A Logical Characterization of Extended TAGs

Uwe Mönnich
Theoretische Computerlinguistik
Universität Tübingen
Wilhelmstr. 19
72074 Tübingen
uwe.moennich@uni-tuebingen.de

Abstract

Context-free tree grammars, originally introduced by Rounds ((Rounds, 1970)), are powerful grammar devices for the definition of tree languages. In the present paper, we consider a subclass of the class of context-free tree languages, namely the class of monadic simple context-free tree languages. For this class of context-free tree languages, a faithful rendering of extended TAGs, we show that it can be given a simple logical characterization in terms of monadic second-order transductions.

1 Introduction

The monadic simple context-free tree languages belong to the class of mildly context-sensitive languages. The intuitive notion of mild context-sensitivity has led to two competing candidates claiming to be an exact formal rendering of the intentions Joshi (1985) was trying to capture when introducing this concept. On the one hand multiple context-free grammars and their equivalents exhibiting a variety of wildly different specifications provide impressive evidence that this precise counterpart of the informal description of mild context-sensitivity constitutes a natural class. On the other hand the well-nested subclass of the multiple context-free grammars has recently been advertised as a formalization more in accordance with Joshi's original intentions ((Kanazawa, 2009; Kuhlmann, 2007)). Both candidates have counterparts in the realm of tree languages and it is in this context that they are easily recognized to provide a mathematically precise framework for the characterization of two leading linguistic models, minimalist syntax and tree adjoining grammars (cf. (Harkema, 2001), (Michaelis, 2001), (Kepser and Rogers, 2007), (Mönnich, 1997; Mönnich, 2007)).

The tree languages in question are the multiple regular tree languages ((Raoult, 1997)) and the simple context-free tree languages ((Engelfriet and Maneth, 2000)). Both language families are proper subfamilies of the tree languages generated by context-free hyperedge-replacement graph grammars and the latter family is identical with the output languages of logical tree-to-tree transductions applied to regular tree languages. The obvious question that poses itself is whether the two restricted rule formats or their corresponding tree transducers, finite-copying top-down tree transducers and simple macro/attributed tree transducers, respectively, can be given an equivalent logical characterization in terms of restrictions on the logical formulas defining the relations in the target structures of logical transductions. This is indeed the case. A central result in a paper by Bloem and Engelfriet (2000) states that the tree languages which are the output of finite-copying top-down tree transducers applied to regular tree languages are exactly the output tree languages of logical tree transducers which are direction preserving in the sense that edges in the output trees correspond to directed paths in the input trees. As a first step towards an analogous result for simple context-free tree languages the main theorem of the paper shows that their monadic subclass, which provides a formalization of the extended version of classical tree adjoining grammars ($TAGs$), has indeed an easy logical characterization that puts them on the same footing as their multiple regular counterparts and thereby closes the gap that

Proceedings of the 11th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+11), pages 37–45,
Paris, September 2012.

has remained with respect to a model-theoretic description of $TAGs$.

The logical approach to the specification of language classes involves a lot of advantageous properties that have paved the way to its application to linguistic issues. Of particular importance in the present context is the restricted translational power of logical transductions. Monadic second-order definable tree translations are, by definition, of linear size increase. An output tree is at most k times as large as its input tree where k denotes the cardinality of the set of copy names. This bound on the copying power of transduction devices that are motivated by the model-theoretic idea of semantic interpretation is the main reason why the output languages of tree transductions definable in terms of monadic second-order logic satisfy in a particular perspicuous way the crucial criteria Joshi has suggested for the family of mildly context-sensitive languages.

One of our reviewers pointed out that there are recent attempts at relaxing the linearity condition of multiple context-free grammars and thus arriving at a larger class of languages for which it is claimed that they are still in accordance with the intuitions behind the notions of mild context-sensitivity. This extended class allows for a limited amount of copying (Cf.(Bourreau et al., To appear; Kallmeyer, 2010)). Whether such a formalization violates the criterion that poses a bound on cross-serial dependencies seems to be an open question. We favor a strong interpretation of this criterion and therefore are inclined to consider the realm of languages covered by logical translations as the the currently leading contender for an exact specification of Joshi's proposal.

There are two main sources that have influenced the ideas reported in this paper. Apart from the fundamental work on graph structure and monadic second-order logic due to Courcelle and Engelfriet (2012) we have to mention a previous attempt at giving a logical description of linear inside-out context-free tree languages (cf. (Kolb et al., 2000; Kolb et al., 2003)). This attempt relied on a particular technique of regularizing context-free tree grammars and does not lend itself to a treatment of arbitrary regular tree languages as input. The other principal source is provided by the characterization of tree transductions that are specifiable in monadic second-order logic in terms of attributed tree transducers with look-ahead. This result was established by Bloem and Engelfriet (2000) and constitutes together with our own earlier proof of the equivalence between simple context-free tree grammars and simple attributed tree transducers (Mönnich, 2010) the basis for the main result of the present paper.

We have been at pains to expound the central notions of this paper in an informal way. Our emphasis has been on motivating examples and connections with recent work on syntax-directed semantics. We hope not to have traded formal rigour for transparency in the proof sketches below.

2 Preliminaries

This section defines familiar notions from the theory of syntax-directed semantics together with its model-theoretic counterpart, the theory of monadic second-order transductions.

For any set A, A^* is the set of all strings over A. ε is the empty string, $|w|$ is the length of a string w. N denotes the set $\{0, 1, 2, 3, \dots\}$ of nonnegative integers.

A *single-sorted* or *ranked alphabet* is a finite set Σ given with a mapping $rank : \Sigma \to N$ (the *rank* mapping). We usually write $\Sigma^{(n)}$ to denote the (unique) set of operators of rank $n \in N$; we also write $\sigma^{(n)}$ to indicate that $rank(\sigma) = n$. The elements of $\Sigma^{(0)}$ are also called *constants*. The set of trees T_Σ is defined recursively as follows. Each constant of Σ, i.e., each symbol of rank 0, is a tree. If σ is of rank k and $t_1, \dots, t_k$ are trees, then $\sigma(t_1, \dots, t_k)$ is a tree. A *tree language* $L \subseteq T_\Sigma$ over Σ is a subset of T_Σ. With each tree $t \in T_\Sigma$ we can associate a string $s \in \Sigma^{(0)*}$ by reading the leaves of t from left to right. This string is called the *yield* of t, denoted by $yd(t)$. More formally, $yd(t) = t$ if $t \in \Sigma^{(0)}$, and $yd(t) = yd(t_1) \cdots yd(t_k)$ whenever $t = \sigma(t_1, \dots, t_k)$ with $k \geq 1$. The yield of tree language L is defined straightforwardly as $yd(L) = \{yd(t) | t \in L\}$.

If A is a set (of symbols) disjoint from Σ, then $T_\Sigma(A)$ (alternatively $T(\Sigma, A)$) denotes the set of trees $T_{\Sigma \cup A}$ where all elements of A are taken as constants. Let $X = \{x_1, x_2, x_3, \dots\}$ be a fixed denumerable set of *input variables* and $Y = \{y_1, y_2, y_3, \dots\}$ be a fixed denumerable set of *parameters*. Let $X_0 = Y_0 = \emptyset$ and, for $k \geq 1, X_k = \{x_1, \dots, x_k\} \subset X$, and $Y_k = \{y_1, \dots, y_k\} \subset Y$. For $k \geq 0, m \geq 0, t \in T_\Sigma(X_k)$, and $t_1, \dots, t_k \in T_\Sigma(X_m)$, we denote by

$t[t_1, \ldots, t_k]$ the result of *substituting* t_i for x_i in t. Note that $t[t_1, \ldots, t_k]$ is in $T_\Sigma(X_m)$. Note also that for $k = 0$, $t[t_1, \ldots, t_k] = t$.

Definition 1. *A* context-free tree *(CFT) grammar is a tuple $G = (\mathcal{F}, \Omega, S, P)$ where $\mathcal{F}$ and Ω are ranked alphabets of non-terminals and terminals, respectively, $S \in \mathcal{F}^{(0)}$ is the start symbol and P is a finite set of productions of the form*

$$F(y_1, \ldots, y_m) \to \xi$$

where $F \in \mathcal{F}$ and ξ is a tree over $\mathcal{F}$, Ω and Y_m.

If for every $F \in \mathcal{F}^{(m)}$ each $y \in Y_m$ occurs exactly once on the right-hand side of the corresponding rule then the context-free tree grammar is called *simple in the parameters (sp)*. The family of tree languages which is generated by context-free tree grammars which are simple in their parameters is designated as CFT_{sp}. Of particular interest to us is the situation where all the non-terminals in a simple context-free tree grammar are at most of arity 1. We call this class of grammars *monadic simple* context-free grammars $CFT_{mon,sp}$.

Attributed tree transducers are a variant of attribute grammars in which all attribute values are trees. Besides *meaning names* which transmit information in a top-down manner, attributed tree transducers contain explicit *context names* which allow information to be passed up from a node to its mother. Consequently, arbitrary tree walks can be realized by attributed tree transducers.

Definition 2. *An* attributed tree transducer *(ATT) is a tuple*

$$A = (Syn, Inh, \Sigma, \Omega, \alpha_m, R),$$

where Syn and Inh are disjoint alphabets of synthesized *and* inherited *attributes, respectively, Σ and Ω are ranked alphabets of* input *and* output *symbols, respectively, α_m is a synthesized attribute, and R is a finite set of rules of the following form: For every $\sigma \in \Sigma^{(m)}$, for every $(\gamma, \rho) \in ins_\sigma$ (the set of inside attributes of σ), there is exactly one rule in R_σ:*

$$(\gamma, \rho) \to \xi$$

where $\xi \in T_{\Omega \cup out_\sigma}$ and out_σ is the set of outside attributes of σ. Rules where ξ is (γ', ρ') are called copy rules.

Definition 3. *For every $\sigma \in \Sigma^{(m)}$, the set of* inside *attributes is the set $ins_\sigma = \{(\alpha, \pi) | \alpha \in Syn\} \cup \{(\beta, \pi i) | \beta \in Inh, i \leq m\}$ and the set of* outside *attributes is the set $out_\sigma = \{(\beta, \pi) | \beta \in Inh\} \cup \{(\alpha, \pi i) | \alpha \in Syn, i \leq m\}$. π and ρ are path variables ranging over node occurrences in the input tree.*

$ATTs$ with rules R_σ at an input symbol σ in which each outside attribute occurs exactly once are called *simple* attributed tree transducers. We denote this class by $ATT_{ss,si}$.

The *dependencies* between attribute occurrences in an input tree s can be represented with the help of R_σ. An instance of an attribute occurrence (α', π') depends on another occurrence (α, π) if σ labels node u in s, R_σ contains the rule $(\alpha', \pi') \to \xi$ and (α, π) labels one of the leaves in ξ. The *dependency graph* $D(s)$ of an input tree $s \in T_\Sigma$ consists of the set of attribute occurrences together with the dependencies according to the rules in R. Reversing the direction of these dependencies leads to the notion of a *semantic graph* $S(s)$ of an input tree $s \in T_\Sigma$.

An attributed tree transducer is *noncircular* if the paths of attribute dependencies are noncircular. It is well known that noncircular ATTs have unique *decorations dec*, functions which assign each attribute occurrence a tree over $\Omega \cup out_\sigma$ in accordance with the productions R_σ.

Definition 4. *The* transduction *realized by a noncircular attributed tree transducer A is the function*

$$\tau_A = \{(s, t) | s \in T_\sigma, t \in T_\Omega, t = dec_s(\alpha_m, \epsilon)\}$$

Declarative tree transductions are inspired by the model-theoretic technique of semantic interpretation (Rabin, 1965). The idea is to define a relational structure inside another structure in terms of monadic second-order formulas. Both the input and the output structures are finite trees regarded as finite models.

The language to be used for the specification of properties and relations satisfied by finite tree structures is a straightforward extension of first-order logic: monadic second-order logic (MSO). The language of this logic contains variables that range over subsets of the universe of discourse and quantifiers that bind these (monadic) predicate variables.

Given a ranked signature Σ the monadic second-order language over trees in T_Σ uses

atomic formulas $lab_\sigma(x)$ ($\sigma \in \Sigma$), $child_i(x, y)$, $x = y$ and $x \in X$ to convey the idea that node x has label σ, that node y is the i-th child of node x, that x and y are the same node and that node x is a member of the set of nodes X.

Besides this extension of the classical first-order logic the concept of a monadic second-order definable tree transducer ($MSOTT$) differs in two further aspects from the method of semantic interpretation as originally introduced by Rabin. First, an MSO formula ϕ serves to define the domain of the transducer. The second modification of the original method of semantic interpretation provides for a fixed number k of disjoint copies of the input tree. It is inside these disjoint copies that the output tree is to be defined.

Definition 5. *Given two ranked alphabets Σ and Ω and a finite set C of copy names, a* monadic second-order definable *tree transducer T from T_Σ to T_Ω is specified by the following formulas of the monadic second-order language over Σ:*

(i) a closed formula φ, the domain *formula*

(ii) formulas $\nu_c(x)$ with $c \in C$, the node *formulas*

(iii) formulas $\psi_{\delta,c}(x)$ with $c \in C$ and $\delta \in \Omega$, the labelling *formulas*

(iv) formulas $\chi_{i,c,d}(x, y)$ with $c, d \in C$ and $i \leq$ maximal arity of symbols in Ω, the edge *formulas*

In sharp contrast with the syntax-directed transformation devices a logic based tree transducer T does not translate its input trees in a recursive top-down manner. The translation τ_T realized by such a declarative transducer has to be defined in terms of the familiar ingredients of a relational structure.

Definition 6. *The* tree translation τ_T *realized by a monadic second-order definable tree transducer T from T_Σ to T_Ω is a partial function $\tau_T : T_\Sigma \to T_\Omega$ defined as follows. The domain of τ_T is $\{s \in T_\Sigma \mid s \models \varphi\}$. For every $s \in T_\Sigma$ in its domain $\tau_T(s)$ is the tree structure $t \in T_\Omega$ such that:*

$D_t = \{(c, x) \in C \times D_s \mid s \models \nu_c(x)\}$
is the tree domain of t

$E_t = \{((c, x), i, (d, y)) \in D_t \times ar(\Omega) \times D_t \mid s \models \chi_{i,c,d}(x, y)\}$ *is the edge relation of t where $ar(\Omega)$ denotes the rank of Ω*

$L_t = \{((c, x), \delta) \in D_t \times \Omega \mid s \models \psi_{c,\delta}(x)\}$
is the labeling function of t

Logic based transducers are called *relabeling* if they just relabel the nodes of an input tree. Of particular interest regarding the logical analysis of monadic simple context-free tree grammars are logic based tree transducers that preserve or reverse the direction of the paths in the input tree in their definitions of edges of output trees. This family of tree transducers is designated by $MSOTT_{dir,rev}$. We depart slightly from this definition in allowing defining upwards paths between a leaf node and the daughter of a dominating branching node. In this situation we speak of a *slight modification* of $MSOTT_{dir,rev}$.

3 From $CFT_{mon,sp}$ to $1S, 1I - ATT_{ss,si}$

The proof of the logical characterization of monadic simple context-free tree languages is based on a procedural characterization of simple context-free tree languages as the output languages of simple attributed tree transducers with one synthesized attribute only (Cf. Mönnich, 2010). The simulation of attributed tree transducers by monadic second-order tree transducers along the lines of Bloem and Engelfriet (2000) then leads to the logical characterization of the (formal representation of) extended $TAGs$ in terms of extremely simple edge definitions on the input trees.

The translation below of a given $CFT_{mon,sp}\, G$ into an equivalent $ATT_{sp}\, A$ is inspired by the proofs of Lemma 5.11 in (Fülöp and Vogler, 1998) and of Lemma 6.1 in (Engelfriet and Maneth, 1999).

Example 1. *Consider the $CFT_{mon,sp}\, G = \langle\{S, S', \overline{S}, E, \overline{a}, \overline{b}, \overline{c}, \overline{d}\}, \{a, b, c, d, \varepsilon, S_t, S_t^0\}, S', P\rangle$ with P given as follows:*

$$
\begin{array}{ll}
S' \longrightarrow S_t(\overline{a}, S(\overline{S}(E)), \overline{d}) & \overline{a} \longrightarrow a \\
S(y) \longrightarrow S_t(\overline{a}, S(\overline{S}(y)), \overline{d}) & \overline{b} \longrightarrow b \\
S(y) \longrightarrow S_t^0(y) & \overline{c} \longrightarrow c \\
\overline{S}(y) \longrightarrow S_t(\overline{b}, y, \overline{c}) & \overline{d} \longrightarrow d \\
 & E \longrightarrow \varepsilon
\end{array}
$$

This grammar generates the language $L = \{a^n b^n, c^n, d^n\}$. A derivation of the string $aabbccdd$ is shown in figure 1. We simplified the presentation in the sense that the last step involves the simultaneous application of several expansion rules.

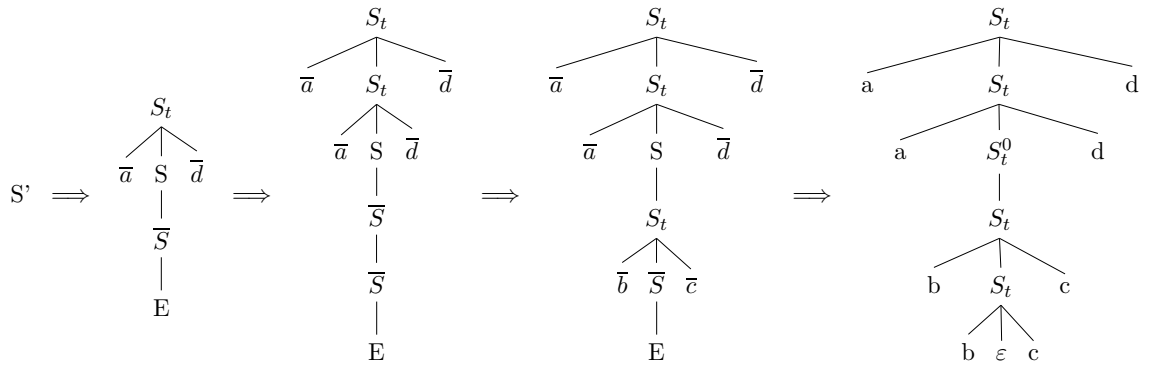

Figure 1: Derivation of yield *aabbccdd*

.

Inspecting the rules of the example grammar it turns out that they exhibit a particular normal form with a terminal symbol as head and a possibly empty string of non-terminals. This is not an accidental feature of our grammar, but a general characteristic of monadic simple context-free tree grammars. Context-free tree grammars in which the root of the right-hand side of each rule is labelled with a terminal symbol are in *Greibach* normal form. It is well known that there are context-free tree languages that do not admit grammars in Greibach normal form. In the case of tree languages generated by monadic simple context-free tree grammars, however, the classical proof for context-free word grammars goes through without any modification because of the isomorphism between strings and monadic trees. The preceding considerations lead to the following lemma:

Lemma 1. *For any monadic context-free tree grammar G, there is a monadic context-free tree grammar G' in Greibach normal form such that*

$$L(G') = L(G)$$

Given these preparations we arrive at the following translation procedure from monadic simple context-free tree grammars to simple attributed tree transducers.

Lemma 2. *For every $CFT_{mon,sp}$ G, there is an $1S, 1I - ATT_{ss,si}$ A_G that outputs the same language when applied to the derivation trees of G.*

Proof For a given $CFG_{mon,sp}$ $G = (\mathcal{F}, \Omega, S, P)$ an $1S, 1I - ATT_{ss,si} = (Syn, Inh, \Sigma, \Omega, \alpha_m, R)$ A_G that outputs the same language is defined from the rules of G in the following way.

- $Syn = \{0, 1\}$ with $\alpha = \alpha_m$ at the root node
- $Inh = \{y\}$
- Every symbol in the derivation trees is assigned one synthesized attribute.
- If $p : N \to \xi$ is an element of P then R'_p is specified for both the synthesized and inherited attributes by structural induction on the right-hand side ξ:

  $$(q, \pi) \to \vartheta(\xi),$$

  where $q = 0$ or $q = 1$ depending on the arity of N and ϑ substitutes a non-terminal M in ξ by $(q, \pi i)$ if the non-terminal M occurs in the i-th non-terminal position in ξ and y by (y, π)

  $$(y_j, \pi_i) \to \vartheta'(\xi'),$$

  where ξ' occurs in the argument position of some non-terminal L in ξ that itself occupies the i-th non-terminal position on the right-hand side of p and ϑ' is identical with ϑ except for erasing every y in ξ'.

⊣

Example 2. *Applying the construction just outlined to the context-free tree grammar of the last example we obtain the following attributed tree transducer $A = (Syn, Inh, \Sigma, \Omega, q_0, R')$:*

- $Syn = \{0, 1\}$
- $Inh = \{y\}$
- $\Sigma = \{p_0, \ldots, p_9\}$
- $\Omega = \{a, b, c, d, \varepsilon, S_t, S_t^0\}$

- $q_0 = 0$
- $R' = \bigcup_{p_i} R'_{p_i}$

$$
\begin{aligned}
R'_{p_1} &= \{(0,\pi) \to S_t((0,\pi 1),(1,\pi 2),(0,\pi 4)),\\
&\qquad (y,\pi 2) \to (1,\pi 3)\}\\
R'_{p_2} &= \{(1,\pi) \to S_t((0,\pi 1),(1,\pi 2),(0,\pi 4)),\\
&\qquad (y,\pi 2) \to (1,\pi 3),\\
&\qquad (y,\pi 3) \to (y,\pi)\}\\
R'_{p_3} &= \{(1,\pi) \to S_t^0((y,\pi))\}\\
R'_{p_4} &= \{|1,\pi) \to S_t((0,\pi 1),(y,\pi)(0,\pi 2))\}\\
R'_{p_5} &= \{(0,\pi) \to a\}\\
R'_{p_6} &= \{(0,\pi) \to b\}\\
R'_{p_7} &= \{(0,\pi) \to c\}\\
R'_{p_8} &= \{(0,\pi) \to d\}\\
R'_{p_9} &= \{(1,\pi) \to \varepsilon\}
\end{aligned}
$$

Given the constructed attributed tree transducer A that is equivalent to the previously considered monadic simple context-free tree grammar G we can now repeat the example derivation displayed in figure 1. We follow the conventional graphical representation for drawing attributed derivation trees together with the dependencies obtaining between synthesized and inherited attributes. Occurrences of synthesized and inherited attributes in conjunction with their tree values appear to the right and left, respectively, of the labelled nodes of the input tree. Dependencies among the attribute occurrences are indicated by arrows. Since dependency graphs indicate the connection between an attribute leaf and the tree which is to be substituted for it by building the output tree bottom-up we will depart from the conventional graphical representation in this respect and adopt instead the tradition of semantic graphs which construct the output tree top-down and therefore are direction preserving as far as relations between attribute values are concerned.

Under the stated conventions the graphical representation of the information transport in terms of the constructed attributed tree transducer A that corresponds to the example derivation of figure 1 looks as shown in figure 2. We have again simplified the presentation in the sense that the application of the "barred" rules is contracted into one single step.

4 Equivalence of $CFT_{mon,sp}$ with (a slight modification of) Non-Copying $MSOTT_{dir,rev}$

It was mentioned above that attributed tree transducers are attribute grammars with all their attribute values restricted to trees and their semantic functions to substitution of trees for dependent leaves. Second–order substitution for internal nodes of trees is achieved through the upward information transport that is made possible by the inherited attributes. An analysis of the paths in the semantic dependency graph of the attributed tree transducer that corresponds to a monadic simple context-free tree grammar in *Greibach* normal form reveals that these paths are either direction preserving as in the case of minimalist grammars or direction reversing.

Lemma 3. *For every $CFT_{mon,sp}$ G there is (a slight modification of) an equivalent non-copying reduced $MSOTT_{dir,rev}$ T.*

Proof (Sketch) The main idea of the proof is a careful case analysis of the translation procedure that produces an equivalent attributed tree transducer from a given monadic simple context-free tree grammar G. Inspection of this translation procedure in the proof of Lemma 2 reveals the following types of right-hand sides in the rules of a monadic simple context-free tree grammar. W.l.g we consider only terminals with at most arity one:

- $F \longrightarrow a$ The synthetic meaning attribute gets the value a at the node corresponding to the application of this rule.
- $F \longrightarrow a(N)$ The synthetic meaning attribute gets the value $a(\alpha,\pi)$ establishing a dependency on the synthetic value of the daughter.
- $F \longrightarrow a(N(M))$ In addition to the previous case a further dependency is established on the value of the sibling node corresponding to the application of a production with left-hand side M. This dependency is mediated by a copy rule at the inherited context attribute of the node corresponding to an application of a production with left-hand side N.

Iteration of the second case leads to further top-down semantic dependencies. If monadic instead

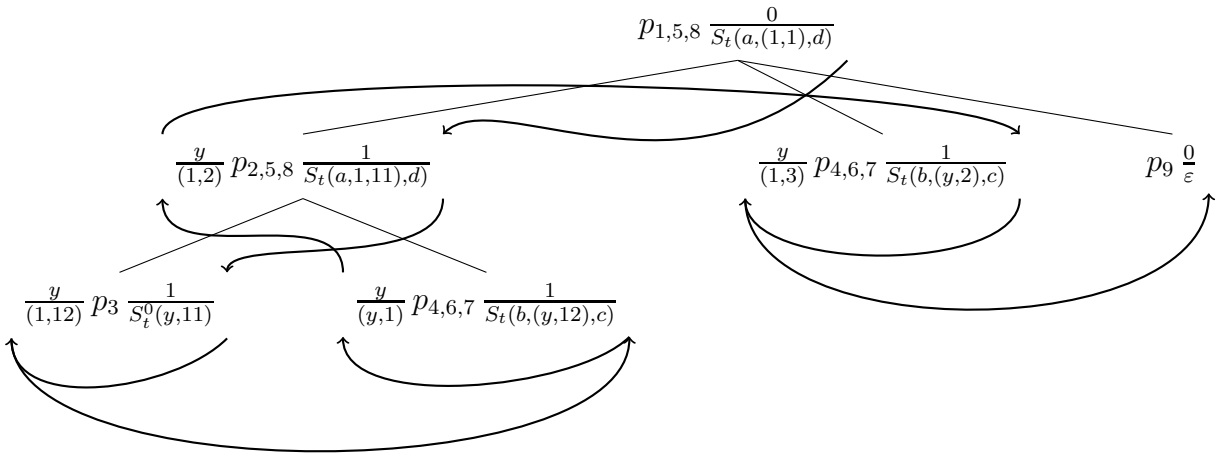

Figure 2: Attributed derivation tree with semantic dependency relations

of nullary non-terminals $F(y)$ are rewritten the parameter y mediates an one-step upwards information transport by means of inherited copy rules.

From this case analysis it follows immediately that all inherited rules are copy rules whereas all synthetic rules add material to the output tree. Furthermore, all upwards dependencies are mediated by an inherited rule establishing a dependency on the synthetic value of a sibling node.

Based on this intermediate simple ATT we can now define the equivalent MSO tree transducer T by specifying the edge formulas $\chi_{1,\gamma,\gamma'}(x,y)$ that represent the dependencies between the attributes and the node formulas $\psi_{d,\gamma}(x)$ that define the labels of the output tree. We assume again for simplicity that the terminals of the given grammar G are of arity at most one and that the copy rules introduce a new output symbol id. The set of copy nodes consists of the synthetic and the inherited attribute. The edge formulas mirror the information transport of rules of the form

$$(*) \qquad (\gamma,\rho) \to a(\gamma',\rho')$$

where (γ,ρ) is an inside, (γ',ρ') an outside attribute and a an output symbol or the new symbol id. Such an edge formula $\chi_{1,\gamma,\gamma'}(x,y)$ is the disjunction of all formulas $\exists z(lab_\sigma(z) \wedge edge_j(z,x) \vee edge_{j'}(z,y))$ for all input symbols σ with j,j' equal to 0 or 1 and $(\gamma,\pi j)$ depending on $(\gamma',\pi j')$ in R_σ. In the situation where $j,j'=0$ $edge_0(x,y)$ is shorthand for $x=y$.

The node formula $\psi_{d,\gamma}(x)$ is the disjunction of all formulas $\exists z(lab_\sigma(z) \wedge edge_j(z,x))$ for all input symbols σ where the same stipulations hold as for the edge formulas and R_σ contains the rule (*).

This easy transfer of the translation technique developed by Bloem and Engelfriet (2000) to the context of simple tree transducers with only one synthesised and one inherited attribute reveals immediately that the defined monadic second-order tree translation fulfills the condition of being *direction preserving/reversing* apart from the information flow to and fro between the the synthetic and inherited copies of the input trees.

Bloem and Engelfriet show how to prune all occurrences of these transitions. Since this makes the "inherited" copy of the input tree superfluous we can erase it completely and arrive thus at a *reduced* non-copying MSO-transduction in the sense that every node is the head or tail of an edge. The pruning step has introduced a slight modification of the upward paths by linking some leafs to the first daughter of a dominating branching node. $\dashv$

Applying the construction just outlined to the information transport illustrated by figure 2 we obtain the defined edges in the output of an MSO-transduction shown in figure 3.

Lemma 4. *For every (slight modification of a) non-copying reduced $MSOTT_{dir,rev}\,T$ there is an equivalent $CFT_{mon,sp}\,G_T$.*

Proof (Sketch) We adapt again to the present situation the method of proof developed by Bloem and Engelfriet (2000). We assume with them that the root of the defined output tree of a given direction preserving/reversing MSO tree transducer T is identical to the root of the input tree. We as-

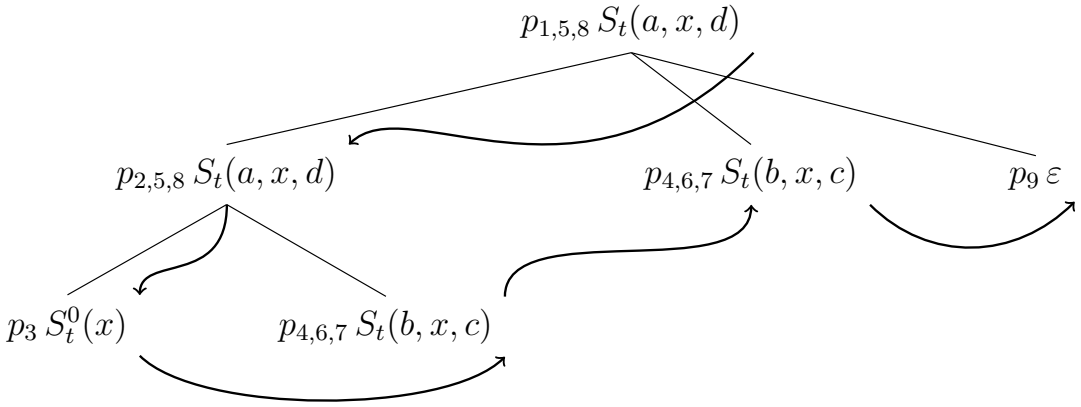

Figure 3: MSO-defined edges

sume furthermore that edges originate or end at every node of the input tree. For nodes occurring between endpoints of edges that correspond to direction preserving paths this assumption could be avoided by eliminating spurious rules like $N \longrightarrow M$ from the equivalent context-free tree grammar. Finally we observe that the upward edges of the output tree may still be established between nodes that are not immediate neighbours.

This non-local feature can be excluded by applying the inverse of the pruning action of the previous lemma. If there is a non-local upward edge from the leaf of the input tree to a sibling of the first daughter of a dominating branching node then we define a local path on an "inherited" copy of the input tree that connects the endpoints of the original non-local upward edge.

The specification of the equivalent ATT relies on information about the configuration of the (local) edges in its immediate neighbourhood. This information can be stored in the node labels by means of a relabeling $MSOTT$ T' that extends each label by a vector of truth values of the formulas for top-down and bottom-up edges and for edges between nodes and their twins in the other copy. To illustrate the simplest case for a unary output symbol a and a "synthesized" copy name γ : If the formula $\psi_{a,\gamma}(x)$ is true at a node with extended label σ' and the only true edge formula is $\chi_{1,\gamma,\gamma}(x,y)$ then $R_{\sigma'}$ contains just $(\gamma,\rho) \to a(\gamma,\rho 1)$.

From this intermediate attributed tree transducer the equivalent monadic simple context-free tree grammar G_T is then defined as follows where we use the notational conventions introduced in the preliminary section (cf. (Mönnich, 2010)):

- $\mathcal{F} = \Sigma$ where the arity of non-terminals is either zero or one depending on the occurrence of an inherited attribute assigned to them in the input tree.
- $\Omega = \Omega$
- $S = \{\sigma^{(0)}\}$ with $\sigma \in \Sigma$ labeling the root of an input tree.
- For every $\sigma \in \Sigma$ we construct a rule

  $$\sigma(y_1,\ldots,y_n) \to t$$

  where $t = COMP(\xi)$ and ξ is the right-hand side of the only synthesized attribute α in R_σ. The right-hand sides of rules in R_σ are designated by $rhs(\gamma\pi,\sigma)$ in the following:

  (i) If $\xi = \alpha\pi i$ then

  $$\sigma(y_1,\ldots,y_n) \to t$$

  where $t = COMP(\xi)$ and ξ is the right-hand side of the only synthesized attribute α in R_σ. The right-hand sides of rules in R_σ are designated by $rhs(\gamma\pi,\sigma)$ in the following:

  (i) If $\xi = \alpha\pi i$ then

  $$COMP(\xi) = \rho(t)$$

  where ρ labels the ith daughter of σ and

  $$t = COMP(rhs(\beta\pi i,\sigma))$$

  (ii) If $\xi = \beta\pi$ then

  $$COMP(\xi) = y$$

(iii) If $\xi = f(\xi_1, \ldots, \xi_r)$ for $f \in \Omega^{(r)}$

$$\text{COMP}(\xi) = f(COMP(\xi_1), \ldots, COMP(\xi_r))$$

By a routine inspection it is easy to verify that the resulting grammar G_A is indeed simple and that it generates exactly the output language of T. $\dashv$

By lemmas 4 and 4 we obtain our main result.

Theorem 1. *The monadic simple context-free tree languages are exactly the output languages (of a slight modification) of non-copying direction preserving/reversing MSO definable tree transductions.*

5 Envoi

The question of how to extend the logical characterization given in the present paper to the full class of simple context-free languages or to the family of well-nested tree languages, for that matter, is still open. Our conjecture is that a similar characterization holds for these languages. The main step towards such a result would consist in establishing a Greibach normal form for this larger class of languages.

References

Roderick Bloem and Joost Engelfriet. 2000. A Comparison of Tree Transductions Defined by Monadic Second-Order Logic and by Attribute Grammars. *J. Comp. System Sci.*, 61:1–50.

Pierre Bourreau, Sylvain Salvati, and Laura Kallmeyer. To appear. On io-copying and mildly context-sensitive formalisms. In *Proceedings of Formal Grammar 2012. Lecture Notes in Computer Science.*

Bruno Courcelle and Joost Engelfriet. 2012. *Graph Structure and Monadic Second-Order Logic. A Language-Theoretic Approach.* Cambridge University Press.

Joost Engelfriet and Sebastian Maneth. 1999. Macro Tree Transducers, Attribute Grammars, and MSO Definable Tree Translations. *Information and Computation*, 154:34–91.

Joost Engelfriet and Sebastian Maneth. 2000. Tree Languages Generated by Context-Free Graph Grammars. In Hartmut Ehrig et al., editor, *Graph Transformation*, number 1764 in LNCS, pages 15–29. Springer.

Zoltán Fülöp and Heiko Vogler. 1998. *Syntax-Directed Semantics - Formal Models Based on Tree Transducers*. Springer, New York and Berlin.

Hendrik Harkema. 2001. *Parsing Minimalist Languages*. Ph.D. thesis, University of California at Los Angeles.

Laura Kallmeyer. 2010. On mildly-context-sensitive non-linear rewriting. *Research on Language and Computation*, 8:341–363.

Makoto Kanazawa. 2009. The convergence of well-nested mildly context-sensitive grammar formalisms. In *14th Conference on Formal Grammar*. Slides available at http://research.nii.jp/~kanazawa/.

Stephan Kepser and James Rogers. 2007. The equivalence of tree adjoining grammars and monadic linear context-free tree grammars. In M. Kracht, G. Penn, and E. Stabler, editors, *Mathematics of Language 10*.

Hans-Peter Kolb, Uwe Mönnich, and Frank Morawietz. 2000. Descriptions of cross-serial dependencies. *Grammars*, 3(2/3):189–216.

Hans-Peter Kolb, Jens Michaelis, Uwe Mönnich, and Frank Morawietz. 2003. An operational and denotational approach to non-context-freeness. *Theoretical Computer Science*, 293:261–289.

Marco Kuhlmann. 2007. *Dependency Structures and and Lexicalized Grammars.* Ph.D. thesis, Saarland University.

Jens Michaelis. 2001. *On Formal Properties of Minimalist Grammar*, volume 13 of *Linguistics in Potsdam*. Universität Potsdam.

Uwe Mönnich. 1997. Adjunction as substitution. In G-J. M. Kruijff, G. Morill, and R. Oehrle, editors, *Formal Grammar '97*, pages 169–178.

Uwe Mönnich. 2007. Minimalist syntax, multiple regular tree grammars and direction preserving tree transductions. In J. Rogers and S. Kepser, editors, *Proceedings Model Theoretic Syntax at 10*.

Uwe Mönnich. 2010. Well-nested tree languages and attributed tree transducers. In Srinivas Bangalore, Robert Frank, and Maribel Romero, editors, *TAG+10. Proceedings of the 10th International Workshop on Tree Adjoining Grammars and Related Formalims.*

Michael O. Rabin. 1965. A simple method for undecidability proofs and some applications. In Y. Bar-Hillel, editor, *Logic Methodology and Philosophy of Science II*, pages 58–68. North-Holland, Amsterdam.

Jean-Claude Raoult. 1997. Rational Tree Relations. *Bull. Belg. Math. Soc.*, 4:149–176.

William C. Rounds. 1970. Tree-oriented proofs of some theorems on context-free and indexed languages. In *2nd Annual ACM Symposium on Theory of Computing*, pages 109–116.