Chapter 10

Minimalist Syntax, Multiple Regular Tree Grammars and Direction Preserving Tree Transductions[1]

Uwe Mönnich
Linguistics Department, University of Tübingen
Tübingen, Germany
um@sfs.uni-tuebingen.de

10.1 Introduction

Model-theoretic syntax deals with the logical characterization of complexity classes. The first results in this area were obtained in the early and late Sixties of the last century. In these results it was established that languages recognised by finite string and tree automata are definable by means of monadic second-order logic (MSO).

To be slightly more precise, the classical results, just mentioned, can be seen as providing translation procedures that relate logical specifications with finite automata equivalent with respect to the defined language classes. Along this way, Büchi (1960) and Elgot (1961) have shown that regular string languages represented through finite (string) automata can be expressed by sentences in the (weak) MSO logic with one successor. For tree languages an analogous result is well known: a tree language is definable in weak MSO logic with multiple successors if and only if it is recognizable by a finite tree automaton (Thatcher and Wright 1968; Doner 1970).

All these approaches suffer from a lack of expressive power in that the family of regular tree languages properly includes all other language families that are captured by the logical formalisms that have been considered in model-theoretic syntax. It is due to this lack of expressive power that grammatical phenomena like cross-serial dependencies in languages like Swiss German or Bambara are beyond the reach of the kind of logical apparatus currently applied to natural language syntax.

We shall therefore propose a hybrid solution to the problem of how to account for mildly context-sensitive phenomena with the help of tree logic. The limited expressive power of this logic in its original set-up makes it impossible to formulate the solution in a way that would deal directly with the problematic phenomena, but we can give these phenomena a slightly different appearance whereby they do become regular and as such definable in tree logic.

The paper is structured as follows. Section 10.2 recalls basic notions from logic, tree grammars and tree translation we need for our discussion in the rest of the paper. In Section 3 we prove that minimalist syntax is equivalent to direction preserving MSO transductions. We conclude with some general remarks and point out some open problems.

[1]We are grateful to a number of people who have helped us understand the connections between syntax-directed semantics and logic based transductions, including: Tom Cornell, Hap Kolb, Stephan Kepser, Jens Michaelis and Frank Morawietz. This work was partially supported by the German Research Council through a grant to the Collaborative Research Center 441 at Tübingen University.

10.2 Preliminaries

The introductory section has tried to motivate the method of semantic interpretation and to explain its application to the theory of natural syntax. 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. We assume that the reader has seen an exposition of the basic concepts of universal algebra.

10.2.1 Syntax-Directed Semantics

Macro tree transducers (MTT) are a model of tree transformation that transduces in a recursive top-down fashion an input tree into an output tree, handling context information in an implicit way. The elements of context information do not have explicit names, but are passed along as parameters of the states in this kind of translation device.

Definition 10.2.1. *A* macro tree transducer *is a tuple* $M = (Q, \Sigma, \Omega, q_0, R)$, *where* Q *is a ranked alphabet of* states, Σ *and* Ω *are ranked alphabets of* input *and* output *symbols, respectively,* $q_0 \in Q_0$ *is the* initial *state and* R *is a set of* rules *of the following form:*

$$(q, \sigma(x_1, \ldots, x_m))(y_1, \ldots, y_n) \rightarrow \xi$$

where $q \in Q^{(n)}$, $\sigma \in \Sigma^{(m)}$ *and* $\xi \in T_{\langle Q, X_m \rangle \cup \Omega}(Y_n)$.

Remark 10.2.2. If every state in Q has rank zero, then M is a *top-down transducer (TOP)*. Macro tree transducers

can therefore be regarded as a context-sensitive extension of top-down transducers.

Macro tree transducers can realize translations that are of double exponential size increase. A subtree of an input tree s can be processed arbitrarily many times by a macro tree transducer M depending on the number of occurrences of an input variable x_i in the right-hand side of a rule of M that rewrites the mother of the particular subtree in a particular state. Restricting the contribution to an output tree that is provided by this copying power of macro tree transducers leads to the notion of *finite-copying* macro tree transducers.

Definition 10.2.3. *Let M be a macro tree transducers with input alphabet Σ that is simple in the parameters. If there is a number $k \in \mathbb{N}$ such that for every input $s \in T_\Sigma$ and node u of s the length of the state sequence of s at node u $|sts_M(s,u)| \leq k$, then M is finite-copying (fc).*

Definition 10.2.4. *Disregarding the input of a macro tree transducer one obtains a context-free tree (CFT) grammar. A CFT grammar is a tuple $G = (\mathcal{F}, \Omega, S, P)$ where $\mathcal{F}$ and Ω are ranked alphabets of nonterminals and terminals, respectively, $S \in \mathcal{F}_0$ is the start symbol and P is a finite set of productions of the form*

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

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

The family of tree languages which is generated by context-free tree grammars which are simple in their parameters is designated as CFT_{sp}.

A grammar $G = \mathcal{F}, \Omega, S, P)$ is called a *regular tree (REGT)* grammar if $\mathcal{F} = \mathcal{F}^{(0)}$, i.e., if all nonterminals are of rank 0.

A further grammatical formalism is defined for the generation of tree tuples. This is an extension of the notion of regular tree grammar, i.e., all the nonterminals are of rank 0, but they range over tuples of trees instead of single trees only.

Definition 10.2.5. *A grammar $G = (\mathcal{F}, \Omega, S, P)$ is called a* multiple regular tree *(MREGT) grammar if $\mathcal{F} = \mathcal{F}^{(0)}$, i.e., if all nonterminals are of rank 0, each nonterminal has assigned a tuple index ≥ 1, the start symbol S has tuple index 1 and the productions are of the form*

$$F \rightarrow (\xi_1, \ldots, \xi_n)$$

where n is the tuple index of F, $\xi_i (1 \leq i \leq n)$ is a tree over $\mathcal{F} \times \{1, \ldots, m\}$ and Ω. It is assumed that m is the maximal tuple index of a nonterminal in $\mathcal{F}$ and that each component $\langle F, k \rangle$ of a nonterminal leaf label occurs exactly once in the ξ_i.

The special case in which the right-hand sides of the productions are tuples of words over $\mathcal{F}$ and Ω is referred to a s a *multiple context-free* (MCF) grammar.

10.2.2 Semantic Interpretations

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 definitional power of monadic second-order tree transducers is highly restricted. The output string languages of these tree transducers defined over regular tree families are mildly context-sensitive (Engelfriet and Heyker, 1991), as will be discussed in the next section.

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.

Definition 10.2.6. *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 10.2.7. *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 $\tau_T : T_\Sigma \rightarrow T_\Omega$ defined as follows. The domain of τ_T is $\{s \in T_\Sigma \mid s \models \varphi\}$. For every $s \in T_\Sigma$ in the domain of τ_T $\tau_T(s)$ 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,
$E_t = \left\{ \begin{array}{l} ((c,x),i,(d,y)) \in D_t \times ar(\Omega) \times D_t \mid \\ s \models \chi_{i,c,d}(x,y) \end{array} \right\}$
is the edge relation of t,
where $ar(\Omega)$ *denotes the arity of* Ω,
$L_t = \{((c,x),\delta) \in D_t \times \Omega \mid s \models \psi_{c,\delta}(x)\}$
is the labelling function of t.

10.3 Minimalist Syntax

The syntactic theory in question is the version of minimalism as presented by Stabler (1997). Minimalist grammars reflect the transition within the transformational tradition from an approach which relies on an explicit formulation of well-formedness conditions on derivations and representations to a procedural specification of the derivational process itself. In this procedural specification the structure building operations are assumed to be determined by the syntactic features of the structures that form the arguments of these operations. Given this operational orientation of feature-driven minimalist syntax it may already be surprising that it allows for a logical specification by means of a particular form of semantic interpretation.

In this section we will show that minimalist syntax can be described in terms multiple regular grammars and, based on this description, sketch its relation with the subclass of direction preserving semantic interpretation where we rely on previously established results on the close connection between multiple regular grammars, top-down tree transducers and direction preserving logic based tree translations. For reasons of space we suppress a formal exposition of Minimalist Grammar (MG) along the lines of Michaelis et al. (2001).

The main insight behind the construction by Michaelis (2001a) of an equivalent multiple context-free string grammar for a given minimalist grammar consists in the realization that it is possible to code the tree structure relevant for the application of the operations of *merge* and *move* by means of nonterminal symbols of the resulting target grammar. These symbols range over yields of tuples of the relevant subtrees. Our proof omits the yield step and retains the tuples of subtrees. What remains to be done is the verification that the action of *merge* and *move* can be simulated by appropriate tree rearrangements that are permitted within the framework of multiple regular tree grammars.

The particular details of the relevant structural aspects that go into the coding of the nonterminals of the multiple regular tree grammar to be constructed are a direct consequence of the definition of the two operations *merge* and *move*. Occurrences of selection features and of their corresponding syntactic features can only be deleted by an application of the operation *merge* if they form the start feature of the head-label of some expression. Besides this structural information the nonterminal has to comprise the additional information as to whether the head with the starting selection feature is part of a complex tree or not, because this has a direct effect on the ordering of the expression resulting from an application of *merge*. In order to be a candidate for the operation *move* an expression has to display a head-label whose starting feature is a licensor. In addition, the expression has to contain a subtree which is a maximal projection and whose head-label starts with a matching licensee feature. This subtree of the expression has to be the only subtree fulfilling the condition of being a maximal projection with a particular licensee feature as the start category of its head-label.

Summarizing the conditions on a nonterminal of the resulting multiple regular tree grammar that collectively succeed in coding all structural aspects decisive for the application of one of the operations *merge* or *move* we are led to the following specifications. Assume that the given minimalist grammar G contains a set of n licensees $(-l_i)_{l \leq i < n}$. Each nonterminal of the resulting multiple tree grammar $G' = (N, \Sigma, P, F, S)$ is then represented by an $n+2$-tuple, where the first component is a suffix of one of the lexically given strings of syntactic categories, except those suffixes that start with one of the licensees $-l_i$ which form the next n components, the last component consisting of the feature *simple* or *complex*. This set of nonterminals N is certainly finite since it is constructed as a finite product of finite sets.

Theorem 10.3.1. *For every minimalist grammar G, there exists a strongly equivalent multiple regular grammar G'.*

In the last paragraphs we have indicated that minimalist grammars are captured by a slight extension of regular tree grammars. This extension employs productions in which the right-hand sides are a "leaf-linked" forest, i.e., finite tuples of trees with some of the leaves connected by means of secondary relations. This type of productions is one of the two ways considered in formal language theory of defining special subclasses of rules in context-free hyperedge replacement grammars with the purpose of limiting the generated graph languages to families of trees. The other subclass is characterized by the restriction that the right-hand sides of these particular expansion rules have to be just trees.

In the remaining part of this section we will try to fulfil our promises regarding minimalist syntax. Fortunately, the characterization of minimalist grammars in terms of multiple regular tree grammars furnishes the missing link in the chain of known equivalences leading from natural language syntax to model-theoretic interpretation via a link provided by automata-theoretic translation. We will first recall the relationship between multiple regular tree grammars and top-down tree transducers and then conclude this part of our discussion by providing a declarative definition of tree translations achieved through top-down transducers.

Theorem 10.3.2 (Raoult, 1997). *Multiple regular tree languages are the same as the output of finite-copying top-down tree transducers.*

Raoult (1997) presents a detailed verification that the construction illustrated by means of our current example provides for any given multiple regular tree grammar a strongly equivalent finite-copying top-down tree transducer. He shows furthermore that an analogous result holds in the other direction. It is possible to specify an equivalent multiple regular tree grammar for any given finite-copying top-down tree transducer. Since our principal objective is to give a purely model-theoretic account of a grammatical framework that derives its inspiration from the basic idea that constituents move we will omit a discussion of this result.

Up to this moment we have merely moved from one system of formal language theory to the next one. It is now that we finally proceed to the logical model of tree transductions that we have to face the question what sort of logical definitions are to account for the tree transformations performed by the structure building operations *merge* and *move*. Implicitly, we have already answered this question with the move to the model of top-down tree transducers, as we will explain.

It has been known for some time that top-down tree transducers coincide with attributed tree transducers with synthesized attributes only (Courcelle and Franchi-Zannettacci, 1982a,b). This correspondence still holds if one considers the restricted case of the output of finite-copying top-down tree transducers and single use attributed tree transducers with synthesized attributes only, respectively, when applied to the family of regular tree languages (Engelfriet and Maneth, 1999). This last restricted family of single use attributed tree transducers with synthesized attributes only, in its turn, is equivalent to the family of direction preserving monadic second-order tree transducers when applied to the family of the regular tree languages (Bloem and Engelfriet, 2000).

Theorem 10.3.3. *For every minimalist grammar G, there exists a strongly equivalent direction preserving tree transducer T definable in monadic second-order logic.*

10.4 Conclusion

The last section has closed the last gap in the chain of equivalences leading from minimalist grammars to a restricted notion of grammar morphism. An important rôle was played by the notion of finite-copying top-down tree transducers connecting multiple regular tree grammars and direction preserving logical tree translations. This special device of syntax-directed interpretation transforms input trees into output trees in a strict recursive top-down manner without regard for any context information. That such a context-free account was possible for a grammatical framework firmly entrenched in the transformational tradition is due to the special form of minimal link condition embodied in the definition of the *move* operation by which it was required that there is exactly one maximal projection of licensee feature $-x$. This formulation provides the basis for the decomposition of minimalist expression trees into tuples of trees that are the appropriate input for the kind of rearrangements performed by multiple regular tree grammars. A similar analysis is not possible for the second-order operations of tree substitution permitted by the framework of tree adjoining grammars. Tree translations equivalent to this model of natural language syntax cannot be defined solely in terms of subtrees. Elements of context information have to be passed along either implicitly in terms of state parameters or explicitly in terms of inherited attributes.

Bibliography

Bloem, Roderick 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.

Büchi, J. Richard (1960). Weak Second-order Arithmetic and Finite Automata. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, **6**:66–92.

Courcelle, Bruno and Paul Franchi-Zannettacci (1982a). Attribute grammars and recursive program schemes I. *Theor. Comput. Sci.*, **17**:163–191.

Courcelle, Bruno and Paul Franchi-Zannettacci (1982b). Attribute grammars and recursive program schemes II. *Theor. Comput. Sci.*, **17**:235–257.

Devlin, Keith (1991). *Logic and Information*. Cambridge University Press.

Doner, John (1970). Tree Acceptors and Some of Their Applications. *Journal of Computer and System Sciences*, **4**:406–451.

Elgot, Calvin C. (1961). Decision Problems of Finite Automata Design and Related Arithmetics. *Trans. Amer. Math. Soc.*, **98**:21–51.

Engelfriet, Joost and Linda Heyker (1991). The string-generating power of context-free graph grammars. *Journal of Computing Systems Science*, **43**:328–360.

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

Engelfriet, Joost and Sven Skyum (1976). Copying Theorems. *Information Processing Letters*, **4**:157–161.

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

Michaelis, Jens (2001b). Transforming linear context-free rewriting systems into minimalist grammars. In P. de Groote, G.F. Morrill, and C. Retoré, eds., *Logical Aspects of Computational Linguistics (LACL 2001)*, volume 2099 of *LNAI*, pp. 228–244. Berlin, Springer.

Michaelis, Jens, Uwe Mönnich, and Frank Morawietz (2001). On minimalist attribute grammars and macro tree transducers. In Christian Rohrer, Antje Roßdeutscher, and Hans Kamp, eds., *Linguistic Form and its Computation*, pp. 287–326. CSLI.

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

Rabin, Michael (1977). Decidable theories. In Jon Barwise, ed., *Handbook of Mathematical Logic*, pp. 595–629. North-Holland.

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

Stabler, Edward P. (1997). Derivational minimalism. In C. Retoré, ed., *Logical Aspects of Computational Linguistics (LACL '96)*, volume 1328 of *LNAI*. Springer, Berlin, Heidelberg.

Thatcher, James W. and Jesse B. Wright (1968). Generalized finite automata theory with an application to a decision problem of second-order logic. *Mathematical Systems Theory*, **2**(1):57–81.