E. Csuhaj-Varjú, Z. Ésik (Eds.): Automata and Formal Languages.
AFL 2008, Balatonfüred, Hungary, May 27-30, 2008. Proceedings, pages 158 - 169.

Subregularly Tree Controlled Grammars and Languages

Jürgen Dassow and *Bianca Truthe*
Otto-von-Guericke-Universität Magdeburg, Fakultät für Informatik
PSF 4120, D-39016 Magdeburg, Germany

Abstract

Tree controlled grammars are context-free grammars where the associated language only contains those terminal words which have a derivation where the word of any level of the corresponding derivation tree belongs to a given regular language. We present some results on the power of such grammars where we restrict the regular languages to some known subclasses of the family of regular languages.

1 Introduction

It is a well-known fact that the most investigated class of formal languages, the regular and context-free languages, are not able to cover all phenomena which are known from natural languages, programming languages etc. Thus, there have been introduced many grammars with a context-free core and some mechanism which controls the sequences of rules in a derivation or the applicability of a rule etc. (see [2] and [7]). One such control mechanism was introduced by Culik II and Maurer in [1] where the structure of the derivation trees is restricted by the requirement that all words belonging to a level of the derivation tree have to be in a given regular language. Păun proved that the generative power of these grammars, called tree controlled grammars, coincides with that of context-sensitive grammars (if erasing rules are forbidden) or arbitrary phrase structure grammars (if erasing rules are allowed). Among the classical decision problems only membership is decidable for context-sensitive languages which is known to be PSPACE-complete. But if one restricts the underlying context-free grammars to be unambiguous, then the membership problem can be solved in quadratic time and a lot of important non-context-free languages can be generated. Thus it is a natural question to consider restricted versions of tree controlled grammars. In this paper we discuss restrictions for the regular languages that contain the words of the levels of the derivation trees.

As classes of control languages, we regard some subclasses of the family of regular languages. These subfamilies are formed by

- finite languages,
- nilpotent languages (which are accepted by automata which do not change the state after a fixed number of transitions),
- combinational languages (which are accepted by automata modelling circuits),
- ordered languages (where the transitions of the accepting automata preserve an order on the state set),

- suffix-closed (or multiple-entry or fully-initial languages) languages (which are accepted by automata where the computation can start in any state),
- commutative languages (which are closed under permutations of the letters in a word),
- circular languages (which are closed under cyclic shifts of the letters in a word),
- non-counting (or star-free) languages (which can be described by expressions using only union, concatenation, and complement).

In most cases we show that the obtained language family coincides with some known language family as the context-sensitive languages, the E0L languages or matrix languages (of finite index); in the remaining cases we present some lower bounds for the generative power.

2 Definitions

Throughout the paper, we assume that the reader is familiar with the basic concepts of formal language theory; for details we refer to [7], [6], and [2].

For an alphabet $V = \{a_1, a_2, \ldots, a_m\}$ and a word $w \in V^*$, we define the Parikh vector $\pi_V(w) = (n(a_1), n(a_2), \ldots, n(a_m))$ where $n(a_i)$, $1 \leq i \leq m$, is the number of occurrences of a_i in w.

By FIN, REG, CF, CS, and RE we denote the families of finite, regular, context-free, context-sensitive languages, and recursively enumerable languages. For a language L over V, we set

$$Suf(L) = \{y \mid xy \in L \text{ for some } x \in V^*\},$$
$$Comm(L) = \{a_{i_1} \ldots a_{i_n} \mid a_1 \ldots a_n \in L, n \geq 1,\ \{i_1, i_2, \ldots, i_n\} = \{1, 2, \ldots, n\}\},$$
$$Circ(L) = \{a_{i+1}a_{i+2} \ldots a_n a_1 a_2 \ldots a_i \mid n \geq 1,\ 1 \leq i \leq n,\ a_1 a_2 \ldots a_n \in L\}.$$

It is known that $Suf(L)$ is regular for a regular language L.

With any derivation in a context-free grammar G, we associate a derivation tree. With any derivation tree t of height k and any number $0 \leq j \leq k$, we associate the words of level j and the sentential form of level j which are given by all nodes of depth j read from left to right and all nodes of depth j and all leaves of depth less than j read from left to right, respectively.

Obviously, if w and v are sentential forms of two successive levels, then $w \Longrightarrow^* v$ holds and this derivation is obtained by a parallel replacement of all nonterminals occurring in w.

A *tree controlled grammar* is a quintuple $G = (N, T, P, S, R)$ where

- (N, T, P, S) is a context-free grammar with a set N of nonterminals, a set T of terminals, a set P of context-free non-erasing rules, and an axiom S,
- R is a regular set over $(N \cup T)^*$.

The language $L(G)$ generated by a tree controlled grammar $G = (N, T, P, S, R)$ consists of all words $z \in T^*$ such that there is a derivation tree t where z is the word obtained by reading the leaves from left to right and the words of all levels of t – besides the last one – belong to R.

Let X be a subfamily of REG. Then we denote the family of languages generated by tree controlled grammars $G = (N, T, P, S, R)$ with $R \in X$ by $\mathcal{TC}(X)$.

Example 1. *As an example we consider the tree controlled grammar*

$$G_1 = (\{S\}, \{a\}, \{S \to SS, S \to a\}, S, \{S\}^+).$$

Since no level can contain the symbol S as well as a terminal a, first one has to replace any S by SS for a certain time before rewriting any S by a. Therefore the levels of an allowed derivation tree consist of the words $S, SS, SSSS, \ldots, S^{2^n}, a^{2^n}$ for some $n \geq 0$. Thus $L(G_1) = \{a^{2^n} \mid n \geq 0\}$.

Example 2. *The tree controlled grammar*

$$G_2 = (\{S, A, B, C, D, E\}, \{a, b\}, P, S, \{S, AB, aAbBa, aCba, Cb\})$$

with $P = \{S \to AB, A \to aAb, B \to Ba, A \to ab, B \to a, A \to aCb, C \to Cb, C \to b\}$ generates the language $L(G_2) = \{a^n b^{n+m} a^n \mid n \geq 1,\ m \geq 0\}$.

In [5] (see also [2]), it has been shown that a language L is generated by a tree controlled grammar if and only if it is generated by a context-sensitive (or monotone) grammar.

Theorem 3. ([5], [2]) $\mathcal{TC}(REG) = CS$. □

In this paper, we are interested in those language families which can be obtained from tree controlled grammars $G = (N, T, P, S, R)$ where the regular language belongs to some special subfamily of the family of regular languages. From the definition, the next statement follows immediately.

Lemma 4. *If $X \subseteq Y \subseteq REG$, then $\mathcal{TC}(X) \subseteq \mathcal{TC}(Y)$.* □

We consider the following restrictions for regular languages. Let L be a language and $V = alph(L)$ the minimal alphabet of L. We say that L is

- *combinational* iff it can be represented in the form $L = V^*A$ for some subset $A \subseteq V$,
- *definite* iff it can be represented in the form $L = A \cup V^*B$ where A and B are finite subsets of V^*,
- *nilpotent* iff L is finite or $V^* \setminus L$ is finite,
- *commutative* iff $L = Comm(L)$,
- *circular* iff $L = Circ(L)$,
- *suffix-closed* (or *fully initial* or *multiple-entry* language) iff $xy \in L$ for some words $x, y \in V^*$ implies $y \in L$ (or equivalently, $Suf(L) = L$),
- *non-counting* (or star-free) iff there is an integer $k \geq 1$ such that, for any words $x, y, z \in V^*$, $xy^k z \in L$ if and only if $xy^{k+1}z \in L$,
- *power-separating* iff for any $x \in V^*$ there is a natural number $m \geq 1$ such that either $J_x^m \cap L = \emptyset$ or $J_x^m \subseteq L$ where $J_x^m = \{x^n \mid n \geq m\}$,
- *ordered* iff L is accepted by some finite automaton $\mathcal{A} = (Z, V, \delta, z_0, F)$ where $(Z, \preceq)$ is a totally ordered set and, for any $a \in V$, $z \preceq z'$ implies $\delta(z, a) \preceq \delta(z', a)$.

It is obvious that combinational, definite, nilpotent and ordered languages are regular, whereas non-regular languages of the other above mentioned types exist.

By *COMB*, *DEF*, *NIL*, *COMM*, *CIRC*, *SUF*, *NC*, *PS*, and *ORD* we denote the families of all combinational, definite, nilpotent, regular commutative, regular circular, regular suffix-closed, regular non-counting, regular power-separating, and ordered languages, respectively. The relations between these language families are investigated e. g. in [4] and [8] and can be given by Figure 1. Moreover, we add the family *MON* of all languages of the form V^*, where V is an alphabet (languages of *MON* are target sets of monoids).

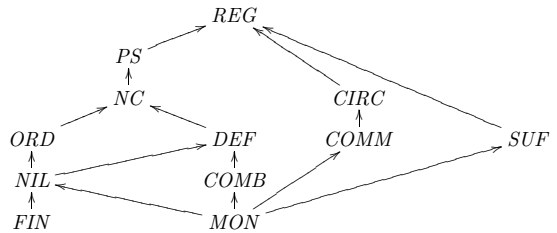

Figure 1: Hierarchy of subregular languages (an arrow from X to Y denotes $X \subsetneq Y$, and if two families are not connected by a directed path then they are incomparable)

A *matrix grammar* is a quadruple $G = (N, T, M, S, Q)$ where

- N, T, and S are specified as in a context-free (or tree controlled) grammar,
- $M = \{m_1, m_2, \ldots, m_r\}$ is a finite set of finite sequences m_i of context-free non-erasing rules, i. e., $m_i = (A_{i,1} \to v_{i,1}, A_{i,2} \to v_{i,2}, \ldots, A_{i,r_i} \to v_{i,r_i})$ for $1 \leq i \leq r$, (the elements of M are called matrices), and
- Q is a subset of the productions occurring in the matrices of M.

The application of a matrix m_i is defined as a sequential application of the rules of m_i in the given order where a rule of Q can be ignored if its left-hand side does not occur in the current sentential form, i. e., $x \Longrightarrow_{m_i} y$ holds iff there are words w_j, $1 \leq j \leq r_i + 1$ such that $x = w_1$, $y = w_{r_i+1}$ and, for $1 \leq j \leq r_i$,

$$w_j = x_j A_{i,j} y_j \quad \text{and} \quad w_{j+1} = x_j v_{i,j} y_j$$

or

$$w_j = w_{j+1} \text{ and } A_{i,j} \text{ does not occur in } w_j \text{ and } A_{i,j} \to v_{i,j} \in Q.$$

The language $L(G)$ generated by G consists of all words $z \in T^+$ such that there is a derivation

$$S \Longrightarrow_{m_{i_1}} v_1 \Longrightarrow_{m_{i_2}} v_2 \Longrightarrow_{m_{i_3}} \ldots \Longrightarrow_{m_{i_t}} v_t = z$$

for some $t \geq 1$.

For any matrix grammar $G = (N, T, M, S, Q)$, any derivation

$$D : S = w_0 \Longrightarrow_{m_{i_1}} w_1 \Longrightarrow_{m_{i_2}} w_2 \Longrightarrow_{m_{i_3}} \ldots \Longrightarrow_{m_{i_n}} w_n$$

in G and any word $z \in L(G)$, we define the indexes by

$$\begin{aligned} Ind(G,D) &= \max\{\#_N(w_i) \mid 0 \le i \le n\}, \\ Ind(G,z) &= \min\{Ind(G,D) \mid D \text{ is a derivation of } z\}, \\ Ind(G) &= \sup\{Ind(G,z) \mid z \in L(G)\}. \end{aligned}$$

By MAT and MAT_{fin}, we denote the families of all languages which can be generated by matrix grammars and matrix grammars G with a finite index $Ind(G)$, respectively.

An *extended interactionless L system* (abbreviated as E0L system) is a quadruple $G = (V, T, P, w)$ where V is an alphabet, T is a non-empty subset of V, w is a non-empty word over V and P is a finite subset of $V \times V^+$ such that, for any $a \in V$, there is at least one element $a \to v$ in P.

We say that $x \in V^+$ directly derives $y \in V^+$, written as $x \Longrightarrow y$, if $x = x_1 x_2 \dots x_n$ for some $n \ge 1$, $x_i \in V$, $1 \le i \le n$, $y = y_1 y_2 \dots y_n$ and $x_i \to y_i \in P$ for $1 \le i \le n$, i. e., any letter of x is replaced according to the rules of P. Thus the derivation process in an E0L system is purely parallel. By $\Longrightarrow^*$ we denote the reflexive and transitive closure of $\Longrightarrow$. The language $L(G)$ generated by G is defined as

$$L(G) = \{z \mid w \Longrightarrow^* z, z \in T^+\}.$$

By *E0L* we denote the families of languages which are generated by E0L systems.

3 Generating the Family of Context-Sensitive Languages

The context-sensitive languages are all generatable by tree controlled grammars with regular control sets that are suffix-closed, circular or ordered.

Theorem 5. $\mathcal{TC}(SUF) = CS$

Proof. Let L be a context-sensitive language over an alphabet T without the empty word (the modifications for the case with the empty word are left to the reader). Then

$$L = \bigcup_{a \in T} \{a\} L_a,$$

where $L_a = \{w \mid aw \in L\}$ is a context-sensitive language. By Theorem 3, for $a \in T$, there is a tree controlled grammar $G_a = (N_a, T, P_a, S_a, R_a)$ with $L(G_a) = L_a$.

We consider the tree controlled grammar $G = (N, T, P, S, R)$ with

$$\begin{aligned} N &= \{S\} \cup \bigcup_{a \in T} (\{A_a, C_a\} \cup N_a), \\ P &= \bigcup_{a \in T} (\{S \to A_a S_a, A_a \to A_a, A_a \to C_a, C_a \to a\} \cup P_a), \\ R &= \{S\} \cup \bigcup_{a \in T} Suf(\{A_a\} R_a \cup \{C_a\} T^*). \end{aligned}$$

By definition, R is suffix-closed and regular.

Obviously, the words w_i, $0 \le i \le n$, defined by the level i of a derivation tree of G have the forms $w_0 = S$, $w_1 = A_a w_1'$, $w_2 = A_a w_2', \dots, w_{n-1} = A_a w_{n-1}'$, $w_n = C_a z$, where $w_i' \in R_a$ for $1 \le i \le n-1$ and the generated word is $az \in \{a\} L_a \subseteq L$. Therefore it easily follows that $L(G) = L$. Thus $L \in \mathcal{TC}(SUF)$ which proves $CS \subseteq \mathcal{TC}(SUF)$. On the other hand, by Theorem 3 and Lemma 4, $\mathcal{TC}(SUF) \subseteq \mathcal{TC}(REG) = CS$. □

For the next theorem, we use the same idea[1].

Theorem 6. $\mathcal{TC}(CIRC) = CS$.

Proof. Let L be a context-sensitive language over an alphabet T. Then

$$L = L_1 \cup \bigcup_{(a,b) \in T^2} \{a\} L_{ab} \{b\},$$

where $L_1 = \{\, w \in L \mid |w| \le 1 \,\}$ is a finite language and $L_{ab} = \{w \in T^* \mid awb \in L\}$ is a context-sensitive language for any pair $(a, b) \in T^2$. By Theorem 3, for $(a, b) \in T^2$, there is a tree controlled grammar $G_{ab} = (N_{ab}, T, P_{ab}, S_{ab}, R_{ab})$ with $L(G_{ab}) = L_{ab}$.

We consider the tree controlled grammar $G = (N, T, P, S, R)$ with

$$\begin{aligned} N &= \{S\} \cup \bigcup_{x \in T} \{[_x, \tilde{[}_x,]_x, \tilde{]}_x\} \cup \bigcup_{(a,b) \in T^2} N_{ab}, \\ P &= \{\, S \to w \mid w \in L_1 \,\} \cup \bigcup_{(a,b) \in T^2} (\{S \to [_a S_{ab}]_b\} \cup P_{ab}) \\ &\quad \cup \bigcup_{x \in T} \{[_x \to [_x, [_x \to \tilde{[}_x, \tilde{[}_x \to x,]_x \to]_x,]_x \to \tilde{]}_x, \tilde{]}_x \to x\}, \\ R &= \{S\} \cup \bigcup_{(a,b) \in T^2} Circ(\{[_a\} R_{ab} \{]_b\} \cup \{\tilde{[}_a\} T^* \{\tilde{]}_b\}). \end{aligned}$$

By definition, R is circular and regular.

Obviously, the words w_i, $0 \le i \le n$, defined by the level i of a derivation tree of G have the forms $w_0 = S$ and $w_1 \in L_1$ or

$$w_0 = S,\ w_1 = [_a w_1']_b,\ w_2 = [_a w_2']_b, \dots, w_{n-1} = [_a w_{n-1}']_b,\ w_n = \tilde{[}_a z \tilde{]}_b,$$

where $w_i' \in R_{ab}$ for $1 \le i \le n-1$ and the generated word is $azb \in \{a\} L_{ab} \{b\} \subseteq L$. Therefore it easily follows that $L(G) = L$. Thus $L \in \mathcal{TC}(CIRC)$ which proves $CS \subseteq \mathcal{TC}(CIRC)$. By Theorem 3 and Lemma 4, we also have $\mathcal{TC}(CIRC) \subseteq CS$. □

Theorem 7. $\mathcal{TC}(ORD) = CS$

Proof. Let L be a context-sensitive language. Then there is a context-sensitive grammar $G = (N, T, P, S)$ in Kuroda normal form, i. e., $P = P_1 \cup P_2$ where all rules of P_1 are of the form $AB \to CD$, all rules of P_2 are of the form $A \to BC$ or $A \to B$

[1] Thanks to an anonymous referee.

or $A \to a$ with $A, B, C, D \in N$ and $a \in T$, such that $L(G) = L$. We construct the tree controlled grammar $\overline{G} = (\overline{N}, T, \overline{P}, S', R)$ with

$$\begin{aligned}
\overline{N} &= \{X' \mid X \in V\} \cup \{X_p \mid p \in P,\ X \in V\} \\
&\quad \cup \{A_{1,p}, B_{2,p} \mid p = AB \to CD \in P_1\} \cup \{A_{1,p} \mid p = A \to v \in P_2\}, \\
\overline{P} &= \{X' \to X_p, X_p \to X' \mid X \in V, p \in P\} \\
&\quad \cup \{A' \to A_{1,p},\ B' \to B_{2,p},\ A_{1,p} \to C',\ B_{2,p} \to D' \mid p = AB \to CD \in P_1\} \\
&\quad \cup \{A_{1,p} \to Y_1'Y_2' \dots Y_n' \mid p = A \to Y_1Y_2 \dots Y_n \in P_2\} \cup \{X' \to X \mid X \in T\}, \\
R &= \{X' \mid X \in V\}^+ \cup \bigcup_{p=AB\to CD\in P_1} \{X_p \mid X \in V\}^*\{A_{1,p}B_{2,p}\}\{X_p \mid X \in V\}^* \\
&\quad \cup \bigcup_{p=A\to v\in P_2} \{X_p \mid X \in V\}^*\{A_{1,p}\}\{X_p \mid X \in V\}^*.
\end{aligned}$$

We first note that each level of a derivation tree of G contains only primed letters or only letters belonging to the same rule p indicated by the index p or contains only terminals. Let $X_1'X_2' \dots X_n'$ be a word belonging to some level of a derivation tree. Then in order to get the next level we have to choose a rule $p \in P$ and any letter X' to replace by X_p or $X_{1,p}$ or $X_{2,p}$. We now discuss the case $p = AB \to CD \in P_1$. Then the new level has the form

$$(X_1)_p(X_2)_p \dots (X_r)_p A_{1,p}B_{2,p}(X_{r+3})_p(X_{r+4})_p \dots (X_n)_p$$

for some r, $0 \le r \le n-2$. Moreover, the word of the following level is

$$X_1'X_2' \dots X_r'C'D'X_{r+3}'X_{r+4}' \dots X_n'.$$

Thus we have simulated a derivation step

$$X_1X_2 \dots X_rABX_{r+3}X_{r+4} \dots X_n \Longrightarrow X_1X_2 \dots X_rCDX_{r+3}X_{r+4} \dots X_n$$

in the grammar G. Analogously, we can show that for $p \in P_2$ also a simulation of a derivation step in G is performed.

Since we start the derivation in $\overline{G}$ with S' and the only way to terminate a derivation in $\overline{G}$ is a simultaneous replacement of all letters X' by $X \in T$, it is easy to see that G and $\overline{G}$ generate the same language L.

Moreover, R is an ordered language. It is easy to construct an ordered automaton $(\overline{N}, Z, z_0, \delta, F)$ that accepts R (one moves to a larger state if one reads $A_{1,p}$, the same holds for $A_{2,p}$, and uses an ordered set of rules in the context-sensitive grammar which is transferred to the corresponding states); due to space limitations we omit the formal construction. It can be found in [3].

Hence, we have $CS \subseteq \mathcal{TC}(ORD)$. The converse inclusions follows from Theorem 3 and Lemma 4 as above. □

From Theorem 3 and Lemma 4, the next statement follows immediately.

Corollary 8. $\mathcal{TC}(NC) = \mathcal{TC}(PS) = CS$. □

Hence, we obtain all context-sensitive languages also with non-counting or power-separating control languages.

4 Generation of Subfamilies of CS

In this section, we consider further families of control languages and give some characterizations of the generated language families by other classes.

Theorem 9. $\mathcal{TC}(COMM) = MAT$.

Proof. i) $MAT \subseteq \mathcal{TC}(COMM)$.

Let L be a matrix language. By [2], Definition 1.3.2 and Lemma 1.3.7, there is a matrix grammar $G = (N, T, M, S, Q)$ such that

$$N = \{S, F\} \cup N_1 \cup N_2,\ N_1 \cap N_2 = \emptyset,\ S, F \notin N_1 \cup N_2,$$

any matrix m of M has one of the following forms

(a) $m = (A \to v_1, B \to C)$ with $A \in N_1$, $v_1 \in (N_1 \cup T)^+ \cup \{F\}$ and $B, C \in N_2$ or
(b) $m = (A \to v_1, B \to a)$ with $A \in N_1$, $v_1 \in (N_1 \cup T)^+$, $B \in N_2$ and $a \in T$ or
(c) $m = (S \to AB)$ with $A \in N_1$ and $B \in N_2$ or $(S \to a)$ with $a \in T$,

the set Q consists of all rules $A \to F$ occurring in a matrix of the form (a), and $L(G) = L$ holds.

We note that all non-terminated sentential forms of G – besides S – end with a letter of N_2. Let

$$\begin{aligned}
\overline{N} &= \{\overline{S}\} \cup \{X' \mid X \in N\} \cup \{X_m \mid m \in M,\ X \in N\} \\
&\quad \cup \{A_{m,1}, B_{m,2} \mid m = (A \to v_1, B \to v_2)\}.
\end{aligned}$$

We define the homomorphism $h : N \cup T \to \overline{N} \cup T$ by $h(X) = X'$ for $X \in N$ and $h(a) = a$ for $a \in T$ and construct the tree controlled grammar $\overline{G} = (\overline{N}, T, P, \overline{S})$ with

$$\begin{aligned}
P &= \{\overline{S} \to a \mid (S \to a) \in M\} \cup \{\overline{S} \to A'B' \mid (S \to AB) \in M\} \\
&\quad \cup \{X' \to X_m \mid m \in M,\ X \in N\} \cup \{X_m \to X' \mid X \in N,\ m \in M\} \\
&\quad \cup \{A' \to A_{m,1}, B' \to B_{m,2} \mid m = (A \to v_1, B \to v_2) \in M\} \\
&\quad \cup \{A_{m,1} \to h(v_1), B_{m,2} \to h(v_2) \mid m = (A \to v_1, B \to v_2) \in M\}
\end{aligned}$$

and the control set

$$\begin{aligned}
R = \{\overline{S}\} &\cup \bigcup_{m=(A\to v_1,B\to v_2)\in M, v_1\neq F} Comm(\{X_m \mid X \in N\}^*\{A_{m,1}B_{m,2}\}) \\
&\cup \bigcup_{(A\to v_1,B\to v_2)\in M} Comm(\{X' \mid X \in N\}^*(\{h(v_1)h(v_2)\} \cup \{\lambda\})) \\
&\cup \bigcup_{m=(A\to F,B\to C)\in M} Comm(\{X_m \mid X \in N, X \neq A\}^+\{B_{m,2}\}) \\
&\cup \bigcup_{m=(A\to F,B\to C)\in M} Comm(\{X' \mid X \in N, X \neq A\}^+\{C'\}).
\end{aligned}$$

It is easy to see that R is a commutative regular language (since one has only to check that besides a finite number of occurrences of some letters all other letters

belong to $\{X_m \mid X \in N\}$ or $\{X' \mid X \in N\}$, respectively, what can be done by a finite automaton).

Since any nonterminal of $\overline{G}$ is replaced in any step and any primed or indexed version of a letter in N_2 is replaced by an indexed or primed version of a letter of N_2 or a terminal, it is obvious that any word of a certain level of a derivation tree, which is not a terminal word, ends with a primed or indexed version of a letter of N_2.

Let us consider a derivation in $\overline{G}$. After applying a rule to $\overline{S}$ we get a terminal belonging to $L(\overline{G})$ as well as to $L(G)$ or a word $A'B'$ which is the sentential form and the word of the first level. We have to replace A' and B' in parallel and the result has to be $A_{m_1}B_{m_2}$ for some matrix $m \in M$. The word of the next level will be $h(v_1)h(v_2)$. Thus we have simulated a rewrite step $A'B' \Longrightarrow h(v_1)h(v_2)$ which corresponds to an application of m to AB which yields v_1v_2.

Now let $w \in Comm(\{X' \mid X \in N\}^*(\{h(v_1)h(v_2)\} \cup \{\lambda\}))$ be the word of a level i of some derivation tree of $\overline{G}$. Then let w' be the sentential form of this level and let us assume that $w' = h(w'')$ for some sentential form w'' of G. If $w \in T^*$, then we have $w' \in T^*$ and thus $w' \in L(\overline{G})$ and $w' = w'' \in L(G)$.

Let $w = v_1A'_1v_2A'_2 \dots v_nA'_n$ with $v_i \in T^*$ and $A_i \in N$ for $1 \leq i \leq n$. Then

$$(A_1)_m(A_2)_m \dots (A_{t-1})_m(A_t)_{m,1}(A_{t+1})_m \dots (A_{n-1})_m(A_n)_{m,2}$$

for some t, $1 \leq t \leq n-1$, is the word of level $i+1$, if $m = (A_t \to v_1, A_n \to v_2) \in M$ and $v_1 \notin \{\lambda, F\}$, or

$$(A_1)_m(A_2)_m \dots (A_{n-1})_m(A_n)_{m,2}$$

is the word of level $i+1$, if $m = (A \to F, A_n \to v_2) \in M$ and $A \neq A_i$ for $1 \leq i \leq n-1$. We only discuss the former case; the latter can be handled analogously. The word of level $i+2$ is $A'_1A'_2 \dots A'_{t-1}h(v_1)A'_{t+1} \dots A'_{n-1}h(v_2)$ which is in R, too. Moreover, let v' be the sentential form of the level $i+2$ and $v' = h(v'')$ for some $v'' \in (N \cup T)^*$. Obviously, $w'' \Longrightarrow_m v''$ holds in G since we replaced non-terminals by themselves or by $h(v_1)$ and $h(v_2)$ and have not changed the terminals. Hence any derivation in $\overline{G}$ simulates a derivation in G and $L(\overline{G}) \subseteq L(G)$ follows.

By analogous considerations, one can show that any derivation in G can be simulated which gives $L(G) \subseteq L(G')$. Therefore, we have $L = L(G) = L(G')$ and $MAT \subseteq \mathcal{TC}(COMM)$.

ii) $\mathcal{TC}(COMM) \subseteq MAT$.

Let $L \in \mathcal{TC}(COMM)$. There is a tree controlled grammar $G = (N, T, P, S, R)$ with $L(G) = L$ where $R \subseteq (N \cup T)^*$ is a regular commutative language. Let $N = \{A_1, A_2, \dots, A_m\}$ and $T = \{a_1, a_2, \dots, a_k\}$. Then the Parikh vectors have the form

$$(n(A_1), n(A_2), \dots, n(A_m), n(a_1)n(a_2), \dots, n(a_k)).$$

Since R is regular, it is semi-linear; and because it is commutative, there are natural numbers $n \geq 1$, $r_i \geq 0$ for $1 \leq i \leq n$, and $\#(N \cup T)$-dimensional vectors p_i and $q_{i,j}$, $1 \leq i \leq n$, $1 \leq j \leq r_i$, such that

$$R = \pi^{-1}_{N \cup T}(\bigcup_{i=1}^{n} \{p_i + \sum_{j=1}^{r_i} \alpha_{i,j}q_{i,j} \mid \alpha_{i,j} \in \mathbb{N} \text{ for } 1 \leq j \leq r_i\})$$

where $\pi_{N \cup T}$ is the Parikh mapping.

With any letter $a \in T$, we associate two new letters A_a and A'_a and define a homomorphism $h : (N \cup T)^* \to (N \cup \{A_a \mid a \in T\})^*$ by

$$h(A) = A \text{ for } A \in N \quad \text{and} \quad h(a) = A_a \text{ for } a \in T.$$

We define the matrix grammar $G' = (N', T \cup \{\$\}, M, \overline{S}, Q)$ where

$$N' = \{\overline{S}, F, C, C', D\} \cup N \cup \{A' \mid A \in N\} \cup \bigcup_{a \in T} \{A_a, A'_a\} \cup \bigcup_{i=1}^{n} \{B_i, B_{i,1}, B_{i,2}, \dots, B_{i,r_i}\},$$

Q consists of all rules with right-hand side F, and M consists of the matrices constructed as follows:

- $(\overline{S} \to B_iS)$
  (initial rules which choose an index i corresponding to a set

  $$H_i = \{p_i + \sum_{j=1}^{r_i} \alpha_{i,j}q_{i,j} \mid \alpha_{i,j} \in \mathbb{N} \text{ for } 1 \leq j \leq r_i\}),$$

- for any vector $p_i = (n(A_1), n(A_2), \dots, n(A_m), n(a_1)n(a_2), \dots, n(a_k))$, $1 \leq i \leq n$, we set

  $(B_i \to B_{i,j}, (A_1 \to A'_1)^{n(A_1)}, (A_2 \to A'_2)^{n(A_2)}, \dots, (A_m \to A'_m)^{n(A_m)},$
  $(A_{a_1} \to A'_{a_1})^{n(a_1)}, (A_{a_2} \to A'_{a_2})^{n(a_2)}, \dots, (A_{a_k} \to A'_{a_k})^{n(a_k)})$ for $1 \leq j \leq r_i$,
  $(B_i \to C, (A_1 \to A'_1)^{n(A_1)}, (A_2 \to A'_2)^{n(A_2)}, \dots, (A_m \to A'_m)^{n(A_m)},$
  $(A_{a_1} \to A'_{a_1})^{n(a_1)}, (A_{a_2} \to A'_{a_2})^{n(a_2)}, \dots, (A_{a_k} \to A'_{a_k})^{n(a_k)}),$

  and any vector $q_{i,j} = (n(A_1)', n(A_2)', \dots, n(A_m)', n(a_1)'n(a_2)', \dots, n(a_k)')$, $1 \leq i \leq n$, $1 \leq j \leq r_i$, we set

  $(B_{i,j} \to B_{i,j'}, (A_1 \to A'_1)^{n(A_1)'}, (A_2 \to A'_2)^{n(A_2)'}, \dots, (A_m \to A'_m)^{n(A_m)'},$
  $(A_{a_1} \to A'_{a_1})^{n(a_1)'}, (A_{a_2} \to A'_{a_2})^{n(a_2)'}, \dots, (A_{a_k} \to A'_{a_k})^{n(a_k)'})$ for $1 \leq j' \leq r_i$,
  $(B_{i,j} \to C, (A_1 \to A'_1)^{n(A_1)'}, (A_2 \to A'_2)^{n(A_2)'}, \dots, (A_m \to A'_m)^{n(A_m)'},$
  $(A_{a_1} \to A'_{a_1})^{n(a_1)'}, (A_{a_2} \to A'_{a_2})^{n(a_2)'}, \dots, (A_{a_k} \to A'_{a_k})^{n(a_k)'}),$

  and $(C \to D, A_1 \to F, A_2 \to F, \dots, A_m \to F, A_{a_1} \to F, A_{a_2} \to F, \dots, A_{a_m} \to F)$
  (if w is the scattered subword of the sentential form build by the letters of N', by these rules we check whether or not $h^{-1}(w)$ belongs to $\pi^{-1}_{N \cup T}(H_i)$; note that $h^{-1}(w)$ is the word of some level),

- $(D \to D, A' \to h(w))$ for $A \to w \in P$,
  $(D \to D, A'_a \to a)$ for $a \in T$,
  $(D \to B_i, A'_1 \to F, A'_2 \to F, \dots, A'_m \to F, A'_{a_1} \to F, A'_{a_2} \to F, \dots, A'_{a_m} \to F)$
  for $1 \leq i \leq n$
  (we simulate the generation of the next level and choose an index, again),

- $(D \to \$)$
  (this rule can only be applied if no nonterminal – besides D – is in the sentential form, because after its use no other matrix can be applied).

By the explanations given to the matrices, the generated languages is $\$L$. Thus $\$L \in MAT$. By the closure properties of MAT (see [2]), we get $L \in MAT$. □

The matrix languages with finite index are exactly those generated by finitely tree controlled grammars.

Theorem 10. $\mathcal{TC}(FIN) = MAT_{fin}$. □

Proof. The inclusion $MAT_{fin} \subseteq \mathcal{TC}(FIN)$ can be proved analogously to the proof of Theorem 9. If $Ind(G) = k$ we have only to take

$$\bigcup_{i=0}^{k} \{X' \mid X \in N\}^i \text{ and } \bigcup_{i=0}^{k} \{X_m \mid X \in N\}^i$$

instead of $\{X' \mid X \in N\}^*$ and $\{X_m \mid X \in N\}^*$. Note that the normal form result carries over to finite index grammars, too.

The inclusion $\mathcal{TC}(FIN) \subseteq MAT_{fin}$ can be proved by constructing a matrix grammar that simulates a tree controlled grammar (see [3]). The number of non-terminals in a sentential form of the matrix grammar equals the number of non-terminals in a sentential form of a certain level of a derivation tree of the tree controlled grammar. Hence, the constructed matrix grammar is of finite index. □

We now present some results on the power of control by the families MON, $COMB$, DEF, and NIL.

Theorem 11.
i) $E0L = \mathcal{TC}(MON) \subseteq \mathcal{TC}(COMB) \subseteq \mathcal{TC}(DEF)$.
ii) *$E0L$ is properly included in* $\mathcal{TC}(DEF)$. □

Proof. i) For a proof of the equality $E0L = \mathcal{TC}(MON)$, we refer to [3]. The inclusions follow from Lemma 4.

ii) The inclusion holds by i). The properness follows from $FIN \subseteq DEF$ and Example 2: $L(G_2) \in \mathcal{TC}(FIN) \subseteq \mathcal{TC}(DEF)$ but $L(G_2) \notin E0L$ (see [6]). □

Theorem 12. $\mathcal{TC}(FIN) \subset \mathcal{TC}(NIL)$ *and* $\mathcal{TC}(MON) \subset \mathcal{TC}(NIL)$. □

Proof. All inclusions follow from Lemma 4. The first inclusion is proper because the tree controlled grammar from Example 1 generates a non-semi-linear set, whereas all languages in $\mathcal{TC}(FIN) = MAT_{fin}$ are semi-linear by [2], Lemma 3.1.5. The properness of the last inclusion follows from $FIN \subseteq NIL$ and Example 2 (analogously to Theorem 11.ii). □

5 Conclusion

First we summarize our results in Figure 2.

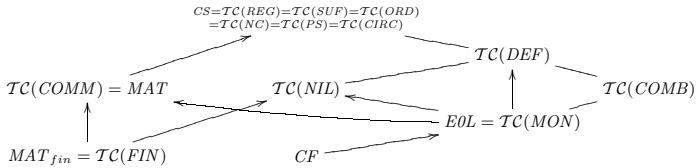

Figure 2: Hierarchy of families of tree controlled languages (an arrow from X to Y denotes $X \subsetneq Y$; two families which are not connected by a directed path, are not necessarily incomparable)

Further we remember that $E0L$ and MAT_{fin} are known to be incomparable. The strictness of some inclusions and the incomparability of some families remain as an open problem.

In the paper we have considered grammars/systems without erasing rules. By $\mathcal{TC}_\lambda(X)$ we denote the families of tree controlled grammars with erasing rules and control sets in X. We note that all proofs of the paper remain valid, if we allow erasing rules and take matrix grammars, E0L systems etc. with erasing rules. Therefore taking into consideration that matrix grammars with erasing rules generate all recursively languages and $\mathcal{TC}_\lambda(REG) = RE$ (see [2]), then we get the following results.

Theorem 13.
i) $RE = \mathcal{TC}_\lambda(REG) = \mathcal{TC}_\lambda(SUF) = \mathcal{TC}_\lambda(ORD) = \mathcal{TC}_\lambda(NC)$
$= \mathcal{TC}_\lambda(PS) = \mathcal{TC}_\lambda(COMM) = \mathcal{TC}_\lambda(CIRC)$,
ii) $MAT_{fin} = \mathcal{TC}_\lambda(FIN) \subset \mathcal{TC}_\lambda(NIL) \subseteq RE$ *and* $\mathcal{TC}_\lambda(MON) \subset \mathcal{TC}_\lambda(NIL)$,
iii) $CF \subset E0L = \mathcal{TC}_\lambda(MON) \subseteq \mathcal{TC}_\lambda(COMB) \subseteq \mathcal{TC}_\lambda(DEF) \subseteq RE$ *and*
$\mathcal{TC}_\lambda(MON) \subset \mathcal{TC}_\lambda(DEF)$. □

References

[1] K. Culik II and H. Maurer, Tree controlled grammars. *Comput.* **19** (1977) 129–139.

[2] J. Dassow and Gh. Păun, *Regulated Rewriting in Formal Language Theory*. EATCS Monographs on Theoretical Computer Science 18, Springer-Verlag, 1989.

[3] J. Dassow and B. Truthe, Subregularly Tree Controlled Grammars and Languages. Technical Report, Otto-von-Guericke-Univ. Magdeburg, Fakultät für Informatik, 2008.

[4] I. M. Havel, The theory of regular events II. *Kybernetika* **6** (1969) 520–544.

[5] Gh. Păun, On the generative capacity of tree controlled grammars. *Computing* **21** (1979) 213–220.

[6] G. Rozenberg and A. Salomaa, *The Mathematical Theory of L Systems*. Academic Press, 1980.

[7] G. Rozenberg and A. Salomaa (Eds.), *Handbook of Formal Languages*, Vol. I – III. Springer-Verlag, Berlin, 1997.

[8] B. Wiedemann, Vergleich der Leistungsfähigkeit endlicher determinierter Automaten. Diplomarbeit, Universität Rostock, 1978.