



Languages Generated Using an Abstract Catenation

ANDREI POPESCU

Department of Fundamentals of Computer Science
University of Bucharest, Romania
Academiei 14, Bucharest, Romania
E-mail: uuomul@yahoo.com

Abstract

We present some results regarding languages that are context-free or regular w.r.t. a non-deterministic binary operation on words that is left variable, only asking a few general properties like associativity. This could be part of a unified approach to *intertextuality*, the results referring, in particular, to important instances of "putting texts together" operations like *concatenation* and *shuffle* (with all their variations - *distributed*, *on trajectories* etc.).

Keywords: intertextuality, non-deterministic operation on words, abstract catenation, pumping lemma.

1 Introduction

Formal languages, organized in Chomsky-like hierarchies, have been studied mainly using the classical concatenation operation on words. Closure properties, membership criteria or equivalent descriptions have been obtained for classes of languages.

However, for different purposes, like the modeling of some linguistic or non-deterministic computation aspects, other operations on words had to be considered. A whole variety of such operations were introduced and studied in [2],[3],[4],[5],[6]: distributed catenation, shuffle, shuffle on trajectories, distributed shuffle, mix etc. In [2] (see also [1], [7]), it is pointed out the linguistic significance of such operations for the phenomenon called *intertextuality* - the contamination of a text by another - encountered in both literature and science. Indeed (besides their well-acknowledged adequacy in treating concurrency or non-determinism in computer science), these operations provide combinatorial ways to put texts together, all of them being steps further towards a mathematical treatment of the above phenomenon. The mentioned papers study generating or accepting structures for languages, with similar functionality as the classical ones, just that they use these new operations instead of the quite inflexible concatenation. Some of the beauty (and usefulness) of the classical results

can be found here too, where similar closure properties, pumping lemmas etc. are proved.

Of course, all of these operations exhibit specific behavior, which individualizes each of them. But what will keep our interest in this paper are exactly their common aspects, in trying to consider a generic way to put texts together (or, we can say, a generic way to "shuffle" words) by taking as working hypotheses some very general and natural properties of an operation on words.

We shall treat the case of a binary operation that assigns to a couple of words a finite language (we do this to catch in our framework also the non-deterministic operations like shuffle). Among the properties we might consider are:

- associativity¹;
- commutativity;
- the empty word is the identity element;
- uniformity (all the words from the result have the same length);
- increasing of the result length with the arguments length;
- extensivity (the increase of length when applying the operation);
- length preservation (the length of the result words are the sums of the arguments lengths);
- recursiveness;
- "easy" recursiveness (for example, there exists a Turing machine with three tapes such that the arguments are scanned from the first two tapes and, at a sole passing, the result is written on the third tape.)²

It is worth saying that all the mentioned operations (that could be thought as hypostases of intertextuality), including concatenation, have all the above properties, except commutativity and "easy" recursiveness.

Some operation-independent properties, taking diverse abstract assumptions, will be proved in this paper.

2 Preliminaries

For a set A , $\mathcal{P}(A)$ ($\mathcal{P}_f(A)$) denotes the set of parts (finite parts) of A . We will sometimes identify a singleton set $\{a\}$ with the element a ; this is the reason why, when dealing with a binary operation $*$ on $\mathcal{P}(A)$ or $\mathcal{P}_f(A)$, $a, b \in A$ and $B \subseteq A$, it makes sense to write $a * B$ or $a * b$.

We fix Σ to be an alphabet and a function $\bullet : \Sigma^* \times \Sigma^* \longrightarrow \mathcal{P}_f(\Sigma^*) \setminus \{\emptyset\}$. We can immediately extend \bullet to a binary operation (denoted the same)

$$\bullet : \mathcal{P}(\Sigma^*) \times \mathcal{P}(\Sigma^*) \longrightarrow \mathcal{P}(\Sigma^*)$$

by putting

$$P \bullet Q = \bigcup_{x \in P, y \in Q} x \bullet y.$$

¹Of course, associativity is stated about the extension of this operation to languages.

²This particular "ease" condition makes sense only for "deterministic" operations.

All along the paper, we shall assume that \bullet is associative and has $\{\lambda\}$ as identity.

Consider the following properties (the variables x, y, x_1, x_2, y_1, y_2 range over Σ^* ; for a word x , $|x|$ denotes its length):

- (E) $(\forall u \in x \bullet y, v \in y \bullet x) |y| \leq |u| \& |y| \leq |v|$;
- (SE) $x \neq \lambda \Rightarrow (\forall u \in x \bullet y, v \in y \bullet x) |y| < |u| \& |y| < |v|$
- (SL) The words from $x \bullet y$ have the same length;
- (LP) $(\forall u \in x \bullet y) |u| = |x| + |y|$;
- (R) \bullet is recursive³.

In what follows, we shall deal with usual grammars of type \mathcal{L}_2 or \mathcal{L}_3 , for which we accept λ -productions. They shall have the form $G = (N, \Sigma, S, P)$ with fixed terminal alphabet Σ , such that N , S and P are the nonterminal alphabet, the start symbol and the productions set respectively.

We extend \bullet to

$$\bullet : (\Sigma \cup N)^* \times (\Sigma \cup N)^* \longrightarrow \mathcal{P}_f((\Sigma \cup N)^*)$$

defined, for $v_1, v_2 \in (\Sigma \cup N)^*$, by

$$v_1 \bullet v_2 = x_1(y_1 \bullet y_2)x_2,$$

where $v_1 = x_1y_1$, $v_2 = y_2x_2$, y_1 is the longest suffix of v_1 formed only by terminals and y_2 is the longest prefix of v_2 formed only by terminals.

Remark 2.1 In the particular deterministic case (when \bullet is just a binary operation on Σ^*), $((\Sigma \cup N)^*, \bullet, \lambda)$ is the direct sum of monoids between $(\Sigma^*, \bullet, \lambda)$ and (N^*, \cdot, λ) , where \cdot is the concatenation.

The *one-step rewrite* relation \Rightarrow_G is defined as usual, just that we consider \bullet instead of concatenation:

$v_1 \Rightarrow_G v_2$ iff $v_1 \in x_1 \bullet x \bullet x_2$, $v_2 \in x_1 \bullet y \bullet x_2$ and $x \rightarrow y \in P$.

The *rewrite relation* \Rightarrow_G^* is the reflexive-transitive closure of the one-step rewrite, while the *generated language* $L_\bullet(G)$ is the set of words from Σ^* for which there exists a rewrite starting from S .

3 \bullet -CF languages

Let $G = (N, \Sigma, S, P)$ be a CF grammar. The *rewrite trees* are labelled trees with emphasized root, ordered list of sons for each vertex and labels from $N \cup \Sigma^*$. They are defined recursively (using a fully parenthesized notation of the form "root(son1,son2,...)"), in the usual way:

- (1) for any production $A \rightarrow w$, with $w \in \Sigma^*$, the two-vertex tree, having the root labeled with A and the other vertex with w , is a rewrite tree;

³Here we talk about the initial function \bullet , that defined on $\Sigma^* \times \Sigma^*$, and not its extension to $\mathcal{P}_f(\Sigma^*) \times \mathcal{P}_f(\Sigma^*)$.

(2) if $\mathcal{A}_1, \dots, \mathcal{A}_n$ are rewrite trees having the roots labeled by A_1, \dots, A_n (from N) and $A \rightarrow v_1 A_1 \dots v_n A_n v_{n+1}$ (with $v_1, \dots, v_{n+1} \in \Sigma^*$) is in P , then $k(p_1, \mathcal{A}_1, \dots, p_n, \mathcal{A}_n, p_{n+1})$, where the nodes k, p_1, \dots, p_{n+1} are labeled with A, v_1, \dots, v_{n+1} , is a rewrite tree.

On rewrite trees (as for usual labeled trees), we consider the *subtree relation* \leq defined as follows: $\mathcal{B} \leq \mathcal{C}$ iff $\mathcal{B} = \mathcal{C}$ or, in the recursive process of building \mathcal{C} , \mathcal{B} appears, sometime, at step (2) form above, as one of the \mathcal{A}_i -s. The *depth* of a rewrite tree is the length of the longest path inside the tree. The *frontier* of a rewrite tree \mathcal{A} , denoted $Fr(\mathcal{A})$, is the list of labels of its leaves, from left to right. If $Fr(\mathcal{A})$ is (v_1, \dots, v_n) , $\bullet Fr(\mathcal{A})$ denotes the language $v_1 \bullet \dots \bullet v_n$. Because of the asociativity of \bullet , the following holds:

Proposition 3.1 Let $G = (N, \Sigma, S, P)$ be a CF grammar and $w \in \Sigma^*$. Then $w \in L_\bullet(G)$ iff there exists an S-rewrite tree (that is, root labeled with S) \mathcal{A} such that $w \in \bullet Fr(\mathcal{A})$.

If $A \in N$, we call *A-erasing* (or, simply, *erasing*) the mapping $h_A : (\Sigma \cup N)^* \rightarrow \mathcal{P}_f((\Sigma \cup N)^*)$ defined by:

$$h_A(v_1 A v_2 A \dots v_{n-1} A v_n) = v_1 \bullet v_2 \bullet \dots \bullet v_n ,$$

where A does not appear in any v_i .

Notice that $h_A \circ h_B = h_B \circ h_A$ for all $A, B \in N$, so, if $\Gamma = \{A_1, \dots, A_n\} \subseteq N$, we can denote by h_Γ the composition of h_{A_1}, \dots, h_{A_n} (in particular, h_\emptyset is the identity).

3.1 Elimination of λ -productions

Proposition 3.2 Assume (SL). Let $L = L_\bullet(G)$, where G is a CF grammar. Then there exists a CF grammar G' without λ -productions such that $L_\bullet(G') \subseteq L$ and $L \setminus L_\bullet(G') \subseteq \{\lambda\}$.

Proof:

Suppose $G = (N, \Sigma, S, P)$. Let $\Omega = \{A \in N / A \Rightarrow_G^* \lambda\}$.

Let $G' = (N, \Sigma, S, P')$, where

$$P' = \{A \rightarrow \beta / A \rightarrow \alpha \in P, \beta \in h_\Gamma(\alpha) \setminus \{\lambda\}, \Gamma \subseteq \Omega\} .$$

The fact that $L_\bullet(G') \subseteq L_\bullet(G)$ is obvious, since any one-step rewrite in G' can be simulated by a rewrite in G .

Now, let \mathcal{A} be an S-rewrite tree in G of a word $w \in \Sigma^+$. For any subtree \mathcal{B} (with root labelled, say, with B) of \mathcal{A} maximal w.r.t. the subtree relation such that $\lambda \in \bullet Fr(\mathcal{B})$,⁴ the following hold:

- $\mathcal{B} \neq \mathcal{A}$, because $w \neq \lambda$;
- $B \in \Omega$;

⁴Because of (SL), $\lambda \in \bullet Fr(\mathcal{B})$ means $\bullet Fr(\mathcal{B}) = \{\lambda\}$.

- if \mathcal{C} (with root labeled by C) is the (strict) subtree of \mathcal{A} which is the parent of \mathcal{B} , there exists the production $C \rightarrow \alpha_1 B \alpha_2$ in P , with $\lambda \notin \alpha_1 \bullet \alpha_2$, that gives the first level of the "expansion" of \mathcal{C} out of C .

So, knowing that $B \in \Omega$, there exists $\alpha \in \alpha_1 \bullet \alpha_2$ such that $C \rightarrow \alpha$ is in P' and $w \in v_3 \bullet v_1 \bullet v_2 \bullet v_4$, with $A \Rightarrow_G^* \alpha_3 C \alpha_4$, $\alpha_i \Rightarrow_G^* v_i$, $i \in \{1, 2, 3, 4\}$.

Thus, if, for each \mathcal{B} and \mathcal{C} as above, the subtree \mathcal{B}

- is replaced in \mathcal{A} by a sole vertex labeled with λ if α_1 ends with a nonterminal or α_2 starts with a nonterminal

or

- is deleted from \mathcal{A} otherwise,

we get a subtree \mathcal{A}' with $\bullet Fr(\mathcal{A}') = \bullet Fr(\mathcal{A})$ and which is an S -rewrite tree in G' . Hence $w \in L_\bullet(G')$, so $L \setminus L_\bullet(G') \subseteq \{\lambda\}$.

q.e.d.

Remark 3.1 If we assume (E), then any CF grammar without λ -productions does not have λ in $L_\bullet(G)$, thus the conclusion of the last proposition could be formulated with $L_\bullet(G') = L \setminus \{\lambda\}$.

3.2 Elimination of renamings

Given a grammar G , a *renaming* in G is a production $A \rightarrow B$ such that $A, B \in N$.

Proposition 3.3 Let $L = L_\bullet(G)$, where $G = (N, \Sigma, S, P)$ is a CF grammar. Then $L = L_\bullet(G')$, with G' a CF grammar without renamings. Moreover, if G does not have λ -productions, neither does G' .

Proof:

Define

$$\Theta = \{(A, B) \in N \times N / (\exists A_1, \dots, A_n \in N) A = A_1 \Rightarrow_G \dots \Rightarrow_G A_n = B\}.$$

Let $G' = (N, \Sigma, S, P')$, where

$$P' = \{A \rightarrow \alpha / \alpha \notin N, (\exists B \in N)(A, B) \in \Theta, B \rightarrow \alpha \in P\}.$$

Remark that P' contains, among others, all the non-renamings from P . It is clear that G' does not have renamings and $L_\bullet(G') \subseteq L$. Conversely, let \mathcal{A} be an S -rewrite tree in G of some terminal word w . Every maximal path in \mathcal{A} which consists only of renamings can be deleted, the first vertex of it following to be connected to the subtree of \mathcal{A} to which the last vertex of it was connected - in this way we obtain an S -rewrite tree in G' for w .

Finally, from the definition of P' , one can see that G' has λ -productions iff G does. *q.e.d.*

Corollary 3.1 Assume (SL). Any language that can be \bullet -generated by a CF grammar G can also be \bullet -generated (eventually loosing λ) by a CF grammar G' without λ -productions and renamings. If, in addition, we assume (R), the transformation of G into G' can be performed automatically.

Proof:

The first part is an immediate consequence of Propositions 2 and 3.

Now, in order to prove that the transformation from Proposition 2 can be automatized, we need to show that the set $\Omega = \{A \in N / A \Rightarrow_G^* \lambda\}$ from the proof of Proposition 2 can be computed. For this, we consider the string $(Y_n)_{n \in \mathbb{N}}$ defined recursively by

$$Y_0 = \{A \in N / A \rightarrow \lambda \in P\},$$

$$Y_{n+1} = Y_n \cup \{A \in N / A \rightarrow A_1 \dots A_n \in P, A_1, \dots, A_n \in Y_n\}.$$

This string has an $n \in \mathbb{N}$ such that $Y_n = Y_{n+1}$ and thus, for each $m \geq n$, $Y_m = Y_n$. Because of (E), the construction of Y_n gives the only way of rewriting λ , so $\Omega = Y_n$. Hence Ω is computable.

Finally, the relation Θ from Proposition 3 is computable because it is the reflexive-transitive closure of a given relation on a finite set.

q.e.d.

Remark 3.2 Any CF grammar can be transformed into a \bullet -equivalent one that does not have the start symbol in the right side of any production (this can be done, for instance, by adding a new start symbol). In addition, this property is invariant to the transformations from the last two propositions. Hence, under the (SL) hypothesis, any \bullet -CF language can be \bullet -generated by a CF grammar that does not have renamings and has at most one λ -production, $S \rightarrow \lambda$.

Corollary 3.2 Assume (SL), (SE) and (R). Then the membership of a word to a \bullet -CF language is decidable.

Proof:

The membership of λ to L is decided by checking whether S is in Ω or not, where Ω is the set from the proof of Proposition 2. According to Corollary 1 and Remark 2, we can automatically construct (starting from the initial grammar G of the language L) a CF grammar G' without renamings and λ -productions that generates $L \setminus \{\lambda\}$. Because of (SE) and (R), it is decidable, for a nonempty word w , whether G' rewrites it or not (because each one-step-rewrite increases the length).

q.e.d.

3.3 The pumping lemma

Proposition 3.4 Let L be a \bullet -CF language. There exist the natural numbers m and q such that, for any $w \in L$ with $|w| > m$, there exist $v_1, v_2, v_3, v_4, v_5 \in \Sigma^*$ such that:

- (1) $w \in v_1 \bullet v_3 \bullet v_5 \bullet v_4 \bullet v_2$;
 - (2) Any word from $v_3 \bullet v_5 \bullet v_4$ has length $\leq q$;
 - (3) $v_3 \neq \lambda$ or $v_4 \neq \lambda$;
 - (4) $(\forall i \in \mathbb{N}) v_1 \bullet (v_3)^{\bullet i} \bullet v_5 \bullet (v_4)^{\bullet i} \bullet v_2 \subseteq L$,
- where $v^{\bullet i}$ means $v \bullet v \dots \bullet v$ (i times).

Proof:

Let $G = (N, \Sigma, S, P)$ be a CF grammar for L . The greatest number of nonterminals which appear in a production determines the greatest number of sons that a vertex in a rewrite tree can have. Consequently, if n is the cardinal of N , there exists a finite number of words that have an S -rewrite tree of depth $\leq n$. Let m be the maximal length of such a word. Analogously, let q be the maximal length of a word that has a rewrite tree (not necessary with root labeled by S) of depth $\leq n + 1$. We show that m and q satisfy the required properties.

Let $w \in L$ of length $> m$. We consider a rewrite tree \mathcal{A} for w with minimal number of edges. But the depth of \mathcal{A} will be $> n$ and we take a path $d_1 \dots d_p c_1 \dots c_{n+1} a$, with $p \geq 0$, formed by vertexes from \mathcal{A} , where d_i, c_i are labeled with nonterminals and a with a terminal word. There exist, among $\{1, \dots, n + 1\}$, two different indices $i < j$ such that c_i and c_j are labeled with the same nonterminal, say A . We denote by \mathcal{B} and \mathcal{C} the subtrees of \mathcal{A} which correspond to the vertexes c_i and c_j respectively. We have $\mathcal{C} < \mathcal{B} < \mathcal{A}$, where $<$ is the subtree relation. It follows that $Fr(\mathcal{A})$ is a list of words from Σ^* , which has $Fr(\mathcal{B})$ as a sublist; and $Fr(\mathcal{B})$ in turn, has $Fr(\mathcal{C})$ as a sublist. Thus, if we let L_i , with $i \in \{1, 2, 3, 4\}$, be the languages obtained by applying the operation \bullet to the "difference lists", we obtain:

$$w \in \bullet Fr(\mathcal{A}) = L_1 \bullet (\bullet Fr(\mathcal{B})) \bullet L_2 = L_1 \bullet (L_3 \bullet (\bullet Fr(\mathcal{C})) \bullet L_4) \bullet L_2$$

so there exist $v_1 \in L_1, v_3 \in L_3, v_3 \in L_3, v_5 \in \bullet Fr(\mathcal{C}), v_4 \in L_4, v_2 \in L_2$, such that

$$w \in v_1 \bullet v_3 \bullet v_5 \bullet v_4 \bullet v_2 ,$$

$$v_3 \bullet v_5 \bullet v_4 \in \bullet Fr(\mathcal{B}) .$$

Now, the depth of \mathcal{B} is $\leq n + 1$, because it is obtained exactly on the path $c_i \dots c_{n+1} a$, otherwise it is contradicted the maximality of this path in \mathcal{A} . Thus $|v_3 \bullet v_5 \bullet v_4| \leq q$. Moreover, if, by absurd, $v_3 = v_4 = \lambda$, we would obtain an S -rewrite tree for w by replacing in \mathcal{A} the subtree \mathcal{B} with \mathcal{C} , which would have fewer edges than \mathcal{A} , contradicting the minimality of \mathcal{A} . Hence $v_3 \neq \lambda$ or $v_4 \neq \lambda$.

At last, it is easy to see that, replacing \mathcal{B} with \mathcal{C} in \mathcal{A} for $i = 0$ and replacing \mathcal{C} with \mathcal{B} in \mathcal{A} $i - 1$ times for the other i -s, we get:

$$\forall i \in \mathbb{N}, \quad v_1 \bullet v_3^{\bullet i} \bullet v_5 \bullet v_4^{\bullet i} \bullet v_2 \subseteq L .$$

q.e.d.

Remark 3.3 If we assume (E), it follows that any word w from Σ has a finite number of \bullet -decompositions (v_1, \dots, v_n) such that $w \in v_1 \bullet \dots \bullet v_n$. So we can modify, in the proof of the previous proposition, the choice of q , taking it to be the maximum between the sums of length of words from a \bullet -decomposition of a word w that has a rewrite tree of depth $\leq n + 1$. We thus obtain a variant of the pumping lemma, in which point (2) is replaced by

$$(2') \quad |v_3| + |v_5| + |v_4| \leq q.$$

Both variants generalize the classical pumping lemma.

Corollary 3.3 Assume (LP) . Let G be a CF grammar. Then $L = L_\bullet(G)$ is infinite iff there exists $z \in L$ such that $m < |z| \leq m + q$, where m and q are the numbers from Proposition 4.

Proof:

Let $z \in L$ such that $m < |z| \leq m + q$. According to the pumping lemma, $z \in v_1 \bullet v_3 \bullet v_5 \bullet v_4 \bullet v_2$ and, for all $i \in \mathbb{N}$, $v_1 \bullet (v_3)^{\bullet i} \bullet v_5 \bullet (v_4)^{\bullet i} \bullet v_2 \subseteq L$. In addition, we know that $v_3 \neq \lambda$ or $v_4 \neq \lambda$, hence, because of (LP) , the $v_1 \bullet (v_3)^{\bullet i} \bullet v_5 \bullet (v_4)^{\bullet i} \bullet v_2$ -s are nonvoid and mutually disjoint. Thus, L is infinite.

Conversely, suppose that L is infinite. So there exist $z \in L$, $|z| > m$. If $|z| \leq m + q$, the proof is finished. Otherwise, from the pumping lemma, $z \in v_1 \bullet v_3 \bullet v_5 \bullet v_4 \bullet v_2$, with $v_1 \bullet v_5 \bullet v_2 \subseteq L$ and $|v_3| + |v_5| + |v_4| \leq q$; hence also $|v_3| + |v_4| \leq q$. Take $x \in v_1 \bullet v_5 \bullet v_2$. We have $|x| = |v_1| + |v_5| + |v_2| > m + q - q = m$; we continue the above reasoning with x , a word from L strictly shorter than z (because $v_3 \neq \lambda$ or $v_4 \neq \lambda$).

q.e.d.

Corollary 3.4 Assume (LP) and (R) . Then it is decidable whether a \bullet -CF language is infinite or not.

Proof:

The numbers m and q from the pumping lemma are obviously computable. Because (LP) implies (SL) and (SE) , by Corollary 2, we can decide, for each word with length between m and $m + q$, whether it is in L . If there is at least one, the language is infinite, otherwise it is finite.

q.e.d.

4 \bullet -regular languages

Remark 4.1 Regarding the elimination of λ -productions, one can easily see that the transformation from Proposition 2 does not change the presumptive regularity of the grammar G : if G is regular, then so is G' . Thus, regular versions of Proposition 2 and Remark 2 also hold.

Making an adaptation of the proof of the pumping lemma for \bullet -CF, we obtain the corresponding result for \bullet -regular languages:

Proposition 4.1 Assume (E) . Let L be a \bullet -regular language. Then there exists the natural number p such that, for each $w \in L$ with $|w| > p$, there exist x, y, z such that:

- (1) $w \in x \bullet y \bullet z$;
- (2) $0 < |y| \leq p$;
- (3) $(\forall i \in \mathbb{N}) x \bullet y^{\bullet i} \bullet z \subseteq L$;

Proof:

We look at the proof of the pumping lemma for \bullet -CF languages and use the same notations as there. We take p to be the maximum of m and q .

Now, we notice that for a rewrite tree in a regular grammar that has k non-leaf vertexes,

- $k - 1$ of them have two sons, the left one being a single vertex tree labeled by a terminal word,
- one of them has only one son, a single vertex tree labeled by a terminal word.

This is the reason why $v_2 = v_4 = \lambda$, so $v_3 \neq \lambda$. Since $|v_3 \bullet v_5| \leq p$, by (E), it follows that $|v_3| \leq p$. Now, in order to get the desired result, we take $x = v_1$, $y = v_3$ and $z = v_5$.

q.e.d.

Corollary 4.1 Assume (SE). Let L be a \bullet -regular language. Then $L \neq \emptyset$ iff there exists $w \in L$ with $|w| \leq p$, where p is the number from Proposition 5.

Proof:

One direction is obvious.

For the other, suppose $w \in L$. If $|w| \leq p$, the proof is ready. Otherwise, $w \in x \bullet y \bullet z$, with $y \neq \lambda$ and $x \bullet z \subseteq L$. Take $w' \in x \bullet z$. From (SE), $|w'| < |w|$ and we repeat the above reasoning with w' .

q.e.d.

Corollary 4.2 Assume (R) and (SE). Then it is decidable whether a \bullet -regular language is empty or not.

Proof:

Let L be a \bullet -regular language. Then we can compute the number p from the pumping lemma. By Corollary 2, we can determine the words from L of length $\leq p$. If we have not found any, by Corollary 5, the language is empty.

q.e.d.

5 Normal forms

Next, we assume \bullet to be a deterministic operation, that is an associative binary operation on Σ^* , with identity λ .

An *atom* of Σ^* is a word w such that there does not exist $x, y \in \Sigma^+$ with $w = x \bullet y$.

The following propositions regarding reduction to normal forms also hold. The reductions are performed very similarly to the classical ones.

Proposition 5.1 (Chomsky normal form) Assume (SE).

I. Let G be a CF grammar. Then there exists a \bullet -equivalent CF grammar G' such that all its productions are of the form $A \rightarrow BC$ or $A \rightarrow w$, where A, B, C are nonterminals and w is a terminal atom word.

II. Let G be a regular grammar. Then there exists a \bullet -equivalent regular grammar G' such that all its productions are of the form $A \rightarrow wB$ or $A \rightarrow w$, where A, B are nonterminals and w is a terminal atom word.

Proposition 5.2 (Greibach normal form) Let G be a CF grammar. Then there exists a \bullet -equivalent CF grammar G' such that all its productions have the form $A \rightarrow A_1 \dots A_n$ (with $A \neq A_1$) or $A \rightarrow w$, where A, A_1, \dots, A_n are nonterminals and w is a terminal atom word.

6 Concluding remarks

As shown in this paper, many properties regarding the structure or decision of languages generated by CF or regular languages are not committed to a particular catenation used in the generating process, but only to some very general properties like associativity or constant length of the result words, naturally related to the above mentioned phenomenon of intertextuality. The great generality of most of the results also resides in the fact that we allowed the catenation to be non-deterministic, capturing operations like shuffle. On the other hand, the fact that we didn't leave the field of formal languages (the abstract operation \bullet is still on words) enabled us treat some specific issues like pumping lemmas or decidability.

Intertextuality is still to be modeled and studied. Other generating (e.g. contextual grammars) or accepting structures tend to behave like the ones discussed here, only asking a few properties for catenation. This would be an area for future research.

References

- [1] P. Delany, G.P. Landow (eds.), "Hypermedia and Literary Studies", *The MIT Press*, Cambridge, Mass. and London, England, 1991.
- [2] M. Kudlek, S. Marcus, A. Mateescu, "Contextual Grammars with Distributed Catenation and Shuffle", *TUCS Technical Report*, 103, 1997.
- [3] M. Kudlek, A. Mateescu, "Distributed Catenation and Chomsky Hierarchy", *Lecture Notes in Computer Science*, Springer-Verlag, 965, 313-322, 1995.
- [4] M. Kudlek, A. Mateescu, "Rational and Algebraic Languages with Distributed Catenation", *Developments in Language Theory II*, eds. J. Dassow, G. Rozenberg and A. Salomaa, World Scientific, Singapore, 129-138, 1996.
- [5] M. Kudlek, A. Mateescu, "Rational, Linear and Algebraic Languages with Mix Operation", *TUCS Technical Report*, 105, 1997.
- [6] M. Kudlek, A. Mateescu, "On distributed catenation", *Report Fac. Comp. Sci.*, Hamburg, FBI-Bericht, 182/95, 1995.
- [7] S. Marcus, *Algebraic Linguistics; Analytical Models*, Academic Press, New York and London, 1967.
- [8] A. Mateescu, "On (left) Partial shuffle". *Proceedings of Results and Trends in Theoretical Computer Science, LNCS*, Springer, 812, 264-278, 1994.
- [9] A. Salomaa, *Formal Languages*, Academic Press, New York, London, 1973.