^s is equivalent with T, we have $D_1[t_1'] = D_1^s[t_1'^s]$ and $D_1[t_1''] = D_1^s[t_1''^s]$ where $\xi_1(t_1') \Rightarrow_{T^s}^* q_1^s(t_1'^s)$ and $\xi_1(t_1'') \Rightarrow_{T^s}^* q_1^s(t_1''^s)$. (Note that $t_1'^s$ and $t_1''^s$ are uniquely determined respectively. If not, T^s had two output trees for $CC_l'[\xi_1(t_1'), \xi_2(t_2'), \xi_3(t_3')]$ and thus it was not single-valued. Since T and T_s are equivalent, however, T^s is also single-valued.) Therefore, D_1^s also must have x_1, and there is a tree $r_1' \in \bar{\mathcal{T}}_\Delta(\{x_1\})$ such that $D_1[x_1 \leftarrow r_1'] = D_1^s$ or $D_1 = D_1^s[x_1 \leftarrow r_1']$. Similarly, for both $j = 2, 3$, there is a tree $r_j' \in \bar{\mathcal{T}}_\Delta(\{x_j\})$ such that $D_j[x_j \leftarrow r_j'] = D_j^s$ or $D_j = D_j^s[x_j \leftarrow r_j']$. Thus, we have $t't_r'[r_1, r_2, r_3] = u_0^s(u_3^s u_{12}^s)[r_1'', r_2'', r_3'']$ where r_j and r_j'' are r_j' or x_j, and $r_j = x_j$ if and only if $r_j'' = r_j'$. By proposition 5 and $t't_r'[r_1, r_2, r_3] = t'((t_{p12}t_{s12})\theta')[r_1, r_2, r_3]$, we have $t_{s12}\theta'[r_1, r_2, r_3]$ is the minimal suffix of $t't_r'[r_1, r_2, r_3]$ containing x_3. Thus, for any decomposition $t't_r'[r_1, r_2, r_3] = t_p t_s$, t_s must contain x_3. However, $u_0^s(u_3^s u_{12}^s)[r_1'', r_2'', r_3''] = (u_0^s u_3^s[x_3 \leftarrow r_3''])u_{12}^s[r_1'', r_2'']$, and $u_{12}^s[r_1'', r_2'']$ does not contain x_3. This is a contradiction.

For the if part, assume that condition (X) holds. Let $Q_{\mathrm{fin}} = \{q \in Q \mid \mathrm{rng}(T(q)) \text{ is finite}\}$ and $Q_{\mathrm{fin}}^{nf} = Q_{\mathrm{fin}} - Q_f$. Note that $U(T)$ is included in Q_{fin} because

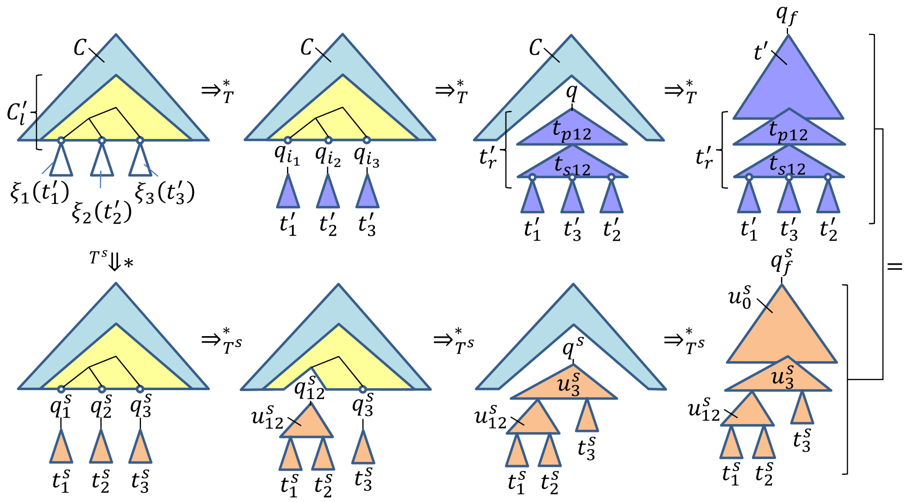

Fig. 4. Transduction by T and T^s

$\mathrm{rng}(T(q)) = \{\bot\}$ for $q \in U(T)$. Also, let $\tilde{Q}^{nf}_{\mathrm{fin}} = \{(q,t) \mid q \in Q^{nf}_{\mathrm{fin}}, t \in \mathrm{rng}(T(q))\}$. We first construct a reduced s-xbot^{-e} T_f, equivalent with T, such that $\mathrm{rng}(T_f(q))$ is infinite or the singleton $\{\bot\}$ for any non-final state $q \notin Q_f$ of T_f. More specifically, we construct $T_f = (\tilde{Q}^{nf}_{\mathrm{fin}} \cup (Q - Q^{nf}_{\mathrm{fin}}), \Sigma, \Delta, Q_f, \delta')$ such that δ' is the smallest set satisfying the following condition: Let $C_l[q_1(x_1), \ldots, q_k(x_k)] \to q(t_r) \in \delta$ be an arbitrary rule.

- If $q \in Q^{nf}_{\mathrm{fin}}$, then $C_l[(q_1,t_1)(x_1), \ldots, (q_k,t_k)(x_k)] \to (q, t_r[t_1, \ldots, t_k])(\bot) \in \delta'$ for all $t_i \in \mathrm{rng}(T(q_i))$ $(i \in [k])$.
- If $q \in Q - Q^{nf}_{\mathrm{fin}}$, let θ_l and θ_r be arbitrary substitutions such that for each $i \in [k]$, if $\mathrm{rng}(T(q_i))$ is infinite, $\theta_l(x_i) = q_i(x_i)$ and $\theta_r(x_i) = x_i$; otherwise, $\theta_l(x_i) = (q_i, t_i)(x_i)$ and $\theta_r(x_i) = t_i$ for any $t_i \in \mathrm{rng}(T(q_i))$. Then, $C_l\theta_l \to q(t_r\theta_r) \in \delta'$.

Note that T_f is reduced and $U(T_f) = \tilde{Q}^{nf}_{\mathrm{fin}}$. Thus, for every rule $C_l[q_1(x_1), \ldots, q_k(x_k)] \to q(t_r) \in \delta'$, $x_i \in \mathrm{var}(t_r)$ if and only if $\mathrm{rng}(T_f(q_i))$ is infinite for $i \in [k]$, that is, condition (X1) is satisfied. Namely,

> For every $C_l[q_1(x_1), \ldots, q_k(x_k)] \to q(t_r) \in \delta'$ and any three variables $x_{i_1}, x_{i_2}, x_{i_3} \in \mathrm{var}(t_r)$, if $\mathrm{nca}(p_1,p_2) \succ \mathrm{nca}(p_1,p_3)$ where $\{p_j\} = pos_{x_{i_j}}(C_l)$ for $j \in [3]$, then the minimal suffix $t_s \in \mathcal{T}_\Sigma(X_k)$ such that $t_r = t_p t_s$ for some $t_p \in \bar{\mathcal{T}}_{\Sigma \cup X_k - \{x_{i_1}, x_{i_2}\}}(\{x_*\})$ does not contain x_{i_3}.

For $V \subseteq \mathrm{var}(t_r)$, let $\mathsf{dp}_{t_r}(V) = pos_{x_*}(t_p)$ where $t_p \in \bar{\mathcal{T}}_{\Sigma \cup X_k - V}(\{x_*\})$ is the maximal prefix of t_r with respect to V such that $t_r = t_p t_s$ for some $t_s \in \mathcal{T}_\Sigma(X_k)$.

Next, we decompose trees C_l and t_r in the both sides of each rule of T_f with respect to the confluence positions of variables in C_l. A position p of C_l w.r.t t_r is a *confluence position* if $\mathrm{var}(C_l|_{pi}) \cap \mathrm{var}(t_r) \subsetneq \mathrm{var}(C_l|_p) \cap \mathrm{var}(t_r)$ for all $i \in [\mathsf{rk}(\lambda_{C_l}(p))]$. Let

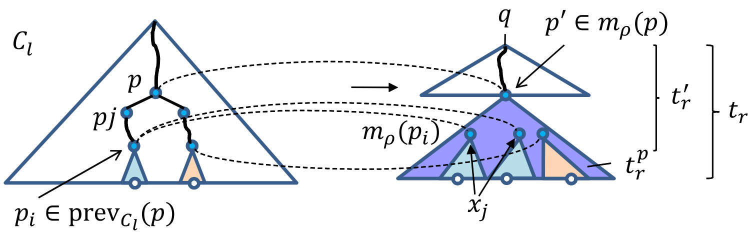

Fig. 5. Construction of t_r^p

$CP(C_l)$ be the set of all confluence positions of C_l. For $p \in CP(C_l)$, let $\mathrm{prev}_{C_l}(p)$ be the set of all the immediate previous confluence positions, that is, $\mathrm{prev}_{C_l}(p) = \{p' \mid p \prec p', \neg\exists p'' \in CP(C_l) : p \prec p'' \prec p'\}$.

By the assumption (X), for each rule $\rho = (C_l[q_1(x_1), \ldots, q_k(x_k)] \to q(t_r)) \in \delta'$, there is a mapping $m_\rho : CP(C_l) \to 2^{pos(t_r)}$ such that $m_\rho(p) = \mathsf{dp}_{t_r}(\mathrm{var}(C_l|_p))$ for each $p \in CP(C_l)$. We can construct an s-bot $T' = (\tilde{Q}^{nf}_{\mathrm{fin}} \cup (Q - Q^{nf}_{\mathrm{fin}}) \cup Q_m, \Sigma, \Delta, Q_f, \delta'')$ from T_f where

- $Q_m = \{q^\rho_p \mid \rho = (C_l[q_1(x_1), \ldots, q_k(x_k)] \to q(t_r)) \in \delta', p \in pos_\Sigma(C_l) - \{\epsilon\}\}$,
- δ'' is the smallest set such that for each rule $\rho = (C_l[q_1(x_1), \ldots, q_k(x_k)] \to q(t_r)) \in \delta'$,
  - For each $p \in CP(C_l)$, let $\sigma_p = \lambda_{C_l}(p) \in \Sigma^{(k_p)}$, then

$$\sigma_p(q^\rho_{p1}(x_1), \ldots, q^\rho_{pk_p}(x_{k_p})) \to q^\rho_p(t^p_r) \in \delta''$$

    where $t^p_r = t'_r|_{p'}$ for any $p' \in m_\rho(p)$ and t'_r is the tree obtained from t_r by replacing all subtrees at positions in $m_\rho(p_i)$ with x_j such that $pj \preceq p_i$ for all $p_i \in \mathrm{prev}_{C_l}(p)$ (See Fig. 5), and
  - If $\epsilon \notin CP(C_l)$, let $\sigma_\epsilon = \lambda_{C_l}(\epsilon) \in \Sigma^{(k_\epsilon)}$, then

$$\sigma_\epsilon(q^\rho_1(x_1), \ldots, q^\rho_{k_\epsilon}(x_{k_\epsilon})) \to q^\rho_\epsilon(t'_r) \in \delta''$$

    where t'_r is the tree obtained from t_r by replacing all subtrees at positions in $\mathsf{dp}_{t_r}(\mathrm{var}(t_r))$ with x_j such that $\mathrm{var}(t_r) \subseteq \mathrm{var}(C_l|_j)$.
  - For each $p \in pos_\Sigma(C_l) - (CP(C_l) \cup \{\epsilon\})$, let $\sigma_p = \lambda_{C_l}(p) \in \Sigma^{(k_p)}$, then
    * $\sigma_p(q^\rho_{p1}(x_1), \ldots, q^\rho_{pk_p}(x_{k_p})) \to q^\rho_p(x_i) \in \delta''$ if $C_l|_{pi}$ has at least one variable for some $i \in [k_p]$,
    * $\sigma_p(q^\rho_{p1}(x_1), \ldots, q^\rho_{pk_p}(x_{k_p})) \to q^\rho_p(\perp) \in \delta''$ otherwise.

  where $q^\rho_\epsilon = q$, and $q^\rho_p = q_i$ for each $x_i \in X_k$ and $p \in pos_{x_i}(C_l)$.

T' is equivalent with T_f because the rules obtained by decomposing each rule $\rho \in \delta'$ can simulate ρ exactly. □

Therefore, condition (X) is a necessary and sufficient condition for an s-xbot^{-e} to have an equivalent s-bot T.