A New–Old Class of Linguistically Motivated Regulated Grammars

S. Marcus, C. Martín-Vide, V. Mitrana, Gh. Păun

Romanian Academy, Romania
Research Group in Mathematical Linguistics, Rovira i Virgili University, Spain
Faculty of Mathematics, University of Bucharest, Romania
Institute of Mathematics of the Romanian Academy, Romania

Abstract

The oldest restriction in the derivation of context-free grammars is believed to be the matrix control mechanism, introduced by S. Abraham in 1965. However, in a paper (with solid linguistic motivations) of I. Bellert, published in the same year, there are *in nuce* the ideas of several control mechanisms considered later, as well as an idea which we do not know to have been explored. In short, conditions about some of the paths from the root to leaf nodes in the derivation trees of a context-free grammar are considered. Here we investigate this type of control, with emphasis on a class which generates a mildly context-sensitive family of languages (in particular, a useful pumping lemma is provided).

1 Introduction

Regulated rewriting is one of the most developed branches of formal language theory, with roots in the mid-sixties and with several dozens of control mechanisms well investigated – see (Dassow and Păun 1989). The motivation started from the observation that many languages of interest, natural and programming languages included, are not context-free. Generating such languages by context-sensitive grammars is not a practical solution, because at the context-sensitive level we have many difficulties (e.g., many problems are undecidable), and, more important from a linguistic point of view, the derivation in a context-sensitive grammar cannot be described by a sufficiently easy to manipulate graph structure. The first restriction in the derivation of context-free grammars is believed to be that introduced by S. Abraham, in 1965, the *matrix grammars*. In short, sequences of rewriting rules are given, which are applied together in a derivation step (this considerably increases the power of context-free grammars).

It was a surprise for us to recently discover the paper by I. Bellert, also published in 1965 (submitted on November 19, 1964) in a quite visible journal, *Information and Control*, with a provocative title, where a class of regulated context-free grammars was introduced which were never – to our knowledge – studied after that. In particular, this paper is not mentioned in (Salomaa 1973), the first monograph having a chapter about regulated rewriting.

Two things are rather interesting in the paper (Bellert 1965). First, it is very well motivated, starting from the assertion that natural language is not context-free (this was a lenghtily debated issue in linguistics: it was "clear" in the sixties, based on arguments of Chomsky, Bar-Hillel, and others; later these arguments were contested, but it is agreed now that natural language contains at least typical constructions which are not context-free; the reader can consult the papers we have included in the bibliography, most of them with self-explanatory titles). However, the paper not only provides a theoretical study of this assertion, but, after introducing "relational grammars", a consistent case study is examined, using the new formalism for modelling several constructions from Polish. Bellert applies her relational grammar to the generation of Polish kernel sentences. In this application, 13 relations are considered, mainly concerning the agreement in gender, number or case between the noun and the predicate.

Second, the definition of relational grammars contained several different ideas which remind us of restrictions on derivation which were widely investigated later, under different names. Informally, the framework is the following one. One considers a context-free grammar $G = (N, T, S, P)$ and a way to select from all possible derivation trees in G only those trees with certain properties, specified by a set of *tuple-grammars* and a *relation* on the rules of these tuple-grammars. More specifically, one gives several k-tuples of context-free grammars, $(G_1, \dots, G_k)$, of the form $G_i = (N_i, N \cup T, S_i, P_i)$. Note that these grammars have as the terminal alphabet the total alphabet of G. The grammars from such a tuple work in parallel, over separate sentential forms, also observing the restriction imposed by a relation $\rho \subseteq P_1 \times \dots \times P_k$. In a derivation of the k-tuple of grammars $(G_1, \dots, G_k)$ we have to use rules related by ρ. The k-tuples $(w_1, \dots, w_k)$ of strings generated in this way by $(G_1, \dots, G_k)$ are then used for selecting from the derivation trees of G only those trees which have paths starting from a given node which are marked by $w_1, \dots, w_k$.

Rather complex, but the reader can see here the idea of tuple grammars, investigated later in a series of papers by several authors, the idea of a matrix grammar (in the tuples of rules specified by the relation ρ), the idea of tree controlled grammars (we refer to Dassow and Păun (1989) for references), and to some extent, the idea of multi-component TAGs (Joshi 1987), where a set of auxiliary trees may be adjoined to an elementary tree. Of course, there are also essential differences between all these grammars.

We do not persist here in this direction, but we observe that the restriction from Bellert (1965) also contains an idea which we do not know to have been investigated in the regular rewriting area: to impose some restriction on the paths present in a derivation tree of a context-free grammar. This is the new–old restriction which we discuss here.

In short, we take two context-free grammars, G_1, G_2, where G_2 generates a language over the total alphabet of G_1. A string w generated by G_1 is accepted only if there is a derivation tree τ of w with respect to G_1 such that there exists a path in τ, from the root to a leaf node, which is marked by a string which is in $L(G_2)$. We say that such a pair $\gamma = (G_1, G_2)$ is a *path-controlled grammar.*

For an intuitive representation of this idea, one should have in mind the fact that a string w is generated by $\gamma = (G_1, G_2)$ by considering two derivation trees, τ_1, τ_2, the first one with respect to G_1 and the second one with respect to G_2; the first tree has w as the frontier, while the second tree is "orthogonal" to the first tree and its frontier string describes a path in the first tree. Figures 1, 2 contain illustrations of this idea.

Here we mainly investigate the generative power and the linguistically oriented properties of path-controlled grammars. When one of the grammars G_1, G_2 is regular, we do not obtain new families of languages, but more interesting are the cases when the two grammars are both linear or context-free. We prove that in the first case we get a family of mildly context-sensitive languages (a pumping lemma is used for obtaining the bounded growth property, while the polynomial parsability is obtained for the case when G_1 has a bounded degree of ambiguity), strictly included into the family of languages generated by matrix grammars (actually, this is true also for the case of using context-free grammars), not able to "count to five", but able to cope with the three basic non-context-free contructions: replication, crossed dependencies, and multiple agreements.

We close this introduction by also pointing out the remote similarity of path-controlled grammars (G_1, G_2), where the grammar G_1 is linear, with a variant of contextual grammars (Marcus 1969), the so-called external contextual grammars with a control language (Păun 1997), where contexts are added in the ends of a string, like in a linear grammar, in a sequence prescribed by a language on a set of labels for these contexts.

2 Path-Controlled Grammars

We use the standard formal language theory notions and notations, as available in many monographs. In particular, V^* is the free monoid generated by the alphabet V under the operation of concatenation, λ is the empty string, $|x|$ is the length of the string $x \in V^*$, and REG, LIN, CF are the families of regular, linear, and context-free languages, respectively. A grammar is given in the form $G = (N, T, S, P)$, where N is the nonterminal alphabet, T is the terminal alphabet, S is the axiom, and P is the set of rewriting rules. As usual, when comparing two languages, the empty string is ignored, that is, we consider L_1 equal to L_2 if $L_1 - \{\lambda\} = L_2 - \{\lambda\}$.

Given a context-free grammar $G = (N, T, S, P)$, with derivations in G we associate derivation trees in the well-known manner. Remember that each node in such a tree is marked by an element of N, with the exception of leaves, which are marked with elements of T; the root is marked with S. Let $S \Longrightarrow^* w$ be a terminal derivation in G and τ its associated tree. Each path from the root of τ to a leaf is described by a string of the form $SA_1, A_2, \ldots A_r a$, with $A_i \in N, 1 \leq i \leq r, r \geq 0$, and $a \in T$. We denote by $path(\tau)$ the language of all these strings, describing paths in τ, by $path(x)$ the union of all the languages $path(\tau)$, where τ is a derivation tree for x in G, and by $path(G)$ the union of all these languages.

Proposition 1 *If G is a context-free grammar, then* $path(G) \in REG$.

Proof. Starting from $G = (N, T, S, P)$, we construct the regular grammar

$G' = (\{S'\} \cup \{[A] \mid A \in N\}, N \cup T, S', P')$, with

$$\begin{aligned} P' &= \{S' \to S[S]\} \cup \{[A] \to B[B] \mid A \to uBv \in P,\ u,v \in (N \cup T)^*\} \\ &\cup \ \{[A] \to a \mid A \to uav \in P,\ u,v \in (N \cup T)^*\}. \end{aligned}$$

The equality $path(G) = L(G')$ is obvious, hence $path(G) \in REG$. ■

Let us now introduce the control mechanism we have announced: a *path-controlled* grammar is a pair $\gamma = (G, G')$, where $G = (N, T, S, P)$ and $G' = (N', N \cup T, S', P')$ are context-free grammars. The language generated by γ is

$$L(\gamma) = \{w \in L(G) \mid path(w) \cap L(G') \neq \emptyset\}.$$

That is, we accept a string w generated by G only if there is a derivation tree for w such that at least one path in this tree is described by a string which can be generated by G'. In other words, the language generated by γ is the yield of the tree language obtained from the derivation trees of G_1 which are "accepted" by G_2.

We denote by $PC(F_1, F_2)$ the family of languages $L(\gamma)$ generated by path-controlled grammars $\gamma = (G, G')$ with G of type F_1 and G' of type F_2, where F_1, F_2 can be one of *REG, LIN, CF*.

In Proposition 5 we will consider three (standard) examples, so we pass directly to investigating the power of path-controlled grammars.

When the control on paths is imposed only by regular grammars, we do not increase the power of regular, linear, or context-free grammars:

Proposition 2 $F = PC(F, REG)$, *for all* $F \in \{REG, LIN, CF\}$.

Proof. By taking G' a grammar which generates the language $(N \cup T)^*$, for any $\gamma = (G, G')$ one obtains $L(\gamma) = L(G)$, that is, $F \subseteq PC(F, REG)$ for all families F.

Also the converse inclusion is true: Let $\gamma = (G, G')$ be a path-controlled grammar with $G = (N, T, S, P)$ of type F and G' regular. Let $FA = (K, N \cup T, s_0, fin, \delta)$ be a deterministic finite automaton for the language $L(G')$ with the mapping δ totally defined. We construct the grammar $G'' = (N \times K, T \cup \{c\}, (S, s_0), P'')$, where c is a new symbol,

$$\begin{aligned} P'' = \{(A,s) \to X_1', \ldots X_n', \mid A \to X_1 \ldots X_n \in P, n \geq 1, X_i \in N \cup T, \text{ and} \\ X_i' = \begin{cases} (X_i, \delta(A,s)), & \text{if } X_i \in N, \\ X_i c, & \text{if } X_i \in T, \delta((A,s), X_i) \in fin, 1 \leq i \leq n\}. \\ X_i, & \text{otherwise}, \end{cases} \end{aligned}$$

From this definition, one sees that the nonterminals of G'' together with the rules from P'' check the existence of strings describing paths in the derivation tree under construction which are in $L(G')$. Clearly, a path exists in a derivation tree of frontier w, with respect to G, if and only if a string having at least one occurrence of the new letter c, which can be reduced to w by removing all the occurrences of c, appears in the language generated by G''. We now define a gsm which removes all the occurrences of c in the input string and terminates successfully if at least one occurrence of c is removed. Consequently, the image of $L(G'')$ by this gsm is exactly $L(\gamma)$. Since all the families mentioned in the theorem statement are closed under gsm mappings, the proof is complete. ■

When the generating grammar of a path-controlled grammar is regular, the obtained language can be produced by a grammar of the same type as the control grammar.

Proposition 3 *$PC(REG, F) \subseteq F$, for all $F \in \{LIN, CF\}$.*

Proof. We present a proof for $F = LIN$ only, the case when $F = CF$ can be handled in a similar way. Let $\gamma = (G, G')$ be a path-controlled grammar with $G = (N, T, S, P)$ regular and G' a linear grammar such that $L(G') \subseteq \{S\}N^*T$. We consider the linear grammar $G_1 = (N_1, \{(A, B), (A, a) \mid A, B \in N, a \in T\}, S_1, P_1)$, $N_1 \cap N = \emptyset$, which generates the language

$$L(G_1) = \{(S, A_1)(A_1, A_2) \dots (A_{n-1}, A_n)(A_n, a) \mid SA_1A_2 \dots A_n a \in L(G')\}.$$

We now construct the linear grammar $G_2 = (N_2, T, S_2, P_2)$, where $N_2 = N \cup N_1 \cup \{S_2\}$ and P_2 contains the following rules:

1. $S_2 \to S_1$, and $A \to S_1$, for all $A \in N$,
2. $S_2 \to Aa$, for all $A \to a \in P$, and $B \to Aa$, for all $A \to aB \in P$,
3. $A \to s(x)$, where $A \to x \in P_1$ and s is a finite substitution defined by $s((A, B)) = \{a \in T \mid A \to aB \in P\}$, $s((A, a)) = \{a \in T \mid A \to a \in P\}$, and $s(X) = \{X\}$, for all $X \in N_2$.

We obtain $L(G_2) = L(\gamma)$. If $w \in L(\gamma)$, then there exists a derivation

$$S \Longrightarrow a_1A_1 \Longrightarrow a_1a_2A_2 \Longrightarrow \dots \Longrightarrow a_1a_2 \dots a_{n-1}A_{n-1} \Longrightarrow$$
$$a_1a_2 \dots a_nB_1 \Longrightarrow a_1a_2 \dots a_na_{n+1}B_2 \Longrightarrow \dots \Longrightarrow a_1a_2 \dots a_{n+m} = w$$

and $SA_1A_2A_{n-1}a_n \in L(G')$. It follows that $(S, A_1)(A_1, A_2) \dots (A_{n-1}, a_n) \in L(G_1)$. Moreover, the following derivation exists in G_2:

$$S_2 \Longrightarrow B_m a_{n+m} \Longrightarrow \dots \Longrightarrow B_1 a_{n+1} a_{n+2} \dots a_{n+m} \Longrightarrow$$
$$S_1 a_{n+1} a_{n+2} \dots a_{n+m} \Longrightarrow^* a_1 \dots a_n a_{n+1} \dots a_{n+m} = w.$$

The converse inclusion is obtained in the same way. ■

There are linear languages which cannot be generated by any path-controlled grammar $\gamma = (G, G')$, with G a regular grammar. Indeed, let us consider the linear language $L = \{a^nb^n \mid n \geq 1\} \cup \{a\}$, and assume that $\gamma = (G, G')$ generates L.

Clearly, Sa must be in $L(G')$, hence any derivation tree in G has a path described by a string, Sa, in $L(G')$, that is $L(\gamma) = L(G)$, a contradiction. However, the following result holds.

Proposition 4 *If L is a language in $F \in \{LIN, CF\}$ without words of length one, then $L \in PC(REG, F)$.*

Proof. Let us consider the grammar $G = (N, T, S, P)$, and the morphism $h : T^* \to \{[a] \mid a \in T\}^*$ defined by $h(a) = [a]$ for $a \in T$. For each $a, b \in T$, let $\partial^r_{ab}(L) = \{w \mid wab \in L\}$. We consider the language

$$L' = \{S\}(\bigcup_{a,b \in T} h(\partial^r_{ab}(L))\{(a)b\}).$$

Clearly, $L' \in F$; let $G' = (N', \{[a], (a) \mid a \in T\} \cup T \cup \{S\}, S, P')$ be a grammar of type F for this language. We also consider the regular grammar $G = (\{[a], (a) \mid a \in T\} \cup \{S\}, T, S, P)$, with the following rules:

$$P = \{S \to a[a],\ S \to a(a),\ [a] \to b[b],\ [a] \to b(b),\ (a) \to b \mid a, b \in T\}.$$

The grammar G generates the language T^* (remember that we ignore the empty string), and from all the derivation trees only paths which are described by strings of the form $S[a_1][a_2]\dots[a_n](a_{n+1})b$, for $a_i \in T, 1 \le i \le n+1, b \in T$, where $n \ge 1$, are in the language L'. By the definition of L', the string $a_1a_2\dots a_na_{n+1}b$ belongs to L. However, this is exactly the string generated in the grammar G by the tree which has a path marked with $S[a_1][a_2]\dots[a_n](a_{n+1})b$. Consequently, for $\gamma = (G, G')$ we have $L(\gamma) = L$. ■

Thus, from the generative point of view, the path-controlled grammars γ of types (F_1, F_2) with at least one of F_1, F_2 equal to REG are not very interesting. This is definitely not the case when F_1, F_2 are equal to LIN or to CF. It is known that the set of all derivation trees of a context-free grammar is a regular tree language and the yield of any regular tree language is context-free, see (Thatcher 1967). Next proposition states that the set of derivation trees of a linear grammar accepted by another linear grammar is not a regular tree language anymore.

At the same time we point out that the three usually discussed non-context-free constructions from natural language, duplication, crossed dependencies, and multiple agreements (up to four related positions) can be handled by path-controlled grammars of type (LIN, LIN).

Proposition 5 *The languages $L_1 = \{a^nb^nc^nd^n \mid n \ge 1\}$, $L_2 = \{a^nb^ma^nb^m \mid n, m \ge 1\}$, $L_3 = \{xcx \mid x \in \{a,b\}^+\}$ are in $PC(LIN, LIN)$.*

Proof. Let us consider the linear grammars:

$$G_1 = (\{S, B, D\}, \{a, b, c, d\}, S, P_1), \text{ with}$$
$$P_1 = \{S \to aSd, S \to aBd, B \to bBc, B \to D, D \to bc\},$$

$$\begin{aligned}
G_2 &= (\{S, A, B, C, D\}, \{a, b\}, S, P_2), \text{ with} \\
P_2 &= \{S \to aS, S \to aB, B \to Bb, B \to A, A \to bA, A \to C, \\
&\quad C \to Ca, C \to D, D \to a\}, \text{ and} \\
G_3 &= (\{S, A, B, C\}, \{a, b, c\}, S, P_3), \\
P_3 &= \{S \to Aa,\ A \to Sb,\ S \to B,\ B \to aC,\ C \to bB,\ B \to c\},
\end{aligned}$$

as well as the linear grammars G'_1, G'_2, G'_3 generating the languages

$$\begin{aligned}
&C_1 = \{S^n B^n Db \mid n \geq 1\}, \quad C_2 = \{S^n B^m A^m C^n Da \mid n, m \geq 1\}, \\
&C_3 = \{w\, h(mi(w))c \mid w \in \{S, A\}^*\},
\end{aligned}$$

respectively, where h is the morphism defined by $h(S) = B, h(A) = C$, and $mi(x)$ is the mirror image (the reversal) of x.

One can see that we have the equalities $L((G_i, G'_i)) = L_i, i = 1, 2, 3$.

The first two cases are left to the reader and we discuss here only the last one. Consider a terminal derivation in the grammar G_3. If the path from S to c in the associated derivation tree is in the language C_3, then it is described by a string of the form $S\alpha_1\alpha_2\ldots\alpha_k B\beta_k\ldots\beta_2\beta_1 c$, where $\alpha_i \in \{S, A\}, \beta_i \in \{B, C\}$, such that $\beta_i = h(\alpha_i), 1 \leq i \leq k$. This means that the string $\beta'_k\ldots\beta'_2\beta'_1 c\alpha'_k\ldots\alpha'_2\alpha'_1$ is in the language $L(\gamma)$, where $\beta'_i = \alpha'_i = a$ if $\beta_i = S$ (hence $\alpha_i = B$), and $\beta'_i = \alpha'_i = b$ if $\beta_i = A$ (hence $\alpha_i = C$). This proves that all strings in $L(\gamma)$ are of the form $wcw, w \in \{a, b\}^*$. ■

However, the families $PC(F, F), F \in \{LIN, CF\}$, are not "too large". First, let us point out an "upper approximation" for them, the classic family MAT of languages generated by matrix grammars.

Proposition 6 $PC(CF, CF) \subseteq MAT$.

Proof. Let $\gamma = (G, G')$ be a path-controlled grammar with $G = (N, T, S, P)$, $G' = (N', N \cup T, S', P')$ being context-free grammars; without loss of the generality we may assume that $L(G') \subseteq \{S\}N^*T$. We define the matrix grammar $G'' = (N'', T'', S'', M)$, with

$$\begin{aligned}
N'' &= N \cup N' \cup \{\bar{X}, \hat{X} \mid X \in N \cup T\} \cup \{S''\}, \\
T'' &= T \cup \{[A, B] \mid A \in N, B \in N \cup T\}, \\
M &= \{(S'' \to \bar{S}S')\} \cup \{(r) \mid r \in P\} \cup \{(X \to h(x)) \mid X \to x \in P'\} \\
&\cup\ \{(\bar{A} \to u\bar{X}v, \hat{A} \to [A, X]) \mid A \in uXv \in P, X \in N \cup T\} \\
&\cup\ \{(\bar{a} \to a, \hat{a} \to a) \mid a \in T\}
\end{aligned}$$

where h is a morphism from $(N' \cup N \cup T)^*$ into $(N' \cup \{\hat{X} \mid X \in N \cup T\})^*$ defined by $h(X) = X$, if $X \in N'$, and $h(X) = \hat{X}$, if $X \in N \cup T$.

The idea behing this construction is the following: we start by constructing a string of the form $\bar{S}\hat{A}_1\hat{A}_2\ldots\hat{A}_k\hat{a}$, where $A_1A_2\ldots A_k a \in L(G_2)$; then we start a derivation in the grammar G_1 taking care that always when rewriting a barred symbol $\bar{A}_i$ which introduces a barred symbol $\bar{B}_i$ (always there is exactly one barred symbol), the corresponding symbol $\hat{A}_i$ is replaced by $[A_i, B]$. The derivation is finished only after removing all symbols with a hat. If in the obtained string the couple symbols $[A_i, B_i]$ are such that $B_i = A_{i+1}$, this means that the rules used in the derivation with respect to G_1 which contain barred symbols have followed a path which is described correctly by the string $A_1A_2 \ldots A_k a \in L(G_2)$ constructed initially.

A more formal counterpart of this explanation can be given.

Claim. *A derivation $S \Longrightarrow^* xay, a \in T, x, y \in T^*$, with respect to G, with the derivation tree containing a path described by $A_1A_2 \ldots A_m a \in L(G')$, exists if and only if $x\bar{A}_i y[A_1, A_2][A_2, A_3] \ldots [A_{i-1}, A_i]\hat{A}_i\hat{A}_{i+1} \ldots \hat{A}_m\hat{a} \in L(G'')$.*

We leave to the reader the technical task of proving (by induction on i) this claim, and we now define a gsm M which reads and leaves unchanged the prefix of its input string till the first symbol $[A, B]$ is met, then removes the rest of the input string, checking at the same time the following conditions (if one of them is not satisfied the machine is blocked and no output is produced): (i) For each consecutive symbols $[A, B][C, D]$ on the input tape, $B = C$ must hold. (ii) The last two symbols of the input string are $[A, a]a$ for some $A \in N$ and $a \in T$.

By the previous claim and since each terminal string produced by the matrix grammar is of the form $x[A_1, A_2][A_3, A_4] \ldots [A_{2n-1}, a]b$, $x \in T^*$, $a, b \in T$, $A_i \in N, 1 \leq i \leq 2n-1$, $A_1 = S$, it follows that the image of $L(G'')$ through the gsm M is equal to $L(\gamma)$. Because the family of matrix languages is closed under gsm mappings, the proof is complete. ■

Unfortunately, the fact that each path-controlled language is a matrix language does not say too much, because not so many (linguistically appealing) properties of matrix languages are know; in particular, no pumping property is known for matrix languages. Fortunately, such a property can be directly obtained for languages in the family $PC(CF, CF)$, with a stronger form for $PC(LIN, LIN)$.

Proposition 7 *If $L \subseteq V^*, L \in PC(CF, CF)$, then there are two constants p and q such that each string $z \in L$ with $|z| > p$ can be written in the form $z = u_1v_1u_2v_2u_3v_3u_4v_4u_5$, such that $0 < |v_1v_2v_3v_4| \leq q$ and $u_1v_1^iu_2v_2^iu_3v_3^iu_4v_4^iu_5 \in L$ for all $i \geq 1$.*

Proof. Consider a path-controlled grammar $\gamma = (G, G')$, with context-free components $G = (N, T, S, P), G' = (N', N \cup T, S', P')$.

It is clear that without any loss of the generality, we may suppose that the grammar G' contains no λ-rule and no chain rule (rule of the form $A \to B$, for A, B nonterminals) – we deal with the language $L(G')$, not directly with the grammar G', hence we can apply classic erasing and chain rule reduction algorithms in order to obtain a grammar equivalent with G' and having these properties.

Now, if $L(\gamma)$ is finite, then the proposition holds by default. Assume that $L(\gamma)$ is infinite, and consider an arbitarily long string in $L(\gamma)$. (Because of space restriction, we give here only a partially formal argument – based on the so-called pigeon hole principle –, which can be followed on the graphical representation from Figure 1.) Consider a derivation tree τ for z with respect to the grammar G which has a path described by a string from the language $L(G')$. Denote by δ this path, and let $\alpha = SA_1A_2 \ldots A_k a$ be the string from $L(G')$ which describes it, with $A_i \in N, 1 \le i \le k, a \in T$.

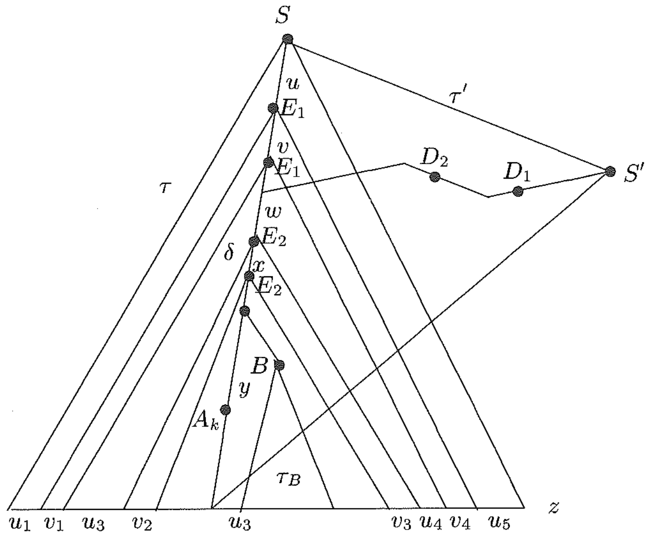

Figure 1

Consider the rules $A_i \to x_i A_{i+1} y_i$ used when passing from A_i to A_{i+1} on this path, as well as the rule $A_k \to x_k a y_k$ used in the last step of the derivation in G corresponding to the path δ. If any $x_i y_i$ contains a nonterminal B such that the substring of z derived from this nonterminal is long enough, then in the subtree τ_B we can find a recurrent nonterminal: there is a path from B to a leaf where a nonterminal $C \in N$ appears twice, hence the derivation in-between the two occurrences of C can be iterated – note that this path, as all paths in the subtree τ_B, is not under the control of G', because this subtree is independent of the path δ and the path δ ensures the correctness of all trees obtained from τ where the path δ is present). In this way, we have obtained a pair of pumped substrings of z in the usual way for context-free languages. The fact that the pumped substrings are bounded in length is ensured by the similar property from the pumping lemma for context-free languages.

Assume now that there is no such a nonterminal B. On the one hand, this means that all derivations starting from the nonterminals which are not appearing on the path δ are of a bounded length – denote this bound by r. On the other hand, the previous assumption implies that the path δ, and hence the string α describing it, are arbitrarily long.

This means that we can find a symbol $D \in N'$ which is repeated arbitrarily many times on a path in the tree τ'. Thus, we can find at least two such occurences of D, let us denote them by D_1, D_2, such that the corresponding subderivations $D_i \Longrightarrow^* \beta_{i,1} D_i \beta_{i,2}$ from τ' have $\beta_{i,1} = E_1, \beta_{i,2} = E_2, i = 1, 2$.

Consequently, the subderivation $D_1 \Longrightarrow^* E_1\beta_{1,1}D_1E_2\beta_{2,1}$ can be iterated in the grammar G' and, also, the derivations corresponding to the substrings $E_1\beta_{1,1}$ and $E_2\beta_{2,1}$ of α can be iterated in G.

Therefore, we have found two substrings v and x of α such that $\alpha = uvwxy$ and the following two assertions hold: (1) $uv^iwx^iy \in L(G'), i \geq 1$, (2) for each i we can iterate a subderivation of the derivation in G described by τ such that the subderivations described by v and x are repeated.

Now, if we iterate the subderivations of τ which correspond to the iteration of v and x in the string α, this will lead to the pumping of four substrings of z, two in the left hand of the path δ and two in the right hand of this path (see again Figure 1). However, we have assumed that the derivations starting from nonterminals which are placed outside the path δ produce only substrings of a length bounded by the constant r. Also the strings v, x pumped in α can be bounded. Therefore, also the substrings of z which are obtained by starting from the nonterminals of v and x are bounded in length by a given constant, that is, the total length of the four pumped substrings of z is bounded by a constant, q. This concludes the proof. (The reader with a mathematical background can easily fill in the combinatorial arguments described in a rather informal manner here, following the classic proof of pumping lemma for context-free grammars). ■

The classic pumping lemma for context-free languages has a special form for linear languages: if $L \in LIN$, then there are $p, q \in \mathbf{N}$ such that each string $z \in L$ with $|z| > p$ can be written in the form $z = uvwxy$, with $0 < |vx| \leq |uvxy| \leq q$, such that $uv^iwx^iy \in L$ for all $i \geq 1$ (that is, the pumped positions can be chosen at a bounded distance from the ends of the string). As it is easy to see from the proof of Proposition 7, the same property can be obtained also for languages in the family $PC(LIN, LIN)$, just by choosing the iterated nonterminals D which are the closest to the root of the tree τ' with respect to G' (then, the iterated substrings of α will be close to the ends of the string, hence two of the iterated substrings of z will be close to the ends of z, too). Thus, we obtain:

Proposition 8 *If* $L \subseteq V^*, L \in PC(LIN, LIN)$, *then there are two constants* p *and* q *such that each string* $z \in L$ *with* $|z| > p$ *can be written in the form* $z = u_1v_1u_2v_2u_3v_3u_4v_4u_5$, *such that* $0 < |v_1v_2v_3v_4| \leq q$, $|u_1v_1v_4u_5| \leq q$, *and* $z_i = u_1v_1^iu_2v_2^iu_3v_3^iu_4v_4^iu_5 \in L$ *for all* $i \geq 1$.

There are several **consequences** of the previous pumping lemmas:

• The inclusion $PC(CF, CF) \subset MAT$ is proper. An example of a matrix language which does not have the pumping property from Proposition 7 is $\{a_1^na_2^na_3^na_4^na_5^n \mid n \geq 1\}$. This language also illustrates the next assertion.

• The path-controlled grammars, even with context-free components, "cannot count to five" (but, as we have seen in Proposition 5, path-controlled grammars with linear components "can count to four").

• The languages in the family $PC(CF, CF)$ (and hence also those from the family $PC(LIN, LIN)$ have the bounded growth property, for each infinite language $L \in PC(CF, CF)$ there is a constant r such that for each string $z \in L$ there is $z' \in L$ such that $|z'| - |z| \leq r$. This is important from a linguistical point of view, as languages with "jumps in length" do not look "natural", see (Joshi 1985).

• The family $PC(LIN, LIN)$ is not closed under concatenation: the language $L = \{a_1^n a_2^n a_3^n a_4^n \mid n \geq 1\}$ is in $PC(LIN, LIN)$ (Proposition 5), but LL is not, because it does not fulfill the condition in the pumping lemma for this family, see Proposition 8.

This last language is not context-free, but there also are context-free languages which are not in $PC(LIN, LIN)$. This can be proved by using another classic necessary condition for a language to be linear: if $L \subseteq V^*, L \in LIN$, then there are two regular languages L_1, L_2 such that $L \subseteq L_1L_2$ and for each string $x \in L_1$ (each string $y \in L_2$) there is a string $y \in L_2$ (a string $x \in L_1$, respectively) such that $xy \in L$.

Proposition 9 *For each language $L \subseteq V^*, L \in PC(LIN, LIN)$, there are three linear languages $L_1 \subseteq V^*\{c\}V^*, L_2 \subseteq V^*\{c\}V^*, L_3 \subseteq V^*$, where $c \notin V$, such that:*

(i) $L \subseteq \{u_1u_2u_3u_4u_5 \mid u_1cu_5 \in L_1, u_2cu_4 \in L_2, u_3 \in L_3\}$,

(ii) *For each string $u_1cu_5 \in L_1$ (for each string $u_2cu_4 \in L_2$, for each string $u_3 \in L_3$) there are a string $u_2cu_4 \in L_2$ and a string $u_3 \in L_3$ (a string $u_1cu_5 \in L_1$ and a string $u_3 \in L_3$, respectively, a string $u_1cu_5 \in L_1$ and a string $u_2cu_4 \in L_2$) such that $u_1u_2u_3u_4u_5 \in L$.*

Proof. Instead of a completely formalized proof, we choose again to use a picture to explain the reasoning. Consider the situation illustrated in Figure 2, for a given path-controlled grammar $\gamma = (G, G')$ with linear grammars G, G', $G = (N, T, S, P)$.

We have a derivation tree τ with respect to the grammar G, where a path δ exists which is described by a string α from $L(G')$. Assume that this path ends when using a rule $B \rightarrow uay$. The language L_3 we look for is the union of all languages generated by the linear grammars $G_B = (N, T, B, P)$, B as above.

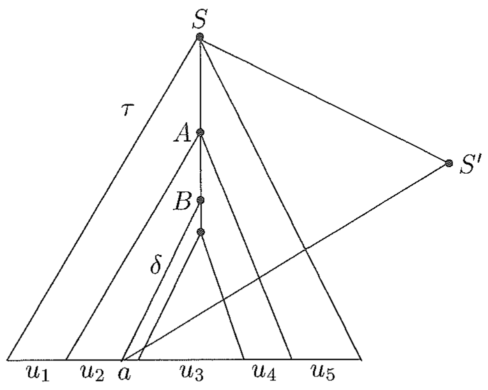

Figure 2

Now, consider the previously mentioned property for the linear language $L(G')$. Two regular languages R_1, R_2 exist such that $L(G') \subseteq R_1R_2$, and all strings from R_1, R_2 are useful, in the sense specified above. We construct two path-controlled grammars $\gamma_1 = (G_1, G_1'), \gamma_2 = (G_2, G_2')$, where

$$G_1 = (N, T \cup \{c\}, S, P \cup \{A \to c \mid A \in N\}),$$
$$G_2 = (N, T \cup \{c\}, S_0, P \cup \{S_0 \to A, A \to c \mid A \in N\}),$$

and G_1', G_2' are regular grammars generating the languages $R_1\{c\}$ and $\{S_0\}(\bigcup_{a \in T} \partial_a^r(R_2))\{c\}$, respectively. Then, $L_1 = L(\gamma_1)$ and $L_2 = L(\gamma_2)$ have the desired properties. First, in view of Proposition 2, the languages L_1, L_2 are linear. Then, each string from $L(\gamma)$ can be obtained by interleaving strings from the three languages L_1, L_2, L_3: from the property of languages R_1, R_2, the path δ is cut into two parts, each one from one of the regular languages R_1, R_2. The definition of the path-controlled grammars γ_1, γ_2 follows this separation according to R_1, R_2. The derivations in G_1 start as any derivation with respect to G from S and end by introducing the symbol c. The derivations in G_2 start from the new axiom S_0, which directly introduces any nonterminal of G, and also ends by introducing the symbol c. These two derivations can be "concatenated", ignoring the symbols c, and can be completed with a derivation corresponding to the linear language L_3, so that the obtained derivation is correct with respect to γ. Because no string from R_1, R_2 is useless, no string from L_1, L_2 is useless; the same is true for the strings of L_3. ■

From this proposition we get two important consequences:

Proposition 10 (i) *$CF - PC(LIN, LIN) \neq \emptyset$.* (ii) *The inclusion $PC(LIN, LIN) \subset PC(CF, CF)$ is proper.*

Proof. The context-free language $\{a^{n_1}b^{n_1}a^{n_2}b^{n_2}\ldots a^{n_6}b^{n_6} \mid n_1, \ldots, n_6 \geq 1\}$ does not have the property from Proposition 9, as one can easily check. ■

Consequently, the family $PC(LIN, LIN)$ is incomparable with the family of context-free languages (and includes the family of linear languages). This corresponds to the speculation that this is probably the case with the classes of grammars which are adequate for modeling the syntax of natural language, see, e.g., (Manaster-Ramer 1999; Marcus, Martin-Vide and Păun 1998).

One of the most important properties of any class of grammars which aims to be considered a model of syntax is the polynomial parsability.

This can be achieved for path-controlled grammars $\gamma = (G, G')$ with linear components and such that G_1 *has a bounded ambiguity* (for each string in the language $L(G)$ there are at most k different derivation trees, for a given constant k) in the following manner. Let $G = (N, T, S, P), G' = (N', N \cup T, S, P')$. Assume that there is no derivations of the form $A \Longrightarrow^* B$ with respect to G (if such derivations exist, then we remove all chain rules from G by the well known algorithm, and this requires a polynomial time). We now construct a gsm M which scans strings from $\{S\}N^*T$ and, nondeterministically, for each symbol $A \in N$ either leaves it unchanged, or removes it provided that tere is a rule $B \rightarrow A$ in P and B was just scanned before A (this can be "remembered by the state of M). Therefore, substrings which correspond to recurrent derivations in G which produce nothing are either erased or not (the second case prevents the possibility to also have productive derivations with the same nonterminals). Consider the language $M(L(G'))$. Because LIN is closed under gsm mappings, this language is linear; let G'' be a grammar for it and consider the path-controlled grammar $\gamma' = (G, G'')$. The equality $L(\gamma) = L(\gamma')$ is obvious (note that we have $L(G') \subseteq M(L(G')))$.

Consider a string $w \in T^*$. We construct as above the path-controlled grammar γ' (this takes a polynomial time). The string w is generated by γ' if and only if there is a tree with respect to G of height at most $card(N)|w|$. Thus, we first decide whether or not $w \in L(G)$ (this can be done in polynomial time). If $w \notin L(G)$, then $w \notin L(\gamma)$. If $w \in L(G)$, then we consider all the derivation trees τ for w with respect to G. Because we have assumed that G has the degree of ambiguity at most k, the number of such trees τ at most k.

For each tree, we have $|w|$ paths from the axiom S to a leaf node. For each such a path we check whether or not it is described by a string in $L(G'')$. Because the height of the derivation trees we consider are bounded, the length of the paths in these trees are bounded, hence the membership of the associated strings to the language $L(G'')$ can be decided in polynomial time (with respect to the length of the strings, which in turn is polynomially bounded with respect to the length of w). If at least such a path, for at least a tree τ', exists, then $w \in L(\gamma)$, otherwise $w \notin L(\gamma)$.

In total, the decision takes a polynomial time. We formulate this important conclusion as a proposition.

Proposition 11 *If $\gamma = (G, G')$ is a path-controlled grammar with linear components G, G' and such that G has a bounded ambiguity, then the parsing of strings in $L(\gamma)$ can be done in polynomial time.*

It is a natural (and interesting) *open problem* whether or not the previous proposition can be extended to all path-controlled grammars, that is, without imposing the bounded ambiguity condition.

3 Final Remarks

Starting from an "old" paper by I. Bellert, which has introduced a (complicated) class of regulated context-free grammars in the aim of formalizing constructions from natural languages (with a detailed application to Polish), we have considered a new type of restriction in derivation: a derivation tree in a context-free grammar is accepted only if it contains a path which is described by a string which can be generated by another context-free grammar. We have found many properties of such devices (generative power, pumping properties, properties related to the mildly context-sensitive concept), but still many problems remain to be solved, both of a language theoretic type – for instance, closure and decidability properties – and of a linguistical type – concerning the adequacy/relevance of these grammars. Irrespective of the answer to such questions, we believe that at least from a historical point of view, this new-old type of regulated mechanism deserves to be investigated and further examined.

As the control by considering only one path from a derivation tree does not look very restrictive (also for capturing linguistics constraints), stronger controls could be of interest. Here are two posibilities: to have *all* path from a derivation tree described by strings in a given language, or, an intermediate case, to select *some* paths (by a "pre-condition", for instance, a regular language) and to impose a condition as above only to the selected paths. The formal study of such variants remains to be carried out; we hope to return to this topic in a forthcoming paper.

References

Abraham, S. (1965). Some questions of phrase-structure grammars, *Computational Linguistics*, 4, pp. 61–70.

Bellert, I. (1965). Relational phrase structure grammar and its tentative applications, *Information and Control*, 8, pp. 503–530.

Chomsky, N. (1956). Three models for the description of language, *IRE Trans. in Information Theory*, IT-2, 3, pp. 113–124.

Chomsky, N. (1957). *Syntactic Structures*, Mouton, The Hague.

Dassow, J. and Păun, Gh. (1989). *Regulated Rewriting in Formal Language Theory*, Springer-Verlag (Berlin).

Higginbotham, J. (1984). English is not a context-free language, *Linguistic Inquiry*, 15, pp. 119–126.

Joshi, A.K. (1985). How much context-sensitivity is required to provide reasonable structural descriptions? Tree adjoining grammars, *in* Dowty, D.R. et al. (eds), *Natural Language Processing: Psycholinguistic, Computational, and Theoretic Perspectives*, Cambridge Univ. Press (New York), pp. 206–250.

Joshi, A.K. (1987). An introduction to tree adjoining grammars, *in* Manaster Ramer, A. (ed), *Mathematics of Language*, John Benjamins (Amsterdam).

Manaster Ramer, A. (1999). Uses and misuses of mathematics in linguistics, *in* Martín-Vide, C. (ed), *Issues in Mathematical Linguistics*, John Benjamin, Amsterdam, pp. 73–130.

Marcus, S. (1969). Contextual grammars, *Rev. Roum. Math. Pures Appl.*, 14, pp. 1525–1534.

Marcus, S., Martin-Vide, C., and Păun, Gh. (1998). Contextual grammars as generative models of natural languages, *Computational Linguistics*, 24, 2, pp. 245–274.

Păun, Gh. (1997). *Marcus Contextual Grammars*, Kluwer (Dordrecht).

Pullum, G.K. (1985). On two recent attempts to show that English is not a CFL, *Computational Linguistics*, 10, pp. 182–186.

Pullum, G.K. and Gazdar, G. (1982). Natural languages and context-free languages, *Linguistics and Philosophy*, 4, pp. 471–504.

Rounds, W.C., Manaster Ramer, A. and Friedman, J. (1987). Finding natural languages a home in formal language theory, *in* Manaster Ramer (ed), *Mathematics of language*, John Benjamins (Amsterdam, Philadelphia), pp. 349–360.

Salomaa, A. (1973). *Formal Languages*, Academic Press (New York).

Shieber, S.M. (1985). Evidence against the context-freeness of natural languages, *Linguistics and Philosophy*, 8, pp. 333–343.

Thatcher, J.W. (1967). Characterizing derivation trees of context-free grammars through a generalization of finite automata theory, *J. Comput. Systems Sci.*, 1, pp. 317–322.