International Journal of Foundations of Computer Science

CONTEXT-FREE GRAMMARS WITH LINKED NONTERMINALS

ANDREAS KLEIN

Institut für Mathematik, Universität Kassel,
Heinrich Plett Straße 40, D-34132 Kassel, Germany
klein@mathematik.uni-kassel.de

MARTIN KUTRIB

Institut für Informatik, Universität Giessen,
Arndtstraße 2, D-35392 Giessen, Germany
kutrib@informatik.uni-giessen.de

Received (Day Month Year)
Accepted (Day Month Year)
Communicated by (xxxxxxxxxx)

We investigate a new type of finite copying parallel rewriting system, i.e., grammars with linked nonterminals, which extend the generative capacity of context-free grammars. They can be thought of as having sentential forms where some instances of a nonterminal may be linked. The context-free-like productions replace a nonterminal together with its connected instances. New links are only established between symbols of the derived subforms. A natural limitation is to bound the degree of synchronous rewriting. We present an infinite degree hierarchy of separated language families with the property that degree one characterizes the regular and degree two characterizes context-free languages. Furthermore, the hierarchy is a refinement of the known hierarchy of finite copying rewriting systems. Several closure properties known from equivalent systems are summarized.

Keywords: Generalized context-free grammars; finite copying parallel rewriting systems; degree hierarchy.

1. Introduction

Context-free grammars are one of the most important and most developed parts of formal language theory. However, in many situations we are confronted with naturally non-context-free languages. "The world is not context-free": a comprehensive discussion of this observation giving "seven circumstances where context-free grammars are not enough" can be found in [5]. So there is considerable interest in grammars based on context-free rewriting rules that extend the generative capacity but have similar properties. Several approaches restrict the use of productions in context-free derivations. They are considered in the framework of regulated rewriting. A detailed presentation is the monograph [5], and [6], which include further references for the following cited related results.

Particularly related are grammars with partially parallel substitution mode. One development are Indian parallel grammars where a substitution step consists of the substitution of all instances of one nonterminal according to one and the same rule. It has turned out that the generated language family is incomparable with context-free languages. The so-called k-grammars are parallel rewriting systems where the degree of synchronous rewriting is bounded by some constant k. Besides the first step in such systems exactly k nonterminals have to be rewritten during a derivation step. The rules and nonterminals may be different. Hierarchies depending on k have been shown for k-grammars with and without erasing productions.

The next restriction concerns the choice and the places of applications of the productions. In scattered context grammars the context-free rewriting rules are grouped together into matrices. The rules in a matrix must be applied simultaneously during one step. The nonterminals to be rewritten in the sentential form have to appear in the ordering given by the rules in the matrices. A slight modification where the ordering condition is relaxed yields the unordered scattered context grammars. The language families of unordered grammars with and without erasing rules are properly included in the languages generated by their ordered variants, respectively. Ordered grammars with scattered context and erasing rules are characterizing the recursively enumerable languages. There are several other developments, e.g., k-simple grammars where also a hierarchy depending on k is known.

Common to all of these restrictions is that the conditions for applying a production at some time can be verified by inspecting the sentential form at the same time. If the requirement of applying all rules in a matrix simultaneously is relaxed, unordered scattered context grammars become unordered vector grammars. The matrices are now called vectors, and the requirement is that once a production in a vector has been applied, all productions must be applied during the derivation. Thus, in general it cannot be decided whether a production can be applied or not by inspecting just one sentential form. This can be seen as a vertical context condition. Stronger definitions are given in [15]. Unordered vector grammars have been introduced in [4]. For a subclass without λ-productions and unit-productions it was shown in [19] that the generated languages are belonging to the complexity class LOGCFL. In [17] this result has been improved to arbitrary languages generated by unordered vector grammars.

Here we investigate grammars with linked nonterminals which are a natural extention of regular and context-free grammars. Basically, the idea is to consider sentential forms of context-free-like grammars where some occurrences of one and the same nonterminal may be linked. A derivation step replaces a nonterminal together with its connected instances, whereby new links are only established between nonterminals of the derived subforms. A natural condition is the limitation of the maximal number of nonterminals linked up, i.e., of the synchronous rewriting. This restriction leads to grammars of a certain degree. It turned out that these rewriting systems are equivalent [13] to parallel rewriting systems which have been called finite copying in [8, 16]. Further equivalent systems of this type are deterministic tree-

walking transducers [1], finite copying deterministic top-down tree-to-string transducers [8], context-free string generating hypergraph-replacement grammars [11], string based linear context-free rewriting systems [20], multiple context-free grammars [21], local unordered scattered context grammars [16], multi-component tree adjoining grammars [12], and finite copying lexical-functional grammars [18].

Another related approach is to limit the number of nonterminals which may simultaneously appear in a sentential form. These grammars of finite index have extensively be investigated. E.g., for matrix grammars with context-free rules an infinite hierarchy has been shown [5]. But by means of different closure properties, these grammars and grammars with linked nonterminals generate different languages.

The idea of linking nonterminals is motivated by the following observation. Some word $a_1 \cdots a_n$ can be seen as a set of couples $\{(a_1, 1), \ldots, (a_n, n)\}$ bringing together letters and positions. If one letter appears at several positions, we could write $(a, i_1, \ldots, i_p)$ instead of $(a, i_1), \ldots, (a, i_p)$. So, the tuples can be seen as in some sense generalized or linked nonterminals that are to be rewritten in a context-free fashion. A formal definition of this notion is given in the next section. In Section 3 it will be shown that there exists an infinite degree hierarchy of separated language families. The hierarchy has the considerable property that degree one characterizes the regular and degree two characterizes the context-free languages. So we obtain an in some sense unified generalization of both families. Furthermore, the hierarchy is a refinement of the known hierarchy of finite copying rewriting systems. From the known equivalent systems we obtain that all families belong to the complexity class LOGCFL which is in P and in the context-sensitive languages. Furthermore, closure properties are summarized.

2. Grammars with Linked Nonterminals

We denote the positive integers $\{1, 2, ...\}$ by $\mathbb{N}$ and the set $\mathbb{N} \cup \{0\}$ by $\mathbb{N}_0$. The empty word is denoted by λ. For the length of w we write $|w|$. We use $\subseteq$ for inclusions and $\subset$ for strict inclusions.

Due to the basic motivation and the underlying idea we have terminal and nonterminal symbols, but we do not regard links between terminals. More formally, let N and T be two finite sets and $w \in (N \cup T)^*$. In order to express the links between symbols from N in w, we use a set of tuples whose components identify the corresponding positions. So, $(a_1 \cdots a_n, C)$ is a word with linked symbols from N if and only if $a_1 \cdots a_n \in (N \cup T)^*$ and C is a subset of the tuples which partition $\{1, \ldots, n\}$ such that for each tuple $(i_1, \ldots, i_q) \in C$ there is some $X \in N$ such that $a_{i_1} = \cdots = a_{i_q} = X$. The set of all links obeying these restrictions for a given word w is denoted by $C_N(w)$. Now we are prepared to define grammars based on this concept.

Definition 1. *A* grammar with linked nonterminals *is a system* $\langle N, T, S, P \rangle$*, where*

1. N *is the finite set of* nonterminals*,*

2. T *is the finite set of* terminals,
3. $S \in N$ *is the* axiom (starting symbol),
4. P *is the finite set of* productions *each of one of the forms* $(S \to w, C)$ *where* $w \in (N \setminus \{S\} \cup T)^*$, $C \in C_N(w)$, *or* $(X \to w_1, \ldots, w_p, C)$ *where* $X \in N$, $p \in \mathbb{N}$, $w_j \in (N \setminus \{S\} \cup T)^*$ *for* $1 \leq j \leq p$, $C \in C_N(w_1 \cdots w_p)$ *such that all* $(i_1, \ldots, i_q) \in C$ *satisfy* $q \leq p$.

Since the starting symbol does not appear on the right-hand side of any production, productions of the form $(S \to w, C)$ are only possible during the first derivation step. They establish possibly some links. Productions of the form $(X \to w_1, \ldots, w_p, C)$ mean that the instances of a p-fold linked symbol X are rewritten by $w_1, \ldots, w_p$. Since $C \in C_N(w_1 \cdots w_p)$ new links are only possible between nonterminals in the derived subforms. The condition $q \leq p$ implies that – besides the first derivation step – the maximal number of nonterminals in a link cannot grow. Thus, a production of the form $(X \to w_1, \ldots, w_p, C)$ is *applicable* to

$$u = (u_1 \cdots u_m, D), \quad u_j \in N \cup T,\ 1 \leq j \leq m, \quad D \in C_N(u_1 \cdots u_m)$$

if there exists $(i_1, \ldots, i_p) \in D$ such that $u_{i_1} = \cdots = u_{i_p} = X$. The application yields $v = (v_1 \cdots v_n, E)$, where

$$v_1 \cdots v_n = u_1 \cdots u_{i_1-1} w_1 u_{i_1+1} \cdots u_{i_2-1} w_2 u_{i_2+1} \cdots w_p u_{i_p+1} \cdots u_m$$

and E contains exactly the links derived from C and $D \setminus \{(i_1, \ldots, i_p)\}$ by adjusting the symbol positions accordingly (cf. Example 2). As usual we write $u \Rightarrow v$ if u directly derives v, and $\Rightarrow^*$ for the reflexive and transitive closure of $\Rightarrow$.

The language generated by such a grammar $\mathcal{G}$ is

$$L(\mathcal{G}) = \left\{ w \mid w \in T^*, \quad (S, \emptyset) \Rightarrow^* (w, \emptyset) \right\}$$

The condition that a link is always between one and the same nonterminal makes life easier but is not really a restriction or even a limitation. Since derivations are in a context-free-like fashion, a grammar without this property can always be transformed to an equivalent one obeying the condition. The only thing to do is to rename uniquely linked up nonterminals that appear on the left-hand side of a production.

Example 2. *The grammar* $\langle \{A, S\}, \{a, b, c\}, S, P \rangle$ *with*

$$P = \{\ (S \to AA, \{(1,2)\}), \quad (A \to aAb, cA, \{(2,5)\}), \quad (A \to ab, c, \emptyset)\ \}$$

generates the language $\{a^n b^n c^n \mid n \in \mathbb{N}\}$. *The derivation for* $n = 4$ *is*

$$\begin{aligned}(S, \emptyset) &\Rightarrow (AA, \{(1,2)\}) \Rightarrow (aAbcA, \{(2,5)\}) \\ &\Rightarrow (aaAbbccA, \{(3,8)\}) \Rightarrow (aaaAbbbcccA, \{(4,11)\}) \\ &\Rightarrow (aaaabbbbcccc, \emptyset)\end{aligned}$$

More figurative the derivation together with the current links may be represented as:

$$S \Rightarrow \overbrace{AA} \Rightarrow a\overbrace{AbcA} \Rightarrow aa\overbrace{AbbccA} \Rightarrow \cdots$$

Similarly, a grammar for the language $\{ww \mid w \in \{a,b\}^+\}$ can be constructed.

A natural condition is the limitation of the maximal number of linked up nonterminals. This restriction leads to grammars of a certain degree.

Definition 3. *Let $m \in \mathbb{N}$ be a constant. A grammar with linked nonterminals*

1. *is* of degree $2m$*, if for its productions $(S \to w, C)$ all $(i_1, \ldots, i_q) \in C$ satisfy $q \leq m$ (i.e., the maximal number of nonterminals in a link is bounded by m).*
2. *It is even* of degree $2m-1$*, if in addition for its productions $(S \to w, C)$ resp. $(X \to w_1, \ldots, w_m, C)$ a link $(i_1, \ldots, i_m) \in C$ implies $i_m = |w|$ resp. $i_m = |w_1 \cdots w_m|$ (i.e., i_m is the position of the rightmost symbol on the right-hand side).*

Roughly speaking, the degree of a grammar determines the maximal number of subwords that may be lengthened during one derivation step in a linked fashion (a nonterminal may have left and right neighboring subwords). For odd degrees the rightmost linked nonterminal is forced to have only left neighbors if the number of symbols in that link is maximal. This is in some sense a generalization of right-linearity. In general, for a grammar of degree k the maximal number k of nonterminals in one link can be derived during the first transition step. In subsequent steps this number cannot be increased. In case of even degrees there may occur more than one link with k nonterminals. In case of odd degrees for maximal links it is requested that the rightmost symbol on the right-hand side of a production belongs to the link. This implies that there may occur at most one maximal link in any sentential form. But nevertheless, there may occur other links which are not maximal.

For example, the language $\{a^n b^n c^n \mid n \in \mathbb{N}\}$ in Example 2 is generated by a grammar of degree three. The construction can be adapted such that for any $k \in \mathbb{N}$ the language $L_k = \{a_1^n \cdots a_k^n \mid n \in \mathbb{N}\}$ over the alphabet $\{a_1, \ldots, a_k\}$ is generated by a grammar of degree k.

The family of all languages which can be generated by grammars with linked nonterminals is denoted by $\mathscr{L}(\text{SN})$. If the productions are restricted to degree k, we use the notion $\mathscr{L}(k\text{-SN})$.

Corollary 4. *Let $k \in \mathbb{N}$ be a constant, then $L_k \in \mathscr{L}(k\text{-SN})$.*

3. Generative Capacity

Taking a closer look at the definitions leads to the observation that in case of degree two on the right-hand sides of the productions there is always a single word. Moreover, the maximal number of nonterminals in a link is one. This means that there are no useful links at all and, hence, the grammar is a context-free one.

The situation in case of degree one is more restrictive. If a nonterminal appears on the right-hand side of some production, then it has to be the rightmost symbol since again the maximal number of symbols linked together is one. This implies

that there is at most one nonterminal on each right-hand side. Together it follows that the grammar is right-linear. We denote the regular languages by $\mathscr{L}(\mathrm{REG})$ and the context-free languages by $\mathscr{L}(\mathrm{CF})$.

Corollary 5. *$\mathscr{L}(1\text{-SN}) = \mathscr{L}(\mathrm{REG})$ and $\mathscr{L}(2\text{-SN}) = \mathscr{L}(\mathrm{CF})$.*

The corollary relates the generative capacity of degree one and two to the regular and context-free languages, respectively. A fundamental result concerning arbitrary degrees is immediately derived from the definition and a well-known result about context-free languages. It has also been shown in terms of equivalent finite copying parallel rewriting systems (e.g. [7]).

Lemma 6. *Each of the languages in $\mathscr{L}(\mathrm{SN})$ or $\mathscr{L}(k\text{-SN})$ for some $k \in \mathbb{N}$ is semi-linear.*

Proof. Let $\mathcal{G}$ be some grammar under consideration. The right-hand sides of its productions can be rearranged such that linked symbols are placed joint together. In general, the resulting grammar $\mathcal{G}'$ generates a different language, but $L(\mathcal{G})$ and $L(\mathcal{G}')$ are letter-equivalent. Since the linked nonterminals of $\mathcal{G}'$ appear joint together, they can be replaced by just one single nonterminal, respectively, without changing the generated language. Now we have a context-free grammar $\mathcal{G}''$ such that $L(\mathcal{G}'')$ and $L(\mathcal{G})$ are letter-equivalent. In [14] it has been shown that every context-free language is semi-linear. Trivially, letter-equivalence preserves semi-linearity. □

3.1. *Strictly Monotone Derivations*

In order to investigate the relationships between degrees k and $k+1$, in particular, to prove an infinite hierarchy, we need a tool for proving negative results. This will be a pumping argument. In order to prove this and other results it is helpful to simplify an arbitrary grammar. A production $(X \to w_1, \ldots, w_p, C)$ is said to be *strictly monotone* if it inserts more symbols than it replaces, i.e., if $|w_1 \cdots w_p| > p$. Otherwise we call a production *non-increasing*.

The next lemma removes non-increasing productions, possibly with the exceptions $(S \to a, \emptyset)$, for $a \in T \cup \{\lambda\}$, if the empty word or some words of length one have to be generated.

Lemma 7. *For every grammar $\mathcal{G}$ of degree $k \in \mathbb{N}$ there exists an equivalent grammar $\mathcal{G}'$ of degree k whose only non-increasing productions are of the form $(S \to a, \emptyset)$, where $a \in T \cup \{\lambda\}$.*

3.2. *Hierarchy*

The following pumping lemma is a useful tool for separating language families since it allows to prove negative results. It is in some sense weaker than others since it contains no statement about the usual ordering in which the repeated subwords appear.

Lemma 8. *Let $\mathcal{G}$ be a grammar of degree $k \in \mathbb{N}$. Then there exists a constant $n \in \mathbb{N}$ such that every $w \in L(\mathcal{G})$ with $|w| \geq n$ may be written as $x_0 y_1 x_1 y_2 \cdots y_k x_k$, where $1 \leq |y_1 y_2 \cdots y_k| \leq n$, and for all $i \in \mathbb{N}$ there exists a word $w' \in L(\mathcal{G})$ such that w' is in some order a concatenation of the (sub)words $x_0, \ldots, x_k$ and i times y_j, for each $1 \leq j \leq k$.*

Proof. Since $\mathcal{G}$ has only finitely many productions, for every long enough word w from $L(\mathcal{G})$ there exists a derivation such that some production(s) with the same left-hand side are applied at least twice. Moreover, the second application is on symbols derived from the first application (cf. Figure 1). Obviously, for a given grammar the necessary word length for such a situation can be calculated. It defines the constant n.

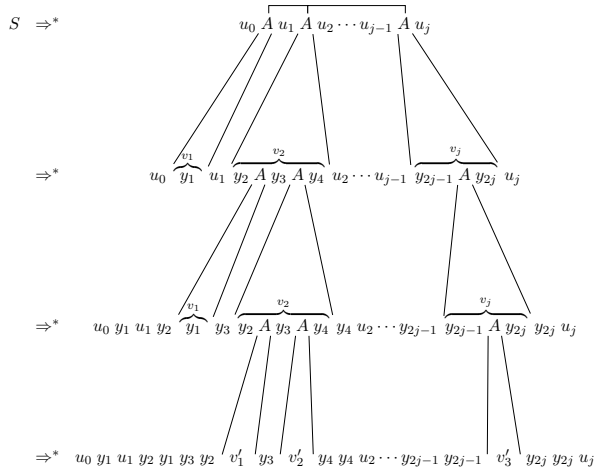

Fig. 1. Example of a derivation scheme of Lemma 8.

Assume for a moment the degree of $\mathcal{G}$ is even. Then for the core derivation we have

$$S \Rightarrow \cdots \Rightarrow u_0 A u_1 A u_2 \cdots u_{j-1} A u_j \Rightarrow \cdots \Rightarrow u_0 v_1 u_1 v_2 u_2 \cdots u_{j-1} v_j u_j \Rightarrow \cdots \Rightarrow w,$$

where $A \in N$, $1 \leq j \leq \frac{k}{2}$, $u_0, \ldots, u_j \in T^*$, and $v_1 \cdots v_j$ contains a j-fold linked A and some terminal symbols. Without loss of generality we may assume that u_i are terminal words and v_i are terminal words with the exception of the linked A, since the derivation of other nonterminals can be finished without affecting the A.

The word v_1 is derived from the first of the linked A, v_2 from the second and so on. Since $v_1 \cdots v_j$ contain a j-fold linked A, there must exist derivations in which this loop appears arbitrary times. In order to determine the pumped portions of w, we have to consider the positions of the A in the words $v_1, \ldots, v_j$.

We define subwords y_m, $1 \leq m \leq 2j$, of $v_1 \cdots v_j$ as follows. From left to right a y_m starts either at the beginning of a word v_l or immediately after a nonterminal A. A y_m ends either at the end of a word v_l or immediately before a nonterminal A. Clearly, some of the y_m may be empty. But due to the fact that we may assume strictly monotone productions, at least one of the y_m is not empty.

Next we define the subwords x_l which are not pumped. These are the already derived subwords $u_0, \ldots, u_j$ and, in addition, the terminal subwords $v'_1, \ldots, v'_j$ to which the j-fold linked A is finally derived. They are numbered from left to right according to their appearance. Again, some of the $x_0, \ldots, x_{2j}$ may be empty.

For the even degree case the lemma follows since j is at most $\frac{k}{2}$ and, thus, we found subwords $x_0, \ldots, x_k$ and $y_1, \ldots, y_k$ as claimed.

In case of an odd degree, j is at most $\lceil \frac{k}{2} \rceil$. For $j < \lceil \frac{k}{2} \rceil$ the lemma has been shown. For $j = \lceil \frac{k}{2} \rceil$ the subword u_j must have been derived during the first step. Afterwards, during the pump-loops the rightmost symbol is always the unique nonterminal which is j-fold linked.

Therefore, y_{2j} is always empty and at most $2j - 1$ portions can be pumped. Since k is odd, we obtain $2j - 1 = 2\lceil \frac{k}{2} \rceil - 1 = 2\frac{k+1}{2} - 1 = k$, and the lemma follows for the odd degree case, too. □

We apply the pumping lemma to the languages $L_k = \{a_1^n \cdots a_k^n \mid n \in \mathbb{N}\}$ of Example 2 and Corollary 4.

Lemma 9. *Let $k \in \mathbb{N}$ be a constant, then L_{k+1} does not belong to $\mathscr{L}(k\text{-SN})$.*

Proof. Assume L_{k+1} belongs to $\mathscr{L}(k\text{-SN})$. Let n be the constant of Lemma 8 and consider the word $w = a_1^n \cdots a_{k+1}^n$. Since we may pump at most k portions of w, the result would not be a word in L_{k+1}. □

The lemma immediately implies the hierarchy of grammars with linked nonterminals.

Theorem 10. *Let $k \in \mathbb{N}$ be a constant, then $\mathscr{L}(k\text{-SN}) \subset \mathscr{L}((k+1)\text{-SN})$ and $\mathscr{L}(k\text{-SN}) \subset \mathscr{L}(\text{SN})$.*

Proof. The inclusions $\mathscr{L}(k\text{-SN}) \subseteq \mathscr{L}((k+1)\text{-SN}) \subseteq \mathscr{L}(\text{SN})$ are for structural reasons. The strictness follows from Lemma 8, Example 2, Corollary 4, and the identity $\mathscr{L}(\text{SN}) = \bigcup_{k=1}^{\infty} \mathscr{L}(k\text{-SN})$. □

The languages L_k are witnesses for the hierarchy. On the other hand, for any $k \in \mathbb{N}$ the language L_k belongs to the family $\mathscr{L}(\text{SN})$. Thus, in some sense the infinite hierarchy converges to $\mathscr{L}(\text{SN})$:

$$\mathscr{L}(1\text{-SN}) \subset \mathscr{L}(2\text{-SN}) \subset \cdots \subset \mathscr{L}(k\text{-SN}) \subset \cdots \subset \mathscr{L}(\text{SN})$$

3.3. *Comparison with* LOGCFL

We now turn to the question where the hierarchy ends up. To this end, we can compare $\mathscr{L}$(SN) with other families. Since $\mathscr{L}$(1-SN) and $\mathscr{L}$(2-SN) are equal to Chomsky families, a natural candidate for comparisons is the family of context-sensitive languages $\mathscr{L}$(CS). The inclusion $\mathscr{L}(\text{SN}) \subseteq \mathscr{L}(\text{CS})$ follows immediately from the fact that after the first step the derivations according to a grammar of arbitrary degree are strictly monotone. So they can be simulated by a linearly space bounded nondeterministic Turing machine what implies context-sensitivity.

The inclusion is even strict: $\mathscr{L}(\text{SN}) \subset \mathscr{L}(\text{CS})$. For example the language $L = \{a^n(b^nc^n)^+ \mid n \in \mathbb{N}\}$ is context-sensitive. By using the pumping lemma it is easy to see that L cannot be generated by any grammar of any degree $k \in \mathbb{N}$. Since L is not semi-linear, the same result follows alternatively from Lemma 6.

In order to strengthen this inclusion, $\mathscr{L}$(SN) must be compared with other language families properly included in $\mathscr{L}$(CS). An interesting candidate is LOGCFL, the class of languages which are log-space reducible to $\mathscr{L}$(CF). LOGCFL is properly contained in P and $\mathscr{L}$(CS), respectively, and has several characterizations. A collection of problems in LOGCFL can be found in [3]. The next theorem has been shown in terms of equivalent finite copying parallel rewriting systems (e.g. [7]).

Theorem 11. $\mathscr{L}(\text{SN}) \subset \text{LOGCFL}$

The following lemma can alternatively be used to prove the strictness of the inclusion $\mathscr{L}(\text{SN}) \subset \text{LOGCFL}$. In addition, it generalizes a well-known result for context-free languages and is a useful tool for proving negative results.

Lemma 12. *Every unary language belonging to* $\mathscr{L}$(SN) *is regular.*

Proof. In the proof of Lemma 6 it has been shown that every language $L \in \mathscr{L}(\text{SN})$ is letter-equivalent to some context-free language L'. For unary languages this implies $L = L'$. Since every unary context-free language is regular [9], the lemma follows. □

For example, the LOGCFL-language $\{a^{2^n} \mid n \in \mathbb{N}\}$ does not belong to $\mathscr{L}$(SN). Finally, the lemma holds for all families $\mathscr{L}(k\text{-SN})$, too.

4. Closure Properties

Several closure properties of the families $\mathscr{L}$(SN) and $\mathscr{L}(k\text{-SN})$ are known by equivalent systems. It turned out that these properties are similar to the properties of context-free languages. For example, the next theorem has been shown in [16] in terms of local unordered scattered context grammars.

Theorem 13. *Let* $k \in \mathbb{N}$ *be a constant, then* $\mathscr{L}(k\text{-SN})$ *and* $\mathscr{L}$(SN) *are full AFLs closed under substitution, but not closed under intersection and complementation*

References

[1] Aho, A. V. and Ullman, J. D. *Translations on a context-free grammar*. Inform. Control 19 (1971), 439–475.

[2] Borodin, A., Cook, S. A., Dymond, P. W., Ruzzo, W. L., and Tompa, M. *Two applications of inductive counting for complementation problems*. SIAM J. Comput. 18 (1989), 559–578.

[3] Cook, S. A. *A taxonomy of problems with fast parallel algorithms*. Inform. Control 64 (1985), 2–22.

[4] Cremers, A. B. and Mayer, O. *On matrix languages*. Inform. Control 23 (1973), 86–96.

[5] Dassow, J. and Păun, G. *Regulated Rewriting in Formal Language Theory*. Springer, Berlin, 1989.

[6] Dassow, J., Păun, G., and Salomaa, A. *Grammars with controlled derivations*. In Rozenberg, G. and Salomaa, A. (eds.), *Handbook of Formal Languages 2*. Springer, Berlin, 1997, pp. 101–154.

[7] Engelfriet, J. *Context-free graph grammars*. In Rozenberg, G. and Salomaa, A. (eds.), *Handbook of Formal Languages 3*. Springer, Berlin, 1997, pp. 125–213.

[8] Engelfriet, J., Rozenberg, G., and Slutzki, G. *Tree transducers, L systems, and two-way machines.*. J. Comput. System Sci. 20 (1980), 150–202.

[9] Ginsburg, S. and Rice, H. G. *Two families of languages related to ALGOL*. J. ACM 9 (1962), 350–371.

[10] Ginsburg, S., Greibach, S. A., and Harrison, M. A. *One-way stack automata*. J. ACM 14 (1967), 389–418.

[11] Habel, A. and Kreowski, H.-J. *Some structural aspects of hypergraph languages generated by hyperedge replacement.*. Theoretical Aspects of Computer Science (STACS 1987), LNCS 274, 1987, pp. 207–219.

[12] Joshi, A. K., Levy, L. S., and Takahashi, M. *Tree adjunct grammars.*. J. Comput. System Sci. 10 (1975), 136–163.

[13] Neukamm, J. *Gekoppelt parallele kontextfreie Ersetzung*. Diploma thesis, Institut für Informatik, Universität Giessen, 2005.

[14] Parikh, R. J. *On context-free languages*. J. ACM 13 (1966), 570–581.

[15] Rambow, O. *Imposing vertical context conditions on derivations*. Developments in Language Theory II. At the Crossroads of Mathematics, Computer Science and Biology, 1996, pp. 257–266.

[16] Rambow, O. and Satta, G. *Independent parallelism in finite copying parallel rewriting systems.*. Theoret. Comput. Sci. 223 (1999), 87–120.

[17] Satta, G. *The membership problem for unordered vector languages*. Developments in Language Theory II. At the Crossroads of Mathematics, Computer Science and Biology, 1996, pp. 267–275.

[18] Seki, H., Nakanishi, R., Kaji, Y., Ando, S., and Kasami, T. *Parallel multiple context-free grammars, finite-state translation systems, and polynomial-time recognizable subclasses of lexical-functional grammars*. Association for Computational Linguistics (ACL 1993), 1993, pp. 130–139.

[19] Sudborough, I. H. *The complexity of the membership problem for some extensions of context-free languages*. Internat. J. Comput. Math. 6 (1977), 191–215.

[20] Vijay-Shanker, K., Weir, D. J., and Joshi, A. K. *Characterizing structural descriptions produced by various grammatical formalisms*. Association for Computational Linguistics (ACL 1987), 1987, pp. 104–111.

[21] Weir, D. J. *Characterizing mildly context-sensitive grammar formalisms*. Ph.D. thesis, Department of Computer and Information Science, University of Pennsylvania, 1988.