Context-Free Languages*

Jean Berstel Luc Boasson

June 8, 2008

*This is Chapter 2 of J. van Leeuwen (ed) Handbook of Theoretical Computer Science, Elsevier, 1990

Introduction

This chapter is devoted to the presentation of some recent results about context-free languages. There have been many results that appeared in the last years, both on classical topics and in new directions of research. The choice of the material to be presented was guided by the idea to emphasize on results which are not presented in any of the available textbooks [2, 12, 32, 33, 38, 55].

Context-free languages and grammars were designed initially to formalize grammatical properties of natural languages. They subsequently appeared to be most suitable for the formal description of the syntax of programming languages. This led to a considerable development of the theory. The recent research is oriented toward a more algebraic treatment of the main topics, in connection with mathematical theories; it also pursues investigations about famous open problems, such as the equivalence of deterministic pushdown automata, or the existence of principal cones with a principal cone of nongenerators.

Most of the theorems given in this chapter have been proved in the last five years. It appears (as systematically indicated in the text) that some of them constitute answers to questions listed in [3]. It should be observed that nearly all the conjectures of [3] are now solved (even if some of the solutions are not given here). As usual, these answers raise new questions, some of which are mentioned below.

Even when restricted to recent results, we had make a choice for the material to be presented. In the first section, we first illustrate the algebraic development of the theory by showing the existence of an invariant for context-free languages, namely the Hotz group. Then we give an account of recent refinements to the proof of inherent ambiguity by a clever investigation of generating functions.

Section 2 is devoted to iteration. We first prove the iteration lemma of Bader and Moura; we then discuss the interchange lemma and some of its applications, mainly to square-free words. Finally we prove that a context-free language which has only degenerated iterative pairs is in fact regular.

Section 3 describes the state of our knowledge concerning generators in cones of context-free languages. The main conjecture, namely that the cone of nongenerators of the context-free languages is not principal, is still open. New informations are: this cone is not the substitution closure of any strict subcone, and it is not generated by any family of deterministic context-free languages. New results concerning the "geography" of the context-free cones are also reported.

In the final section, we give an account of the theory of context-free groups, that is the groups for which the word problem is context-free. We give a global characterization, a description in terms of Caley graphs, and a relation to the theory of "ends".

There are major topics not presented here. Among them, the theory of rewriting systems has been the object of a recent monograph by Jantzen [41]. Connections to infinite words are only scarcely sketched in the text. For an overview, see [13] and [61]. The decidability of equivalence of deterministic context-free languages has made considerable progress in the last years. However, it seems not yet ripe for a systematic treatment, and the interested reader is referred to [51, 59, 62].

1 Languages

1.1 Notation and examples

1.1.1 Grammars

A *context-free grammar* $G = (V, A, P, S)$ is composed of a finite alphabet V, a subset A of V called the *terminal alphabet*, a finite set $P \subset (V - A) \times V^*$ of *productions*, and a distinguished element $S \in V - A$ called the *axiom*. A letter in $V - A$ is a *nonterminal* or *variable*.

Given words $u, v \in V^*$, we write $u \to v$ (sometimes subscripted by G or by P) whenever there exist factorizations $u = xXy$, $v = x\alpha y$, with (X, α) a production. A *derivation* of length $k \geq 0$ from u to v is a sequence $(u_0, u_1, \ldots, u_k)$ of words in V^* such that $u_{i-1} \to u_i$ for $i = 1, \ldots, k$, and $u = u_0$, $v = u_k$. If this hold, we write $u \overset{k}{\to} v$. The existence of some derivation from u to v is denoted by $u \overset{*}{\to} v$. If there is a proper derivation (i.e. of length ≥ 1), we use the notation $u \overset{+}{\to} v$. The *language generated by* G is the set

$$L(G) = \{w \in A^* \mid S \overset{*}{\to} w\}\,.$$

If X is a variable in G, we write

$$L_G(X) = \{w \in A^* \mid X \overset{*}{\to} w\}\,.$$

Thus $L(G) = L_G(S)$. A language L is called *context-free* if it is the language generated by some context-free grammar.

Consider a derivation $u = u_0 \to u_1 \to \cdots \to u_k = v$. It is a *leftmost* derivation if, for any derivation step $u_i \to u_{i+1}$, the variable in u_i which is replaced is the leftmost occurrence of a variable in u_i. Rightmost derivations are defined symmetrically. A *derivation tree* in G is a rooted, planted labelled tree. The internal nodes of the tree are labelled with variables. Leaves are labelled with elements in $A \cup \{1\}$, subject to the following condition. Let s be a node, and let $s_1, s_2, \ldots, s_n$ be the children of s, ordered from left to right. If X is the label of s, and if Y_i is the label of s_i, then $(X, Y_1 \cdots Y_n) \in P$. Moreover, if $n \geq 2$, then none of the Y_i's is the empty word. It is well-known that there is a bijection between derivation trees with root X and leaves in $A \cup \{1\}$, and leftmost derivations (rightmost derivations) from X into words over A.

Two grammars are *equivalent* if they generate the same language.

1.1.2 Examples

There are several convenient shorthands to describe context-free grammars. Usually, a production (X, α) is written $X \to \alpha$, and productions with same lefthand side are grouped together, the corresponding righthand sides being separated by a '+'. Usually, the variables and terminal letters are clear from the context. The language generated by a context-free grammar is denoted by the list of productions enclosed in a pair of brackets, the axiom being the first lefthand side of a production.

(i) The *Dyck languages*. Let $A = \{a_1, \ldots, a_n\}$, $\bar{A} = \{\bar{a}_1, \ldots, \bar{a}_n\}$ be two disjoint alphabets. The *Dyck language* over $A \cup \bar{A}$ is the language

$$D_n^* = \langle\, S \to TS + 1\,;\; T \to a_1 S \bar{a}_1 + \cdots + a_n S \bar{a}_n \,\rangle$$

The notation is justified by the fact that D_n^* is indeed a submonoid of $(A \cup \bar{A})^*$. It is even a free submonoid, generated by the language of *Dyck primes*

$$D_n = \langle\, T \to a_1 S \bar{a}_1 + \cdots + a_n S \bar{a}_n\,;\; S \to TS + 1 \,\rangle\,.$$

If $n = 1$, we write D^* and D instead of D_1^* and D_1.

The *two-sided Dyck language* over $A \cup \bar{A}$ is the language

$$\hat{D}_n^* = \langle\, S \,\to\, TS + 1\,;\; T \,\to\, \sum_{i=1}^{n} a_i S \bar{a}_i + \sum_{i=1}^{n} \bar{a}_i S a_i \,\rangle\,.$$

Again $\hat{D}_n^*$ is a free submonoid of $(A \cup \bar{A})^*$ generated by the set $\hat{D}_n$ of *two-sided Dyck primes*. This set is also context-free, and is generated by T in the following grammar

$$\begin{aligned}
T \,&\to \sum_{i=1}^{n} T_i + \sum_{i=1}^{n} \bar{T}_i\,,\\
T_i \,&\to a_i S_i \bar{a}_i\,; \quad \bar{T}_i \,\to\, \bar{a}_i \bar{S}_i a_i \qquad (i = 1, \ldots, n)\,,\\
S_i \,&\to 1 + \sum_{j=1}^{n} T_j S_i + \sum_{j \neq i} \bar{T}_j S_j \qquad (i = 1, \ldots, n)\,,\\
\bar{S}_i \,&\to 1 + \sum_{j \neq i} T_j \bar{S}_i + \sum_{j=1}^{n} \bar{T}_j \bar{S}_i \qquad (i = 1, \ldots, n)\,.
\end{aligned}$$

Again, we write $\hat{D}^*$ and $\hat{D}$ instead of $\hat{D}_1^*$ and $\hat{D}_1$.

There is an alternative way to define these languages as follows. Consider the congruence δ (resp.$\hat{\delta}$) over $A \cup \bar{A}$ defined by

$$a_i \bar{a}_i \equiv 1 \bmod \delta\ (i = 1, \ldots, n)\,,\ a_i \bar{a}_i \equiv \bar{a}_i a_i \equiv 1 \bmod \hat{\delta},\ (i = 1, \ldots, n)\,.$$

Then $D_n^* = \{w \in (A \cup \bar{A})^* \mid w \equiv 1 \bmod \delta\}$ and $\hat{D}_n^* = \{w \in (A \cup \bar{A})^* \mid w \equiv 1 \bmod \hat{\delta}\}$. Moreover, the quotient $(A \cup \bar{A})^*/\hat{\delta}$ is a group, called *the free group generated by* A and denoted by $F(A)$.

(ii) The *Lukasiewicz language* over a set $A = A_0 \cup A_1 \cdots \cup A_n$ partitioned into subsets A_i of symbols of "arity" i is the language

$$\langle S \,\to\, A_0 + A_1 S + \cdots + A_n S^n \rangle$$

The most well-known case is when $A_0 = \{b\}, A_2 = \{a\}$, and the other sets are empty. This gives the language

$$Ł = \langle S \,\to\, b + aSS \rangle\,.$$

(iii) The languages of *completely parenthesized expressions*

$$E_n = \langle S \,\to\, \sum_{k=1}^{n} a_k S b S c_k + d \rangle\,.$$

For $n = 1$, we write E instead of E_1: $E = \langle S \,\to\, aSbSc + d \rangle$.

(iv) The set of *palindromes* over an alphabet A

$$Pal = \langle S \,\to\, \sum_{a \in A} aSa + \sum_{a \in A} a + 1 \rangle$$

is the set of words $w \in A^*$ with $w = w^\sim$ where $w^\sim$ denotes the *reversal* of w. Related to this set are the *symmetric languages* Sym_n defined over the alphabet $\{a_1, \ldots, a_n, \bar{a}_1, \ldots, \bar{a}_n\}$ by

$$Sym_n = \langle S \,\to\, \sum_{i=1}^{n} a_i S \bar{a}_i + 1 \rangle\,.$$

Contrary to previous conventions, Sym will denote the language Sym_2.

It is interesting to observe that the languages

$$\{w\#w^{\sim} \mid w \in A^*\} \text{ and } \{w\#w' \mid w' \neq w^{\sim}\}$$

(with $\#$ not in A) are both context-free. On the contrary, the language $Copy = \{w\#w \mid w \in A^*\}$ is not context-free (as can be shown by one of the pumping lemmas given below); however, the language $\{w\#w' \mid w' \neq w\}$ *is* context-free.

(v) The *Goldstine language* G over $\{a, b\}$ is the set G of words

$$a^{n_1}ba^{n_2}b\cdots a^{n_p}b$$

with $p \geq 1, n_i \geq 0$, and $n_j \neq j$ for some j, $1 \leq j \leq p$. To see that this language is context-free we start with the context-free language

$$\{a^pb^qc \mid q \neq p+1,\ q, p \geq 0\}$$

and then apply the substitution

$$a \mapsto a^*b\,, \quad b \mapsto a\,, \quad c \mapsto b(a^*b)^*\,.$$

The language G is the result of this substitution. Since rational (and even context-free) substitution preserves context-freeness, the language G is context-free.

Observe that G is related to the infinite word

$$\mathbf{x} = aba^2ba^3b\cdots a^nba^{n+1}b\cdots$$

Let indeed

$$\text{Co-Pref}(\mathbf{x}) = \{w \mid w \text{ is not a prefix of } \mathbf{x}\}\,.$$

Then G is just composed of those words in Co-Pref$(\mathbf{x})$ which end with the letter b. Further, consider the context-free language

$$\{a^nb^p \mid p > n+1,\ n \geq 0\}$$

and then apply the substitution

$$a \mapsto a^*b\,, \quad b \mapsto a\,.$$

Let H be the resulting language. Then

$$\text{Co-Pref}(\mathbf{x}) = Ga^* \cup H\,.$$

This shows that Co-Pref$(\mathbf{x})$ is a context-free language.

1.1.3 Ambiguity

A grammar $G = (V, A, P, S)$ is *unambiguous* if every word in $L(G)$ has exactly one left most derivation. It is equivalent to say that there is only one derivation tree for each word, whence only one right derivation. A language is *unambiguous* if there is an unambiguous grammar to generate it, otherwise it is called *inherently ambiguous*.

Ambiguity is undecidable. However there are techniques that work in special cases to prove inherent ambiguity. One method is by using iteration lemmas in a clever way. (see e.g. Harrison [38]). This can be used for instance to prove that the language $\{a^nb^pc^q \mid n = p \text{ or } p = q\}$ is inherently ambiguous. The same method applies sometimes to unambiguity relatively to a subclass. Let us just give one example. A grammer is *linear* for if, for every production, the righthand side contains at most one occurrence of a variable. A language is linear if there exists a linear grammar that generates it.

Consider the following language over $\{a, b, \#\}$:

$$M = \{a^n b^n \# a^p b^q \mid n, p, q \geq 1\} \cup \{a^p b^q \# a^n b^n \mid n, p, q \geq 1\}\,.$$

This language is linear. However, it can be shown that every linear grammar generating M is ambiguous. On the other hand

$$M = \{a^n b^n \# a^p b^q \mid n, q, p \geq 1\} \cup \{a^n b^m \# a^p b^p \mid n, m, p \geq 1, n \neq m\}$$

is the disjoint union of two (non linear) languages, which both are unambiguous, thus M is unambiguous.

We shall see another way to attack ambiguity below.

1.1.4 Reduced grammars, normal forms

There exist a great number of normal forms for grammars. These normal forms have mainly theoretical interest and are of little help in pratical applications such as parsing. Reduction is a first step toward these normal forms.

A grammar $G = (V, A, P, S)$ is *reduced* if the following 3 conditions are fulfilled :

(i) for every nonterminal X, the language $L_G(X)$ is nonempty ;
(ii) for every $X \in V - A$, there exist $u, v \in A^*$ such that $S \stackrel{*}{\rightarrow} uXv$;
(iii) the axiom S appears in no righthand side of a production.

It is not difficult to see that for every grammar G with $L(G) \neq \emptyset$, an equivalent reduced grammar can effectively be constructed. A variation of this construction which sometimes is useful, requires that $L_G(X)$ is infinite for every variable X. A production is called an *ε-production* if its righthand side is the empty word. At least one ε-production is necessary if the language generated by the grammar contains the empty word. It is not too difficult to construct, for every context-free grammar G, an equivalent grammar with no ε-production excepted a production $S \rightarrow 1$ if $1 \in L(G)$. The final special kind of grammars we want to mention is the class of proper grammars. A grammar G is *proper* if it has neither ε-productions nor any production of the form $X \rightarrow Y$, with Y a variable. Again, an equivalent proper grammar can effectively be constructed for any grammar G if $L(G) \not\ni 1$. The two most common normal forms for context-free grammars are the so called Chomsky normal form and the Greibach normal form. A grammar $G = (V, A, P, S)$ is in *Chomsky normal form* if every production $X \rightarrow \alpha$ satisfies $\alpha \in A \cup (V - A)^2$. It is in *Greibach normal form* if $\alpha \in A \cup A(V - A) \cup A(V - A)^2$. For every context-free grammar G with $1 \notin L(G)$ equivalent grammars in Chomsky normal form and in Greibach normal form can effectively be constructed. A less usual normal form is the *double Greibach normal form* where every production $X \rightarrow \alpha$ satisfies

$$\alpha \in A \cup A^2 \cup A[(V - A) \cup (V - A)^2]A\,.$$

There again, for every context-free grammar G with $1 \notin L(G)$, an equivalent grammar in double Greibach normal form can be effectively constructed (Hotz [39]). A very large variety of such normal forms exists (Blattner, Ginsburg [14]).

1.1.5 Systems of equations

Let $G = (V, A, P, S)$ be a context-free grammar. For each variable X, let P_X be the set of righthand sides of productions having X as lefthand side. With our notation, the

set of productions can be written as

$$X \to \sum_{p \in P_X} p \qquad (X \in V - A)$$

or simply as

$$X \to P_X \qquad (X \in V - A)\,.$$

The *system of equations* associated to the grammar G is the set of equations

$$X = P_X \qquad (X \in V - A)\,.$$

A solution of this system is a family $L = (L_X)_{X \in V-A}$ of subsets of A^* such that

$$L_X = P_X(L) \qquad (X \in V - A)\,,$$

with the notation

$$P_X(L) = \bigcup_{p \in P_X} p(L)$$

and $p(L)$ being the product of languages obtained by replacing, in p, each occurrence of a variable Y by the language L_Y. Solutions of a system of equations are ordered by component-wise set inclusion. Then one has :

1.1. THEOREM (Schützenberger [57]). *Let $G = (V, A, P, S)$ be a context-free grammar. The family $L = (L_X)$ with $L_X = L_G(X)$ is the least solution of the associated set of equations. If the grammar G is proper, then the associated system has a unique solution.*

1.2. EXAMPLE. The grammar $S \to aSS + b$ is proper. Thus the Lukasiewicz language $Ł$ is the unique language satisfying the equation $Ł = aŁŁ \cup b$.

1.3. EXAMPLE. The grammar $S \to S$ generates the empty set which is the least solution of the equation $X = X$. Every language is indeed solution of this equation.

For more details along these lines see [43], and [56].

1.1.6 Pushdown automata

A *pushdown automaton* (pda) $\mathcal{A} = (A, V, Q, \delta, v^0, q^0, Q')$ is composed of a finite terminal alphabet A, a finite nonterminal alphabet V, a (nondeterministic) transition function δ from $(A \cup \{\varepsilon\}) \times Q \times V$ into the finite subsets of $Q \times V^*$, an initial pushdownstore symbol v^0 in V, an initial state q^0 in Q and a set Q' of terminal states, a subset of Q. A *configuration* of $\mathcal{A}$ is a triple $c = (\gamma, q, x)$ in $V^* \times Q \times A^*$. The automaton moves directly from configuration $c = (\gamma, q, x)$ into configuration $c' = (\gamma', q', x')$, denoted $c \vdash\!\!\!- \; c'$ iff
— either $\gamma = \gamma_1 v$ $(v \in V)$, $x = ax'$ $(a \in A)$, $\gamma' = \gamma_1 m$ $(m \in V^*)$ and $\delta(a, q, v) \ni (q', m)$; this is a "a-move".
— or $\gamma = \gamma_1 v$ $(v \in V)$, $x = x'$, $\gamma' = \gamma_1 m$ $(m \in V^*)$ and $\delta(\varepsilon, q, v) \ni (q', m)$; this is a "$\varepsilon$-move".

We denote $\overset{*}{\vdash\!\!\!-}$ the reflexive and transitive closure of the relation $\vdash\!\!\!-$, and we define the *language recognized by empty store* by $\mathcal{A}$ as

$$\mathit{Null}(\mathcal{A}) = \{x \in A^* \mid (v^0, q^0, x) \overset{*}{\vdash\!\!\!-} (1, q, 1) \quad q \in Q\}\,,$$

and the *language recognized by terminal state* by $\mathcal{A}$ as

$$T(\mathcal{A}) = \{x \in A^* \mid (v^0, q^0, x) \overset{*}{\vdash\!\!-} (\gamma, q', 1) \quad \gamma \in V^*,\ q' \in Q'\}\,.$$

The context-free languages are then characterized in terms of pda's: $L \subset A^*$ is context-free iff there exists a pda $\mathcal{A}$ such that $L = T(\mathcal{A})$ (resp. $L = Null(\mathcal{A})$). Moreover, this result holds even if $\mathcal{A}$ is restricted to be *realtime* (= involves no ε-moves in its transition function).

A pda is *deterministic* (is a dpda) iff, for each q in Q and each v in V,
— *either* $\delta(\varepsilon, q, v)$ is a singleton in $Q \times V^*$ and, for each a in A, $\delta(a, q, v) = \emptyset$;
— *or* $\delta(\varepsilon, q, v) = \emptyset$ and, for each a in A, the set $\delta(a, q, v)$ is either empty or a singleton.

A context-free language L is *deterministic* iff there exists a dpda $\mathcal{A}$ such that $L = T(\mathcal{A})$. It should be noted that, contrarily to what happens for nondeterministic pda's, the family of languages recognized by empty store by a dpda $\mathcal{A}$ forms a strict subfamily of the deterministic languages. Similarly, the family of realtime dpda's gives raise to a strict subfamily of deterministic languages. (See [38] for all these classical results)

1.2 Hotz group

One of the most interesting questions concerning the relation between grammars and languages is whether two grammars are equivalent, i.e. generate the same language. Since this question is undecidable, one may look for weaker formulations of this question, i.e. properties which are implied by the equivalence of context-free grammars. One such invariant has been discovered by G. Hotz [40] and is the topic of this section.

Consider any set A. The *free group* $F(A)$ over A is the quotient monoid

$$F(A) = (A \cup \bar{A})^*/\hat{\delta}$$

where $\bar{A} = \{\bar{a} \mid a \in A\}$ is a disjoint copy of A and where $\hat{\delta}$ is the congruence generated by the relations $a\bar{a} \equiv \bar{a}a \equiv 1, \quad (a \in A)$ (see also Example (i) in Section 1.1.2.

Let $G = (V, A, P, S)$ be a context-free grammar. The *Hotz group* of G is the group

$$\mathcal{H}(G) = F(V)/[P]\,,$$

where $[P]$ is the group congruence generated by P, that is $u \equiv v \bmod [P]$ iff u and v can be obtained one from each other by successive application of productions or their inverses in *both* directions : $u \equiv v \bmod [P]$ iff there exist $k \geq 0$ and $w_0, \ldots, w_k$ such that $u = w_0, v = w_k$, and for $i = 0, \cdots, k-1$,

$$w_i \to w_{i+1} \quad \text{or} \quad w_{i+1} \to w_i \quad \text{or} \quad \bar{w}_i \to \bar{w}_{i+1} \quad \text{or} \quad \bar{w}_{i+1} \to \bar{w}_i\,.$$

1.4. EXAMPLE. For the grammar $G = \langle S \to 1 + aSb \rangle$, the congruence is generated by $S \equiv 1$, $S \equiv aSb$; clearly $a^n b^p \equiv a^{n-p}$, for $n, p \in \mathbf{Z}$. Thus $\mathcal{H}(Q) = \mathbf{Z}$.

1.5. THEOREM. *Let G_1 and G_2 be two reduced context-free grammars. If $L(G_1) = L(G_2)$, then $\mathcal{H}(G_1) \cong \mathcal{H}(G_2)$.*

This result means that the Hotz group, defined for a grammar, is in fact a property of the language generated by the grammar. It is well-known that other algebraic objects are associated with a formal language. The most frequently quoted is the syntactic monoid (for a discussion, see Perrin's Chapter [52]).

Theorem 1.5 is an immediate consequence of the following intrinsic characterization of the Hotz group. Let $L \subset A^*$ be a language. The *collapsing group* of L is the quotient

$$\mathcal{C}(L) = F(A)/[L \times L]$$

of the free group over A by the finest (group) congruence such that L is contained in a single class.

1.6. THEOREM. *Let L be a context-free language, and let G be a reduced context-free grammar generating L; then the collapsing group of L and the Hotz group of G are isomorphic, i.e. $\mathcal{C}(L) \cong \mathcal{H}(G)$.*

PROOF. Let $G = (V, A, P, S)$. For convenience, we denote the congruence $\mathrm{mod}[P]$ by $\sim$, and the congruence $\mathrm{mod}[L \times L]$ by $\equiv$. Observe that $\sim$ is defined over $F(V)$, and $\equiv$ is only defined over $F(A)$.

We first show that if $u, v \in F(A)$, then

$$u \equiv v \Longleftrightarrow u \sim v\,.$$

For this, consider words $w, w' \in L$. Then $S \overset{*}{\rightarrow} w'$. Consequently $w \sim S$, and $S \sim w'$, whence $w \sim w'$. By induction, it follows that for $u, v \in F(A)$,

$$u \equiv v \Rightarrow u \sim v\,.$$

Conversely, consider words $w, w' \in L_G(X)$ for some variable X. Since the grammar is reduced, uwv, $uw'v \in L$ for some words $u, v \in A^*$. Thus $uwv \equiv uw'v$, and since we have a group congruence, it follows that $w \equiv w'$. Thus each $L_G(X)$ is contained in a single class of the collapsing congruence.

Consider now, for each variable X, a fixed word $\varphi(X) \in L_G(X)$, and extend φ to a morphism from V^* into A^*, and then from $F(V)$ into $F(A)$, by setting $\varphi(a) = a$ for $a \in A$. In view of the previous discussion, given any production $X \rightarrow \alpha$ in P, one has $\varphi(X) \equiv \varphi(\alpha)$. Assume now that $u, v \in F(V)$, and $u \sim v$. Then there are $k \geq 0$, and $w_0, \ldots, w_k \in F(V)$, such that $u = w_0, v = w_k$, and $\varphi(w_i) \equiv \varphi(w_{i+1})$ for $i = 0, \ldots, k-1$. Consequently $\varphi(u) \equiv \varphi(v)$, and since $\varphi(u) = u$, $\varphi(v) = v$, this shows that $u \equiv v$. Thus, our claim is proved.

Let p be the canonical morphism

$$p : F(V) \rightarrow F(V)/\sim\ (\cong \mathcal{H}(G))\,.$$

In order to complete the proof, it suffices to observe that $\mathcal{H}(G) = p(F(A))$, since for each $w \in V^*$, one has $w \sim \varphi(w)$, and $\varphi(w) \in F(A)$. □

The concept of collapsing group appears in [64]. A more systematic formulation of Theorem 1.6 is given by Frougny et al. [30]. For recent developments on these lines, see [23, 24].

The relation between the congruence $[P]$ and the so-called NTS languages is the following : rather than considering the group $F(V)/[P]$, we may consider the quotient monoid $V^*/[P]$ also called the *Hotz monoid*. Obviously, this is no more an invariant. However, it may happen, for some grammar $G = (V, A, P, S)$, that the congruence class of each variable X is exactly the set of sentential forms generated by X :

$$\{x \in V^* \mid x \equiv X \bmod [P]\} = \{x \in V^* \mid X \overset{*}{\rightarrow} x\}\,.$$

In that case, the grammar G is called an *NTS-grammar*. Some striking results concerning these grammars are (Sénizergues [58]):

(i) languages generated by NTS-grammars are deterministic context-free languages;
(ii) the equivalence of NTS-grammars is decidable;
(iii) given a context-free grammar G, it is decidable whether G is an NTS-grammar.

There remains an interesting open problem: is the family of NTS-languages (i.e. languages generated by NTS-grammars) closed under inverse morphism? This question seems to be related to another one concerning a weakening of the NTS condition. A grammar $G = (V, A, P, S)$ is called *pre-NTS* if, for each variable X, the restriction of its congruence class to terminal words is exactly the language generated by X, thus if $L_G(X) = \{x \in A^* \mid X \equiv x \bmod [P]\}$. Clearly, any NTS-grammar is pre-NTS. The question is whether the converse holds for languages.

1.3 Ambiguity and transcendence

As already mentioned above, a proof that a given context-free language is inherently ambiguous by means of combinational arguments is rather delicate. The reason for this is that one has to show that *every* grammar is ambiguous. Another reason is that pumping lemmas like those described in the next section only concern local structure.

A fundamental technique for proving ambiguity is based on the use of generating functions. This technique has recently been refined and successfully employed by Flajolet [28, 29]. Let $L \subset A^*$ be any language over a finite alphabet A. The *generating function* of L is given by the series

$$f_L(z) = \sum a_n z^n \,,$$

where

$$a_n = \mathrm{Card}\{w \in L \mid |w| = n\} \,.$$

Since $a_n \leq \mathrm{Card}(A)^n$ for $n \geq 0$, the series $f_L(z)$ is an analytic function in the neighbourhood of the origin, and its radius of convergence ρ satisfies $\rho \geq 1/\mathrm{Card}(A)$. The basic result for the study of ambiguity is the following classical theorem of Chomsky and Schützenberger:

1.7. THEOREM. *Let $f_L(z)$ be the generating function of a context-free language L. If L is unambiguous, then $f_L(z)$ is an algebraic function.*

This result means of course that if $f_L(z)$ is transcendental and L is context-free, then L is inherently ambiguous. In order to use this statement, is suffices to rely on well-known classical families of transcendental functions or to use more or less easy criteria for transcendence. We just give some examples.

1.8. PROPOSITION. *The Goldstine language G is inherently ambiguous.*

PROOF. Consider indeed a word which is not in G. Then either it ends with the letter a, or it is one of the words in the set

$$F = \{1, ab, aba^2b, aba^2ba^3b, \cdots\} \,.$$

The generating function of the words over $A = \{a, b\}$ ending with the letter a is $z/(1-2z)$. The generating function of the set F is

$$1 + z^2 + z^5 + z^9 + \cdots = \sum_{n \geq 1} z^{n(n+1)/2-1} \,.$$

Thus

$$f_G(z) = \frac{1-z}{1-2z} - \frac{1}{z} \sum_{n \geq 1} z^{n(n+1)/2} = \frac{1-z}{1-2z} - \frac{g(z)}{z} \,.$$

Now $f_G(z)$ is transcendental iff $g(z)$ is so. And indeed, $g(z)$ is trancendental. To see this, recall that an algebraic function has a finite number of singularities. On the other hand, a powerful "gap theorem" [54] states that if for some series $h(z) = \sum_{n\geq 0} a_n z^{c_n}$ the exponents satisfy $\sup(c_{n+1} - c_n) = \infty$, then $h(z)$ admits its circle of convergence as a natural boundary. This holds for our function $g(z)$ which thus has infinitely many singularities and is not algebraic. □

Two points should be observed. First, the fact that the generating function is algebraic holds for a language as well as for its complement. Thus, there is no way to get any converse of the Chomsky-Schützenberger theorem. Next, the technique of natural boundaries is quite general for proving transcendence of functions.

1.9. PROPOSITION. *Let $A = \{a, b\}$, and let C be the language of products of two palindromes:*

$$C = \{w_1 w_2 \mid w_1 w_2 \in A^*,\ w_1 = w_1^{\sim},\ w_2 = w_2^{\sim}\}\,.$$

The language C is inherently ambiguous.

A first proof of this proposition is due to Crestin [21] who has proved that its ambiguity is unbounded i.e. that there exists no bound for the number of leftmost derivations for some word in any grammar. Kemp [42] computed the generating function of C which is

$$f_C(z) = 1 + 2\sum_{m\geq 1} \mu(m) \frac{z^m(1+z^m)(1+2z^m)}{(1-2z^{2m})^2}\,,$$

where $\mu(m) = \prod_{p|m}(1-p)$, the product being over all prime divisors of m. A delicate analysis by Flajolet [29] shows that $f_C(z)$ has singularities at the points $2^{1/2m}e^{ij/m}$. Thus there are infinitely many singularities and $f_C(z)$ is not algebraic. □

1.10. PROPOSITION. *The language*

$$L = \{w \in A^* \mid |w|_a \neq |w|_b \text{ or } |w|_b \neq |w|_c\}$$

over $A = \{a, b, c\}$ is inherently ambiguous.

PROOF. The complement M of this language is

$$M = \{w \in A^* \mid |w|_a = |w|_b = |w|_c\}$$

and its generating function is

$$f_M(z) = \sum_{n\geq 0} \frac{(3n)!}{(n!)^3} z^{3n} = \sum a_n z^{3n}\,.$$

In order to show that $f_M(z)$ is not algebraic, we observe that by Stirling's formula

$$a_n \sim 3^{3n}\frac{\sqrt{3}}{2\pi n}\,.$$

On the other hand, Flajolet [29] shows that an asymptotic equivalent of the form β^n/n is characteristic for transcendental functions. Thus $f_M(z)$ and $f_L(z)$ are transcendental. □

Flajolet's paper [29] contains a lot of other examples of inherently ambiguous languages, and develops a systematic classification of languages which can be handled by analytic methods (See also [6]).

2 Iteration

Iteration is the most direct method in order to prove that a formal language is not in a given family of languages. In general, the iteration is on the words of the language and reflects some property of the way languages in the considered family are constructed. The iteration lemmas in fact give a property of "regularity" which expresses the finiteness of the construction mechanism.

There exist numerous results concerning iteration, depending on the type of languages studied. For a fixed family of languages, there may exist several variations. A bibliography on this topic has been compiled by Nijholt [49].

We are concerned here with iteration lemmas for context-free languages. We give three recent results of different nature. The first (Theorem 1.3) is the conclusion of several statements of increasing precision concerning the constraints for iterative pairs in a context-free language. The second (Theorem 2.1) states a different kind of property : if a language is "rich" i.e. if there are "many" words of given length, then factors of words may be interchange without leaving the language.

The third result (Theorem 3.1) expresses to what amount an iteration property may characterize languages : this is interesting because iteration is only a "local" property of words, and therefore is difficult to relate to global properties of grammars.

2.1 Iteration lemmas

Consider a fixed language $L \subset A^*$. An *iterative pair* in L is a tuple of words $\eta = (x, u, y, v, z)$ with $uv \neq 1$, such that for all $n \geq 0$

$$xu^n yv^n z \in L\,.$$

Given a word $w \in L$, the iterative pair η is a pair *for* w provided $w = xuyvz$; a word w is said to *admit* an iterative pair if there exists an iterative pair for w.

2.1. EXAMPLE. For the language D of Dyck primes over $\{a, \bar{a}\}$, the tuple $(a, a, 1, \bar{a}, \bar{a})$ is an iterative pair. However, $(1, a, 1, \bar{a}, 1)$ is not because $1 \notin D$. The latter is of course an iterative pair in D^*.

An easy way to construct iterative pairs for an infinite algebraic language L generated by some (reduced) grammar $G = (V, A, P, S)$ is the following. Since L is infinite, there exists in G some variable X and some derivation $X \overset{*}{\to} uXv$ for some words $u, v \in A^*$. Since G is reduced,

$$S \overset{*}{\to} xXz\,, \quad X \overset{*}{\to} y$$

for some words x, y, z in A^*. But then

$$S \overset{*}{\to} xu^n yv^n z \quad \text{for all } n \geq 0.$$

Historically the first iteration result concerning context-free languages is the following :

2.2. THEOREM (Bar Hillel, Perles and Shamir [38]). *Let $L \subset A^*$ be a context-free language. There exists an integer N such that any word $w \in L$ of length $|w| \geq N$ admits an iterative pair in L.*

This theorem is a special case of Ogden's refinement, the result stated below. In order to formulate it, we need a definition.

Let $w \in A^*$ be a word of length n. A *position* in w is any integer in $\{1, 2, \ldots, n\}$. Choosing a set of distinguished position in w thus is equivalent to choosing a subset of $\{1, \ldots, n\}$.

2.3. THEOREM (Ogden [38]). *Let $L \subset A^*$ be a context-free language. There exists an integer N such that for any word $w \in L$ and for any choice of at least N distinguished positions in w, there exists an iterative pair $\eta = (x, u, y, v, z)$ for w in L such that, in addition*

(i) *either x, u, y each contain at least one distinguished position or y, v, z each contain at least one distinguished position;*
(ii) *the word uyv contains at most N distinguished position.*

Observe that in (i), both conditions may be satisfied. Theorem 2.2 is a consequence of this result obtained by considering that all positions in a word are distinguished.

This theorem means that the place of the iterating group uyv within the factorization $w = xuyvz$ can be chosen to some extent : assume that the distinguished positions are chosen to be consecutive in w. Then either u or v is entirely composed of letters at distinguished positions.

One cannot expect to have a much more precise information on the position of u and v within the word w, and in particular it is impossible to force both u and v to be at distinguished places. However, the following result of Bader and Moura indicate that, to some extent, both u and v cannot be at certain places.

2.4. THEOREM ((Bader and Moura [7])). *Let L be a context-free language. There exists an integer N such that, for any word w, and for any choice of at least d distinguished and e excluded positions in w with $d > N^{1+e}$, there exists an iterative pair*

$$\eta = (x, u, y, v, z)$$

for w in L such that, in addition

(i) *either x, u, y each contain at least one distinguished position or y, v, z each contain at least one distinguished position ;*
(ii) *the word uv contains no excluded position ;*
(iii) *if uyv contains r distinguished and s excluded positions, then $r \leq N^{1+s}$.*

Before going into the proof, let us give a first example of the use of the theorem, to show how it works.

2.5. EXAMPLE. Let $A = \{a, b\}$, and let

$$L = b^* \cup aa^+b^* \cup \{ab^p \mid p \text{ prime}\}\,.$$

This language is not context-free of course. However Ogden's iteration lemma does not prove it, because there is no way to get rid of pumping the initial a. On the other hand, when the initial a is considered as in an excluded position, then Bader and Moura's iteration lemma ensures that $uv \in b^*$, and then it is easily seen that L is not context-free.

PROOF OF THEOREM 2.4. Let $G = (V, A, P, S)$ be a context-free grammar generating L. Let t be the maximal length of the righthand side of the productions, and let $N = t^{2k+6}$ with k being the number of nonterminals in G.

Let $w \in L$ be a word with e excluded and $d > N^{1+e}$ distinguished positions. Consider a derivation tree for w. A node in the tree is a *branch point* if it has at least

two children which have distinguished descendants. Let P be a path with the greatest number of branch points. Since w has at least $t^{2(k+3)(e+1)}$ distinguished positions, the path P has at least $2(k+3)(e+1)$ branch points. The branch points in path P are grouped into two sets. A *left* branch point is a branch point having a child with a distinguished descendant to the left of path P. Right branch points are defined symmetrically. Observe that a node may be both a left and a right branch point. Clearly P has at least $(k+3)(e+1)$ left or at least $(k+3)(e+1)$ right branch points.

Assume that P has at least $(k+3)(e+1)$ left branch points and divide the lowermost part of P in $e+3$ subpaths. The subpath nearest to the bottom contains $e+1$ left branch points, each of the following $e+1$ "internal paths" has $k+1$ left branch points and the topmost again $e+1$ left branch points (see Fig. 1).

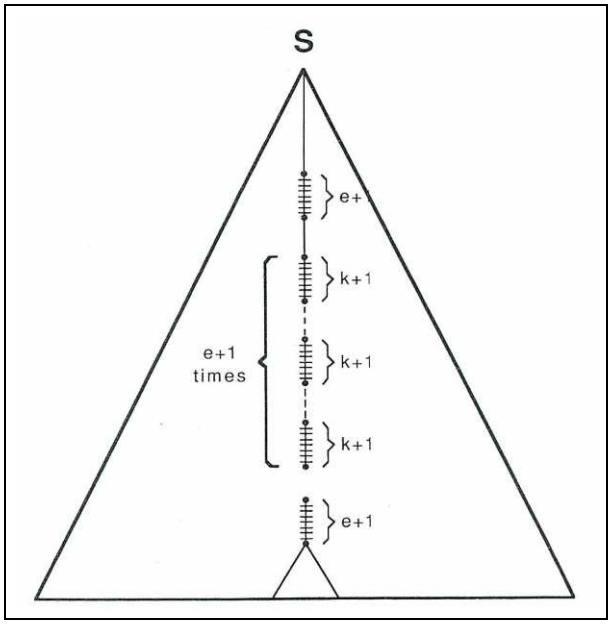

Figure 1: Factorization of a path

In each of the internal subpaths P_i, $(0 \leq i \leq e)$ there must be two branch points with the same label, say X_i. Thus there exist words $u_i, v_i \in A^*$, such that $X_i \stackrel{*}{\rightarrow} u_i X_i v_i$. Moreover, since the node is a left branch point, the word u_i contains at least one distinguished position.

Now observe that there are $e+1$ pairs of words (u_i, v_i) but only e excluded positions. Therefore one of the words $u_i v_i$ no excluded position.

The remaining claims are easily verified. □

2.6. EXAMPLE. As an application of the previous iteration lemma, we consider the following situation. Given an infinite word

$$\mathbf{x} = a_0 a_1 a_2 \cdots a_n \cdots$$

over some alphabet A, the language

$$\text{Co-Pref}(\mathbf{x}) = \{w \mid w \neq a_0 a_1 \cdots a_{|w|-1}\} \subset A^*$$

is the set of all words over A which are not initial segments of $\mathbf{x}$. We consider infinite

words of the form

$$x = a^{f(1)}ba^{f(2)}b\cdots a^{f(n)}b\cdots \tag{1}$$

where $f : \mathbb{N} \to \mathbb{N}$ is some function, and we prove that Co-Pref(x) is not context-free provided f grows rapidly enough. Assume indeed that for some function f, the language Co-Pref(x) is context-free, let N be the integer associated with this language in Theorem 2.4, and consider the word

$$w = a^{f(1)}b^{f(2)}b\cdots ba^{1+f(n)} \in \text{Co-Pref}(\mathbf{x})\,.$$

Declare that all positions in $a^{f(1)}b\cdots a^{f(n-1)}b$ are excluded, and that the last $1+f(n)$ are distinguished. In order to be allowed to apply the theorem, it is required that

$$1 + f(n) > N^{n+\sum_{i=1}^{n-1} f(i)}$$

It is easily seen that this holds for large enough n whenever

$$f(n) \geq 2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}$$

where there are $2n$ stacked exponents. In this case, w has an iterative pair $\eta = (\alpha, u, \beta, v, \gamma)$ with $u\beta v\gamma$ a right factor of the factor $a^{1+f(n)}$. Consequently $\alpha\beta\gamma$ is an initial segment of $\mathbf{x}$, contradicting the theorem.

Let us mention that this brute force technique gives weaker results than Grazon's clever iterated use of Ogden's Lemma [35]. It appears indeed to be rather involved to prove that a language of the form Co-Pref$(\mathbf{x})$ is not context-free. It has been shown by Grazon that all these languages satisfy Ogden's lemma. M.G. Main (personal communication) observed that they also satisfy the interchange lemma given in the next paragraph. A. Grazon shows that for a language Co-Pref$(\mathbf{x})$ with $\mathbf{x}$ of the form (1) to be not context-free, it suffices that $f(n) \geq 2^n$ for $n \geq 1$.

2.2 Interchange lemma

The interchange lemma we describe below gives a different kind of constraint for words of a context-free language. Instead of iteration, the unavoidable property described concerns the possibility to exchange factors of words in some positions without leaving the language. One interesting aspect of this lemma is that it holds for languages which have "many" words of given length. (Observe that this is precisely the case where the classical iteration lemma are difficult to use).

Let L be a language over some alphabet A, and let $n \geq 0$ be an integer. A subset $R \subset L \cap A^n$ is an *interchange* set for L if there exist integers p, q with $0 \leq p+q \leq n$ such that for all u, v, w, u', v', w' with $uvw, u'v'w' \in R$, $|u| = |u'| = p$, $|v| = |v'| = q$ implies $uv'w,\ u'vw' \in L$. If this holds, the integer q is called the *span* of R.

2.7. THEOREM ("interchange lemma", Ogden, Ross, Winklmann [50]). *Let $L \subset A^*$ be a context-free language. There exists a number C, such that, for any integers n, m with $2 \leq m \leq n$ and for any set $Q \subset L \cap A^n$, there exists an interchange set $R \subset Q$ for L of size*

$$\mathrm{Card}(R) \geq \frac{\mathrm{Card}(Q)}{Cn^2}$$

and of span q with $m/2 \leq q \leq m$.

Clearly, the interchange set R may be empty unless $\mathrm{Card}(Q) \geq Cn^2$, which means in pratice that the number of words in $L \cap A^n$ should grow faster than n^2.

PROOF. Let L be generated by a context-free grammar $G = (V, A, P, S)$ in Chomsky normal form. Let n be an integer, and let $Q \subset L \cap A^n$. For all $X \in V - A$, and integers n_1, n_2 with $0 \leq n_1 + n_2 \leq n$, we denote by $Q(n_1, X, n_2)$ the set of words $w \in Q$ such that there is a derivation

$$S \stackrel{*}{\rightarrow} uXv \stackrel{*}{\rightarrow} uxv = w$$

with $|u| = n_1, |v| = n_2$. Clearly each set $Q(n_1, X, n_2)$ is an interchange set with span $n - (n_1 + n_2)$. It is also clear that $Q(n_1, X, n_2) \subset Q$.

Let now m be any integer that satisfies $2 \leq m \leq n$. We claim that

$$Q \subset \bigcup Q(n_1, X, n_2) \tag{2}$$

where the union is over all sets with span $q = n - (n_1 + n_2)$ satisfying the relation $m/2 < q \leq m$.

Let indeed $w \in Q$. Then clearly $w \in Q(n_1, X, n_2)$ for some n_1, X, n_2. It remains to shows that the parameters n_1, X, n_2 can be chosen in such a way that the span of $Q(n_1, X, n_2)$ is in the desired interval. It is clear that the span can always be chosen greater than $m/2$ (take $X = S$). Assume now that span q is strictly greater than m. Then the derivation $X \stackrel{*}{\rightarrow} x$ where $|x| = q$, may be factorized into

$$X \rightarrow YZ \stackrel{*}{\rightarrow} x = yz$$

for some y, z with $Y \stackrel{*}{\rightarrow} y$ and $Z \stackrel{*}{\rightarrow} z$. Clearly one of the words y or z has length strictly greater than $m/2$. Assume it is y. Then w is in $Q(n_1, Y, n_2 + |z|)$ which has span $q - |z|$. The conclusion follows by induction.

Now observe that in Equation 2, the union is over at most $\mathrm{Card}(V - A) \cdot n^2$ terms. Consequently, there is at least one set $R = Q(n_1, X_1, n_2)$ with

$$\mathrm{Card}(R) \geq \frac{\mathrm{Card}(Q)}{\mathrm{Card}(V - A) \cdot n^2} .$$

□

We now apply the Interchange Lemma to prove that a special language is not context-free. For this, we call a *square* any word of the form uu, with u nonempty. A word is *squarefree* if none of its factors is a square. It is easily seen that there are only finitely many squarefree words over 1 or 2 letters. We quote without proof the following result.

2.8. THEOREM (Thue [44]). *The set of squarefree words over a three letter alphabet is infinite.*

For a proof and a systematic exposition of the topic, see [44]. It is easily seen that the set of squarefree words over at least three letters is not context-free. The same question concerning its complement i.e. the set of words containing squares, was stated as conjecture in [3] and was open for a long time. Two different proofs were given, the first (Ehrenfeucht and Rozenberg [26]) based on growth consideration for EOL systems, the second (Ross and Winklmann [53]) contains some preliminary version of the interchange lemma. The proof given here is from [50].

2.9. THEOREM. *The language of words containing a square over an alphabet with at least three letters is not context-free.*

Denote by $\mathcal{C}_n$ the language of words over n letters containing a square. The proof is in two steps. Only the first one is sketched below. The first step is to show that $\mathcal{C}_6$ is not context-frere. Then a so-called square-free morphism (i.e. a morphism preserving square-free words) is applied to show that $\mathcal{C}_3$ also is not context-free (For more details about square-free morphisms, see [8, 18, 22]).

PROOF OF STEP 1 OF THEOREM 2.9. Let us consider the 6 letter alphabet $A = \{\$, 0, 1, a, b, c\}$ and assume that $\mathcal{C}_6 \subset A^*$ is context-free. We choose a large enough integer N and fix a squarefree word over $B = \{a, b, c\}$ of length N, say $v = c_1 c_2 \cdots c_N$ with $c_i \in B$. Next we consider the set

$$Q = \{\$t\$t \mid t = d_0 c_1 d_1 c_2 \cdots c_N d_N,\ d_i \in \{0, 1\}\}\,.$$

Any word in Q is a square and contains no other square. Each word in Q has $4(N+1)$ letters, and $\mathrm{Card}(Q) = 2^{N+1}$. Choose $n = 4(N+1)$ and $m = n/2 = 2(N+1)$ in the interchange lemma. Then there exists an interchange set $R \subset Q$ of size

$$\mathrm{Card}(R) \geq \frac{\mathrm{Card}(Q)}{Cn^2} = \frac{2^{N+1}}{16C(N+1)^2} \tag{3}$$

for some constant C. Moreover, the span q of R satisfies $(N+1) < q \leq 2(N+1)$.

Using the notation of the beginning of the paragraph, consider words uxv, $u'x'v'$ in R, such that $ux'v$ is in $\mathcal{C}_6$. Since $|u| = |u'|$, $|x| = |x'| = q$, the words x et x' have the same letters in B at the same places. But since v is square-free, uxv and $ux'v$ both contain a square only if $x = x'$. This shows that if, uxv, $u'x'v' \in R$ interchange, then $x = x'$. Consequently, the number of words in R is bounded by

$$\mathrm{Card}(R) \leq 2^{N+1-(1+q/2)} \leq 2^{(N+1)/2}$$

since $|x| = q \geq N+1$. From Equation (3), it follows that

$$2^{N+1} 16C(N+1)^2 \leq \mathrm{Card}(R) \leq 2^{(N+1)/2}$$

which is impossible for large enough N. □

There have been several developments around the topic of context-freeness related to square-free words. In [3], a conjecture claims that any context-free language containing the langage $\mathcal{C}_n$ for $n \geq 3$ must be the complement of a finite langage. This conjecture has been disproved by Main [45]. He shows that the set Co-Pref($\mathbf{x}$) is context-free where $\mathbf{x}$ is an infinite word which is square-free (and even overlap-free). An analogue of Proposition 2.5 for words with overlaps (an overlap is a word of the form $uxuxu$ with u nonempty) has been given by Gabarró [31]. For other developments along these lines, see [44].

2.3 Degeneracy

In Section 2.1, we have defined an iterative pair of a word w in a language L as a 5-uple (x, u, y, v, z) such that

(i) $w = xuyvz$;
(ii) $xu^n yv^n z \in L$ for all integers $n \geq 0$.

Such a pair is called *degenerated* if $xu^* yv^* z \subset L$ (for more details, see Section 3.1). The aim of this section is to prove the

2.10. THEOREM. *If all the iterative pairs in a given context-free language are degenerated, then the language is regular.*

It should be observed that this result does not characterize regular sets. For instance, the language $R = \{a^n b^p \mid n, p \geq 0,\ \ n \equiv p \bmod 2\}$ is regular. It has the following pair $(a, a, 1, b, b)$ which is not degenerated because $aa^2bb \notin R$. On the other hand, there do exist nonregular languages having all their iterative pairs degenerated. (Obviously, they cannot be context-free!) Such an example is

$$\{a^n b^n c^n \mid n \geq 1\} \cup a^+ \cup b^+ \cup c^+ .$$

The following definitions and proofs are from [27]. A *type* over A is a word y such that $|y|_a \leq 1$ for any $a \in A$. Note that the number of different types is finite.

A word x over the alphabet A has type y iff there exists a morphism h such that $h(y) = x$ and such that $h(a) \in aA^*a \cup a$ for all letters $a \in A$.

2.11. EXAMPLE. Let $A = \{a, b, c\}$. The word $x = acabc$ is of type abc; just set $h(a) = aca$, $h(b) = b$, $h(c) = c$. It is also of type ac with $h'(a) = a$, $h'(b) = b$, $h'(c) = cabc$.

As seen in the example, a word may have several types. On the other hand, we have the following

2.12. LEMMA. *Every word x over A has a type.*

PROOF. The proof is by induction on the number k of different letters in x. If $k = 0$ or $k = 1$, the result is obvious. Assume the result holds for some k and let x be a word using $k + 1$ letters.

Let a be the first letter of x and let x_1 be the longest prefix of x ending with a. Then $x = x_1 x_2$. Since x_2 uses at most k letters, by induction it has a type y and, clearly, ay is a type for x. □

2.11. EXAMPLE (*continued*). Applied to $x = acabc$, this method gives raise to a factorization $x = aca \cdot bc$ and to the morphism $h(a) = aca$, $h(b) = b$, $h(c) = c$. This computation thus gives the type abc.

Given two words x and x' of some type y, we define the *interleaving* x'' of x and x' (according to the associated morphisms h and h') as follows. Let $y = a_1 a_2 \cdots a_k$, $x = x_1 x_2 \cdots x_k$, $x' = x'_1 x'_2 \cdots x'_k$ with $h(a_i) = x_i = \bar{x}_i a_i$ and $h'(a_i) = x'_i$; then, the interleaving is $x'' = \bar{x}_1 x'_1 \bar{x}_2 x'_2 \cdots \bar{x}_k x'_k$.

2.13. EXAMPLE. Let $x = acabc$ be of type abc with $h(a) = aca$, $h(b) = b$, $h(c) = c$. Let $x' = ababcbcbc$ be of type abc with $h'(a) = aba$, $h'(b) = bcbcb$, $h'(c) = c$. Then, their interleaving x'' is equal to $acaba \cdot bcbcb \cdot c$. If we were to change h' in $h''(a) = a$, $h''(b) = babcbcb$, $h''(c) = c$, the (new) interleaving would be $aca \cdot babcbcb \cdot c$

We now turn to some constructs on derivation trees for a given grammar G in Chomsky normal form generating a language $L(G)$ having all its iterative pairs degenerated.

Pruned Trees : Given a derivation tree, a fixed path in this tree and two nodes of the path having the same label, we define the *left directly pruned* tree as the tree obtained by erasing all subtrees to the left of the path between the two choosen nodes including the first one. Clearly, in general, the new tree obtained is no more a derivation tree. However, if L has all its pairs degenerated, the produced word still is in L. We define similarly the *right directly pruned* tree (see Fig. 2).

We now extend this construct to several pairs of nodes on the fixed path. To do so, we choose k pairs of nodes $(r_1, s_1) \cdots (r_k, s_k)$ such that

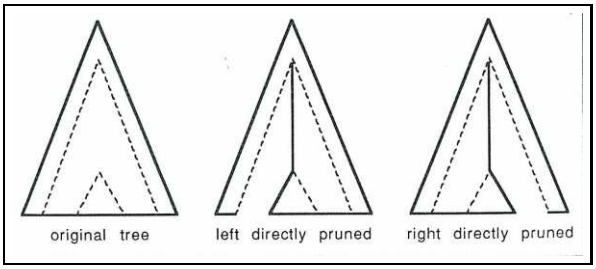

Figure 2: Pruning a tree

(i) for each i, r_i and s_i have the same label,
(ii) for each i, s_i is a descendant of r_i,
(iii) for each i, $r_{i+1} = s_i$ or r_{i+1} is a descendant of s_i.

To each pair (r_i, s_i), we associate an *index* L or R. Then, the corresponding pruned tree is, by definition the tree obtained by successively left directly or right directly pruning the original tree on the pair (r_i, s_i) according to the fact that the index is L or R. Again, the resulting tree may not be a derivation tree. However, as soon as L has all its pairs degenerated, the produced word will belong to the language.

Relabeled Trees : We define a new set of labels for the nodes, by

$$\begin{aligned}\bar{V} =&\{(X, Y, Z, i) \mid X, Y, Z \in (V - A),\ i \in \{0, 1\}\} \\ &\cup \{(X, a) \mid x \in (V, A),\ a \in A\}\,.\end{aligned}$$

Given a derivation tree and a fixed path in it, we relabel the nodes of the path by elements of $\bar{V}$ according to the following rule :

(i) if the node r is labeled by a variable X which is derived in YZ in the tree, the new label of r will be (X, Y, Z, i) with $i = 0$ if Y is on the path and $i = 1$ if Z is on the path.

(ii) if the node r is labeled by the variable X which is derived in $a \in A$ in the tree, the new label of r will be (X, a).

The word of new labels collected on the path from the root to the leaf is called the *spine* of the marked tree.

Further Definitions : First, let μ be the function from $V^+ \times \bar{V}^+$ into the subsets of V^* defined as follows : $\mu(\alpha, \bar{\beta})$ is the set of words γ such that $\alpha\gamma$ is generated in the grammar G by a derivation tree where the path from the root to the last letter of α gives raise to a marked tree with spine $\bar{\beta}$. Now, let δ be the function from V^+ into the subsets of $\bar{V}^+$ defined by

$$\delta(\alpha) = \{\bar{\beta} \in \bar{V}^+ \mid \mu(\alpha, \bar{\beta}) \neq \emptyset\}\,.$$

Finally, for each non empty left factor α of a word in $L(G)$, we define $\Theta(\alpha) = \{\bar{\beta}_0 \in \bar{V}^+ \mid \bar{\beta}_0$ is a type of some $\bar{\beta}$ in $\delta(\alpha)\}$. Clearly, for each α, $\Theta(\alpha)$ is a finite set and the number of such possible sets is finite. We are now ready to prove the crucial lemma needed for the theorem:

2.14. LEMMA. *Let α and α' be two nonempty left factors of $L(G)$ and assume $\Theta(\alpha) = \Theta(\alpha')$; then*

$$\{\gamma \mid \alpha\gamma \in L(G)\} = \{\gamma \mid \alpha'\gamma \in L(G)\}\,.$$

PROOF. Clearly, it suffices to prove

$$\Theta(\alpha) = \Theta(\alpha') \text{ and } \alpha\gamma \in L(G) \Longrightarrow \alpha\gamma' \in L(G)\,. \tag{4}$$

Sketch of the proof of (4). As $\alpha\gamma$ is in $L(G)$, there exists a derivation tree T producing $\alpha\gamma$. In T, choose the path from the root to the last letter of α and build the marked version of T giving raise to its spine $\bar{\beta}$. If $\bar{\beta}$ has type $\bar{\beta}_0$, then $\bar{\beta}_0$ is in $\Theta(\alpha')$ and there exist a word $\bar{\beta}'$ of type $\bar{\beta}_0$ such that $\bar{\beta}' \in \delta(\alpha')$. Hence, we can build a derivation tree T' which has a marked version with spine $\bar{\beta}'$ producing a word $\alpha'\gamma'$ in $L(G)$. So, now we have two derivation trees T and T' producing $\alpha\gamma$ and $\alpha'\gamma'$ respectively, with in each a selected path whose marked versions are of same type $\bar{\beta}_0$. The idea is then to produce a new tree T'' looking like an interleaving of T and T' along the selected paths. Namely, let $\bar{\beta}_0 = \bar{v}_1\bar{v}_2\cdots\bar{v}_k$ and $h(\bar{\beta}_0) = \bar{\beta}$, $h'(\bar{\beta}_0) = \bar{\beta}'$. Let $\bar{\beta}''$ be the interleaving of $\bar{\beta}$ and $\bar{\beta}'$ according to h and h'.

We shall now build the tree T'' by completing this path into a derivation tree. At the same time that we indicate how to build up T'', we indicate on the path $\bar{\beta}''$ some pairs of nodes with an index L or R. This is done is such a way that when we prune T'' according to these pairs of nodes, we get a new tree $\hat{T}''$ producing $\alpha\gamma'$. Then, because the language has all its pairs degenerated, $\hat{T}''$ will produce a word in the language and (4) will be proved. We now describe the construction of T'' together with the

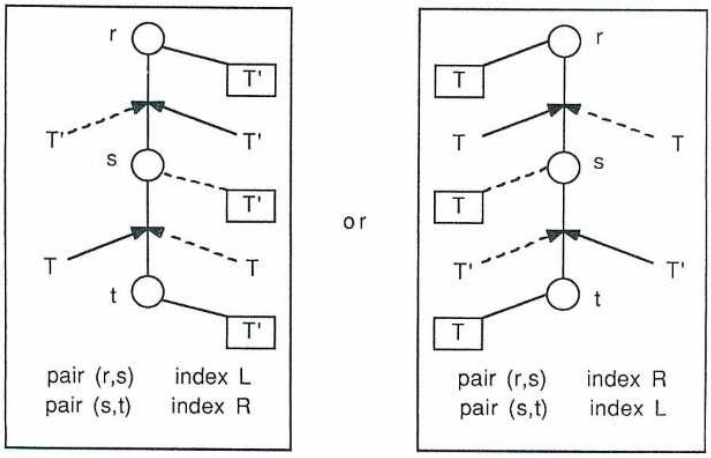

Figure 3: *Case 1* : $h(\bar{v}_i) \neq \bar{v}_i \neq h'(\bar{v}_i)$, $\bar{\beta}'' = \bar{v}_i\bar{\beta}\bar{v}_i\bar{\beta}'\bar{v}_i$

pairs of nodes to be pruned. For this, we go through $\bar{\beta}''$ by segments, each of which is the contribution of a letter $\bar{v}_i$ of $\bar{\beta}_0$. We shall picture what to do by indicating which subtrees (from T or T') have to be added to the right and left of the path $\bar{\beta}''$. The dotted arrows will show what is dropped out after the pruning. The reader will check that, in each case, the pruned tree obtained $\hat{T}''$ produces a part of γ' on the right of the path and a part of α on the left of it. Hence $\hat{T}''$ produces $\alpha\gamma'$ as already announced.

There are four cases according to the fact that the images through h and h' of a letter $\bar{v}_i$ is equal to $\bar{v}_i$ or not. In each case, we have two symmetric situations, according to the fact that the selected path leaves the node through its left or right son (see Figures 3–6). □

Now, we can prove the announced result.

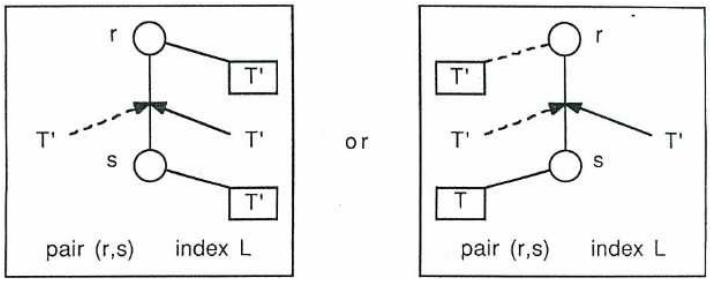

Figure 4: *Case 2* : $h(\bar{v}_i) = \bar{v}_i \neq h'(\bar{v}_i)$, $\bar{\beta}'' = \bar{v}_i\bar{\beta}'\bar{v}_i$

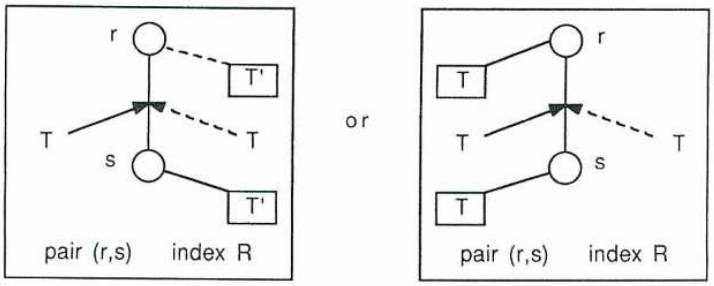

Figure 5: *Case 3* : $h(\bar{v}_i) \neq \bar{v}_i = h'(\bar{v}_i)$, $\bar{\beta}'' = \bar{v}_i\bar{\beta}\bar{v}_i$

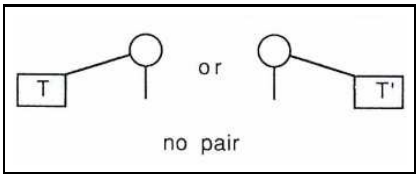

Figure 6: *Case 4* : $h(\bar{v}_i) = \bar{v}_i = h'(\bar{v}_i)$, $\bar{\beta}'' = \bar{v}_i$

PROOF OF THE THEOREM. By the lemma, the nonempty left factors of L fall into a finite number of classes modulo L : there are at most as many classes as the number of different sets $\Theta(\alpha)$. On the other hand, there is at most one new class containing the empty word. As all the words which are not left factors of L fall in one specific class, the equivalence modulo L has at most 2 more classes than the number of possible sets $\Theta(\alpha)$. Hence, this equivalence is of finite index which shows that L is regular. □

3 Looking for nongenerators

3.1 Preliminaries

A transduction from A^* to B^* is any function from A^* to the set of subsets of B^*. Such a mapping is a *rational transduction* iff its graph is rational. Many characterizations of rational transductions have been given (see [12]). We will here use the following :

3.1. THEOREM. *A transduction τ from A^* to B^* is rational iff there exist an alphabet C, a regular set R over C and two alphabetical morphisms g and h from C^* to A^* and*

B^* respectively, such that for all $x \in A^*$,

$$\tau(x) = h(g^{-1}(x) \cap R)\,.$$

This is known as Nivat's theorem. (A morphism is alphabetical or length-decreasing if it maps a letter into either a letter, or the empty word.) As usual, we will use transductions on languages : the image of a language L is the union of the images of each word in L. From Theorem 3.1, it is clear that any context-free language will have a context-free image through a rational transduction. One of the basic properties of rational transductions is the:

3.2. THEOREM. *Given two rational transductions τ_1: $A^* \to B^*$ and τ_2: $B^* \to C^*$, their composition $\tau_2 \circ \tau_1$ is a rational transduction from A^* to C^*.*

Usually, this is stated as "rational transductions are closed under composition" and is known as Elgot and Mezei's theorem. Now, given two languages L_1 and L_2, it may happen that there exists a rational transduction τ such that $L_2 = \tau(L_1)$. Then we say that L_1 *dominates* L_2 and denote this by $L_1 \geq L_2$. Note that because of Theorem 3.2, this relation is transitive. If it happens then that $L_1 \geq L_2$ and $L_2 \geq L_1$, we say that L_1 and L_2 are *rationally equivalent*, in symbols $L_1 \approx L_2$. Again, because of Theorem 1.2, this relation is transitive. Hence, it is an equivalence relation. Two languages are *incomparable* if no relation holds between them.

3.3. EXAMPLE. The Dyck languages D_2^* and $\hat{D}_2^*$ are equivalent. The Dyck language D_2 dominates D_1^*. These languages are not equivalent : D_1^* does not dominate D_2^*. The languages D_1^* and $\hat{D}_1^*$ are incomparable.

The main motivation in using rational transductions for comparing context-free languages comes from the idea that if $L_1 \geq L_2$, then L_1 should be at least as "complicated" as L_2. This idea is more or less formalized in the framework of iterative pairs.

In Section 2.1, an *iterative pair* of a word w in a language L is defined as a tuple (x, u, y, v, z) such that $w = xuyvz$ and $xu^nyv^nz \in L$ for all $n \geq 0$. The classical pumping lemma for context-free languages (see Section 2.1) ensures that if L is context-free, any long enough word in L will have such a pair. In general the set of possible exponents

$$\{(n, m) \in \mathbb{N} \times \mathbb{N} \mid xu^nyv^mz \in L\}$$

contains, by definition, the diagonal $\{(n, n) \mid n \geq 0\}$. If it reduces to that set, the iterative pair is called *strict* ; if on the contrary the set of exponents is the whole set $\mathbb{N} \times \mathbb{N}$, the iterative pair is *degenerated*. Arbritrary intermediate situations may arise.

3.4. EXAMPLE. (i) Let D^* be the Dyck language over $A = \{a, \bar{a}\}$. Then $x = aa\bar{a}a\bar{a}\bar{a}$ admits the following pair $\pi = (a, a, \bar{a}a\bar{a}, \bar{a}, 1)$. It is a strict iterative pair.

(ii) Let $S_< = \{a^nb^m \mid 1 \leq m \leq n\}$. The word $x = aaabb$ admits $\pi = (a, a, ab, b, 1)$ as an iterative pair. Note that π is neither strict nor degenerated. On the other hand $(a, aa, 1, bb, 1)$ is *not* a pair because exponent 0 yields $a \notin S_<$. We leave to reader to check that $S_<$ has no strict iterative pair.

3.5. THEOREM. *Let L_1 and L_2 be two context-free languages such that $L_1 \geq L_2$; if L_2 has a strict iterative pair then so does L_1.*

This result shows that strict pairs cannot be created by rational transductions. This is extensively used to show that $L_1 \geq L_2$ does not hold. For instance, going back to the above example, Theorem 3.5 shows that $S_<$ does not dominate D^*.

Theorem 3.5 has been considerably strengthened by the consideration of systems of pairs. A system consists in several iterative pairs in the same word, each of which can be pumped up independently. We will not give here any details on this interesting extension (see [12]).

The notion of rational transduction naturally leads to consider families of languages closed under this operation. Such a family is called a *rational cone*. We have already remarked that the image of a context-free language through any rational transduction is context-free, so we can state : the family Alg of context-free languages is a rational cone. The same holds, for instance, for Rat and Lin, which are respectively the families of regular and linear languages. If we look at some family $\mathcal{L}$ (such as the deterministic languages) which fails to be a rational cone, we can define the least rational cone containing $\mathcal{L}$. It is denoted by $\mathcal{T}(\mathcal{L})$ and called the rational cone generated by $\mathcal{L}$. If $\mathcal{L}$ is the family of deterministic languages, then $\mathcal{T}(\mathcal{L})$ will be Alg again.

On the other hand, given a rational cone, $\mathcal{L}$ we may look for the smallest possible set of languages $\mathcal{G}$ such that $\mathcal{L} = \mathcal{T}(\mathcal{G})$. Whenever $\mathcal{G}$ can be chosen to be one single language G, the cone $\mathcal{L}$ is said to be a *principal rational cone* with generator G; it is denoted $\mathcal{T}(G)$. From the Chomsky-Schützenberger theorem, we derive that the cone Alg is principal and that D_2^* is a generator. Note then that, as two equivalent languages generate the same cone, the following languages are generators of Alg as well : D_2, $\hat{D}_2^*$, D_n^*, D_n, $\hat{D}_n^*$, $\hat{D}_n$ $(n \geq 2)$, E, E_n. So, for instance, we may write $\mathsf{Alg} = \mathcal{T}(E)$. The family of linear languages turns out to be a principal cone too. The languages Pal or Sym are generators of Lin. The family Rat is principal too. Any nonempty regular language is a generator of it. Moreover Rat is included in any rational cone. Besides all this, as we have mentioned that D_1^* does not dominate D_2^*, we know that $\mathcal{T}(D_1^*)$ is a strict subfamily of Alg. It is the family Rocl of restricted one counter languages, those languages which can be recognized by a pushdown automaton with one symbol in the pushdown alphabet and no zero test (see [12]). Now the question of building up some nonprincipal cones is raised. If we do not accept "trivial" constructions such as the union of two principal cones with incomparable generators, we get that any set of generators of a nonprincipal cone must be infinite. To prove that such cones exist and to build them, the easiest method is to use substitution.

Given a language L over the alphabet A and, for each letter a of A a language L_a, we define the substitution σ as the morphism from A^* into its subsets given by $\sigma(a) = L_a$ for each $a \in A$. If all the languages L_a are in some family $\mathcal{L}$, then σ is an $\mathcal{L}$-substitution. So, a rational substitution is a substitution such that each L_a is regular. (Note that in this case, a substitution is a rational transduction). Given two families of languages $\mathcal{L}$ and $\mathcal{M}$, we can define the substitution of $\mathcal{M}$ in $\mathcal{L}$ by $\mathcal{L} \square \mathcal{M} = \{\sigma(L) \mid L \in \mathcal{L}, \sigma \text{ is an } \mathcal{M}\text{-substitution } \}$. Again, substitution and rational transductions nicely mix together.

3.6. THEOREM. *If $\mathcal{L}$ and $\mathcal{M}$ are two rational cones, then so is $\mathcal{L} \square \mathcal{M}$. Moreover, if $\mathcal{L}$ and $\mathcal{M}$ are principal, so is $\mathcal{L} \square \mathcal{M}$.*

The proof of Theorem 3.6 uses a very special substitution called the *syntactic substitution* : given a language L over A and a language M over B with $A \cap B = \emptyset$, the syntactic substitution of M in L, denoted by $L \uparrow M$, is the substitution defined by $\sigma(a) = aM$ for $(a \in A)$. This special substitution gives raise to a crucial lemma :

3.7. LEMMA ("Syntactic lemma"). *Given two languages L and M and two rational cones $\mathcal{L}$ and $\mathcal{M}$, then $L \uparrow M \in \mathcal{L} \,\square\, \mathcal{M}$ implies either $L \in \mathcal{L}$ or $M \in \mathcal{M}$.*

EXAMPLE (*application*). We want to prove that if $\mathsf{Alg} = \mathcal{L} \,\square\, \mathcal{M}$, then either $\mathcal{L}$ or $\mathcal{M}$ is equal to Alg. For this we use the syntactic lemma with two copies of E :

$$E \uparrow E' \in \mathcal{L} \,\square\, \mathcal{M} \;\Rightarrow\; E \in \mathcal{L} \text{ or } E' \in \mathcal{M}$$

and the result is proved.

Another application allows us to build nonprincipal cones.

3.8. LEMMA. *Given a cone $\mathcal{L}$, either $\mathcal{L} \,\square\, \mathcal{L} = \mathcal{L}$ or the smallest substitution closed rational cone $\mathcal{T}^\sigma(\mathcal{L})$ containing $\mathcal{L}$ is not principal.*

Take now $\mathcal{L} = \mathsf{Lin}$. As $\mathsf{Lin} \,\square\, \mathsf{Lin}$ is not included in Lin, the cone $\mathcal{T}^\sigma(\mathsf{Lin}) = \mathcal{T}^\sigma(Sym)$ is not principal. This cone is the family of quasi-rational languages (or of non-expansive languages) and is denoted by Qrt. The same conclusion holds with $\mathcal{L} = \mathsf{Rocl}$ giving raise to the nonprincipal cone Ict of iterated-counter languages. We may even take $\mathcal{L} = \mathsf{Lin} \cup \mathsf{Rocl}$ and get a nonprincipal cone of socalled Greibach languages, denoted by Gre. For each of these cones, we get a generating family using the syntactic substitution. For instance, Qrt will be generated by $\{Sym, Sym \uparrow Sym, \ldots, Sym \uparrow (Sym \cdots \uparrow Sym), \ldots\}$. Up to now, we have used two methods to get rational cones. We choose a family $\mathcal{L}$ and look either at the cone $\mathcal{T}(\mathcal{L})$ it generates, or at the substitution closed cone $\mathcal{T}^\sigma(\mathcal{L})$ it generates. However, there exist other methods. We shall present here two new ways of getting cones, each of them raising more questions than it anwers !

Given a principal cone $\mathcal{L}$, we may distinguish the family of those languages which are generators of $\mathcal{L}$ as well as the family of those languages in $\mathcal{L}$ which are not. It is easy to see that this second family is a rational cone $\mathcal{N}\mathcal{L}$ which is the largest subcone of $\mathcal{L}$. In the particular case of $\mathcal{L} = \mathsf{Alg}$, this largest subcone is rather denoted by Nge and one of the most popular conjecture in this framework is

CONJECTURE. Nge *is a nonprincipal rational cone.*

Several attempts to prove this conjecture have been made. A great number of them tried to use the substitution operation to build up Nge in a similar way than we built Qrt. We now know that such an attempt cannot succeed (see Section 3.3). Besides, in the sequel, we will show that Nge differs from all the already presented cones in that sense that there is no family of deterministic languages which can generate Nge (see Section 4). It should be noted that for any principal rational cone $\mathcal{L}$ (excepted the cone Rat) the question of the principality of $\mathcal{N}\mathcal{L}$ is open ($\mathcal{N}\mathsf{Rat} = \{\emptyset\}$). Along the same lines, we can mention that nobody knows if there exists or not a rational cone $\mathcal{L}$ which contains only Rat as a strict subcone. Such a cone has to be principal. So we have

QUESTION. Does there exist a principal rational cone $\mathcal{L}$ such that $\mathcal{N}\mathcal{L} = \mathsf{Rat}$?

A second method for getting new rational cones comes from the following obvious observation : given two rational cones $\mathcal{L}$ and $\mathcal{M}$, their intersection $\mathcal{L} \cap \mathcal{M}$ is a rational cone. Here again we do not know much about such cones and we mainly have open questions rather than results. For instance, if we take $\mathcal{L} = \mathsf{Lin}$ and $\mathcal{M} = \mathsf{Rocl}$, we only have that $\mathsf{Lin} \cap \mathsf{Rocl} \supseteq \mathcal{T}(S)$, where $S = \{a^n b^n \mid n \geq 0\}$. This inclusion has recently been proved to be strict by Brandenburg [19]. However, we still do not know whether

this intersection is a principal cone. In Section 3.5, we present some results in this area showing that such intersections seem to be, in general, larger than was thought at the beginning (Note that here again some attemps have been made to describe Nge as the intersection of two rational cones (see [36])).

3.2 Generators

We state here one of the most important characterizations of generators of the cone of context-free languages. The proofs of the following results are very technical and will not be given there. They can be found in [10]. Over the fixed alphabet $A = \{\#_1, \#_2, a, b, c, d\}$, we define the language E' by $E' = \#_1 E \#_2$. Then we have

3.9. THEOREM. *A language L over B is a generator of the cone of context-free languages iff there exist a morphism h from A^* to B^* and two regular sets R over A and K over B such that*

(i) $h(E') = L \cap K$
(ii) $h^{-1}(L) \cap R = E'$
(iii) $|h^{-1}(w) \cap R| = 1$ *for all words* $w \in L$.

This is known as Beauquier's theorem. From this result we get

3.10. COROLLARY. *For any generator L, there exists a regular set K such that $L \cap K$ is an unambiguous generator.*

Theorem 3.9 can be stated in a slightly different way :

3.11. THEOREM. *A language $L \subset B^*$ is a generator iff there exist six words x, y, α, β, γ, $\delta \in B^*$ and a regular set $K \subset B^*$ such that*

$$L \cap K = \langle S \to xTy,\ T \to \alpha T \beta T \gamma + \delta \rangle$$

A recent improvement shows that $\alpha, \beta, \gamma, \delta$ can be chosen to be a (biprefix) code [11].

Essentially, these results show that, in any generator, there is an encoded version of the language E. We now present an application.

3.12. PROPOSITION. *The cone* Alg *of context-free languages has no commutative generator.*

PROOF. Let L be a commutative language over B. Suppose that L is a generator. Then Theorem 3.9 holds. Set $h(\#_1) = x$, $h(\#_2) = y$, $h(a) = \alpha$, $h(b) = \beta$, $h(c) = \gamma$, $h(d) = \delta$.

Since $u = \#_1 a a^n d(bdc)^n b a^n d(bdc)^n c \#_2$ is in E' for all integers n, we can choose n large enough to find in each block of a's an iterative factor a^λ in the regular set R. Then $z = \#_1 a a^{n+\lambda} d(bdc)^n b a^{n-\lambda} d(bdc)^n c \#_2$ is in R. Moreover, it has the same commutative image than u. Hence $h(z)$ is in L and $h^{-1}(h(z))) \in h^{-1}(L) \cap R$. Now z is in $h^{-1}(h(z))$ and by (ii) of Theorem 3.9 it should also be in E' which is not true. □

COMMENT. Corollary 3.10 stated above leads naturally to the following notion : a language L is *strongly ambiguous* if for any regular set K such that $L \cap K$ and L are equivalent, $L \cap K$ is ambiguous. So, we know that no generator of Alg is strongly ambiguous. We can even extend this notion as follows : a language L is *intrinsically*

ambiguous if there is no language equivalent to L which is unambiguous. Such languages do exist. For instance, the language

$$\{a^n ba^m ba^p ba^q \mid (n \geq q \text{ and } m \geq p) \text{ or } (n \geq m \text{ and } p \geq q)\}$$

is intrinsically ambiguous (see [9]). We leave to the reader to check that, generally, the classical examples of ambiguous languages are not intrinsically ambiguous (not even strongly ambiguous).

3.3 Nongenerators and substitution

The aim of this section is to prove the following theorem.

3.13. THEOREM. *The cone* Nge *of nongenerators of context-free languages is not the substitution closure of any strict subcone.*

This result puts an end to any attempt of proving that Nge is nonprincipal by showing that this cone is the substitution closure of simpler subcones. It implies for instance that the family of Greibach languages is strictly included in Nge. Theorem 3.13 will follow from the following more general result :

3.14. THEOREM. *For any given context-free language L, there exists a context-free language $L^{\uparrow}$ such that*

(i) *$L^{\uparrow}$ is a nongenerator if L is so;*
(ii) *the rational cone generated by $L^{\uparrow}$ contains the substitution closure of the cone generated by L, i.e. $\mathcal{T}(L^{\uparrow}) \supset \mathcal{T}^{\sigma}(L)$.*

Theorem 3.14 has other consequences, such as the following corollary which answers Conjecture 8 in [3] :

3.15. COROLLARY. *There does exist a principal rational cone of nongenerators containing the family* Qrt *of non-expansive languages.*

The proofs given here come from [15]. Let us turn first to the proof of Theorem 3.14. We start by some definitions necessary to construct the language $L^{\uparrow}$ associated to L.

Given a word x in D over the alphabet $A = \{a, \bar{a}\}$, we define the *height* of an occurrence (x_1, α, x_2) in x (with $x = x_1 \alpha x_2$, $\alpha \in A$) by $|x_1\alpha|_a - |x_1\alpha|_{\bar{a}}$. Then, the height of the word x is the maximum of all heights over the occurrences in x. It is easy to check that we can compute the heights of the occurrences from left to right in a sequential manner.

Namely, if the current height is k, add 1 if you read a letter a and substract 1 if you read a letter $\bar{a}$. On the other hand, the set of words in D of height at most k is a regular set. Hence, there exists a gsm-mapping which, when reading a word x in D of height at most k, produces the word y obtained by indicing the letters by their heights. It will be denoted by num_k.

3.16. EXAMPLE. Consider the word $x = aa\bar{a}aa\bar{a}\bar{a}aa\bar{a}\bar{a}$. The height of the third a is 2. The height of x is 3. Then $num_2(x) = \emptyset$ and

$$num_3(x) = a_1 a_2 \bar{a}_1 a_2 a_3 \bar{a}_2 \bar{a}_1 a_2 \bar{a}_1 \bar{a}_0 \,.$$

Note that in $num_k(x)$, a letter a_i matches a letter $\bar{a}_{i-1}$.

Given the alphabet $C = A \cup B$, with $B \cap A = \emptyset$, we consider the projection p from C^* into A^*. A word x over C is called a *D-word* if $p(x)$ is in D and x begins and ends with a letter in A. The height of an occurrence in such a D-word is then defined in the same way than for a word in D. Again, we define the gsm-mapping num_k producing from a D-word of height at most k, the word y obtained by indicing the letters of x by their height.

3.17. EXAMPLE. Let $B = \{b, c\}$. The word $x = aabc\bar{a}baba bc\bar{a}bc\bar{a}c\bar{a}cabc\bar{a}\bar{a}$ is a D-word; further, $num_2(x) = \emptyset$ and

$$num_3(x) = a_1a_2b_2c_2\bar{a}_1b_1a_2b_2a_3b_3c_3\bar{a}_2b_2c_2\bar{a}_1c_1a_2b_2c_2\bar{a}_1\bar{a}_0\,.$$

Given a context-free language L over B, we consider first the marked version of L which is the language $M = aL\bar{a} \cup \{1\}$ over $C = A \cup B$. We then denote by M_i the copy of M obtained by indicing all the letters of L by i, the letter a by i and the letter $\bar{a}$ by $i-1$. Thus, in M_i the letter a_i will match $\bar{a}_{i-1}$. We use these copies to define

$$M_{(1)} = M_1, M_{(2)} = M_1 \Uparrow M_2, \ldots,$$
$$M_{(k)} = M_1 \Uparrow (M_2(\cdots \Uparrow M_k \cdots)), \ldots,$$

where $\Uparrow$ stands for a substitution very similar to the syntactic substitution. The substitution $\Uparrow$ is defined as follows : the image of any b_i is b_iM_{i+1} when b is in B, the image a_i is a_iM_{i+1} and the image of $\bar{a}_{i-1}$ is just $\bar{a}_{i-1}$. Clearly, $\Uparrow$ is so near from the usual syntactic substitution that the family $\{M_{(1)}, M_{(2)}, \ldots, M_{(k)}, \ldots\}$ generates the substitution closed rational cone $\mathcal{T}^\sigma(M)$ generated by M, which is the same than $\mathcal{T}^\sigma(L)$. Using the morphism h from $(\cup_{i\geq 1}C_i)^*$ onto C^* which just erases the indices, we define $M^{(k)} = h(M_{(k)})$ and $M^{(\infty)} = \cup_{k\geq 1}M^{(k)}$. (For instance, if $L = \{bc\}$, the word x of Example 3.2 will be in $M^{(3)}$.) We then get our first proposition.

3.18. PROPOSITION. *The language $M^{(\infty)}$ is context-free.*

To prove the proposition, note first the

OBSERVATION. *$x \in M^{(\infty)}$ if and only if $x \in M = M^{(1)}$ or*

$$x = ay_1az_1\bar{a}y_2az_2\bar{a}\cdots y_{n-1}az_{n-1}\bar{a}y_n\bar{a}$$

with y_1, $y_n \in B^*$, $y_i \in B^+$ for $2 \leq i \leq n-1$, $az_i\bar{a} \in M^{(\infty)}$ and $ay_1y_2\cdots y_{n-1}y_n\bar{a} \in M$.

This observation is proved by a straightforward induction on the integer k for which $x \in M^{(k)}$.

PROOF OF PROPOSITION 3.18. Using the observation, we get that $M^{(\infty)}$ can be generated by the following generalized context-free grammar which has infinitely many productions : $\langle S \rightarrow ay_1Sy_2S\cdots y_{n-1}Sy_n\bar{a}+1 \mid ay_1y_2\cdots y_{n-1}y_n\bar{a} \in M\rangle$. Clearly, the set of right members is context-free and it is well known that such a generalized grammar does generate a context-free language. □

Still using the above observation, it is easy to prove that any word x in $M^{(\infty)}$ is a D-word over C. Moreover, if x is of height at most k, then $num_k(x)$ is in $M_{(k)}$. So, we get $num_k(M^{(\infty)}) = num_k(M^{(k)}) = M_{(k)} - \{1\}$.

Now define $M^{(+)}$ to be the set of those words over C which are not D-words, we note that $M^{(+)}$ is context-free. Set $L^\uparrow = M^{(+)} \cup M^{(\infty)}$. Clearly $L^\uparrow$ is context-free. Moreover, since num_k equals $\emptyset$ on $M^{(+)}$, we get $num_k(L^\uparrow) = num_k(M^{(\infty)}) = M_{(k)} - \{1\}$. This shows that the cone of $L^\uparrow$ does contain all the $M_{(k)}$. So, we can state the following proposition.

3.19. PROPOSITION. *$L^{\uparrow}$ is a context-free language such that $\mathcal{T}(L^{\uparrow}) \supset \mathcal{T}^{\sigma}(L)$.*

Note that this proposition is the condition (ii) of Theorem 3.14. So, we now turn to condition (i) and prove this proposition.

3.20. PROPOSITION. *$L^{\uparrow}$ is a generator of the cone* Alg *of context-free languages iff L is.*

Before starting the proof, we introduce a notation. Given a language P over an alphabet disjoint from A, let $\lhd P \rhd$ denote the language

$$\lhd P \rhd = \{a^n y \bar{a}^n \mid n \geq 1,\ y \in P\}.$$

Let us show first the following lemma.

3.21. LEMMA. *Given a language P, if $\lhd P \rhd$ is in $\mathcal{T}(L^{\uparrow})$, then there exists an integer k such that P is in $\mathcal{T}(M^{(k)})$.*

PROOF (*outline*). The detailed proof is rather technical. We just outline it. As $\lhd P \rhd$ is in $\mathcal{T}(L^{\uparrow})$, we may write $\lhd P \rhd = g(f^{-1}(L^{\uparrow}) \cap R)$.

Let k be the number of states of a finite automaton recognizing R; we want to show that all the words $a^n y \bar{a}^n$ with $n > k$ are images of words in $M^{(k)}$. For this, we choose a word z of minimal length in $f^{-1}(L^{\uparrow}) \cap R$ such that $g(z) = a^n y \bar{a}^n$ for some $n > k$. Then z naturally factorizes into $z_1 z_2 z_3$ such that $g(z_2) = y$ and z_2 is maximal with this property. Moreover, as n is larger than k, z_1 can be factorized in $z_1' u z_1''$ such that $z_1' u^* z_1'' z_2 z_3 \subset R$ and $g(u) \in a^+$. So, for all integers $m \neq 1$, $f(z_1' u^m z_1'' z_2 z_3)$ is not in $L^{\uparrow}$ because $g(z_i u^m z_1'' z_2 z_3)$ is not in $\lhd P \rhd$. As z is of minimal length, $f(u)$ is non empty which implies that $f(z)$ is not in $M^{(+)}$ otherwise all its iterated versions with at most one exception would be in $M^{(+)}$, hence in $L^{\uparrow}$. So $f(z)$ is in $M^{(\infty)}$. Assume then that $f(z)$ is not in $M^{(k)}$. In $x = f(z)$, there exist occurrences of letters of height larger than k. This implies that z can be factorized into $z = \hat{z}_1 \hat{z}_2 \hat{z}_3$ such that $\hat{z}_1 \hat{z}_2^* \hat{z}_3 \subset R$ and $p(f(\hat{z}_2)) = a^s$ for some $s \neq 0$. Hence $f(\hat{z}_1 \hat{z}_2^r \hat{z}_3)$ is in $M^{(+)}$ and $g(\hat{z}_1) g(\hat{z}_2)^* g(\hat{z}_3)$ is in $\lhd P \rhd$. As z is of minimal length, $g(\hat{z}_2)$ is not empty. Then it has to be a factor of y. So, we know that $\hat{z}_2$ is a factor of z_2 and we may write $z = z_1' u z_1'' z_2' \hat{z}_2 z_2'' z_3$ with $f(z_1' u z_1'' z_2' \hat{z}_2^2 z_2'' z_3) \in M^{(+)}$. Then, for all m but at most one, we have $f(z_1' u^m z_1'' z_2' \hat{z}_2^2 z_2'' z_3) \in M^{(+)}$. This contradicts the fact that the image of such a word is in $\lhd P \rhd$ only for $m = 1$. □

We are now ready for the following proof.

PROOFOF PROPOSITION 3.20. Clearly, if L is a generator, so is $L^{\uparrow}$. So, only the converse has to be proved. Assume that $L^{\uparrow}$ is a generator. Then, for any context-free language P, $\lhd P \rhd \in \mathcal{T}(L^{\uparrow})$. In particular, $\lhd E \rhd \in \mathcal{T}(L^{\uparrow})$. By lemma 3.7, there exists an integer k such that $E \in \mathcal{T}(M^{(k)})$ and thus $M^{(k)}$ is a generator. As we know that $M^{(k)}$ and $M_{(k)}$ are equivalent, we have that $M_{(k)}$ is a generator. A simple application of the syntactic lemma shows then that L is generator. □

Clearly, Propositions 3.19 and 3.20 prove Theorem 3.14. We now show how we can derive Theorem 3.13.

PROOFOF THEOREM 3.13. It is known that if $\mathcal{L}$ is a rational cone, its closure under union $\mathcal{L}_{\cup}$ satisfies $\mathcal{T}^{\sigma}(\mathcal{L}) = \mathcal{T}^{\sigma}(\mathcal{L}_{\cup})$. On the other hand, $\mathcal{L}_{\cup} = \mathsf{Nge}$ implies $\mathcal{L} = \mathsf{Nge}$. Assume that Nge is the substitution closure of some subcone $\mathcal{L}$. Then, we may assume that $\mathcal{L}$ is closed under union. Let L be any language in Nge. Then, by (i) of Theorem

3.2, $L^{\uparrow}$ is in Nge. It follows that $\lhd L^{\uparrow} \rhd \in \mathsf{Nge} = \mathcal{T}^{\sigma}(\mathcal{L})$. So, there exists a finite number of languages in $\mathcal{L}$, say $L_1, L_2, \ldots, L_k$ such that $\lhd L^{\uparrow} \rhd \in \mathcal{T}^{\sigma}(\{L_1, L_2, \ldots, L_k\})$. If we consider now L_0 to be the union of disjoint copies of $L_1, L_2, \ldots, L_k$, we get $\lhd L^{\uparrow} \rhd \in \mathcal{T}^{\sigma}(L_0)$ with $L_0 \in \mathcal{L}$. As $\mathcal{T}^{\sigma}(L_0)$ is contained in $\mathcal{T}(L_0^{\uparrow})$, we have $\lhd L^{\uparrow} \rhd \in \mathcal{T}(L_0^{\uparrow})$.

By the lemma, there exists an integer k such that $L^{\uparrow} \in \mathcal{T}(L_0^{(k)})$ and by (ii) of Theorem 3.2, $\mathcal{T}^{\sigma}(L)$ is included in $\mathcal{T}(L^{\uparrow})$. Hence $\mathcal{T}^{\sigma}(L) \subset \mathcal{T}(L^{\uparrow}) \subset \mathcal{T}(L_0^{(k)})$. Then, by the syntactic lemma, $\mathcal{T}(L) \subset \mathcal{T}(L_0)$. As L_0 is in $\mathcal{L}$, we have that any language L in Nge is in $\mathcal{L}$, which means $\mathcal{L} \supset \mathsf{Nge}$. Since the reverse inclusion is obvious, we have $\mathcal{L} = \mathsf{Nge}$. □

CONCLUDING REMARKS. It is worthwhile pointing out that if L is deterministic, so is the language $L^{\uparrow}$ constructed here. Thus for instance, there does exist a principal subcone of Alg containing Qrt which has a deterministic generator. However, the proposed construction leaves the following questions open.

QUESTION 1. Does there exist a principal substitution closed cone strictly included in Alg, larger than Rat ?

QUESTION 2. Does there exist a nongenerator L such that $\mathcal{T}(L) = \mathcal{T}(L^{\uparrow})$?

Note that if we can answer Question 2 positively, we will have a positive answer to Question 1. Note also that if Nge is principal, both questions can be answered positively.

3.4 Nongenerators and determinism

The aim of this section is to prove

3.22. THEOREM. *The cone* Nge *is not generated by any family of deterministic context-free languages.*

This result shows that the cone Nge is very different from all the classical cones of context-free languages (principal or not) which all have deterministic generators. The theorem will follow from the existence of a particular language L which has the following two properties :

(i) L is a non-generator;
(ii) any deterministic language which dominates L is a generator.

The proof proposed here is an adaptation of the one given in [16]. We will start by some general results on dpda's before defining the language L. Then, we will show that L satisfies (i) and (ii) and, finally, prove Theorem 3.22.

Given a $dpda$ (deterministic pushdown automaton) $\mathcal{A}$ with input alphabet A, nonterminal alphabet V and set of states Q, recall (see Section 1.1.6) that a *configuration* of $\mathcal{A}$ is a triple $c = (\gamma, q, x)$ in $V^* \times Q \times A^*$. The right most symbol of γ is the top of the pushdown store. The dpda is *normalized* if any ε-move of $\mathcal{A}$ erases the top of the pushdown store. It is well known that for any dpda, there exists an equivalent normalized dpda (see [38]). So, from now on, we will assume that $\mathcal{A}$ is normalized. As usual, a *computation* of $\mathcal{A}$ is a sequence of configuration $c_i = (\gamma_i, q_i, x_i)$ such that $\mathcal{A}$ goes from c_i to c_{i+1} in one move and where $c_0 = (S_0, q_0, x_0)$ is initial (i.e. S_0 is the

initial pushdown and q_0 the initial state). The computation $c_0, c_1, \ldots, c_n$ is *maximal* if $c_n = (\gamma_n, q_n, 1)$ (which means that the input x_0 has been completely read) and there is no configuration which can possibly follow c_n. It is well known that, as soon as $\mathcal{A}$ is normalized, for each input x over A, there is exactly one maximal associated computation. It will be called *the* computation of $\mathcal{A}$ over x, the configuration c_n is said to be its *result*.

A computation $c_0, c_1, \ldots, c_n$, with $c_i = (\gamma_i, q_i, x_i)$ is said to *contain an iterative pair* if there exist four integers i, j, k, l with $0 \leq i < j < k < l \leq n$, such that the following holds for all m :

(i) $\gamma_i = \gamma S$ for some $\gamma \in V^*$, $S \in V$;
(ii) $\gamma_m \in \gamma V^+$ for $i \leq m \leq j$;
(iii) $\gamma_j = \gamma \mu S$ for some $\mu \in V^*$ and $q_i = q_j$;
(iv) $\gamma_m \in \gamma \mu V^*$ for $j < m < k$;
(v) $\gamma_k = \gamma \mu$;
(vi) $\gamma_m \in \gamma V^+$ for $k < m < l$;
(vii) $\gamma_l = \gamma$ and $q_l = q_k$.

If we look at the words $x_0 = yx_i$, $x_i = ux_j$, $x_j = zx_k$, $x_k = vx_l$, we get that the computations of $\mathcal{A}$ over the words $yu^n zv^n x_l$, $(n \geq 0)$ all lead to the same result. Moreover, as $\mathcal{A}$ is normalized, we known that u is not empty. So, for any suffix t such that $x_0 t$ is accepted by $\mathcal{A}$, the word $x_0 t$ will have $(y, u, z, v, x_l t)$ as an iterative pair. Conversely, if x is a recognized word containing a nondegenerated iterative pair, there exists an iteration of x for which the associated computation contains an iterative pair.

A *prime computation* is a computation containing no iterative pair. A *prime word* is a word on which the computation of $\mathcal{A}$ is prime. We then leave to the reader the proof of

FACT 1. *Given a dpda $\mathcal{A}$, the set $P(\mathcal{A})$ of prime words is regular.*

FACT 2. *For any word x in the regular set $Q(\mathcal{A}) = A^* - P(\mathcal{A})$, there exist infinitely many words whose computations have the same result than the computation over x.*

We are now ready to define our special language L. Let $A = \{a, b, c, d\}$. Over $A \cup \{\#\}$, consider the languages

$$S_< = \{a^n b^p \mid n \geq p\}\,, \quad L_1 = S_< \# E\,,$$
$$L_2 = \{x \# y \mid x, y \in A^+,\ |x| < |y|\}\,.$$

Then, the language L is defined by $L = L_1 \cup L_2$. Clearly, L is context-free. We first prove

3.23. PROPOSITION. *L is not a generator of the cone* Alg.

PROOF. We show that L has no strict iterative pair. Let (x, u, y, v, z) be an iterative pair of $w = xuyvz$ in L. We then look for the occurrence of $\#$ in w. Clearly, it can be neither in u nor in v. So the following situations may arise:

(i) $\#$ lies in x. Then there exists an integer k such that $|u^k y v^k z| > |x|$. Hence, $xu^k u^+ y v^+ v^k z \subset L_2 \subset L$ and (x, u, y, v, z) is not strict.

(ii) $\#$ lies in z. Then $z = z_1 \# z_2$ and, as $xu^n y v^n z_1$ becomes longer than z_2, the word z_2 is in E. Thus (x, u, y, v, z_1) must be a pair of $S_<$ and it cannot be strict.

(iii) $\#$ lies in y. Then $y = y_1 \# y_2$. Clearly, if u or v is empty the pair is not strict. So, we assume $v \neq 1$. Then, there is at most one integer k such that $y_2 v^k z$

is in E. Hence, for all n but possibly one, $xu^n y_1 \# y_2 v^n z$ is in L_2, which implies $xu^n y_1 \# y_2 v^n v^+ z$ is in L_2 and the pair is not strict. □

Before proving that no deterministic nongenerator dominates L, we need a preliminary result on the pairs of L :

3.24. PROPOSITION. *For any word y in E, there exists a word x such that $x\#y \in L$ and such that $x\#y$ has a nondegenerated pair whose iterative elements are within x.*

PROOF. Let N be the integer associated to L by Bader and Moura's Theorem 1.3. Given y in E, choose $x = a^d b^d$ with $d > N^{2+|y|}$. In the word $x\#y$, distinguish the d occurrences of the letter a, and exclude the $|y| + 1$ letters of $\#y$. Bader and Moura's theorem guarantees that $x\#y$ contains an iterative pair within x. Clearly, it is not degenerated. □

We can now prove the following crucial result needed for Theorem 3.22.

3.25. PROPOSITION. *Let T be a deterministic language and h an alphabetic morphism such that $L = h(T)$, then T is a generator of* Alg.

PROOF *Sketch* . Let T be recognized by a dpda $\mathcal{A}$ over B. Let B' be the letters θ such that $h(\theta) = \#$. Recall that $P(\mathcal{A})$ is the regular set of those words over B which have a prime computation and that $Q(\mathcal{A})$ is its complement. We then define the regular set $B^* - h^{-1}(h(P(\mathcal{A})) = K$. This is the set of those words x' on B such that there exists no prime word x'' satisfying $h(x') = h(x'')$. We then define $R = KB'$ and we claim that $E = h(R^{-1}T)$.

(i) $E \subset h(R^{-1}T)$. By Proposition 3.24, for any word y in E, we can find a word x such that $x\#y$ is in L and has an iterative pair within x which is not degenerated. Then, for any words x' and y' and any θ in B' such that $h(x') = x, h(y') = y$ and $x'\theta y' \in T$, the word $x'\theta y'$ has a nondegenerated pair within x'. Hence x' is in K and $x'\theta$ is in R. It then follows that y' is in $R^{-1}T$ and then that $y \in h(R^{-1}T)$.

(ii) $h(R^{-1}T) \subset E$. Let y' be in $R^{-1}T$. Then, there exists $x'\theta$ in R such that $x'\theta y'$ is in T. Choose the shortest such $x'\theta$. The word x' is in K, so it is not prime. Moreover, the iterative pair that it contains has an image by h which is an iterative pair of $h(x'\theta y')$. So, we can find infinitely many words leading to the same result than x'. These words have images under h which are of increasing length because, as x' was choosen as short as possible, the iterative elements in x' have a nonempty image by h. Then, there will exist a word x'' leading to the same result than x' such that $|h(x'')| > |h(y')|$. This implies that $h(y')$ is in E.

So, putting together (i) and (ii), we have $E = h(R^{-1}T)$ and thus T is a generator. □

We can now prove Theorem 3.22.

PROOFOF THEOREM 3.22. Suppose that there exists a family $\{T_1, T_2, \cdots, T_n, \cdots\}$ of deterministic languages generating Nge. Then, there would exist an integer k such that the language L is in $\mathcal{T}(T_k)$. Thus, for some alphabetic morphisms g, h and some regular language R, one would have $L = h(g^{-1}(T_k) \cap R)$. Set $T = g^{-1}(T_k) \cap R$, then $L = h(T)$ with T a deterministic language. By Proposition 3.25, T is a generator of Alg and obviously, so is T_k. Then $\{T_1, \ldots, T_n, \ldots\}$ will generate Alg. □

Observe that this result shows in particular that if Nge is principal, it has no deterministic generator. On the other hand, this proof leaves open the following

QUESTION. May Nge be generated by a family of unambiguous languages ?

3.5 Intersection of principal cones

It is easily verified that the intersection of two cones is again a cone. However, it is not known whether the intersection of two principal cones of context-free languages is again principal. (There is a counter-example by Ullian [63] concerning noncontext-free cones. In his definition, morphisms are required to be nonerasing.) In fact, for no pair of context-free principal cones, the status of the intersection is known, excepted in trivial cases.

Among the various conjectures concerning these problems, two were recently disproved by Brandenburg [19] and Wagner [65]. Consider the languages

$$S = \{a^n b^n \mid n \geq 1\}\,, \quad \mathit{Copy} = \{w\#w \mid w \in \{a,b\}^+\}\,.$$

3.26. THEOREM (Brandenburg [19]). *Let* $B = \{a^i b^j c^m d^n \mid i \neq m$ *or* $j \leq m \leq n\}$. *Then* $B \in \mathsf{Lin} \cap \mathsf{Rocl}$, *but* $B \notin \mathcal{T}(S)$.

This result shows that $\mathsf{Lin} \cap \mathsf{Rocl}$ strictly contains $\mathcal{T}(S)$, disproving the conjecture that these two cones coincide. It leaves open the question whether $\mathsf{Lin} \cap \mathsf{Rocl}$ is principal. The proof is delicate, and we omit it here.

3.27. THEOREM (Wagner [65]). *Let* $W = \{u\#v\#w \mid v \neq u^\sim$ *or* $v = w\}$. *Then* $W \in \mathsf{Lin} \cap \mathsf{Reset}$, *but* $W \notin \mathsf{Ocl}$.

PROOF (*Sketch*). Clearly $\{u\#v\#w \mid v = w\}$ is in Reset. On the other hand, as $\{u\#v \mid v \neq u^\sim\} = \{fag\#g'bf' \mid a,b \in A,\ a \neq b,\ |f| = |f'|\}$, we see that $\{u\#v\#w \mid v \neq u^\sim\}$ is in Reset. Hence $W \in \mathsf{Reset}$. Similarly, as W can be written $\{u\#v\#w \mid v \neq u^\sim$ or $w = u^\sim\}$, we have that $W \in \mathsf{Lin}$ and, consequently, $W \in \mathsf{Lin} \cap \mathsf{Reset}$.

The fact that $W \notin \mathsf{Ocl}$ is more delicate. The argument can be sketched as follows. It is known that any counter pda may be assumed to be realtime [34]. Hence, the height of the pushdown store after reading an input of lenght n is bounded by $k.n$ for some fixed k. It follows that the number of different possible configurations reached then is bounded by $k'.n$ for some k' (remember that the pda is just a counter). On the other hand, after reading $u\#u^\sim\#$, the only possible suffix is $u^\sim$. So, any configuration reached after reading $u\#u^\sim\#$ will accept $u^\sim$ or nothing. This shows that no successfull configuration can be simultaneously reached by $u\#u^\sim\#$ and $u'\#u'^\sim\#$ for $u \neq u'$. But, the number of different such words is 2^n whence the number of different configurations is at most $k'n$. Hence the contradiction. □

Note that Brandenburg [19] shows the same result using, instead of W, the language

$$\{a^i b^j c^m d^m a^r b^s c^p d^q \mid i \neq n \text{ or } j \neq m \text{ or } (i = n = r = q \text{ and } j = m = s = p)\,.$$

In the same direction, we mention the following result.

3.28. THEOREM (Brandenburg [19]). *Let* C *be the language defined by*

$$C = \{a^i b^j c^m d^n \mid i \neq n$$ *or* $(i \geq j$ *and* $j = m)\}\,.$$

Then $C \in \mathsf{Rocl} \cap \mathsf{Reset}$ *and* $C \notin \mathsf{Lin}$.

Finally, let Queue be the cone of languages accepted by queue machine (FIFO languages or simple Post languages).

3.29. THEOREM. *Let* $S^{(2)} = \{a^n b^m c^m d^n \mid n, m \geq 0\}$. *Then* $S^{(2)} \in \mathsf{Lin} \cap \mathsf{Queue}$, *but* $S^{(2)} \notin \mathsf{Ocl}$ *and* $S^{(2)} \notin \mathsf{Reset}$.

3.30. COROLLARY. *The language* $S = \{a^n b^n \mid n \geq 1\}$ *is not a generator of any of the following cones :* $\mathsf{Lin} \cap \mathsf{Rocl}$, $\mathsf{Lin} \cap \mathsf{Reset}$, $\mathsf{Rocl} \cap \mathsf{Reset}$, *and* $\mathsf{Lin} \cap \mathsf{Queue}$.

4 Context-free groups

There exist several relations between context-free languages and groups, such as the Hotz group described in Section 1, or the definition of the two-sided Dyck language as the set of words equivalent to the empty word in the morphism of the free monoid onto the free group. This section describes recent results, mainly by Muller and Schupp [46, 47, 48] on the reverse probleme raised by Anisimov [1] : Consider a presentation of a finitely generated group, and consider the language of all words which are equivalent to the empty word. This language may or may not be context-free. What does it mean about the group that the language *is* context-free ? Before we give some recent answers concerning this question, let us recall some basic concepts.

4.1 Context-free groups

4.1.1 Presentation

A *presentation* of a group G is defined by an alphabet A, and a set R of *relators*. A new alphabet $\bar{A}$, disjoint from A, is chosen, with a bijection $a \mapsto \bar{a}$ between A and $\bar{A}$. This bijection is extended to $A \cup \bar{A}$ by setting $\bar{\bar{a}} = a$. For $w \in (A \cup \bar{A})^*$, $w = a_1 a_2 \cdots a_n$, the word $\bar{w}$ is defined by $\bar{w} = \bar{a}_n \cdots \bar{a}_1$. The relators are pairs (u, v) of words. Since G is wanted to be a group, the *trivial relations* $(a\bar{a}, 1)$, for $a \in A \cup \bar{A}$ are tacitely required to be in R and are omitted in the notation. Then any relation (u, v) is equivalent to $(u\bar{v}, 1)$ and therefore R may also be considered as a subset of $(A \cup \bar{A})^*$. The pair $\langle A; R \rangle$ is a *presentation* of G if $G \simeq (A \cup \bar{A})^*/[R]$, where $[R]$ is the congruence generated by R. The group is finitely generated if A is finite.

4.1. EXAMPLE. The free commutative group over $A = \{a, b\}$ is defined by $R = \{ab = ba\}$ (It is easier to read equations than pairs). In the free monoid over the alphabet $A \cup \bar{A} = \{a, b, \bar{a}, \bar{b}\}$, this relator, together with $a\bar{a} = \bar{a}a = b\bar{b} = \bar{b}b = 1$, induces other relators, such as $\bar{a}\bar{b} = \bar{b}\bar{a}, a\bar{b} = \bar{b}a$ etc... In fact, this free abelian group is isomorphic to $\mathbf{Z}^2$.

4.1.2 Word problem

Let G be a finitely generated group with presentation $\langle A; R \rangle$. The *word problem* of G (more accurately one should say the word problem of the presentation), is the set $W(G)$ of words over $A \cup \bar{A}$ that are equivalent to the empty word.

4.2. EXAMPLE. For the free group $F(A)$ over an n letter alphabet, the word problem is the two-sided Dyck language $\hat{D}_n^*$ which is context-free.

4.3. EXAMPLE. It is easily seen that the word problem for the free abelian group $\langle a, b; ab = ba \rangle$ is the set of words $w \in (A \cup \bar{A})^*$ such that $|w|_a = |w|_{\bar{a}}$ and $|w|_b = |w|_{\bar{b}}$. This language is not context-free.

A finitely generated group G is called *context-free* if there exists a presentation $\langle A; R\rangle$ of G for which the word problem is a context-free language. The following observation states that context-freeness is a property of the group and not of the presentation.

4.4. PROPOSITION. *Let G be a context-free group. Then the word problem of any finitely generated presentation of G is a context-free language.*

Thus, the free group is context-free, and the free abelian group with two generators is not.

4.1.3 A global characterization.

The question to determine the context-free groups was raised by Anisimov. It was solved by Muller and Schupp [47] up to a special argument contributed by Dunwoody [25]. It uses also a theorem of [60] concerning the structure of groups with more than one end (in the sense of Section 4.3 below).

First, we recall that a group G is *virtually free* if G has a free subgroup of finite index.

4.5. THEOREM. *Let G be a finitely generated group. Then G is context-free iff G is virtually free.*

It can be expected that there is a relation between special groups and special context-free languages. One such result has been given by Haring-Smith [37] : the word problem of a finitely presented group is freely generated by a simple context-free language [38], if and only if the group is a free product of a free group of finite rank and of a finite number of finite groups.

4.2 Cayley graphs

In this section, we describe a characterization of context-free groups by a triangulation property of the Cayley graph associated to one of its presentation.

Let G be a finitely generated group and let $\langle A; R\rangle$ be a presentation of G with A finite. The *Cayley graph* $\Gamma(G)$ of the presentation has as vertices the elements of G. The edges are labelled by elements of the alphabet $A \cup \bar{A}$. There is an edge from g to g' labelled by c iff $gc \equiv g' \bmod [R]$. For each edge, there is an inverse edge, from g' to g and labelled by $\bar{c}$.

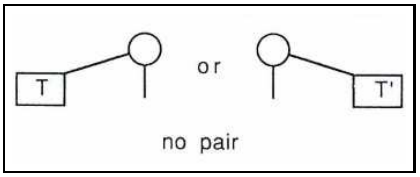

Figure 7: Cayley graph of $\mathbb{Z}$

4.6. EXAMPLE. Figures 7–9 represent the Cayley graphs of the free group with one generator, the free abelian group with two generators and the free group with two generators. In drawing graphs, we represent only one of the pair composed of an edge and of its inverse. Each edge being labelled (by a letter), the label of a path is the

word composed of the labels of its edges. Clearly any label of a closed path (a cycle) is in the word problem of the representation, and conversely. A cycle is *simple* if nonconsecutive edges do not share a common vertex.

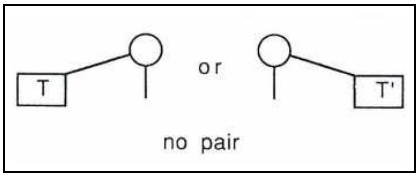

Figure 8: Cayley graph of $\mathbb{Z}^2$

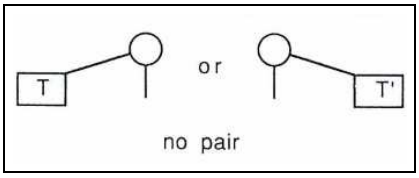

Figure 9: Cayley graph of the free group with two generators

A *diagonal* triangulation T of a simple cycle P is a triangulation of P with the following properties:

(1) The vertices of the triangles are among those of P.

(2) Each new edge has a label over $(A \cup \bar{A})^*$; these labels are such that reading around the boundary of each triangle gives a relation which holds in the group G.

A *bound* for the triangulation is an upper bound to the length of the new labels.

4.7. EXAMPLE. In the free abelian group over $\{a, b\}$, a closed path is given in fat lines in Figure 10, the additional edges are drawn in thin lines. A bound for the triangulation is 3. There is also a triangulation of bound 2, but none of bound 1.

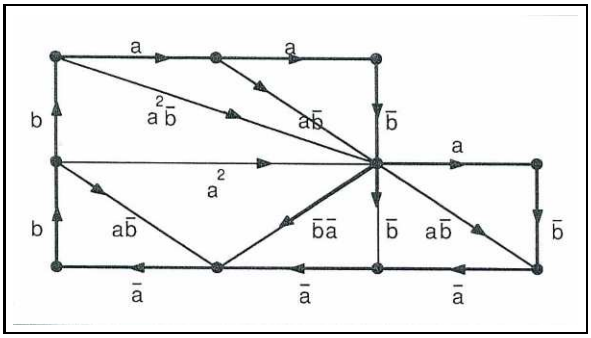

Figure 10: Triangulations in $\mathbb{Z}^2$

The Cayley graph can be *uniformly triangulated* if there is an integer K such that every simple cycle admits a diagonal triangulation with bound K.

4.8. THEOREM. *A finitely generated group G given by a presentation $\langle A; R\rangle$ is context-free if and only if its Cayley graph can be uniformly triangulated.*

PROOF (*Sketch*). Take a reduced context-free grammar $G = (V, A \cup \bar{A}, P, S)$ generating the word problem, in Chomsky normal form (see Section 1.1). For each variable X, let $u_X \in (A \cup \bar{A})^*$ be a fixed word generated by X. Let

$$w = a_1 a_2 \cdots a_n \qquad (a_i \in A \cup \bar{A})$$

be the labels of a simple cycle, and assume $n \geq 4$. Since $S \stackrel{*}{\to} w$ one has

$$S \to XY \stackrel{*}{\to} xy \qquad x = a_1 \cdots a_i, \quad y = a_{i+1} \cdots a_n \,.$$

Introduce a new edge from the starting point of the first edge to the endpoint of the ith edge with label $u_X (= u_Y^{-1})$. Then the relations $u_X = a_1 \cdots a_i$, $u_Y = a_{i+1} \cdots a_n$ hold in G and the cycles are shorter.

The reverse construction is similar. First, a variable X_u is introduced for any word $u \in (A \cup \bar{A})^*$ of length less or equal to the uniform bound. Then productions $A_u \to u$, and $A_u \to A_v A_w$, are introduced whenever $u = vw$ in the group G. Finally, one shows by induction on the length of the cycles that if $u_1, \cdots, u_n$ are the labels of the edges of a cycle, each of bounded length, then $A_1 \stackrel{*}{\to} A_{u_1} A_{u_2} \cdots A_{u_n}$. □

This proof shows that in a context-free group G there always exists a finite subset H with the following property : any word w in the word problem of G can be split into two parts $w_1 w_2$ such that each w_i is a product of elements of H. In some sense, the constructed grammar just takes this set H as set of variables. It is then easily seen that the grammar generates a pre-NTS language. On the other hand, a very similar construction has been used to show that some special languages are NTS [4]. So, it was natural to ask if the above proof could be improved to get the following theorem:

4.9. THEOREM. *Any context-free group has a NTS word-problem.*

No proof of this result is known which uses the triangulation result. However, using a constructive characterization of virtually free groups, the above theorem was recently shown to be a consequence of Theorem 4.5 [5]. Note that theorem 4.9 may be used to construct a set of generators in a context-free group in such a way that the uniform bound of Theorem 4.8is 1.

4.3 Ends

The triangulation result of the previous section gives a "local" characterization. We now quote a more global one.

Let Γ be the Cayley graph of a finitely generated presentation of a group G Denote by $\Gamma^{(n)}$ the subgraph with vertices at distance at most n from the origin (the vertex with label 1), and let k_n be the number of *connected components* in $\Gamma - \Gamma^{(n)}$. The *number of ends* of Γ is $\lim_{n\to\infty} k_n$.

4.10. EXAMPLE. For the Cayley graph of the infinite cyclic group, the number of ends is 2. For the free abelian group, the number of ends is 1; for the free group with two generators, the number of ends is infinite.

4.11. PROPOSITION. *If G is an infinite context-free group, then the Cayley graph of one of its finitely generated presentations has more than one end.*

Stallings [60] gives a complete description of the structure of groups with more than one end.

This result, showing that there must be several ends for a context-free group, will now be opposed to a statement which claims that there must be only few nonisomorphic subgraphs. For this, we need some definitions.

Let $n \geq 0$, and let C be a connected component of $\Gamma - \Gamma^{(n)}$. A *frontier point* of C is a vertex u of C at distance $n+1$ from the origin. If v is a vertex of Γ at distance say $n+1$, we denote by $\Gamma(v)$ the component of $\Gamma - \Gamma^{(n)}$ containing v, and by $\Delta(v)$ the set of all frontier points of $\Gamma(v)$.

An *end-isomorphism* between two subgraphs $\Gamma(u)$ and $\Gamma(v)$ is a label preserving graph isomorphism that maps $\Delta(u)$ onto $\Delta(v)$. The Cayley graph is *context-free* if the set $\{\Gamma(v) \mid v \text{ a vertex of } \Gamma\}$ has only finitely many isomorphism classes under end-isomorphism.

4.12. THEOREM. *A group is context-free if and only if its Cayley graph is context-free.*

4.13. EXAMPLE. Consider first the infinite cyclic group. There are exactly two isomorphism classes. For the free abelian group with two generators the number of frontier points of the unique connected component of $\Gamma - \Gamma^{(n)}$ grows quadratically and therefore all these components are non isomorphic. Thus the Cayley graph is not context-free. For the free group over two generators, again there are four isomorphism classes.

There is a strong relation between context-free graphs and pushdown automata. Muller, Schupp [48] proved this last theorem.

4.14. THEOREM. *A graph is context-free if and only if it is the complete transition graph of a pushdown automaton.*

Acknowlegments

We acknowledge helpful discussions with Jean-Michel Autebert, Christophe Reutenauer and Paul Schupp.

References

[1] A. V. ANISIMOV, Group languages, *Kibernetica* **4** (1971), 18–24.

[2] J. M. AUTEBERT, *Langages algébriques*, Masson, 1987.

[3] J. M. AUTEBERT, J. BEAUQUIER, L. BOASSON, M. NIVAT, Quelques problèmes ouverts en théorie des langages, *RAIRO Informatique théorique* **13** (1979), 363–379.

[4] J. M. AUTEBERT, J. BEAUQUIER, L. BOASSON, G. SÉNIZERGUES, Langages de parenthèses, langages NTS et homomorphismes inverses, *RAIRO Informatique théorique* **18** (1984), 327–344.

[5] J. M. AUTEBERT, L. BOASSON, G. SÉNIZERGUES, Groups and NTS languages, *J. Comput. System Sci.* **35** (1987), 213–267.

[6] J. M. AUTEBERT, P. FLAJOLET, J. GABARRÓ, Prefixes of infinite words and ambiguous context-free languages, *Inform. Proc. Letters* **25** (1987), 211–216.

[7] C. BADER, A. MOURA, A generalization of Ogden's lemma, *J. Assoc. Comput. Mach.* **29** (1982), 404–407.

[8] D. R. BEAN, A. EHRENFEUCHT, G. F. MCNULTY, Avoidable patterns in strings of symbols, *Pacific J. of Math.* **85** (1979), 261–294.

[9] J. BEAUQUIER, Ambiguïté forte, in: *Proc. 5th ICALP* , Udine, *Lecture Notes Comput. Sci.* **62** (1978), 52–62.
[10] J. BEAUQUIER, Générateurs algébriques et systèmes de paires itérantes, *Theoret. Comput. Sci.* **8** (1979), 293–323.
[11] J. BEAUQUIER, F. GIRE, On context-free generators, *Theoret. Comput. Sci.* **51** (1987), 117–127.
[12] J. BERSTEL, *Transductions and Context-Free Languages*, Teubner Verlag, 1979.
[13] J. BERSTEL, Properties of infinite words : recent results, in: R. Cori, B. Monien (eds), *Symp. Theoret. Aspects Comput. Sci. '89*, *Lecture Notes Comput. Sci.* **?** (1989), ?–?.
[14] M. BLATTNER, S. GINSBURG, Position restricted grammar forms and grammars, *Theoret. Comput. Sci.* **17** (1982), 1–27.
[15] L. BOASSON, Non-générateurs algébriques et substitution, *RAIRO Informatique théorique* **19** (1985), 125–136.
[16] L. BOASSON, A. PETIT, Deterministic languages and nongenerators, *RAIRO Informatique théorique* **21** (1987), 41–57.
[17] R. V. BOOK, S. A. GREIBACH, C. WRATHALL, Reset machines, *J. Comput. Syst. Sci.* **19** (1973), 256–276.
[18] F. J. BRANDENBURG, Uniformely growing k-th powerfree homomorphisms, *Theoret. Comput. Sci.* **23** (1983), 69–82.
[19] F. J. BRANDENBURG, Intersections of some families of languages, in: *Proc. 13th ICALP*, Rennes, *Lecture Notes Comput. Sci.* **226** (1986), 60–68. Also: Counters are not in the GLB of pushdown and queues, Techn. Report MIP 8513, University of Passau (1985).
[20] D. E. COHEN, Groups of cohomological dimension one, *Lecture Notes in Math.* **245**, Springer-Verlag, 1972.
[21] J. P. CRESTIN, Un langage non ambigu dont le carré est d'ambiguïté non bornée, in: *Proc. 1st ICALP*, Paris, North-Holland, 1973, 377–390.
[22] M. CROCHEMORE, A sharp characterization of square-free morphisms, *Theoret. Comput. Sci.* **18** (1982), 221–226.
[23] V. DIEKERT, Investigations on Hotz groups for arbitrary grammars, *Acta Informatica* **22** (1986), 679–698.
[24] V. DIEKERT, A. MÖBUS, Hotz-isomorphism theorems in formal language theory, in: R. Cori, M. Wirsing (eds), *Symp. Theoret. Aspects Comput. Sci. '88*, *Lecture Notes Comput. Sci.* **294** (1988), 126–135.
[25] M. J. DUNWOODY, The accessibility of finitely presented groups, *Inventiones Mathematicae* **81** (1985), 449–457.
[26] A. EHRENFEUCHT, G. ROZENBERG, On the separative power of EOL systems, *RAIRO Informatique théorique* **17** (1983), 13–32.
[27] A. EHRENFEUCHT, G. ROZENBERG, Strong iterative pairs and the regularity of context-free languages, *RAIRO Informatique théorique* **19** (1685), 43–56.
[28] P. FLAJOLET, Ambiguity and transcendence, in: *Proc. 12th ICALP* , *Lecture Notes Comput. Sci.* **194** (1985), 179–188.
[29] P. FLAJOLET, Analytic models and ambiguity of context-free languages, *Theoret. Comput. Sci.* **49** (1987), 283–309.
[30] C. FROUGNY, J. SAKAROVITCH, E. VALKEMA, On the Hotz group of a context-free grammar, *Acta Informatica* **18** (1982), 109-115
[31] J. GABARRÓ, Some applications of the interchange lemma, *Bull. EATCS* **25** (1985), 19–21.
[32] S. GINSBURG, *The Mathematical Theory of Context-Free Languages*, McGrawHill, 1966.

[33] S. GINSBURG, *Algebraic and Automata-Theoretic Properties of Formal Languages*, North-Holland, 1975.
[34] S. GINSBURG, J. GOLDSTINE, S. A. GREIBACH, Uniformly erasable families of languages, *J. Comput. Syst. Sci.* **10** (1975), 165–182.
[35] A. GRAZON, An infinite word language which is not co-CFL, *Inform. Proc. Letters* **24** (1987), 81–86.
[36] S. A. GREIBACH, Checking automata and one-way stack languages, *J. Comput. Syst. Sci.* **3** (1969), 196–217.
[37] R. H. HARING-SMITH, Groups and simple languages, *Ph. D. Thesis*, Urbana (1981).
[38] M. HARRISON, *Introduction to Formal Language Theory*, Addison-Wesley, 1978.
[39] G. HOTZ, Normalform transformations of context-free grammars, *Acta Cybernetica* **4** (1978), 65–84.
[40] G. HOTZ, Eine neue Invariante für kontextfreie Sprachen, *Theoret. Comput. Sci.* **11** (1980), 107–116.
[41] M. JANTZEN, *Confluent String Rewriting*, EATCS Monographs Theoret. Comput. Sci. **14**, Springer-Verlag 1988
[42] R. KEMP, On the number of words in the language $\{w \in \Sigma^* \mid w = w^{\sim}\}$, *Discrete Math.* **40** (1980), 225–234.
[43] W. KUICH, A. SALOMAA, *Formal Power Series and Languages*, EATCS Monographs Theoret. Comput. Sci. **5**, Springer-Verlag, 1986.
[44] M. LOTHAIRE, *Combinatorics on Words*, Addison-Wesley, 1983.
[45] M. G. MAIN, An infinite square-free co-CFL, *Inform. Proc. Letters* **20** (1985), 105–107.
[46] D. E. MULLER, P. E. SCHUPP, Pushdown automata, ends, second-order logic and reachability problems, in: *Proc. 13th Annual ACM Symp. on the Theory of Computing*, Milwaukee (1981), 46–54.
[47] D. E. MULLER, P. E. SCHUPP, Groups, the theory of ends, and context-free languages, *J. Comput. System Sci.* **26** (1983), 295–310.
[48] D. E. MULLER, P. E. SCHUPP, The theory of ends, pushdown automata, and second-order logic, *Theoret. Comput. Sci.* **37** (1985), 51–75.
[49] A. NIJHOLT, An annotated bibliography of pumping, *Bull. EATCS* **17** (1982), 34–52.
[50] W. OGDEN, R. ROSS, K. WINKLMANN, An "interchange lemma" for context-free languages, *SIAM J. Comput.* **14** (1985), 410–415.
[51] M. OYAMAGUCHI, The equivalence problem for realtime dpda's, *J. Assoc. Math. Comput* **34**(3), 1987, 731–760.
[52] D. PERRIN, *Finite Automata*, this handbook.
[53] R. ROSS, K. WINKLMANN, Repetitive strings are not context-free, *RAIRO Informatique théorique* **16** (1982), 191–199.
[54] W. RUDIN, *Real and Complex Analysis*, McGraw Hill, 1974.
[55] A. SALOMAA, *Formal Languages*, Academic Press, 1973.
[56] A. SALOMAA, *Formal Languages and Power Series*, this handbook.
[57] M. P. SCHÜTZENBERGER, On a theorem of R. Jungen, *Proc. Amer. Math. Soc.* **13** (1962), 885–889.
[58] G. SÉNIZERGUES, The equivalence and inclusion problems for NTS languages, *J. Comput. Syst. Sci.* **31** (1985), 303–331.
[59] G. SÉNIZERGUES, Some decision problems about controlled rewriting systems, *Theoret. Comput. Sci.* **?** (198?), ?–?.

[60] J. STALLINGS, Groups of cohomological dimension one, *Proc. Symp. Pure Math. American Math. Soc.* **17** (1970), 124–128.
[61] W. THOMAS, *Automata on Infinite Objects*, this handbook.
[62] E. TOMITA, K. SEINO, A direct branching algorithm for checking equivalence of two deterministic pushdown transducers, one of which is realtime strict, *Theoret. Comput. Sci.* **?** (198?), ?–?.
[63] J. S. ULLIAN, Three theorems concerning principal AFL's, *J. Comput. Syst. Sci.* **5** (1971), 304–314.
[64] E. VALKEMA, On some relations between formal languages and groups, *Proc. Categorical Algebraic Methods Comput. Sci. System Th.*, Dortmund (1978), 116–123.
[65] K. WAGNER, On the intersection of the class of linear context-free languages and the class of single-reset languages, *Inform. Proc. Letters* **23** (1986), 143–146.