Grammars with generalized contextfree rules and their tree automata

H. Volger (Univ. of Passau)

Abstract

We develop a uniform approach for studying classes of generalized contextfree grammars and their classes of trees. The rules of the grammars are still contextfree i.e. they have a single nonterminal on the left side. However, the set of right sides of rules with a given nonterminal on the left may be defined implicitly by means of an expression of an abstract logic whose models are finite words. Thus the set of rules may be infinite, in general. But the domain of locality of the derivation trees which consists of a node together with its finite set of successor nodes remains unchanged.

The generalization covers a wide range of examples. The uniform approach has the advantage that different classes of generalized contextfree grammars can be compared by means of a translation between their logics. In this manner a result proved for a class of generalized contextfree grammars over a given logic can be transferred to all less general classes of grammars whose logic can be translated into the given one. Quite a number of results may be extended to generalized contextfree grammars. In particular, the equivalence between classes of trees which consist of derivation trees of a contextfree grammar and those which consist of trees recognized by a tree automaton can be extended to the general case .

However, to deal with further classes of grammars which involve certain context-sensitive mechanisms like (linear) indexed grammars we need a generalization of a different type. We extend the nonterminals by elements from a fixed value structure. Extending likewise the contextfree rules by control conditions on the ocurring values one may control the derivation in the vertical direction. Again we keep the domain of locality of the contextfree grammars.

1 Introduction

The domain of locality of derivation trees of a contextfree grammar consists of a node together with its finite set of successor nodes. As a uniform method to study more general classes of grammars and their associated classes of trees we introduce classes of generalized contextfree grammars which are based on the same domain of locality. Hence the left side of a generalized contextfree rule is still a single nonterminal. However, the set of right sides of rules with a given nonterminal on the left may be defined implicitly by means of an expression whose interpretation is a set of words, which may be infinite in general. Thus one keeps the domain of locality of contextfree grammars and the derivation trees can be defined as usual.

The motivations for this work are twofold. In his dissertation Schneider (1999) has studied the systematic extension of parsing schemata a la Sikkel (1993) from contextfree grammars to more general classes of grammars. On the other hand Palm (1997) has studied in his dissertation several logics for defining classes of trees (cf. also Volger(1999)). The logics are used to express the wellformedness conditions on trees of a principle-based grammar formalism like GB of Chomsky (1981). Describing the transformation from a set of principles formulated in a first-order language for trees to a rule-based grammar he has obtained as intermediate steps generalized contextfree grammars where the right side of the rules are determined by expressions of appropriate logics for words.

We shall consider the following set up. Fix a set $X(S \cup \Sigma)$ of expressions for defining sets of words i.e. sentential forms with S and Σ as finite sets of nonterminal and terminal symbols. Thus X may be viewed as an abstract logic whose expressions have finite words as models. An X-

production is a pair (s, e) where $s \in S$ is a nonterminal and $e \in X(S \cup \Sigma)$ is an X-expression. The words u which satisfy e determine the associated set of rules for s which may be infinite. Now the usual definitions can be extended in a uniform manner to X-contextfree grammars. We obtain classes of grammars which will be weakly equivalent to contextfree grammars whenever all expressions used are required to define contextfree languages.

Our generalization covers a wide range of examples. Beside the classical example of the extended contextfree grammars of Thatcher and Wright (1968) which are based on regular expressions we want to mention here the state-transition grammars of Schneider (1999) which admit an Earley parser, a localized version of the ID/LP-grammars of Gazdar (1985) and the ML-grammars of Schneider (1999) which are based on a propositional modal logic with a modal operator for the right sibling relation. The latter case includes grammars with partially specified contextfree rules.

Quite a number of results for contextfree grammars can be extended to generalized contextfree grammars. In particular, the equivalence up to a projection between classes of trees which consist of derivation trees of a contextfree grammar and those which consist of trees recognized by a tree automaton can be extended to the general case under fairly mild conditions.

In addition, different classes of generalized contextfree grammars can be compared by means of translations. A translation is an interpretation preserving map between the associated sets of expressions and therefore it determines a simulation between the associated classes of grammars. By this method a result proved for one class might be transferred to all classes which can be translated into it.

However, our approach does not cover classes of grammars like (linear) indexed grammars or grammars from the geometric hierarchy of Weir (1992). These classes involve context-sensitive mechanisms like indices or control words which forces us to admit infinite sets of nonterminals.

Therefore another type of generalization has to be used. We will consider attributed words where the nonterminal symbols carry values from a given value structure. Likewise the rules are extended by control conditions on the occurring values to control the derivation in the vertical direction. Again we have kept the domain of locality of the contextfree grammars. It remains to be seen whether both types of generalization can be combined to cover further examples.

2 Grammars with quasi contextfree rules

A contextfree grammar $G = (S, \Sigma, P, s_0)$ may be represented in a slightly nonstandard manner as follows. Replace the finite subset P of $S \times (S \cup \Sigma)^*$ by the map $p : S \to P_{fin}((S \cup \Sigma)^*)$ where the finite subset $p(s) = \{u : (s, u) \in P\}$ of $(S \cup \Sigma)^*$ consists of the right sides of the rules with left side s. More generally, one might admit infinite subsets $p(s)$. To restrict the admissible sets we consider a set of expressions which are interpreted by sets of words and we require that each set $p(s)$ is defined implicitly by an expression e. The pair (s, e) which determines a possibly infinite set of contextfree rules will be called a quasi contextfree rule as the left side is still a single nonterminal. However, it should be noted that the set of symbols remains finite.

2.1 Grammars over a word logic:

Therefore we will introduce a class of abstract logics whose expressions are evaluated over words. For the rest of the paper we fix a set Σ of terminal symbols and for a given finite set S of nonterminal symbols we use the notation $\overline{S}$ for the set of symbols $S \cup \Sigma$.

Definition 1 *$\mathcal{X} = (X, \xi)$ is a* word logic *if for each finite set V we have a set $X(V)$ of* X-expressions *over V, an element $0_{X(V)}$ of $X(V)$ and an* interpretation map *$\xi_V : X(V) \to P(V^*)$ satisfying $\xi_V(0_{X(V)}) = \emptyset$.*

In the following we use the model theoretic notation $w \models_{\mathcal{X}} x$ for $w \in \xi_V(x)$. For later use we introduce the preorder $\leq_{X(V)}$ induced by the subset ordering on $P(V^*)$: $x_1 \leq_{X(V)} x_2$ iff $\xi_V(x_1) \subseteq \xi_V(x_2)$.

Let $\mathcal{X} = (X, \xi)$ be a word logic. As indicated above $\mathcal{X}$-expressions may be used to give an implicit definition of a set of contextfree rules of a grammar.

Definition 2 *$G = (S, \Sigma, p, s_0)$ is called an $\mathcal{X}$-*grammar *if $p : S \to X(\overline{S}^*)$ and $s_0 \in S$. The elements of Σ resp. S are the terminal resp.*

nonterminal symbols of G and s_0 is the start symbol of G whereas p is called the rule map of G. $\mathcal{XG}(S,\Sigma)$ denotes the class of $\mathcal{X}$-grammars over Σ and S.

The graph of the rule map p determines the set of generalized contextfree rules of G which is a subset of $S \times X(S)$. The composition with the interpretation map $\xi_{\overline{S}} : X(\overline{S}) \to P(\overline{S}^*)$ yields $Prod(G) := \{s \to_G u : u \models p(s)\}$, the associated set of *contextfree rules of* G which in general will be an infinite subset of $S \times \overline{S}^*$.

Making use of the rule relation $\to_G$ the derivation relation $\Rightarrow_G$ is defined in the usual manner: $u \Rightarrow_G v$ if there exist u_1, u_2 and $s \to_G w$ in $Prod(G)$ such that $u = u_1 s u_2$ and $v = u_1 w u_2$. All other definitions which are based on $Prod(G)$ and do not require the finiteness of this set can be extended to $\mathcal{X}$-grammars in this manner. In particular, one defines $Lang(G) \subseteq \Sigma^*$, the *language generated by* G and then $\mathcal{XL} := \{Lang(G) : G \in \mathcal{XG}\}$, the class of *$\mathcal{X}$-languages*.

Since the rules considered are still contextfree one can define as usual the class $Trees(G)$ of *derivation trees of* G as a subclass of $Trees(S,\Sigma)$. The latter consists of those finite trees whose leaves resp. inner nodes are labeled with elements from $\Sigma \cup \{\epsilon\}$ resp. S as follows. Thus (T, I) belongs to $Trees(G)$ if T is a tree domain and the map $I : T \to S \cup (\Sigma \cup \{\epsilon\})$ satisfies:

1. $I(\epsilon) = s_0$ and $I(w) \in (\Sigma \cup \{\epsilon\})$ whenever $succ(w) = [\,]$
2. If $w \in T$ with $succ(w) = [w1, \ldots, wk]$ and $k \geq 1$
   then $I(w1) \cdot \ldots \cdot I(wk) \in \xi_{\overline{S}} \cdot p(I(w))$

As usual the yield of (T, I) produces an element of $Lang(G)$. Finally we obtain $\mathcal{XT} := \{Trees(G) : G \in \mathcal{XG}\}$, the class of *$\mathcal{XG}$-recognizable trees*.

2.2 Examples of word logics

The following list of examples illustrates the range of the notion we have introduced. In the first three examples the word logic is determined by a class of languages where each language interprets itself.

(1) **Contextfree grammars**:
The standard example $\mathcal{X}_1 = \mathcal{CF}$ is defined by:

$$X_1(V) := P_{fin}(V^*) \text{ and } \xi_V(L) := L$$

Hence $G = (S, \Sigma, p, s_0)$ is a $\mathcal{X}_1$-grammar if $p : S \to P_{fin}(\overline{S}^*)$. Each $p(s)$ determines a finite set of contextfree rules with left side s i.e. G is a contextfree grammar. Thus $\mathcal{X}_1\mathcal{G}$ is the class $\mathcal{CFG}$ of contextfree grammars and $\mathcal{X}_1\mathcal{L}$ is is the class $\mathcal{CFL}$ of contextfree languages.

(2) **Quasi contextfree grammars**:
The maximal example $\mathcal{X}_2 = \mathcal{QCF}$ is defined by:

$$X_2(V) := P(V^*) \text{ and } \xi_V(L) := L$$

Hence $G = (S, \Sigma, p, s_0)$ is a $\mathcal{X}_2$-grammar if $p : S \to P(\overline{S}^*)$. Here each $p(s)$ determines an arbitrary set of contextfree rules with left side s. For lack of a better name the grammars in $\mathcal{X}_2\mathcal{G}$ have been called quasi contextfree grammars i.e. $\mathcal{X}_2\mathcal{G} = \mathcal{QCFG}$. However, the class $\mathcal{X}_2\mathcal{L} = \mathcal{QCFL}$ of quasi contextfree languages consists of arbitrary languages since any language L can be generated by the $\mathcal{QCF}$-grammar $G_L = (\{*\}, \Sigma, p_L, *)$ with $p_L(*) = L$. This example shows that we obtain more languages than we have bargained for. To get more reasonable classes of grammars the defining class of languages has to be restricted as in the next example.

(3) **Weakly contextfree grammars**:
An intermediate example $\mathcal{X}_3 = \mathcal{WCF}$ is obtained by:

$$X_3(V) := \mathcal{CFL}(V) \text{ and } \xi_V(L) = L$$

Hence $G = (S, \Sigma, p, s_0)$ is a $\mathcal{X}_3$-grammar if $p : S \to \mathcal{CFL}(\overline{S})$. Here $p(s)$ is a contextfree language over $\overline{S}$ i.e. generated by a contextfree grammar with $\overline{S}$ as terminal symbols. For lack of a better name the grammars in $\mathcal{X}_3\mathcal{G}$ have been called weakly contextfree grammars i.e. $\mathcal{X}_3\mathcal{G} = \mathcal{WCFG}$. It can be shown that the languages in $\mathcal{WCFL}$ are again contextfree languages. This follows by a construction which substitutes the rules of the contextfree grammars for each of the languages $p(s)$ into the rules of the $\mathcal{WCF}$-grammar.

In the next two examples the word logic is determined by a class of automata where each automaton is interpreted by its accepted language.

(4) **Extended contextfree grammars**:
The classical example $\mathcal{X}_4 = \mathcal{ECF}$ which motivated our definition is defined as follows, where $NFA(V)$ denotes the class of nondeterministic finite automata over V and $L(A)$ denotes the language accepted by the automaton A:

$$X_4(V) = NFA(V) \text{ and } \xi_V(A) = L(A)$$

Clearly $Rexp(V)$, the set of regular expressions over V, could be used as well. This yields the class $\mathcal{X}_4\mathcal{G} = \mathcal{ECFG}$ of extended contextfree grammars which was introduced by Thatcher and Wright (1968). There it was shown that $\mathcal{X}_4\mathcal{L}$ is again the class of contextfree languages. It should be noted that the family of nondeterministic finite automata in the definition of an $\mathcal{ECF}$-grammar may be replaced by a single automaton with a family of initial states.

(5) **State-transition grammars**:
The following class of grammars introduced by Schneider (1999) constitutes the largest class of grammars for which parsing schemata working from left to right can be defined. To arrive at the definition we have to introduce the underlying class of automata which work by generating symbols rather than consuming symbols. In addition, the new state and the generated symbol may depend on all the symbols generated so far.

$A = (Q, V, \mu, Q_0, Q_f)$ is a *state-transition automaton* over V if $\mu : Q \times V^* \to P(Q \times V)$ is the transition map and $Q_0 \subseteq Q$ resp. $Q_f \subseteq Q$ is the set of initial resp. final states. The transition relation for the configurations is defined for $a \in V \cup \{\epsilon\}$ by:

$$(q, u) \vdash_A (q', ua) \text{ if } \mu(q, u) \ni (q', a)$$

The language generated by A is given by $L(A) = \{u : (q, \epsilon) \vdash_A^* (q', u), q \in Q_0, q' \in Q_f\}$. $STA(V)$ denotes the class of state-transition automata over V.

Now $\mathcal{X}_5 = \mathcal{ST}$ is defined by:

$$X_5(V) = STA(V) \text{ and } \xi_V(A) = L(A)$$

Thus $G = (S, \Sigma, p, s_0)$ $p : S \to ST(\overline{S})$ is a $\mathcal{ST}$-grammar if $p(s)$ is a state-transition automaton over $\overline{S}$ for each $s \in S$. Hence we have obtained $\mathcal{X}_5\mathcal{G} = \mathcal{STG}$, the class of state-transition grammars of Schneider (1999). Again it should be noted that the family of state-transition automata in the definition of a $\mathcal{ST}$-grammar may be replaced by a single automaton with a family of initial state sets. We can show that $\mathcal{X}_5 = STL$ consists of arbitrary languages because any language L can be generated by $A_L := (\{q_1, q_2\}, \overline{S}, \delta, \{q_1\}, \{q_2\})$ where $\delta(q_1, u) := \{(q_1, a) : ua \in prefix(L)\} \cup \{(q_2, a) : ua \in L\} \cup \{(q_2, \epsilon) : u = \epsilon \in L\}$. Here the same remark as for example $\mathcal{X}_2$ applies. To get more reasonable classes of grammars one has to add restrictions on the class of automata.

(6) **Localized ID/LP grammars:**
$\mathcal{X}_6 = \mathcal{IDLP}$ will be defined by $X_6(V) = P_{fin}(M(V)) \times LP(V)$, where $M(V)$ is the set of multisets over V and $LP(V)$ is the set of irreflexive, transitive relations over V. Then $\xi_V(\{M_1, \ldots, M_k\}, R)$ is the set of those words, whose associated multiset is M_i for some i and whose order is admissible for R i.e. the order is disjoint to the converse of R. Thus $G = (S, \Sigma, p, s_0)$ is an $\mathcal{IDLP}$-grammar if $p : S \to MP(\overline{S}) \times IR(\overline{S})$. If $p(s)$ happens to be a fixed relation R for all $s \in S$, we obtain an ID/LP-grammar in the sense of Gazdar et al. (1985). This shows that the new class $\mathcal{X}_6\mathcal{G}$ is a localized version of the class of ID/LP-grammars.

(7) **Modal logic grammars:**
In the example $\mathcal{X}_7 = \mathcal{ML}$ formulas of a modal logic are used to define sets of words. Let $ML[\Rightarrow, V]$ be the propositional modal logic with $\langle\Rightarrow\rangle$ as modal operator for the relation of right sibling and the elements of V as propositional variables. Let $Wmod(x)$ denote the set of those models of the formula x which correspond to words w in V^*. Now the example $\mathcal{X}_7 = \mathcal{ML}$ is defined by:

$$X_7(V) := ML[\Rightarrow, V] \text{ and } \xi_V(x) = Wmod(x)$$

The class $\mathcal{X}_7\mathcal{G} = \mathcal{MLG}$ of modal logic grammars is closely related to the modal(1,*)-grammars of Schneider(1999). More general classes of grammars defined with modal logic have been considered by Blackburn and Spaan (1993) and Palm (1997). – Other languages like propositional linear temporal logic or first-order logic might be used as well.

(8) **Right linear grammars:**
In the next example the class of words itself is restricted. To cover this case the notion of a word logic has to be extended slightly by admitting sets of expressions of the form $X(S, \Sigma)$. In this manner it may depend differently on S

and Σ. The whole approach can be extended as well but the notation becomes more cumbersome.

$\mathcal{X}_8 = \mathcal{RL}$ is defined by:

$$X_8(S,\Sigma) := P_{fin}(\Sigma^* S \cup \Sigma^*) \text{ and } \xi_{S,\Sigma}(L) := L$$

Now $G = (S, \Sigma, p, s_0)$ is a $\mathcal{RLG}$-grammar if $p : S \to P_{fin}(\Sigma^* S \cup \Sigma^*)$. Each $p(s)$ determines a finite set of right linear rules with left side s i.e. G is a right linear grammar. Thus we have $\mathcal{X}_8\mathcal{G} = \mathcal{RLG}$. – Similarly, a word logic for the class of linear grammars could be defined.

Remark: To fit the class of linear indexed grammars of Gazdar (1985) into this set up the set $S \times I^*$ for an index set I would have to be used as set of nonterminals. However, this set will be infinite in general. Admitting infinite sets of nonterminals in a grammar any language L could be generated by a trivial contextfree grammar. To see this introduce an associated nonterminal for each word in L. In a later section we shall discuss a different approach which covers examples like the linear indexed grammars.

2.3 Closure properties

Closure properties of the classes of languages will serve as an example how results can be extended to generalized contextfree grammars. The following property of a word logic $\mathcal{X} = (X, \xi)$ is satisfied in many examples and is very useful in proving closure properties of the class $\mathcal{XG}$ of grammars and the class $\mathcal{XL}$ of languages.

Definition 3 *The word logic* (X, ξ) *is* (weakly) functorial *if for each (injective) map* $f : V_1 \to V_2$ *with finite sets* V_1,V_2 *there exists* $X(f) : X(V_1) \to X(V_2)$ *satisfying the following conditions:*

1. $\xi_{V_2} X(f) = P(\overline{f}^*)\xi_{V_1}$.
$$\begin{array}{ccc} X(V_1) & \stackrel{X(f)}{\longrightarrow} & X(V_2) \\ \downarrow \xi_{V_1} & & \downarrow \xi_{V_2} \\ P(V_1^*) & \stackrel{P(f^*)}{\longrightarrow} & P(V_2^*) \end{array}$$
2. $X(gf) = X(g)X(f)$ *for* $f : V_1 \to V_2, g : V_2 \to V_3$ *and* $X(id_V) = id_{X(V)}$

Thus X is a functor on finite sets and ξ is a natural transformation between X and the functor $P((-)^*)$. In the list of examples above the word logics $\mathcal{X}_1$ - $\mathcal{X}_5$, $\mathcal{X}_8$ are functorial, $\mathcal{X}_6$ is weakly functorial whereas in example $\mathcal{X}_7 = \mathcal{ML}$ the positivity of the formulas seems to be necessary.

The functoriality of a word logic $\mathcal{X}$ ensures that we may assume without loss of generality that the set of nonterminals of an $\mathcal{X}$-grammar is disjoint to a given set. This follows by the lemma below.

Lemma 1 *Let* $f : S \to T$ *be a map and* $G = (S, \Sigma, p, s_0)$, $H = (T, \Sigma, q, t_0)$ *be* $\mathcal{X}$*-grammars for a word logic* $\mathcal{X} = (X, \xi)$ *such that* H *is induced by* G *via* f *i.e. we have* $X(\overline{f})p = qf$ *and* $f(s_0) = t_0$. *Whenever* $\mathcal{X}$ *is functorial or* $\mathcal{X}$ *is weakly functorial and* f *is injective then* $Lang(G) = Lang(H)$ *and* $\overline{f}(Trees(G)) = Trees(H)$.

The proof is based on a chain of equivalences: $f(s) \to_H v$ iff $v \in \xi_{\overline{T}} q f(s)$ iff $v \in P(\overline{f}^*)\xi_{\overline{S}} p(s)$ iff there exists u with $\overline{f}^*(u) = v$ and $u \in \xi_{\overline{S}} p(s)$ iff there exists u with $\overline{f}^*(u) = v$ and $s \to_G u$.

The lemma can be used to show that the functoriality of the word logic ensures the closure under substitution of the associated class of languages.

Proposition 2 *For any word logic* $\mathcal{X} = (X, \xi)$ *which is weakly functorial the classes* $\mathcal{XL}$ *and* $\mathcal{XT}$ *are closed under substitution i.e. for* $G = (S, \Sigma, p, s_0) \in \mathcal{XG}$ *and* $G_a = (S_a, \Sigma_a, p_a, s_{0,a}) \in \mathcal{XG}$ *for* $a \in \Sigma$ *there exists* $\hat{G} = (\hat{S}, \hat{\Sigma}, p, s_0) \in \mathcal{XG}$ *with* $\hat{S} \simeq S \cup \bigcup_{a\in\Sigma} S_a$ *and* $\hat{\Sigma} = \bigcup_{a\in\Sigma} \Sigma_a$ *such that:*

$$Lang(\hat{G}) = Lang(G)\,[a/Lang(G_a) : a \in \Sigma]$$

$$Trees(\hat{G}) = Trees(G)\,[a/Trees(G_a) : a \in \Sigma]$$

By lemma 1 we may assume that the sets S and all the S_a are disjoint and thus $\hat{S} := S \cup \bigcup_{a\in\Sigma} S_a$ is a disjoint union. The rule map $q : \hat{S} \to X(\hat{S} \cup \hat{\Sigma})$ is defined as follows. $q \mid_{S_a} := X(j_a)p_a$ where j_a is the inclusion of $S_a \cup \Sigma_a$ in $\hat{S} \cup \hat{\Sigma}$. $q \mid_S := X(k)p$ where $k : S \cup \Sigma \to \hat{S} \cup \hat{\Sigma}$ is determined by $k(s) = s$ for $s \in S$ and $k(a) = s_{0,a}$ for $a \in \Sigma$. Now the proof follows from the equivalence: $s \to_{\hat{G}} v$ iff there exist u and u_a for $a \in \Sigma$ such that $v = u\,[a/u_a]$, $s \to_G u$ and $s_{0,a} \to_{G_a} u_a$ for $a \in \Sigma$.

As usual the result in proposition 2 gives rise to further closure properties of $\mathcal{XL}$. The corollary below is given without proof.

Corollary 3 *Let $\mathcal{X} = (X, \xi)$ be a word logic which is weakly functorial.*

1. *If for $v_1, v_2 \in V$ there exists $x \in X(V)$ with $\xi(x) = \{v_1, v_2\}$ then $\mathcal{XL}$ is closed under binary unions.*
2. *If for $v_1, v_2 \in V$ there exists $y \in X(V)$ with $\xi(y) = \{v_1 v_2\}$ then $\mathcal{XL}$ is closed under concatenation.*
3. *If for $v_1, v_2 \in V$ there exist $y, z \in X(V)$ with $\xi(y) = \{v_1 v_2\}$ and $\xi(z) = \{\epsilon\}$ then $\mathcal{XL}$ is closed under Kleene iteration.*

Assume that $\mathcal{X}$ admits one letter singletons i.e. for all V there exist $d_V : V \to X(V)$ such that $\xi_V d_V(v) = \{v\}$ for all $v \in V$. Under this assumption we may use in (1) the stronger assumption that X admits binary unions. This requires that for all V and all $x_1, x_2 \in X(V)$ there exists $x \in X(V)$ with $\xi_V(x) = \xi_V(x_1) \cup \xi_V(x_2)$. In the same manner we require in (2) resp. (3) that X admits concatenation resp. Kleene iteration. For example the word logics $\mathcal{X}_1$ - $\mathcal{X}_5$ satisfy these properties. - In a later section we shall consider word logics which admit finitary unions.

2.4 Translations and simulations:

To compare grammars for two different word logics $\mathcal{X} = (X, \xi)$ and $\mathcal{Y} = (Y, \eta)$ we introduce translations between the logics which determine simulations between the associated classes of grammars.

Definition 4 *$F : \mathcal{X} \to \mathcal{Y}$ is a* translation *from $\mathcal{X}$ to $\mathcal{Y}$ if for each finite set V there exists a map $F_V : X(V) \to Y(V)$ which is compatible with the interpretation maps i.e. $\xi_V = \eta_V \cdot F_V$. Thus we have for all $x \in X(V)$:*

$$w \models_{\mathcal{Y}} F_V(x) \text{ iff } w \models_{\mathcal{X}} x \qquad (1)$$

.

A translation $F : \mathcal{X} \to \mathcal{Y}$ determines a *simulation* $\hat{F} : \mathcal{XG} \to \mathcal{YG}$ of the grammars in $\mathcal{XG}$ by grammars in $\mathcal{YG}$ as follows. For a grammar $G = (S, \Sigma, p, s_0)$ in $\mathcal{XG}$ the simulating grammar $\hat{F}(G)$ is given by $(S, \Sigma, \hat{F}(p), s_0)$ where $\hat{F}(p) := F_S \cdot p$ is the composition of the rule map p with the translation map F_S.

Saying that G is *simulated* by $\hat{F}(G)$ is justified by the lemma below since it states that both grammars have the same derivation relation. Thus we may say that the grammars in $\mathcal{XG}$ *can be simulated* by the grammars in $\mathcal{Y}G$ if there exists a translation $F : \mathcal{X} \to \mathcal{Y}$. This will be denoted by $\mathcal{XG} \leq \mathcal{Y}G$. Clearly, the simulatability relation $\leq$ is a preorder since a composition of translations is again a translation and the identity is a translation.

Lemma 4 *Let $F : \mathcal{X} \to \mathcal{Y}$ be a translation. For each grammar $G \in \mathcal{XG}$ we have:*

1. $\to_{\hat{F}(G)} = \to_G$
   i.e. the sets of contextfree rules coincide.
2. $\Rightarrow_{\hat{F}(G)} = \Rightarrow_G$
   i.e. the derivation relations coincide.
3. $Trees(\hat{F}(G)) = Trees(G)$
   i.e. G and $\hat{F}(G)$ are strongly equivalent.
4. $Lang(\hat{F}(G)) = Lang(G)$
   i.e. G and $\hat{F}(G)$ are weakly equivalent.

Making use of equation (1) statement (1) follows by a chain of equivalences: $s \to_{\hat{F}(G)} w$ iff $w \models_{\mathcal{Y}} \hat{F}(p)(s)$ iff $w \models_{\mathcal{Y}} F_{\overline{S}}(p(s))$ iff $w \models_{\mathcal{X}} p(s)$ iff $s \to_G w$. (2) and (3) are a consequence of (1) whereas (4) follows from (2). — The corollary below shows how the simulations can be used.

Corollary 5 *Let $\mathcal{X}$,$\mathcal{Y}$ be word logics.*

1. *$\mathcal{X} \leq \mathcal{Y}$ implies $\mathcal{XT} \subseteq \mathcal{YT}$ and $\mathcal{XL} \subseteq \mathcal{YL}$.*
2. *$\mathcal{X} \leq \mathcal{Y} \leq \mathcal{X}$ implies $\mathcal{XT} = \mathcal{YT}$ and $\mathcal{XL} = \mathcal{YL}$.*
3. *$\mathcal{X} \leq \mathcal{QCF}$*
4. *$CF \leq \mathcal{X}$ implies $\mathcal{CFL} \subseteq \mathcal{XL}$.*
5. *$\mathcal{X} \leq \mathcal{WCF}$ implies $\mathcal{XL} \subseteq \mathcal{WCFL} = \mathcal{CFL}$.*

(2) and (3) in lemma 4 yield (1) and hence (2). (1) is used to prove (3) - (5). To prove (3) one uses the interpretation map as translation. (5) makes use of the equation $\mathcal{WCFL} = \mathcal{CFL}$ which was mentioned in the discussion of the example $\mathcal{WCF}$ above.

Proposition 6 *The classes of generalized contextfree grammars mentioned above are related by the following translations:*

$$\begin{array}{ccc} \mathcal{RL} & \to & \mathcal{IDLP} \\ \downarrow & \swarrow & \downarrow \\ \mathcal{CF} & \to & \mathcal{ML} \\ \downarrow & \swarrow & \downarrow \\ \mathcal{ECF} & \to & \mathcal{WCF} \\ \downarrow & & \downarrow \\ \mathcal{ST} & \to & \mathcal{QCF} \end{array}$$

The translations $\mathcal{ST} \to \mathcal{QCF}$ and $\mathcal{WCF} \to \mathcal{QCF}$ follow by (3) in corollary 5, the maximality of $\mathcal{QCF}$. The translations $\mathcal{RL} \to \mathcal{CF}$, $\mathcal{CF} \to \mathcal{ECF}$, $\mathcal{RL} \to \mathcal{IDLP}$, $\mathcal{IDLP} \to \mathcal{CF}$, $\mathcal{CF} \to \mathcal{ML}$ and hence $\mathcal{IDLP} \to \mathcal{ML}$ are obvious. The translation $\mathcal{ECF} \to \mathcal{WCF}$ uses the fact that regular languages are contextfree. To get the translation $\mathcal{ECF} \to \mathcal{ST}$ one notes that state-transition automata whose transition function do not depend on the word generated so far may be viewed as nondeterministic automata. Adding the further restriction that the associated transition graph is acyclic one obtains again the translation $\mathcal{CF} \to \mathcal{ECF}$. To construct the translation $\mathcal{ML} \to \mathcal{ECF}$ which in turn yields $\mathcal{ML} \to \mathcal{WCF}$ one makes use of a function f_a on regular expressions which satisfies: $L(r) = a \cdot L(f_a(r))$. It splits a letter a off at the left end of a word.

Corollary 7 *The associated classes of languages are related as follows:*

$$\begin{array}{ccc} \mathcal{RLL} & \subseteq & \mathcal{IDLPL} \\ \cap\,| & & \cap\,| \\ \mathcal{CFL} & = & \mathcal{MLL} \\ \| & & \| \\ \mathcal{ECFL} & = & \mathcal{WCFL} \\ \cap\,| & & \cap\,| \\ \mathcal{STL} & = & \mathcal{QCFL} \end{array}$$

This follows from proposition 6 by an application of 5 and the equations $\mathcal{CFL} = \mathcal{WCFL}$ and $\mathcal{STL} = \mathcal{QCFL}$ mentioned earlier discussing the examples $\mathcal{X}_3 = \mathcal{WCF}$ and $\mathcal{X}_5 = \mathcal{ST}$. – In this context it might be useful to have a notion of weak simulation which yields weakly equivalent grammars. Possibly this might be obtained by the use of weak translations which are based on translation maps of the form $F_S : X(S) \to Y(f(S))$.

It should be mentioned that it is fairly easy to find conditions on a word logic $\mathcal{X}$ which guarantee closure properties of the class $\mathcal{XL}$ of languages. Let $\mathcal{X}_0$ be the word logic determined by $X_0(S) = P_{=1}(\overline{S})$. Then $\mathcal{XL}$ is closed under unions whenever $\mathcal{X}_0 \leq \mathcal{X}$. Similarly, $\mathcal{XL}$ is closed under concatenations whenever $\mathcal{X}_0 \leq \mathcal{X}$ and each set $X(S)$ admits concatenations.

3 Generalized tree automata

It is well known that the class of trees which are recognizable by a contextfree grammar coincides with the class of trees which are recognizable by a tree automaton (cf. (Thatcher, 1967)). It remains to be seen whether this equivalence may be lifted to grammars over a word logic. To do this we have to generalize the notion of a tree automaton to an arbitrary word logic.

As a starting point we shall use the following variant of the definition of a nondeterministic top-down tree automaton.

Definition 5 $A = (Q, S, \Sigma, \delta, q_0, s_0)$ *is a (nondeterministic top-down)* tree automaton *if it satisfies:*

1. *Q is finite set of states and q_0 an initial state in Q*
2. *S is finite set of nonterminal symbols and s_0 an initial symbol in S*
3. *$\delta : S \times Q \to P_{fin}(\overline{Q}^*)$ is a transition function which associates with each pair (s, q) a finite set of possible lists (=words) of successor states or terminal symbols.*

3.1 Tree automata over a word logic

Let $\mathcal{X} = (X, \xi)$ be a word logic. Replacing in the above definition CF by $\mathcal{X}$ we obtain $\mathcal{X}$-tree automata as the appropriate generalization.

Definition 6 $A = (Q, S, \Sigma, \delta, q_0, s_0)$ *is a (nondeterministic top-down)* $\mathcal{X}$-tree automaton *if it satisfies:*

1. *Q is a finite set of states and q_0 is an initial state in Q*
2. *S is a finite set of nonterminal symbols and s_0 is an initial symbol in S*
3. *$\delta : S \times Q \to X(Q)$ is a transition function which associates with each pair (s, q)*

by means of $\xi_Q : X(Q) \to P(\overline{Q}^*)$ a set of possible lists (=words) of successor states or terminal symbols.

$\mathcal{XTA}(Q, S, \Sigma)$ denotes the class of $\mathcal{X}$-tree automata with state set Q and symbols $S \cup \Sigma$. In the case $\mathcal{X} = \mathcal{CF}$ we get back the definition above. – The $\mathcal{XTA}$-recognizable trees (T, I) in $Trees(S, \Sigma)$ are defined with the help of run trees (T, K) in $Trees(Q, \Sigma)$ as follows.

Definition 7 *A tree (T, K) in $Trees(Q, \Sigma)$ is called a* run *for the tree $(T, I) \in Trees(S, \Sigma)$ over the $\mathcal{X}$-tree automaton A whenever:*

1. *If $w \in T$ with $suc(w) = [w1, \ldots, wk], k \geq 1$ then $K(w1) \cdot \ldots \cdot K(wk) \in \xi_Q \cdot \delta(I(w), K(w))$*
2. *If $w \in T$ with $suc(w) = [\,]$ then $K(w) = I(w) \in \Sigma \cup \{\epsilon\}$.*

Now a tree (T, I) in $Trees(S, \Sigma)$ is said to be $\mathcal{XTA}$- *recognized* by an $\mathcal{X}$-tree automaton A if there exists a run (T, K) for (T, I) over A. $Trees(A))$ denotes the class of $\mathcal{XTA}$-recognizable trees for A. – In the case $\mathcal{X} = \mathcal{CF}$ we get back the usual definition of recognizable trees.

3.2 The correspondence result

Under certain mild conditions on the word logic $\mathcal{X} = (X, \xi)$ we get the desired correspondence result. It relates derivation trees (=recognizable trees) of an $\mathcal{X}$-grammar with recognizable trees of an associated $\mathcal{X}$-tree automaton. A word logic $\mathcal{X}$ is said to be *finitary* if for each surjective map $f : S_1 \to S_2$ with S_1, S_2 finite the map $X(f)$ has finite preimages i.e. the set $X(f)^{-1}(x)$ is finite for each $x \in X(S_2)$. All the examples of word logics mentioned above except $\mathcal{X}_2 = QCF$ and $\mathcal{X}_5 = STA$ are finitary.

Theorem 8 *Let $\mathcal{X} = (X, \xi)$ be a word logic which is functorial and finitary and admits finitary unions. Then the following holds:*

1. *There exists a map*

$$xta : \mathcal{XG}(S, \Sigma) \to \mathcal{XTA}(S, S, \Sigma)$$

which preserves recognizable trees.

More precisely, for each $G \in \mathcal{XG}$ we have: $(T, I) \in Trees(G)$ implies $(T, I) \in Trees(xta(G))$, where I is a run for (T, I). Hence (T, I) is $\mathcal{XTA}$-recognizable whenever (T, I) is $\mathcal{XG}$-recognizable.

2. *There exists a map*

$$xg : \mathcal{XTA}(Q, S, \Sigma) \to \mathcal{XG}(Q \times S, \Sigma)$$

which preserves recognizable trees up to a projection of the label sets.

More precisely, for each $A \in \mathcal{XTA}$ we have: $(T, I) \in Trees(A)$ with a run (T, K) implies $(T, K, I) \in Trees(xg(A))$. Note that (T, I) is obtained from (T, K, I) by the projection onto $(S \cup \Sigma)$. Hence (T, I) is $\mathcal{XG}$-recognizable up to a projection whenever (T, I) is $\mathcal{XTA}$-recognizable.

Corollary 9 *Let $\mathcal{X} = (X, \xi)$ be a word logic as in theorem 8. Then we have the following result: A tree is a projection of a recognizable tree for an $\mathcal{X}$-grammar iff it is a projection of a recognizable tree for an $\mathcal{X}$-tree automaton.*

The map xta for (1) is obtained as follows. $xta(S, \Sigma, p, s_0)$ is defined as $(S, S, \Sigma, \delta, s_0, s_0)$ where $\delta(s, s) := p(s)$ and $\delta(s, s') := 0_{X(S)}$ for $s \neq s'$. The desired equivalence makes use of $p(I(w)) = \delta(I(w), I(w))$.

To get the map xg we associate with an $\mathcal{X}$-tree automaton $A = (Q, S, \Sigma, \delta, q_0, s_0)$ the $\mathcal{X}$-tree automaton $\hat{A} := (Q \times S, S, \Sigma, \hat{\delta}), (q_0, s_0), s_0)$ where $\hat{\delta}((q, s), s) := \bigcup(y : X(pr_1)(y) = \delta(q, s))$ and $\hat{\delta}((q, s), s') := 0_{X(Q \times S)}$. Hence $\hat{\delta}$ is uniqely determined by the map $\mu : Q \times S \to X(Q \times S)$ with $\mu((q, s)) = \hat{\delta}((q, s), s) = \bigcup(y : X(pr_1)(y) = \delta(q, s))$. Now we obtain $xg(Q, S, \Sigma, \delta, q_0, s_0) := (Q \times S, \Sigma, \mu, (q_0, s_0))$. Making use of the preordering on $\mathcal{X}$-expressions introduced earlier the required equivalence can be proved.

4 Contextfree grammars with controlled rules

To deal with classes of grammars like indexed grammars or linear indexed grammars we need another type of generalization of contextfree grammars. In order to control the derivation in the vertical dimension one introduces additional values for the nonterminals.

We replace words by *attributed words* where the nonterminals carry values from a given structure $\mathcal{A} = (A, (f_i : i = 1, \ldots, l), (r_j : j = 1, \ldots, m), a_0, A_f)$ where a_0 is an initial value and A_f is a set of final values. Now

a set of *attributed nonterminals* is a set of the form $S \times A$ and we obtain words of the form $u_1 s_1[a_1]u_2 \dots u_k s_k[a_k]u_{k+1}$ in $((S \times A) \cup \Sigma)^*$ where $u_1, \dots u_{k+1} \in \Sigma^*$, $s_1, \dots, s_k \in S$ and $a_1, \dots, a_k \in A$. Thus we have replaced $Words(S, \Sigma) = (S \cup \Sigma)^*$ by $\mathcal{A}Words(S, \Sigma) = Words(S \times A, \Sigma)$.

Likewise the contextfree rules will be extended by a control condition.

Definition 8 *An* $\mathcal{A}$-controlled contextfree rule *is of the form* (r, φ) *where* $r = s \to u_1 s_1 u_2 \dots u_k s_k u_{k+1}$ *is a contextfree rule and the control condition* φ *is a conjunction of atomic formulas over* $\mathcal{A}$ *with* $\{x, x_1, \dots, x_k\}$ *as free variables. If* r *is a terminal rule i.e.* $k = 0$ *then we require that* φ *is the formula* $x = a_0$.

Typical control conditions will be connection formulas $r_j(x, x_1, \dots, x_k)$, synthesizing formulas $f_i(x_1, \dots, x_k) = x$ or inheritance formulas $\bigwedge_{1 \le i \le k} x_i = f_i(x)$.

The application of the rule (r, φ) to the attributed word $vs[a]w$ yields the attributed word $vu_1 s_1[a_1]u_2 \dots s_k[a_k]u_{k+1}w$ whenever the local condition $\varphi(a, a_1, \dots, a_k)$ holds in $\mathcal{A}$. In the definition of a derivation tree we require in addition at the leaves the boundary conditions imposed by the terminal rules and that the value at the root is the initial value a_0.

4.1 Linear control conditions

A control condition is said to be *linear* if the condition involves only the value at the node and at a specified successor node usually called the head. More precisely, an $\mathcal{A}$-controlled contextfree rule (r, φ) is *linear* if the control condition φ is of the form $\psi(x, x_i) \wedge \bigwedge_{j \ne i} x_j = a_0$. Thus the nonterminal s_i in r is specifed as head. This shows that the underlying grammars of linearly $\mathcal{A}$-controlled grammars should be considered as headed contextfree grammars.

The grammars in $\mathcal{HCFG}$ are defined as follows. A *headed word* is either a word $u_1 \underline{s} u_2$ where a unique nonterminal s has been specified as head or it is a terminal word w. Thus $HWords(S, \Sigma) := ((S \cup \Sigma)^* \times S \times (S \cup \Sigma)^*) \cup \Sigma^*$ is the set of headed words over S.

Consequently we obtain *headed contextfree rules* if we replace words by headed words in the right side of a rule. Hence G is a *headed contextfree grammar* if its rules are of the form $s \to u_1 \underline{s'} u_2$ or $s \to w$ where $w \in \Sigma^*$. The head introduces a special successor in the local tree, the *projection* onto the head. Hence the derivation trees of a $\mathcal{HCFG}$ will be *trees with projection lines* i.e. trees in $HTrees(S, \Sigma)$. – Below we shall see that the class of headed contextfree grammars itself may be viewed as a class of linearly $\mathcal{B}$-controlled grammars where B is the set of truthvalues.

4.2 Examples of controlled contextfree grammars

(1) **Headed contextfree grammars:**
Let $\mathcal{B} = (B, (\oplus_k : k \in N), 0, \{1\})$ be the head structure where $B = \{0, 1\}$ and $\oplus_k$ is the k-fold exclusive or. The control condition for k is $\oplus_k(x_1, \dots x_k) = 1$. In this case the grammars are basically headed contextfree grammars. The value 1 selects the head nonterminal and a projection line starts with the value 0 and later nodes have the value 1.

(2) **Indexed grammars:**
Let $\mathcal{A} = (I^*, (pop_i : i \in I), nop, (push_i : i \in I), \epsilon, \{\epsilon\})$ be the structure of *stacks over* I. The control conditions for k nonterminals are $\bigwedge_{1 \le j \le k} pop_i(x) = x_j$ resp. $\bigwedge_{1 \le j \le k} x = x_j$ resp. $\bigwedge_{1 \le j \le k} push_i(x) = x_j$. In this case the $\mathcal{A}$-controlled contextfree grammars are the *indexed grammars* of Aho (1968).

(3) **Distributed indexed grammars:**
Let $\mathcal{A}_d = (I^*, (c_k : k \in N), \epsilon, \{\epsilon\})$ be the structure of *distributed stacks over* I where $c_k(u_1, \dots, u_k) = u_1 \dots u_k$ is the k-fold concatenation. The control condition for k nonterminals is $c_k(x_1, \dots, x_k) = x$. In this case the $\mathcal{A}$-controlled contextfree grammars are the *distributed indexed grammars* of Staudacher (1994).

(4) **Unary definite clause grammars:**
To obtain definite clause grammars ((Pereira and Warren, 1980)) we consider the following value structure. It consists of a set of terms together with a relation for term matching for sequences of terms. In this manner we obtain the subclass of unary definite clause grammars i.e. those grammars where all nonterminals (=predicates) have exactly one argument. The case where all operations are at most unary is related to indexed grammars (cf. (Bertsch, 1994)).

The next two examples will be linear.

(5) **Linear indexed grammars:**
Let $\mathcal{A} = (I^*, (pop_i : i \in I), nop, (push_i : i \in$

$I), \epsilon, \{\epsilon\})$ be the structure of *stacks over* I. The linear control conditions are determined by $pop_i(x) = x_j$ resp. $x = x_j$ resp. $push_i(x) = x_j$ where x_j is the head successor. In this case the linearly $\mathcal{A}$-controlled contextfree grammars are just the *linear indexed grammars* of Gazdar (1985).

(6) **Headed contextfree grammars with a contextfree grammar as a control set:** Let $H = (\Delta, T, Q, t_0) \in CFG(\Delta, T)$ be given as controlling grammar. The associated value structure $\mathcal{A}_H$ is defined as follows. $((T \cup \Delta)^*, (ldel_d : d \in \Delta), (app_q : q \in Q), \epsilon, \{\epsilon\})$ where the relation $ldel_d$ removes an element d at the left end i.e. $(u_1, u_2) \in ldel_d$ if $u_1 = du_2$ and the relation $app_{t \to w}$ applies the rule $t \to w$ i.e. $(u_1, u_2) \in app_{t \to w}$ if $u_1 = v_1 t v_2, u_2 = v_1 w v_2$ The linear control conditions are determined by $ldel_d(x, x_j)$ resp. $app_q(x, x_j)$ where x_j is the head successor. In this case the linearly $\mathcal{A}_H$-controlled contextfree grammars are controlled contextfree grammars with H as controlling grammar (cf. (Weir, 1992)). They constitute the second step of the hierarchy introduced by Weir. Further steps in the hierachy will require more complex value structures.

Remark: To obtain arbitrary definite clause grammars where we may have nonterminals with finitely many arguments we have to extend our approach to the case with several attributes or sorts. This would mean that we have a value structure for each sort. – Similar remarks apply to the class of attribute grammars (Knuth (1968)).

5 Summary

As a means to study more general classes of grammars and their classes of trees we have developed a uniform approach to generalized contextfree grammars. We have presented two generalization of the class of contextfree grammars which cover a wide range of examples. Both generalizations yield grammars with a contextfree core since they preserve the domain of locality of contextfree grammars. As a first extension we have considered generalized cf rules. Here the right sides of the rules for a given nonterminal on the left side are defined implicitly by means of expressions which are interpreted by sets of words. This led to $\mathcal{X}$-grammars for a word logic $\mathcal{X}$.

In a second extension we have considered attributed contextfree rules. Here the nonterminals carry values from a given value structure and the application of the rules is controlled by conditions on the associated values. This led to $\mathcal{A}$-controlled as well as linearly $\mathcal{A}$-controlled contextfree grammars for a value structure $\mathcal{A}$. It remains to be seen whether both approaches may be combined to cover further examples.

References

A.V. Aho. 1968. Indexed grammars - an extension of contextfree grammars. *Journal of the Association for Computing Machinery*, 15:647–671.

E. Bertsch. 1994. On the relationship between indexed grammars and logic programs. *Journal of Logic Programming*, 18:81–98.

P. Blackburn and E. Spaan. 1993. A modal perspective on the computational complexity of attribute value grammar. *Journal of Logic, Language and Information*, 2:129–169.

N. Chomsky. 1981. *Lectures on government and binding*. Foris, Dordrecht.

G. Gazdar, E. Klein, G.K. Pullum, and I. Sag. 1985. *Generalized Phrase Structure Grammar*. Blackwell, Oxford.

G. Gazdar. 1985. Applicability of indexed grammars to natural language. Technical Report CSLI-85-34, Center for the Study of Language and Information, Stanford.

J.E. Hopcroft and J.D. Ullman. 1979. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley.

D.E. Knuth. 1968. Semantics of contextfree languages. *Math. Systems Theory*, 2:127 – 145.

A. Palm. 1997. *The expressivity of tree languages*, volume 173. Infix Verlag.

A. Palm. 1999. The expressivity of tree languages for syntactic structure. In H.-P. Kolb and U. Moennich, editors, *The Mathematics of Syntactic Structures*, pages 113 – 151. Mouton de Gruyter.

F.C.N. Pereira and D.H.D. Warren. 1980. Definite clause grammars for language analysis - a survey of the formalism and a comparison with augmented transition networks. *Artificial Intelligence*, 13:231 – 278.

K.-M. Schneider. 1999. *Algebraic construction of parsing schemata.* Ph.D. thesis, Universität Passau, Germany.

K. Sikkel. 1993. *Parsing schemata.* Ph.D. thesis, University of Twente, Netherlands.

P. Staudacher. 1994. Neue Grenzen jenseits der Kontextfreiheit: DI-Grammatiken und DI-Automaten. In S.W. Felix, C. Habel, and G. Rickheit, editors, *Kognitive Linguistik: Repräsentationen und Prozesse*. Westdeutscher Verlag, Opladen.

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

J.W. Thatcher. 1967. Characterizing derivation trees of contextfree grammars through a generalization of finite automata theory. *Journal of Computer and System Sciences*, 1:317–322.

K. Vijay-Shanker and D.J. Weir. 1994. The equivalence of four extensions of context-free grammars. *Mathematical Systems Theory*, 27:511–546.

H. Volger. 1999. Principle languages and principle based parsing. In H.-P. Kolb and U. Moennich, editors, *The Mathematics of Syntactic Structures*, pages 83 – 113. Mouton de Gruyter.

D. Weir. 1992. A geometric hierarchy beyond context-free languages. *Theoretical Computer Science*, 104:235–261.