On the Expressiveness of Deterministic Transducers over Infinite Trees*

Thomas Colcombet[1] and Christof Löding[2]

[1] Institute of Informatics, Warsaw University, Poland
thomas.colcombet@laposte.net
[2] Lehrstuhl Informatik VII, RWTH Aachen, Germany
loeding@informatik.rwth-aachen.de

Abstract. We introduce top-down deterministic transducers with rational lookahead (transducer for short) working on infinite terms. We show that for such a transducer $\widetilde{T}$, there exists an MSO-transduction T such that for any graph G, $\mathit{unfold}(T(G)) = \widetilde{T}(\mathit{unfold}(G))$. Reciprocally, we show that if an MSO-transduction T "preserves bisimilarity", then there is a transducer $\widetilde{T}$ such that for any graph G, $\mathit{unfold}(T(G)) = \widetilde{T}(\mathit{unfold}(G))$. According to this, transducers can be seen as a complete method of implementation of MSO-transductions that preserve bisimilarity. One application is for transformations of equational systems.

1 Introduction

The theory of tree transducers has been widely studied since the 1970s (see e.g. [9]). Tree transducers are abstract machines describing relations between finite terms. Among the numerous known families of transducers one happens to be a good compromise between decidability and expressiveness requirements: the top-down tree transducers with regular lookahead [15]. Those transducers are closed by composition, and preserve the regularity of sets of terms by inverse image.

One application of tree transducers is to implement relations between domains different from trees, in particular graphs. The principle is to attach a semantics from tuple of graphs to graphs of correct arity to each symbol and to use this semantic to evaluate any tree build upon those symbols. The resulting object is a graph called the interpretation of the tree. In this context, tree transducers describe relations between graphs through the trees representing them. Engelfriet studied this approach [16] and as it turns out top-down tree transducers with regular lookahead suit particularly well in this setting.

Top-down tree transducers have also been extended to macro tree transducers [18] which are themselves equivalent to so-called tree-to-graph transducers [19]. Those devices are strictly more expressive than top-down tree transducers. Drewes compares them with respect to translations between algebras [13]. The

* This research has been partially supported by the European Community Research Training Network "Games and Automata for Synthesis and Validation" (GAMES).

representation by monadically definable transformations of those transducers has been studied extensively (see e.g. [17, 4]), however, those results cannot be seen as the finite case counterpart of the results presented in this paper.

We describe in the present paper a similar theory for top-down tree transducers, but working on infinite terms. Such infinite terms can be interpreted into infinite objects as investigated first by Courcelle for graphs [10, 1, 2, 7] (technically the interpretation is extended to infinite terms by a limit passing argument). The same need for transducers appears in this context. However, a slightly different point of view can be adopted : each infinite tree can be obtained as the unfolding of a (possibly infinite) graph. To this respect, interpreting the term is equivalent to solving the graph seen as an equational system. For this reason, we investigate how transducers of terms can be compared with transformations of graphs : this approach produces tools for transforming (possibly infinite) equational systems. This approach is extensively used in [8].

In this paper we introduce top-down transducers with rational lookahead — we say simply transducers from now — working on infinite trees. We define the notion of rationality by means of monadic second-order definability : a set of (possibly infinite) trees is rational if it is the set of tree models of some monadic second-order formula. A transducer is a deterministic device with a finite number of states that reads a (possibly infinite) input tree starting from the root and produces a (possibly infinite) output tree. Each transition consists in either consuming the input root symbol, producing an output symbol, or verifying that the input tree belongs to some rational set (this ability is called the 'lookahead').

Major results concerning the finite tree case still hold for those transducers : we show the closure by composition of transducers and the rationality of the inverse image of a rational set by a transducer. However, an extra hypothesis of determinism of the transducer is necessary in our proof. We also investigate the relationships of those transducers with respect to unfolding and monadic second-order transductions (MSO-transductions for short). We establish that the result of a transducer applied to the unfolding of a graph can also be obtained by the successive application of an MSO-transduction followed by an unfolding. We say in this case that the MSO-transduction implements the transducer. Let us note that such an MSO-transduction is by definition bisimilarity preserving. In fact, a converse to this result also holds and is the most involved proof presented in this work : every MSO-transduction implements a transducer provided that it preserves bisimilarity. For this reason transducers can be understood as the tree theoretic counterpart to MSO-transductions.

Among consequences of those results are that regularity of terms (but not rationality of sets of terms) is preserved by transducers. More generally, term solutions of safe higher-order program schemes of level n are closed under application of transducers [20, 6, 5].

The remainder of the paper is divided as follows. In the next section we give the basic definitions on graphs, MSO-transductions, and transducers. In Section 3 we state some basic properties of deterministic transducers and show that the functions computed by them can also be obtained using MSO-transductions followed by unfolding. In the last section we present the result that MSO-transductions that preserve bisimilarity of graphs can be simulated by deterministic transducers on the unfoldings of the graphs.

2 Definitions

An (edge-labeled) graph G over an alphabet Σ is a pair $G = (V_G, E_G)$ where V_G is the set of vertices and $E_G \subseteq V_G \times \Sigma \times V_G$ is the set of edges. A rooted graph G is of the form $G = (V_G, E_G, r_G)$ where V_G and E_G are as before and $r_G \in V_G$ is the root of G. A directed path in G is a sequence of vertices such that successive vertices u, v in this sequence are connected by an edge $(u, a, v) \in E_G$ for some $a \in \Sigma$. A sequence of vertices is an undirected path if successive vertices u, v in this sequence are connected by an edge $(u, a, v) \in E_G$ or $(v, a, u) \in E_G$. For two vertices $u, v \in V_G$ a connection of u and v is a path from u to v that does not contain any vertex twice.

A (undirected) tree t is a graph such that for each two vertices $u, v \in V_t$ there is exactly one undirected connection between u and v. A rooted tree t is a tree such that for each $v \in V_t$ there is a directed path from the root r_t of t to v. For a rooted graph G we denote by $\mathrm{unfold}(G)$ the rooted tree that is obtained by unfolding G from the root r_G. If we want to unfold G from a vertex v different from the root, then we write this as $\mathrm{unfold}(G, v)$.

For a ranked alphabet $\mathcal{F}$ and $f \in \mathcal{F}$ we write $|f|$ for the arity of f. By $|\mathcal{F}|_{\max}$ we denote the maximal rank of a symbol in $\mathcal{F}$. We represent terms over $\mathcal{F}$ as rooted trees over the alphabet $\Sigma_{\mathcal{F}} = \mathcal{F} \cup \{1, \ldots, |\mathcal{F}|_{\max}\}$. A term over $\mathcal{F}$ is a rooted tree t over $\Sigma_{\mathcal{F}}$ such that

- there is exactly one edge starting from r_t and this edge is labeled with a letter from $\mathcal{F}$,
- if there is an edge $(v, f, v') \in E_t$ for some $f \in \mathcal{F}$, then this is the only edge starting from v and there are exactly $|f|$ edges starting from v' labeled by $1, \ldots, |f|$, respectively, and
- if there is an edge $(v, \ell, v') \in E_t$ with $\ell \in \{1, \ldots, |\mathcal{F}|_{\max}\}$, then there is an edge labeled by a letter from $\mathcal{F}$ starting from v'.

The set of all $\mathcal{F}$-terms is denoted by $\mathcal{T}(\mathcal{F})$.

We say that a rooted graph G represents a term iff $\mathrm{unfold}(G)$ is a term. We are only interested in graphs representing terms. Therefore, in the following an $\mathcal{F}$-graph always means a rooted graph over $\Sigma_{\mathcal{F}}$ that represents a term. So, the $\mathcal{F}$-trees are exactly the $\mathcal{F}$-terms. For two $\mathcal{F}$-graphs G and G' we write $G \sim G'$ if G and G' represent the same term, i.e., if $\mathrm{unfold}(G) = \mathrm{unfold}(G')$. Since $\mathcal{F}$-graphs are deterministic they have the same unfolding iff they are bisimilar. Hence, on $\mathcal{F}$-graphs the relation $\sim$ corresponds to bisimulation equivalence (cf. [22]).

According to the above definition of terms there is a natural partition of the vertices of an $\mathcal{F}$-graph into those vertices being the source of an $\mathcal{F}$-edge and those vertices being the source of edges labeled with natural numbers. We denote the former of these two sets by $V_G^{\mathcal{F}} = \{v \in V_G \mid \exists f \in \mathcal{F}, u \in V_G \ : \ (v, f, u) \in E_G\}$. Since the vertices that are not from $V_G^{\mathcal{F}}$ are those that have to be inserted when passing from the usual representation of terms to our representation we call them auxiliary vertices. The vertices from $V_G^{\mathcal{F}}$ are called main vertices.

MSO-Transductions. For the remainder of this article we fix two ranked alphabets $\mathcal{F}$, $\mathcal{F}'$ and write Σ, Σ' instead of $\Sigma_{\mathcal{F}}$ and $\Sigma_{\mathcal{F}'}$. We assume the standard syntax and semantics of MSO logic over graphs, i.e., quantification over individual vertices (first-order quantification) and quantification over sets of vertices (monadic second-order quantification). For an introduction to MSO logic we refer the reader to [14]. An MSO-transduction T is of the form

$$T = (\Sigma, \Sigma', (\phi_{a,i,j}(x, y))_{a \in \Sigma', i,j \in [1,n]}, (\rho_i(x, y))_{i \in [1,n]}, n)$$

with MSO-formulas $\phi_{a,i,j}(x, y)$ and $\rho_i(x, y)$ over the signature $(E_a)_{a \in \Sigma}$, where each E_a is a binary symbol interpreted as the set of a-labeled edges.

In order to obtain a unique root we require that for all $\mathcal{F}$-graphs G and all $v \in V_G$ there is at most one $u \in V_G$ and $i \in [1, n]$ such that $G \models \rho_i(v, u)$. For each $\mathcal{F}$-graph G we define the graph $T(G)$ over Σ' that is obtained by applying T to G as follows. If there are no $u \in V_G$ and $i \in [1, n]$ with $G \models \rho_i(r_G, u)$, then $T(G)$ is undefined. Otherwise,

- $V_{T(G)} = V \times [1, n]$,
- for $a \in \Sigma'$ and $i, j \in [1, n]$ there is an edge $((v, i), a, (u, j))$ in $E_{T(G)}$ iff $G \models \phi_{a,i,j}(v, u)$, and
- $r_{T(G)} = (u, i)$ for the unique u and i with $G \models \rho_i(r_G, u)$.

Note that our definition of MSO-transduction slightly differs from the standard definition (cf. [11]). We are interested in rooted graphs and thus we need the formulas $\rho_i(x, y)$ to define the roots of the transformed graph. Since furthermore, our main interest is on the unfolding of the transformed graphs, we do not need a formula restricting the domain of T. We sometimes write $T(G, v)$ for some vertex v of G to denote the application of T to the graph G with its root changed to v.

The definition of an MSO-transduction does not enforce that $T(G)$ represents an $\mathcal{F}'$-term when applied to a graph G representing an $\mathcal{F}$-term. Furthermore, we are interested in simulating MSO-transductions by transducers working on terms. Thus, we want to consider MSO-transductions that, when applied to two $\mathcal{F}$-graphs representing the same term, yield two $\mathcal{F}'$-graphs again representing the same term. This is captured by the following definition.

We call an MSO-transduction T bisimilarity preserving iff

- $T(G)$ (if it is defined) is an $\mathcal{F}'$-graph for each $\mathcal{F}$-graph G and
- for all $\mathcal{F}$-graphs G and G', if $G \sim G'$, then $T(G)$ is defined iff $T(G')$ is defined and $T(G) \sim T(G')$.

So, bisimilarity preserving MSO-transductions transform $\mathcal{F}$-graphs into $\mathcal{F}'$-graphs and preserve bisimulation equivalence of graphs. In particular, because of the first condition, all the formulas $\phi_{a,i,j}(x,y)$ in a bisimilarity preserving MSO-transduction T must be deterministic in the following sense. For all $\mathcal{F}$-graphs G and $v \in V_G$ there is at most one $u \in V_G$ such that $G \models \phi_{a,i,j}(v,u)$. We call an MSO-transduction with this property deterministic.

Transducers with Rational Lookahead. A top down tree transducer with rational lookahead (transducer for short) is a tuple $\widetilde{T} = (Q, \mathcal{F}, \mathcal{F}', q_0, \Delta)$ with:

- Q a finite set of states,
- $q_0 \in Q$ the initial state, and
- Δ a finite set of rules of one of the following forms:
  (production rule): $q(x) \to g(q_1(x), ..., q_{|g|}(x))$ with $g \in \mathcal{F}'$, x a variable, and $q_1, \ldots, q_{|g|} \in Q$.
  (consumption rule): $q(f(x_1, ..., x_{|f|})) \to q'(x_i)$ with $f \in \mathcal{F}$, $q, q' \in Q$, and $x_1, \ldots, x_{|f|}$ variables.
  (lookahead rule): $q(x \in L) \to q'(x)$ with L a rational set of $\mathcal{F}$-terms (called lookahead set), $q, q' \in Q$, and x a variable.

Each rule of Δ can be interpreted as a rewrite rule. A lookahead rule $q(x \in L) \to q'(x)$ can only be applied to $q(t)$ if t is a term from L. Hence the lookahead rules allow to 'inspect' the input tree and collect some information about it.

A transducer $\widetilde{T} = (Q, \mathcal{F}, \mathcal{F}', q_0, \Delta)$ is deterministic if for each state $q \in Q$ and each $\mathcal{F}$-term t the set of rules that can be applied to $q(t)$

- either consists of lookahead rules with pairwise disjoint lookahead sets, or
- contains exactly one production rule, or
- contains exactly one consumption rule.

According to the above definition we will speak of production states, consumption states, and lookahead states.

The result $\widetilde{T}(t)$ of applying $\widetilde{T}$ to an $\mathcal{F}$-term t is the term that is obtained from t by applying the rewrite rules of $\widetilde{T}$ 'to the limit', starting from $q_0(t)$. In the formal definition of $\widetilde{T}(t)$ we have to be careful because we cannot simply define the image of an infinite term as the limit of a sequence of images of finite terms. Because of the lookahead the functions computed by deterministic transducers need not to be continuous.

Let $\widetilde{T} = (Q, \mathcal{F}, \mathcal{F}', q_0, \Delta)$ be a deterministic transducer and let $\mathcal{F}'_\perp$ be the ranked alphabet $\mathcal{F}'$ augmented by a new symbol $\perp$ of rank 0. By induction on n we define for each state $q \in Q$ and each infinite term $t \in \mathcal{T}(\mathcal{F})$ the term $\delta_n(q,t) \in \mathcal{T}(\mathcal{F}'_\perp)$ as $\delta_0(q,t) = \perp$,

- if $q(x) \to g(q_1(x), \ldots, q_{|g|}(x)) \in \Delta$, then $\delta_{n+1}(q,t) = g(\delta_n(q_1,t), \ldots, \delta_n(q_{|g|},t))$,
- if $q(f(x_1, ..., x_{|f|})) \to q'(x_i) \in \Delta$, then $\delta_{n+1}(q,t) = \delta_n(q',t_i)$ for $t = f(t_1, \ldots, t_{|f|})$,
- if $q(x \in L) \to q'(x) \in \Delta$ and $t \in L$, then $\delta_{n+1}(q,t) = \delta_n(q',t)$.

If no transition of the transducer can be applied or if the right hand side in the definition of $\delta_{n+1}(q,t)$ is undefined, then $\delta_{n+1}(q,t)$ is undefined.

First note that in each situation at most one rule can be applied because of the determinism of the transducer. Therefore, if $\delta_n(q,t)$ is defined, then it is unique. If we consider the complete partial order $\sqsubseteq$ on $\mathcal{F}'_\perp$-terms with $t' \sqsubseteq t$ if t' is obtained from t by replacing subterms with $\perp$, then one can easily show by induction on n that the sequence $(\delta_n(q,t))_{n\in\mathbb{N}}$ is either increasing (w.r.t $\sqsubseteq$) or undefined from a certain point onward. In the former case we let $\widetilde{T}_q(t)$ be the limit of this sequence and in the latter case $\widetilde{T}_q(t)$ is undefined. Now we can define $\widetilde{T}(t) = \widetilde{T}_{q_0}(t)$.

3 First results about transducers

In this section, we establish that the inverse image of a rational set by a transducer is also rational (Lemma 2). We also show that transducers can be implemented by MSO-transductions (Theorem 1).

According to the current definition, it may happen that for some $\mathcal{F}$-term t and some state q the term $\widetilde{T}_q(t)$ still contains the symbol $\perp$. This phenomenon can be a technical burden for the following proofs. We start this section by normalizing transducers in such a way that this situation does not occur anymore.

Formally, let $\widetilde{T}$ be a transducer from $\mathcal{F}$-terms to $\mathcal{F}'$-terms of states Q, we say that $\widetilde{T}$ is *normalized* if for any state $q \in Q$ and any $\mathcal{F}$-term t, $\widetilde{T}_q(t) \neq \perp$.

For $\widetilde{T}$ to be normalized it is sufficient but not necessary to have a production in each of its cycles. Consider for instance a transducer that would remove all the occurrences of a given symbol — say a of arity 1 — provided that this symbol has only a finite number of occurrences in the term. This transducer contains cycles without production since an unbounded number of a can be removed without producing any output symbol. However, by definition this cannot happen infinitely often.

Lemma 1. *Let $\widetilde{T}$ be a transducer from $\mathcal{F}$ to $\mathcal{F}'$ and $\perp'$ be a new symbol of arity 0. There exists effectively a transducer $\widetilde{T}'$ from $\mathcal{F}$-terms to $(\mathcal{F}' \cup \{\perp'\})$-terms such that $\widetilde{T}' = h \circ \widetilde{T}$ where h replaces every occurrence of the symbol $\perp$ in a term by $\perp'$.*

An important property of deterministic transducers is that their domain is rational. More precisely, as stated in Lemma 2, the inverse image of a rational language by a normalized deterministic transducer is rational.

Lemma 2. *Let $\widetilde{T}$ be a deterministic transducer from $\mathcal{F}$-terms to $\mathcal{F}'$-terms. If L is a rational subset of $\mathcal{T}(\mathcal{F}')$, then the set $\widetilde{T}^{-1}(L)$ is also rational.*

Let us notice that in this proof, the determinism of the transducer is explicitly needed and we don't know if the result remains true without this restriction. This was not the case for transducers of finite trees. It also follows directly from this lemma that the domain of a transducer is rational.

Finally, we aim at establishing Theorem 1 which expresses how a transducer can be simulated by an MSO-transduction before unfolding. First of all we need a result relating lookaheads with MSO-logic. It is a particular case of Courcelle's result in [12].

Lemma 3. *For any rational set of $\mathcal{F}$-terms L, there exists an MSO-formula $\phi(x)$ such that for any $\mathcal{F}$-graph G and any vertex $v \in V_G^{\mathcal{F}}$, $\mathrm{unfold}(G, v) \in L$ iff $G \models \phi(v)$.*

We can now state the main result of this section.

Theorem 1. *Let $\widetilde{T}$ be a normalized deterministic transducer from $\mathcal{F}$-terms to $\mathcal{F}'$-terms. There exists effectively an MSO-transduction T such that for any $\mathcal{F}$-graph G and any vertex $v \in V_G^{\mathcal{F}}$, $T(G, v)$ is defined iff $\widetilde{T}(\mathrm{unfold}(G, v))$ is defined, and in this case $\mathrm{unfold}(T(G, v)) = \widetilde{T}(\mathrm{unfold}(G, v))$.*

The proof consists in using one copy of the graph for each state of the transducer. Then the MSO-formulas put correctly the edges. The Lemmas 2 and 3 combined allow the implementation of lookahead rules.

4 From MSO-Transductions to Transducers

The goal of this section is to show that bisimilarity preserving MSO-transductions can be simulated by deterministic transducers on the unfoldings of graphs. In a first step we use the fact that T is bisimilarity preserving to obtain some kind of normal form for T.

Since rooted graphs are bisimilar to their unfoldings the following simple remark allows us to consider MSO-transductions operating on trees instead of graphs. Formally, this means that if T is a bisimilarity preserving MSO-transduction, then $\mathrm{unfold}(T(G)) = \mathrm{unfold}(T(\mathrm{unfold}(G)))$ for every $\mathcal{F}$-graph G.

To simulate an MSO-transduction by a deterministic transducer we would like the MSO-transduction to 'respect' the type (main or auxiliary) of the vertices since transducers only work on main vertices. Furthermore, transducers work in a top-down fashion. Thus, we want to normalize the MSO-transduction in such a way that the new edges go 'downward' if the transduction is applied to a tree. Under this assumption a deterministic transducer can construct the edges defined by the MSO-transduction by going down the term that is represented by t and using its rational lookahead. The following definition formally captures the properties we need to simulate an MSO-transduction by a deterministic transducer.

Definition 1. *An MSO-transduction $T = (\Sigma, \Sigma', (\phi_{a,i,j}(x, y))_{a,i,j}, (\rho_i(x, y))_i, n)$ is in top-down normal form iff for all $\mathcal{F}$-trees t*

(a) $r_{T(t)} = (r_t, 1)$,
(b) *if $t \models \phi_{g,i_1,i_2}(v_1, v_2)$ and $t \models \phi_{\ell,i_2,i_3}(v_2, v_3)$ for $g \in \mathcal{F}'$, $\ell \in \{1, \ldots, |\mathcal{F}'|_{\max}\}$, and $i_1, i_2, i_3 \in [1, n]$, then $v_1 \sqsubseteq v_3$, and*
(c) *for every $\mathcal{F}$-graph G, if $G \models \phi_{g,i,j}(u, v)$ for $g \in \mathcal{F}'$, then $u \in V_G^{\mathcal{F}}$ and $v \notin V_G^{\mathcal{F}}$, and if $G \models \phi_{\ell,i,j}(u, v)$ for $\ell \in \{1, \ldots, |\mathcal{F}'|_{\max}\}$, then $u \notin V_G^{\mathcal{F}}$ and $v \in V_G^{\mathcal{F}}$.*

The following lemma states that we can always ensure condition (a).

Lemma 4. *For each MSO-transduction T there exists an MSO-transduction T' such that $r_{T'(G)} = (r_G, 1)$ and $\operatorname{unfold}(T(G)) = \operatorname{unfold}(T'(G))$ for each rooted graph G.*

The most intricate thing is to establish condition (b) of Definition 1. First, we construct for a given tree t a new graph $\widehat{t}$ on which the MSO-transduction must satisfy condition (b).

Let t be an $\mathcal{F}$-tree. We can assume that $V_t \subseteq \Sigma^*$ by identifying each element v from V_t with the labeling of the unique path leading from r_t to v. We define the $\mathcal{F}$-graph $\widehat{t}$ by

- $V_{\widehat{t}} = \{\langle u_1, \ldots, u_n\rangle \mid n > 0,\ u_1, \ldots, u_n \in V_t, \text{ and } u_{i-1} \not\sqsubseteq u_i \text{ for all } i \in [2, n]\}$.
- The edges in $E_{\widehat{t}}$ are those of the form $\langle u_1, \ldots, u_n\rangle \xrightarrow{a} \langle u_1, \ldots, u_n a\rangle$ and $\langle u_1, \ldots, u_n, va, v\rangle \xrightarrow{a} \langle u_1, \ldots, u_n, va\rangle$.

Figure 1 shows a part of this construction for a term built from a single binary symbol f. The underlined vertices correspond to the vertices of the original term.

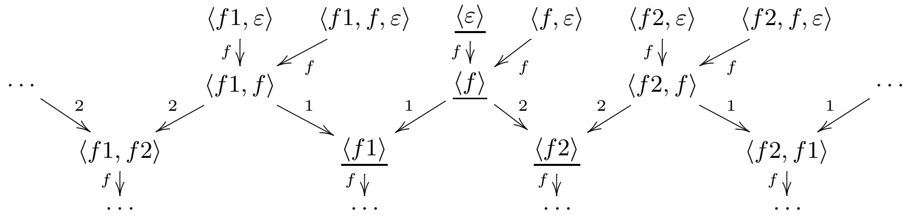

Fig. 1. A part of $\widehat{t}_f$

For $\widehat{u} \in V_{\widehat{t}}$ we denote by $\lambda(\widehat{u})$ the last element in the sequence $\widehat{u}$, i.e., if $\widehat{u} = \langle u_1, \ldots, u_m\rangle$, then $\lambda(\widehat{u}) = u_m$. The following properties are easy to derive from the definition of $\widehat{t}$.

Lemma 5. *For each $\mathcal{F}$-tree t and each vertex $\widehat{v} \in V_{\widehat{t}}$:*

(i) *$\widehat{t}$ is a (undirected) tree.*
(ii) *If $\lambda(\widehat{v}) = \varepsilon$, then there are no edges with target $\widehat{v}$ in $\widehat{t}$. Otherwise, $\widehat{v}$ is the target of exactly two edges in $\widehat{t}$ that have the same label as the only edge with target $\lambda(\widehat{v})$ in t.*
(iii) *$\widehat{v}$ is the source of an a-edge in $\widehat{t}$ iff $\lambda(\widehat{v})$ is the source of an a-edge in t.*
(iv) $\operatorname{unfold}(t, v) = \operatorname{unfold}(\widehat{t}, \langle v\rangle)$ *for each $\mathcal{F}$-tree t and vertex v of t.*

An important property of $\widehat{t}$ is that two vertices $\widehat{u}$ and $\widehat{u}'$ with $\lambda(\widehat{u}) = \lambda(\widehat{u}')$ are indistinguishable, i.e., there is an automorphism of $\widehat{t}$ that maps $\lambda(\widehat{u})$ to $\lambda(\widehat{u}')$. This enforces a certain behavior of deterministic MSO-transductions on $\widehat{t}$ that corresponds to the second property of the top-down normal form.

Lemma 6. *Let T be a deterministic MSO-transduction, t an $\mathcal{F}$-tree, and $\widehat{u}, \widehat{v} \in V_{\widehat{t}}$. If $\widehat{t} \models \phi_{a,i,j}(\widehat{u}, \widehat{v})$, then $\lambda(\widehat{u}) \sqsubseteq \lambda(\widehat{v})$.*

As Lemma 6 shows, property (b) of Definition 1 holds on $\widehat{t}$ for MSO-transductions preserving bisimulation equivalence. To transfer this property to t itself we will make use of the concept of tree-like structures (cf. [24, 3]).

For a graph G the tree-like structure $G^* = (V_G^*, \mathrm{son}, \mathrm{clone}, E_G^*)$ is the structure over the universe V_G^* with the relations son, clone, and E_G^* defined by $\mathrm{son} = \{(w, wv) \mid w \in V_G^*,\ v \in V_G\}$, $\mathrm{clone} = \{wvv \mid w \in V_G^*,\ v \in V_G\}$, and $E_G^* = \{(wv, a, wu) \mid w \in V_G^*,\ (v, a, u) \in E_G\}$. The crucial point of the tree-like structure is that monadic second-order properties can be expressed as monadic second order properties of the original structure.

Theorem 2 (cf. [3]). *For each MSO-sentence ϕ there exists an MSO-sentence ϕ^* such that $G^* \models \phi \Leftrightarrow G \models \phi^*$ for all $\mathcal{F}$-graphs G.*

The two structures $\widehat{t}$ and t^* are closely related, $\hat{t}$ can be interpreted in t^*. This can be used to show a modification of the above result for $\widehat{t}$.

Lemma 7. *For each MSO-formula $\phi(x_1, \ldots, x_m)$ there exists an MSO-formula $\phi^*(x_1, \ldots, x_m)$ such that for all $\mathcal{F}$-trees t:*

(i) If $\widehat{t} \models \phi(\widehat{u}_1, \ldots, \widehat{u}_m)$, then $t \models \phi^*(\lambda(\widehat{u}_1), \ldots, \lambda(\widehat{u}_m))$ for all $\widehat{u}_1, \ldots, \widehat{u}_m \in V_{\widehat{t}}$.
(ii) If $t \models \phi^*(u_1, \ldots, u_m)$, for $u_1, \ldots, u_m \in V_t$, then there are $\widehat{u}_1, \ldots, \widehat{u}_m \in V_{\widehat{t}}$ with $\lambda(\widehat{u}_i) = u_i$ $(1 \leq i \leq m)$ and $\widehat{t} \models \phi(\widehat{u}_1, \ldots, \widehat{u}_m)$.

Now we can establish property (b) of Definition 1.

Lemma 8. *For each bisimilarity preserving MSO-transduction T there exists an MSO-transduction T' satisfying (a) and (b) of Definition 1 with* $\mathrm{unfold}(T(t)) = \mathrm{unfold}(T'(t))$ *for all $\mathcal{F}$-trees t.*

As a last step in the normalization we have to ensure (c) of Definition 1. Since the MSO-transduction might use copies of auxiliary vertices as targets of $\mathcal{F}$-edges and main vertices as targets of other $\mathcal{F}$-edges we cannot simply redirect the latter ones to main vertices within the same copy. For this reason we have to introduce new copies of the original term and redirect edges with wrong type of source vertex or target vertex to these new copies.

Lemma 9. *For each bisimilarity preserving MSO-transduction T there exists an MSO-transduction T' in top-down normal form with* $\mathrm{unfold}(T(t)) = \mathrm{unfold}(T'(t))$ *for all $\mathcal{F}$-trees t.*

Having established the top-down normal form the next goal is to simulate MSO-transductions in this normal form by deterministic transducers. We make use of the well-known equivalence between MSO logic and Rabin automata on infinite trees (cf. [23]). This allows us to pass from the formulas defining the edges in the MS transduction to equivalent automata accepting infinite terms with appropriate markings coding the assignment for the free first-order variables in the formulas. We start by defining marked terms and automata running on them. Due to the lack of space we do not give detailed definitions and assume familiarity with the theory of automata on infinite trees (cf. [23]).

For a set Θ, a Θ-marked $\mathcal{F}$-term (t, μ) consists of an $\mathcal{F}$-term t and a marking function $\mu : \Theta \to V_t$. Let $\mathcal{T}(\mathcal{F}, \Theta)$ be the set of all Θ-marked $\mathcal{F}$-terms. For a set $L \subseteq \mathcal{T}(\mathcal{F}, \Theta)$ of marked terms we define $L|_{\mathcal{F}} = \{t \mid \exists \mu.(t, \mu) \in L\}$.

We fix an MSO-transduction $T = (\Sigma, \Sigma', (\phi_{a,i,j}(x, y))_{a,i,j}, (\rho_i(x, y))_i, n)$ in top-down normal form that was obtained from a bisimilarity preserving MSO-transduction as described in the previous subsection. For all $g \in \mathcal{F}'$, all $i \in [1, n]$, and all $\boldsymbol{j} \in [1, n]^{|g|}$ define

$$\psi_{g,i,\boldsymbol{j}}(x, x_1, \ldots, x_{|g|}) = \exists y \bigvee_{m \in [1,n]} \left(\phi_{g,i,m}(x, y) \wedge \bigwedge_{\ell \in \{1,\ldots,|g|\}} \phi_{\ell,m,j_\ell}(y, x_\ell) \right),$$

where j_ℓ denotes the ℓth component of $\boldsymbol{j}$, i.e., $\boldsymbol{j} = (j_1, \ldots, j_k)$. For $g \in \mathcal{F}'$ we define $\Theta_g = \{g, 1, \ldots, |g|\}$. For $(t, \mu) \in \mathcal{T}(\mathcal{F}, \Theta_g)$ we write by abuse of notation $(t, \mu) \models \psi_{g,i,\boldsymbol{j}}$ iff $t \models \psi_{g,i,\boldsymbol{j}}(\mu(g), \mu(1), \ldots, \mu(|g|))$. Note that T being in top-down normal form implies that $\mu(g), \mu(1), \ldots, \mu(|g|) \in V_t^{\mathcal{F}}$ if $(t, \mu) \models \psi_{g,i,\boldsymbol{j}}$ by Definition 1 (c). This enables us to adapt the usual framework of automata running on infinite trees (cf. [23]) to our representation of marked terms.

Every MSO-formula can be translated into a nondeterministic Rabin tree automaton accepting precisely the models of the formula [21]. Hence, for all $g \in \mathcal{F}'$, all $i \in [1, n]$, and all $\boldsymbol{j} \in [1, n]^{|g|}$ there exists a nondeterministic Rabin tree automaton $\mathcal{A}_{g,i,\boldsymbol{j}} = (Q_{g,i,\boldsymbol{j}}, \mathcal{F} \times 2^{\Theta_g}, Q^{\text{in}}_{g,i,\boldsymbol{j}}, \Delta_{g,i,\boldsymbol{j}}, \Omega_{g,i,\boldsymbol{j}})$ that accepts exactly the Θ_g-marked $\mathcal{F}$-terms (t, μ) with $(t, \mu) \models \psi_{g,i,\boldsymbol{j}}$. We assume that the state sets of these automata are pairwise disjoint. We denote the union of the automata $\mathcal{A}_{g,i,\boldsymbol{j}}$ by the automaton $\mathcal{A} = (Q, \bigcup_{g \in \mathcal{F}'}(\mathcal{F} \times 2^{\Theta_g}), Q_{\text{in}}, \Delta, \Omega)$, where Q, Q_{in}, Δ, and Ω are obtained by taking the union of the respective components of the automata $\mathcal{A}_{g,i,\boldsymbol{j}}$.

The automaton $\mathcal{A}$ is the main ingredient in the construction of the transducer. The idea is to keep track of the states of $\mathcal{A}$ that could have been reached on the input term and to use the rational lookahead to decide which automaton $\mathcal{A}_{g,i,\boldsymbol{j}}$ to use to construct the next edge. We illustrate the work of the transducer with a simple example.

Consider the part of a marked term (t, μ) depicted in the upper left box of Figure 2. The labels g, 1, and 2 on the vertices are the marks. The states q_g, q, q_1, q_2 are part of an accepting run of the automaton $\mathcal{A}_{g,i,j_1,j_2}$ for some i, j_1, j_2. In steps (1) to (5) the work of the transducer is illustrated.

In (1) the transducer is in state $\langle i, C\rangle$. The i indicates in which copy introduced by the MSO-transduction the transducer currently is. The set of states that could have been reached by $\mathcal{A}$ at this point are stored in C. We assume that $q_g \in C$ and using its rational lookahead the transducer can check that the term with the marking depicted in the upper left box is accepted from $\mathcal{A}_{g,i,j_1,j_2}$ with q_g as initial state.

In step (2) the transducer applies a production rule creating the g-edge, the 1-edge, and the 2-edge. Now it has to consume the part of the term until

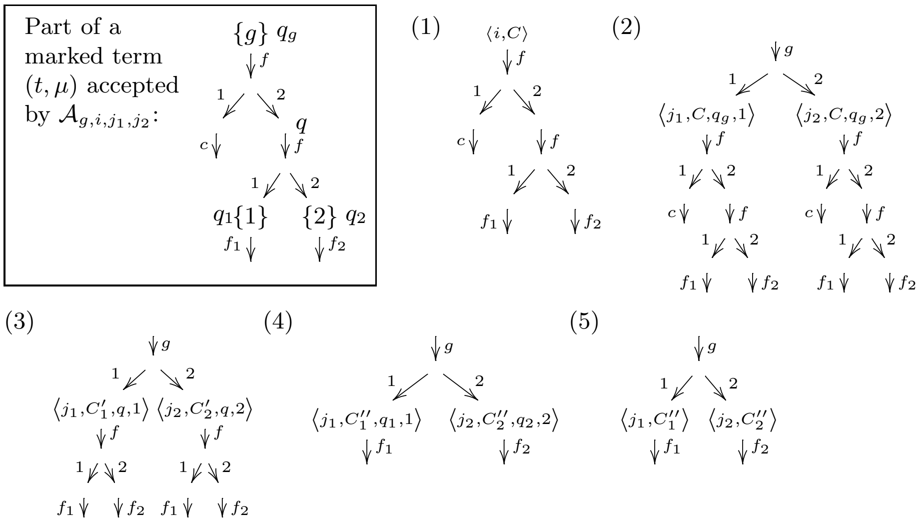

Fig. 2. Illustration of the transducer constructing an edge

reaching the vertex marked with 1 in the left hand side copy and likewise in the right hand side copy for the vertex marked with 2. To that aim it goes to the states $(j_1, C, q_g, 1)$ and $(j_2, C, q_g, 2)$ in the two corresponding copies of t. The last component of the states indicates for which mark the transducer is waiting.

In step (2) the transducer has reached the two corresponding vertices. Note that in step (3) two independent rewritings of the transducer have been applied in parallel and similarly in step (4). In all these steps the transducer implicitly uses its lookahead to decide in which direction to proceed in the term. In each consumption step the sets storing the reachable states of $\mathcal{A}$ are updated. Having reached the desired vertices the transducer switches back to the mode where it checks which edge to construct next. The formal realization of this idea is rather technical and is omitted here.

Lemma 10. *There exists a deterministic transducer* $\widetilde{T}$ *such that* $\mathrm{unfold}(T(t)) = \widetilde{T}(t)$ *for each term* t*.*

Now we can prove the main result of this section.

Theorem 3. *For each bisimilarity preserving MSO-transduction* T *there exists a deterministic transducer* $\widetilde{T}$ *such that* $\mathrm{unfold}(T(G)) = \widetilde{T}(\mathrm{unfold}(G))$ *for each* $\mathcal{F}$*-graph* G*.*

Proof. Let T be a bisimilarity preserving MSO-transduction. By Lemma 9 there is an MSO-transduction T' in top-down normal form such that $\mathrm{unfold}(T(t)) = \mathrm{unfold}(T'(t))$ for all $\mathcal{F}$-trees t. According to Lemma 10 there is a deterministic transducer $\widetilde{T}$ such that $\mathrm{unfold}(T'(t)) = \widetilde{T}(t)$ for each $\mathcal{F}$-tree t. Let G be an $\mathcal{F}$-graph and let $t_G = \mathrm{unfold}(G)$. Then we get $\mathrm{unfold}(T(G)) = \mathrm{unfold}(T(t_G)) = \mathrm{unfold}(T'(t_G)) = \widetilde{T}(t_G)$. $\square$

References

1. K. Barthelmann. On equational simple graphs. Technical Report 9, Universität Mainz, Institut für Informatik, 1997.
2. K. Barthelmann. When can an equational simple graph be generated by hyperedge replacement? In *MFCS '98*, volume 1450 of *LNCS*, pages 543–552. Springer, 1998.
3. D. Berwanger and A. Blumensath. The monadic theory of tree-like structures. In E. Grädel, W. Thomas, and T. Wilke, editors, *Automata, Logics, and Infinite Games*, number 2500 in LNCS, chapter 16, pages 285–301. Springer, 2002.
4. R. Bloem and J. Engelfriet. A comparison of tree transductions defined by monadic second order logic and by attribute grammars. *JCSS*, 61(1):1–50, 2000.
5. A. Carayol and S. Wöhrle. The caucal hierarchy of infinite graphs in terms of logic and higher-order pushdown automata. In *FSTTCS 2003*, volume 2914 of *LNCS*, pages 112–123. Springer, 2003.
6. D. Caucal. On infinite terms having a decidable monadic theory. In *MFCS'02*, volume 2420 of *LNCS*, pages 165–176. Springer, 2002.
7. T. Colcombet. On families of graphs having a decidable first order theory with reachability. In *ICALP 2002*, volume 2380 of *LNCS*. Springer, 2002.
8. T. Colcombet. *Propriétés et représentations de structures infinies.* PhD thesis, IRISA, Rennes, 2004. to appear.
9. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications, 1997.
10. B. Courcelle. The monadic second-order logic of graphs II: infinite graphs of bounded tree width. *Mathematical Systems Theory*, 21:187–221, 1989.
11. B. Courcelle. Monadic second-order graph transductions: A survey. In *TCS*, volume 126, pages 53–75. Elsevier, 1994.
12. B. Courcelle. The monadic second order logic of graphs IX: Machines and their behaviours. In *TCS*, volume 151, pages 125–162, 1995.
13. F. Drewes. The use of tree transducers to compute translations between graph algebras. Report 8/94, Univ. Bremen, 1994.
14. H.-D. Ebbinghaus and J. Flum. *Finite Model Theory.* Perspectives in Mathematical Logic. Springer, Berlin, second edition, 1995.
15. J. Engelfriet. Top-down tree transducers with regular lookahead. *Mathematical Systems Theory*, 10:289–303, 1977.
16. J. Engelfriet. Graph grammars and tree transducers. In S. Tison, editor, *CAAP'94*, volume 787 of *LNCS*, pages 15–36. Springer, 1994.
17. J. Engelfriet and S. Maneth. Characterizing and deciding MSO-definability of macro tree transductions. *LNCS*, 1770:542+, 2000.
18. J. Engelfriet and H. Vogler. Macro tree transducers. *JCSS*, 31:71–146, 1985.
19. J. Engelfriet and H. Vogler. The translation power of top-down tree-to-graph transducers. *JCSS*, 49:258–305, 1994.
20. T. Knapik, D. Niwinski, and P. Urzyczyn. Higher-order pushdown trees are easy. In *FOSSACS'02*, volume 2303 of *LNCS*, pages 205–222. Springer, 2002.
21. M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969.
22. C. Stirling. *Modal and Temporal Properties of Processes.* Graduate Texts in Computer Science. Springer, 2001.
23. W. Thomas. Languages, automata, and logic. In *Handbook of Formal Language Theory*, volume III, pages 389–455. Springer, 1997.
24. I. Walukiewicz. Monadic second-order logic on tree-like structures. *TCS*, 275:311–346, 2002.