Power of controlled insertion and deletion

Lila Kari
Academy of Finland and Department of Mathematics[1]
University of Turku
20500 Turku
Finland

Abstract

The paper investigates classes of languages obtained as the closure of certain atomic languages under some insertion and deletion operations. Each of the classes studied is closed under an insertion operation, a deletion operation and an iterative insertion one. The operations are controlled and have been chosen as stated in order to allow an increase as well as a decrease of the length of the words in the operands. The iterative operation has been included in each class to provide an infinite growth of the strings.

1 Introduction

Language operations have been studied intensively in formal language theory. One of the main goals of the theory is to represent a family of languages as the closure of some atomic languages with respect to some operations. The theory of abstract families of languages (AFL) deals with operations, many operations appear in formal language theory applications, and so on.

This paper analyzes the generative power of some insertion and deletion operations defined in [4]. The operations are generalizations of catenation and left/right quotient and among them we can list *insertion, parallel insertion, controlled insertion, scattered insertion, deletion, parallel deletion, controlled deletion, scattered deletion*. For a comprehensive study of these and other operations as well as of various related topics see [4], [5], [6], [7], [8], [10].

The operations needed in our study are defined in Section 2 and some closure properties necessary for our investigation are proved in Section 3.

In Section 4 we focus on classes of languages which contain the singleton letters, the empty word, the empty set and are closed under an insertion operation,

[1]The work reported here is part of the project 11281 of the Academy of Finland

a deletion operation and an iterative insertion one. Finally, the mirror image operator and the union with the empty word are added for technical reasons.

Our results have the following uniform structure. The classes obtained properly contain the family of regular languages. Moreover, the class whose operations are sequential is properly included in the family of context-free languages. The one whose operations are parallel is included in the family of context-sensitive languages and contains non-context-free languages.

Let Σ be a finite alphabet and Σ^* the set of all words over Σ, including the empty word λ. The length of a word $w \in \Sigma^*$ is denoted by $\lg(w)$. The left quotient of a word u by a word v is defined by "$v\backslash u = w$ iff $u = vw$", and the right quotient u/v is defined analogously. The mirror image of a word u is denoted by $\mathrm{Mi}(u)$. For two languages L_1, L_2 over Σ,

$$L_1 - L_2 = \{u |\ u \in L_1 \text{ and } u \notin L_2\},\ L_1^c = \Sigma^* - L_1.$$

In the sequel REG, CF, CS will denote the family of regular, context-free and context-sensitive languages respectively. For unexplained formal language notions the reader is referred to [9].

2 Controlled insertion and deletion

The simplest and most natural generalization of catenation is the *sequential insertion* (see [4]). Given two words u and v, instead of catenating v at the right extremity of u, the new operation inserts v in an arbitrary place in u. Notice that the sequential insertion of two words is a finite set of words and that their catenation is an element of this set. However, catenation cannot be obtained as a particular case of sequential insertion because we cannot force the insertion to take place at the right extremity of the word. This brings up the notion of *control*: each letter determines what can be inserted after it. The catenation operation will then be obtained by using a particular case of *controlled insertion*.

Let L be a language over the alphabet Σ. For each letter a of the alphabet, let $\Delta(a)$ be a language over an alphabet Σ_a. The Δ- controlled insertion into L (shortly, *insertion*), is defined as:

$$L \leftarrow\!\!\dashv \Delta = \bigcup_{u \in L} (u \leftarrow\!\!\dashv \Delta),\ \text{where } u \leftarrow\!\!\dashv \Delta = \{u_1 a v_a u_2 |\ u = u_1 a u_2, v_a \in \Delta(a)\}.$$

The function $\Delta : \Sigma \longrightarrow 2^{\Sigma'^*}$, where $\Sigma' = \bigcup_{a \in \Sigma} \Sigma_a$ is called a *control function*.

The insertion is an $(n+1)$-ary operation, where n is the arity of Σ, the domain of the control function.

The catenation can be now expressed as $L_1 L_2 = h(L_1 \# \leftarrow\!\!\dashv \Delta)$, where $\Delta(\#) = L_2$, $\Delta(a) = \emptyset$, $\forall a \in \Sigma$ and h is the morphism defined by $h(\#) = \lambda$, $h(a) = a$, $\forall a \in \Sigma$.

Note that the empty word does not occur in the result of the insertion. Indeed, the notion of control implies the presence of at least one letter in the word in which the insertion is performed. Therefore we have

$$(L_1 \leftarrow\!\!\dashv \Delta) = (L_1 - \{\lambda\}) \leftarrow\!\!\dashv \Delta.$$

In the following we shall define a parallel variant of insertion.

Let L be a language over Σ and $\Delta : \Sigma \longrightarrow 2^{\Sigma'^*}$ a control function satisfying $\Delta(a) \neq \emptyset$, $\forall a \in \Sigma$. The Δ-controlled parallel insertion into L (shortly, *parallel insertion*) is defined as:

$$L \Leftarrow\!\!\dashv \Delta = \bigcup_{u \in L} (u \Leftarrow\!\!\dashv \Delta), \text{ where}$$

$$u \Leftarrow\!\!\dashv \Delta = \{a_1 v_1 a_2 v_2 \dots a_k v_k |\ u = a_1 \dots a_k, k \geq 1, a_i \in \Sigma,$$
$$\text{and } v_i \in \Delta(a_i), 1 \leq i \leq k\}.$$

For example, if $L = \{cd, \lambda, bdc, f^2\}$ and Δ is the control function defined by $\Delta(c) = \{a\}$, $\Delta(d) = \{\lambda, e\}$, $\Delta(b) = \{\lambda\}$, $\Delta(f) = \{f\}$ then,

$$L \Leftarrow\!\!\dashv \Delta = \{cad, cade, bdca, bdeca, f^4\}.$$

Note that in the previous definition the control function cannot have the empty set as its value. This condition has been introduced because of the following reasons. If there would exist a letter $a \in \Sigma$ such that $\Delta(a) = \emptyset$, then all the words $u \in L$ which contain a would give $u \Leftarrow\!\!\dashv \Delta = \emptyset$. This means that these words would not contribute to the result of the parallel insertion. Consequently, we can introduce, without loss of generality, the condition $\Delta(a) \neq \emptyset$, $\forall a \in \Sigma$.

For each of the above mentioned variants of insertion, a "dual" deletion operation can be also considered. Let L be a language over the alphabet Σ. For each letter a of the alphabet, let $\Delta(a)$ be a language over Σ. The Δ-controlled deletion from L (shortly, *deletion*) is defined as:

$$L \longmapsto \Delta = \bigcup\nolimits_{u \in L} (u \longmapsto \Delta), \text{ where}$$

$$u \longmapsto \Delta = \begin{array}{l} \{u_1 a u_2 \in \Sigma^* |\ u = u_1 a v u_2 \text{ for some} \\ u_1, u_2 \in \Sigma^*, a \in \Sigma \text{ and } v \in \Delta(a)\}. \end{array}$$

The function $\Delta : \Sigma \longrightarrow 2^{\Sigma^*}$ is called a control function.

For example, if $L = \{abba, aab, bba, aabb\}$ and Δ is the control function $\Delta(a) = b$, $\Delta(b) = a$, then $L \longmapsto \Delta = \{aba, abb, aa, bb, aab\}$. As a language operation, the Δ-controlled deletion has the arity $\text{card}(\Sigma) + 1$.

Notice that if λ belongs to L, λ does not contribute to the result of the deletion:

$$L \longmapsto \Delta = (L - \{\lambda\}) \longmapsto \Delta, \forall\ L \subseteq \Sigma^*, \Delta : \Sigma \longrightarrow 2^{\Sigma^*}.$$

A parallel variant of the deletion will be defined in the sequel. Let $u \in \Sigma^*$ be a word and $\Delta : \Sigma \longrightarrow 2^{\Sigma^*}$ be a control function which does not have $\emptyset$ as its value. The set $u \models\!\!\!\models \Delta$ is obtained by finding all the non-overlapping occurrences of av_a, $v_a \in \Delta(a)$, in u, and by deleting v_a from them. Here v_a may be different for different occurrences of a in the same word. Between any two occurrences of words of the type av_a, $v_a \in \Delta(a)$, in u, no other words of this type may remain.

Let L be a language over an alphabet Σ and $\Delta : \Sigma \longrightarrow 2^{\Sigma^*}$ be a control function such that $\Delta(a) \neq \emptyset$, $\forall a \in \Sigma$. The Δ- controlled parallel deletion from L (shortly, *parallel deletion*) is formally defined as:

$$L \models\!\!\!\models \Delta = \bigcup_{u \in L} (u \models\!\!\!\models \Delta), \text{ where}$$

$$\begin{aligned} u \models\!\!\!\models \Delta = \quad & \{u_1 a_1 u_2 a_2 \dots u_k a_k u_{k+1} |\ k \geq 1, a_j \in \Sigma, 1 \leq j \leq k, \\ & u_i \in \Sigma^*, 1 \leq i \leq k+1, \text{ and there exist } v_i \in \Delta(a_i), 1 \leq i \leq k, \\ & \text{such that } u = u_1 a_1 v_1 \dots u_k a_k v_k u_{k+1}, \text{where} \\ & \{u_i\} \cap \Sigma^* (\cup_{a \in \Sigma} a \Delta(a)) \Sigma^* = \emptyset,\ 1 \leq i \leq k+1.\} \end{aligned}$$

The last line is a formalization of the condition that no word av_a, $v_a \in \Delta(a)$, may occur in u between $a_i v_i$, $1 \leq i \leq k$, $v_i \in \Delta(a_i)$.

For example, if $L = \{abababa, a^3b^3, abab\}$ and $\Delta(a) = b$, $\Delta(b) = a$, then

$$L \models\!\!\!\models \Delta = \{a^4, ab^3, a^2b^2, ab^2a^2, a^3b, a^3b^2, a^2, ab^2\}.$$

The arity of the Δ-controlled parallel deletion is $\text{card}(\Sigma) + 1$.

As in the case of deletion, if the empty word belongs to L, this does not influence the result of the parallel deletion:

$$L \models\!\!\!\models \Delta = (L - \{\lambda\}) \models\!\!\!\models \Delta,\ \forall L \subseteq \Sigma^*, \Delta : \Sigma \longrightarrow 2^{\Sigma^*}, \Delta(a) \neq \emptyset, \forall a \in \Sigma.$$

3 Closure properties

This section contains some closure properties of the families of the Chomsky hierarchy under iterated insertion and iterated parallel insertion. These closure properties will later be used in establishing the position of the investigated classes of languages relative to the Chomsky hierarchy.

Let L be a language over Σ and $\Delta : \Sigma \longrightarrow 2^{\Sigma'^*}$ be a control function. The Δ-controlled insertion of order k into L is defined as

$$\begin{aligned} L \longleftarrow\!\!\!\dashv {}^0 \Delta &= L, \\ L \longleftarrow\!\!\!\dashv {}^{k+1} \Delta &= (L \longleftarrow\!\!\!\dashv {}^k \Delta) \longleftarrow\!\!\!\dashv \Delta,\ k \geq 0. \end{aligned}$$

The Δ-controlled iterated insertion into L (shortly, *iterated insertion*) is then defined as

$$L \longleftarrow\!\!\!\dashv {}^* \Delta = \bigcup_{k=0}^{\infty} (L \longleftarrow\!\!\!\dashv {}^k \Delta).$$

If the control function is $\Delta : \Sigma \longrightarrow 2^{\Sigma'^*}$ and $\Sigma' - \Sigma \neq \emptyset$, we put $\Delta(a) = \emptyset$ for $a \in \Sigma' - \Sigma$.

Our first results show that the family of regular languages is not closed under iterated insertion, whereas the families of context-free and context-sensitive languages are closed under it.

Proposition 1 *The family of regular languages is not closed under iterated insertion.*

Proof. Take $L = \{ab\}$ and the control function Δ defined by:

$$\Delta : \{a, b\} \longrightarrow 2^{\{a,b\}^*}, \Delta(a) = \Delta(b) = \{ab\}.$$

Then $L = \{ab\} \leftarrowtail^* \Delta$ equals the Dyck language of order one, which is not a regular language. □

Proposition 2 *The family of context-free languages is closed under iterated insertion.*

Proof. Let L be a language generated by the context-free grammar $G = (N, \Sigma, S, P)$ and $\Delta : \Sigma \longrightarrow 2^{\Sigma'^*}$ be a control function. The fact whether or not $\lambda \in L$ is irrelevant to the result of the insertion into L. Therefore, if $\lambda \in L$ then $L \leftarrowtail^* \Delta = [(L - \{\lambda\}) \leftarrowtail^* \Delta] \cup \{\lambda\}$. Consequently we can assume, without loss of generality, that L is a λ-free language. Assume that for every $a \in \Sigma$, the language $\Delta(a)$ is generated by the context-free grammar $G_a = (N_a, \Sigma_a, S_a, P_a)$, that the nonterminal sets N, N_a, $a \in \Sigma$, are pairwise disjoint and $\Sigma' = \cup_{a \in \Sigma} \Sigma_a$. Assume further that the grammar G satisfies the following properties (see [9], pp.55-56): *(i)* S does not appear on the right side of any production of P, *(ii)* if $\lambda \in L(G)$ the only production of P with λ as the right side is $S \longrightarrow \lambda$, *(iii)* all the productions of P (except eventually $S \longrightarrow \lambda$) are of the form $A \longrightarrow BC$, $A \longrightarrow a$, $A, B, C \in N$, $a \in \Sigma$. Assume that also the grammars G_a, $a \in \Sigma$ satisfy similar properties.

Construct the context-free grammar:

$$\begin{aligned} G' = \; & (N', \Sigma \cup \Sigma', S, P'), \\ N' = \; & N \cup (\cup_{a \in \Sigma} N_a), \\ P' = \; & P \cup (\cup_{a \in \Sigma} P_a) \cup \\ & \{A \longrightarrow aS_a |\; A \in N', a \in \Sigma, A \longrightarrow a \in P \cup (\cup_{a \in \Sigma} P_a)\}. \end{aligned}$$

It is not difficult to prove that $L(G') = L \leftarrowtail^* \Delta$. □

Proposition 3 *The family of context-sensitive languages is closed under iterated insertion.*

Proof. Let L be a language generated by the context-sensitive grammar $G = (N, \Sigma, S, P)$ and $\Delta : \Sigma \longrightarrow 2^{\Sigma'^*}$ be a control function. The fact whether or

not $\lambda \in L$ is irrelevant to the result of the insertion into L. Therefore, if $\lambda \in L$ then $L{\leftarrow\!\!\dashv}^*\Delta = [(L - \{\lambda\}){\leftarrow\!\!\dashv}^*\Delta] \cup \{\lambda\}$. Consequently we can assume, without loss of generality, that L is a λ-free language. Assume that, for every $a \in \Sigma$, the language $\Delta(a)$ is generated by the context-sensitive grammar $G_a = (N_a, \Sigma_a, S_a, P_a)$, that the nonterminal sets N, N_a, $a \in \Sigma$, are pairwise disjoint and $\Sigma' = \cup_{a\in\Sigma}\Sigma_a$. Assume further that the grammar G satisfies the properties *(i)* and *(ii)* from Proposition 2 together with *(iii)* every rule of P containing a terminal letter is of the form $A \longrightarrow a$ where $A \in N$ and $a \in \Sigma$ (see [9], pp.19-20). Assume that also all G_a, $a \in \Sigma$ satisfy similar properties.

Construct the context-sensitive grammar:

$$\begin{aligned} G' = &\quad (N', \Sigma \cup \Sigma', S, P'), \\ N' = &\quad N \cup (\cup_{a\in\Sigma} N_a) \cup \{\#\}, \\ P' = &\quad P \cup (\cup_{a\in\Sigma}(P_a - \{S_a \longrightarrow \lambda\})) \cup \\ &\quad \{A \longrightarrow a\#S_a\# \mid A \in N', a \in \Sigma, A \longrightarrow a \in P \cup (\cup_{a\in\Sigma} P_a)\}. \end{aligned}$$

Define now the morphism $h : \Sigma'^* \longrightarrow \Sigma'^*$ by $h(\#) = \lambda$, $h(a) = a$ for all $a \neq \#$. It is not difficult to show that

$$h(L(G')) = L{\leftarrow\!\!\dashv}^*\Delta,$$

and that h is a 3-linear erasing with respect to $L(G')$. We conclude that CS is closed under iterated insertion. □

The *iterated parallel insertion* can be defined starting from the parallel insertion. The formal definition can be obtained by replacing in the definition of iterated insertion $\leftarrow\!\!\dashv$ with $\Leftarrow\!\!\dashv$. Observe, however, that the iterated parallel insertion can be defined only when the control function Δ, defined on Σ, has as values languages over the same alphabet Σ.

The following results show that the family of regular and of context-free languages are not closed under iterated parallel insertion but the family of context-sensitive languages is closed under it.

Proposition 4 *The family of regular and the family of context-free languages are not closed under iterated parallel insertion.*

Proof. Take $L = \{a\}$ and the control function Δ defined by $\Delta(a) = a$. Then, $L{\Leftarrow\!\!\dashv}^*\Delta = \{a^{2^n} \mid n \geq 0\}$, which is not a context-free language. □

Proposition 5 *The family of context-sensitive languages is closed under iterated parallel insertion.*

Proof. Let L be a language generated by the context-sensitive grammar $G = (N, \Sigma, S, P)$, and let $\Delta : \Sigma \longrightarrow 2^{\Sigma^*}$ be a control function, $\Delta(a) \neq \emptyset$, $\forall a \in \Sigma$. Assume that, for every $a \in \Sigma$, the language $\Delta(a)$ is generated by the context-sensitive grammar $G_a = (N_a, \Sigma_a, S_a, P_a)$, that the nonterminal sets N, N_a, $a \in$

Σ are pairwise disjoint and that $\cup_{a\in\Sigma}\Sigma_a \subseteq \Sigma$. Assume further that all the above grammars satisfy the requirements of Proposition 3. The fact whether or not $\lambda \in L$ does not affect the result of parallel insertion into L. If λ belongs to L then $L \Leftarrow\!\!\!| \, ^*\Delta = [(L - \{\lambda\}) \Leftarrow\!\!\!| \, ^*\Delta] \cup \{\lambda\}$.Consequently we will assume, without loss of generality, that L is λ-free.

We can construct then the following grammar:

$$\begin{aligned}
G' = \;& (N', \Sigma', S', P'),\\
\Sigma' = \;& \Sigma \cup \{\$, \#\},\\
N' = \;& N \cup (\cup_{a\in\Sigma} N_a) \cup \{S', X\},\\
P' = \;& P \cup (\cup_{a\in\Sigma}(P_a - \{S_a \longrightarrow \lambda\}))\cup\\
& \{S' \longrightarrow XS', S' \longrightarrow \$S\#, X\$ \longrightarrow \$X\}\cup\\
& \{Xa \longrightarrow aX |\ a \in \Sigma, \lambda \in \Delta(a)\}\cup\\
& \{Xa \longrightarrow aS_aX |\ a \in \Sigma\}\cup\\
& \{Xa\# \longrightarrow aS_a\# |\ a \in \Sigma\}\cup\\
& \{Xa\# \longrightarrow a\# |\ a \in \Sigma, \lambda \in \Delta(a)\},
\end{aligned}$$

where $S', X, \$, \#$ are new symbols which do not occur in any of the given alphabets.

Intuitively, the grammar G' works as follows. First, a sentential form of the type $X^n\$w\#$, $w \in L$ is generated, where n represents the number of parallel iterations that will be made into w. X starts to move to the right. When crossing a letter a, it generates at its left a start symbol of $\Delta(a)$. The rules $S_a \longrightarrow \lambda$ are never needed. When X reaches the right extremity of the sentential form, it disapears.

The language $L(G')$ is context-sensitive. Indeed, all rules of G' except the ones of the form $Xa\# \longrightarrow a\#$ are length-increasing. However, the application of such a rule during a minimal derivation of a word $\alpha \in L(G)$ is always preceded by the application of a rule $Xa \longrightarrow aS_aX$. If this wouldn't be the case, our X would represent a dummy iteration step, in which all the inserted words are empty. This would further imply that the derivation for α is not minimal, as α could be obtained with a shorter derivation where the dummy iteration step is omitted.

The rule $Xa \longrightarrow aS_aX$ increases the length of the sentential form by one and the rule $Xa\# \longrightarrow a\#$ decreases its length by one. Combining these observations we conclude that the longest sentential form in a terminal derivation of α has the length smaller than or equal to $\lg(\alpha) + 1$.

Consequently, for all words $\alpha \in L(G')$ the workspace of α is smaller than $2\lg(\alpha)$ and, according to the workspace theorem (see, for example [9], pp.93-97), $L(G')$ is a context-sensitive language.

If we consider now the morphism $h : (\Sigma \cup \{\$, \#\})^* \longrightarrow \Sigma^*$, defined by $h(\$) = h(\#) = \lambda$ and $h(a) = a$ for $a \in \Sigma$ it can be proved that $h(L(G')) = L \Leftarrow\!\!\!| \, ^*\Delta$. Clearly, h is 3-linear erasing with respect to the language $L(G')$. □

4 Power of operations

Let $\mathcal{S}$ be the smallest class of languages which contains $\emptyset$, the language $\{\lambda\}$, the singleton letters and is closed under union with the empty word, mirror image, insertion, iterated insertion and deletion with singletons. The union with lambda has been added because λ cannot occur in the result of insertion and deletion. If this operation wouldn't have been used, the class $\mathcal{S}$ would not contain any language L with $\lambda \in L$, except $\{\lambda\}$.

The following result should be compared with the characterization of REG as the smallest class containing singleton letters, the empty set and closed under union, catenation and catenation closure.

Theorem 1 *$\mathcal{S}$ is contained in the family of context-free languages and properly contains the family of regular languages.*

Proof. In order to show that REG $\subseteq \mathcal{S}$ we will prove the closure of $\mathcal{S}$ under catenation, union and catenation closure.

Catenation. Let L_1, L_2 be two languages in $\mathcal{S}$, over the alphabet Σ. If $\#$ is a new symbol which does not belong to Σ, let Δ_1, Δ_2 be the control functions:

$$\begin{array}{ll} \Delta_1 : \{\#\} \longrightarrow 2^{\Sigma^*}, & \Delta_2 : \Sigma \cup \{\#\} \longrightarrow 2^{\Sigma^*}, \\ \Delta_1(\#) = L_2, & \Delta_2(\#) = L_1,\ \Delta_2(a) = \emptyset, \forall a \in \Sigma. \end{array}$$

The following equality holds:

$$\#L_1L_2 = (\{\#\} \longleftarrow\!\!\!\shortmid\ \Delta_1) \longleftarrow\!\!\!\shortmid\ \Delta_2.$$

The Δ_1-controlled insertion performs the task of catenating the symbol $\#$ and the language L_2. The Δ_2-controlled insertion inserts the language L_1 in the language $\#L_2$, at the right of $\#$, realizing thus the catenation $\#L_1L_2$.

If we define now the control function:

$$\Delta_3 : \Sigma \cup \{\#\} \longrightarrow 2^{(\Sigma \cup \{\#\})^*}, \Delta_3(\#) = \emptyset, \Delta_3(a) = \#, \forall a \in \Sigma,$$

we have that:

$$\begin{array}{lll} L_1L_2 = & \mathrm{Mi}(\mathrm{Mi}(\#L_1L_2) \longmapsto \Delta_3), & \text{if } \lambda \notin L_1 \cap L_2, \\ L_1L_2 = & \mathrm{Mi}(\mathrm{Mi}(\#L_1L_2) \longmapsto \Delta_3) \cup \{\lambda\}, & \text{if } \lambda \in L_1 \cap L_2. \end{array}$$

The role of the Δ_3-controlled deletion is to delete the symbol $\#$ in every word of $\mathrm{Mi}(\#L_1L_2)$. This operation could be performed only after Mi transferred the symbol $\#$ to the right extremity of the words. This transfer was needed because the first letter of a word cannot be erased by deletion. Finally, Mi was used again, in order to obtain the desired language from its mirror image.

The catenation L_1L_2 has been obtained from $L_1, L_2, \{\#\}, \emptyset \in \mathcal{S}$ by using the operations union with lambda, mirror image, insertion and deletion with singletons. Therefore, the class $\mathcal{S}$ is closed under catenation.

Union. We will show first that the union of two letters is a language belonging to $\mathcal{S}$. Indeed, let $\{a\}, \{b\}$ be two singleton letters. Let $\#$ be a letter different from a and b and define the control function Δ_4 by:

$$\Delta_4 : \{a, b, \#\} \longrightarrow 2^{\{a,b,\#\}^*}, \Delta_4(\#) = a, \Delta_4(a) = b, \Delta_4(b) = \emptyset.$$

The following relation holds:

$$\{\#a, \#b\} = \{\#ab\} \mapsto \Delta_4.$$

The Δ_4- controlled deletion was used to obtain a set of two elements from a singleton. The additional symbol $\#$ was needed in order to make possible the deletion at the left extremity of the word ab.

If we define now the control function:

$$\Delta_5 : \{a, b, \#\} \longrightarrow 2^{\{a,b,\#\}^*}, \Delta_5(a) = \Delta_5(b) = \#, \Delta_5(\#) = \emptyset,$$

we obtain the requested set:

$$\{a, b\} = \mathrm{Mi}(\{\#a, \#b\}) \mapsto \Delta_5.$$

The role of the Δ_5-controlled deletion was to delete the symbol $\#$ and the mirror image transferred it to the right extremity of every word, to allow its deletion. Observe that another application of Mi is not needed.

As we have obtained the set $\{a, b\}$ starting from the sets $\emptyset$, $\#$, a, b and $\{\#ab\}$ (which belongs to $\mathcal{S}$ as $\mathcal{S}$ is closed under catenation) and applying only insertion, deletion with singletons and mirror image, we conclude that it belongs to $\mathcal{S}$.

Returning now to the general case, let L_1, L_2 be two languages in $\mathcal{S}$, over the alphabet Σ. Let $\#_1, \#_2$ be two symbols which do not occur in Σ and Δ_6, Δ_7 be the control functions:

$$\begin{array}{ll} \Delta_6 : \{\#_1, \#_2\} \longrightarrow 2^{\Sigma^*}, & \Delta_7 : \Sigma \cup \{\#_1, \#_2\} \longrightarrow 2^{\{\#_1,\#_2\}^*}, \\ \Delta_6(\#_1) = L_1, & \Delta_7(\#_1) = \#_2, \\ \Delta_6(\#_2) = L_2, & \Delta_7(\#_2) = \#_1, \Delta_7(a) = \emptyset, \forall a \in \Sigma. \end{array}$$

The following equality is now obvious:

$$\#_1\#_2L_1 \cup \#_2\#_1L_2 = (\{\#_1, \#_2\} \leftarrow\!\!\!| \, \Delta_6) \leftarrow\!\!\!| \, \Delta_7.$$

Indeed, the Δ_6-controlled insertion inserts L_1 after $\#_1$ and L_2 after $\#_2$, yielding thus $\#_1L_1 \cup \#_2L_2$. The Δ_7-controlled insertion inserts then $\#_2$ after $\#_1$ and $\#_1$ after $\#_2$.

If we further define the control functions :

$$\Delta_8 : \Sigma \cup \{\#_1, \#_2\} \longrightarrow 2^{(\Sigma \cup \{\#_1,\#_2\})^*}, \Delta_9 : \Sigma \cup \{\#_2\} \longrightarrow 2^{(\Sigma \cup \{\#_2\})^*},$$

$$\Delta_8(a) = \#_1, \forall a \in \Sigma \cup \{\#_2\}, \quad \Delta_9(a) = \#_2, \forall a \in \Sigma,$$
$$\Delta_8(\#_1) = \emptyset, \qquad \Delta_9(\#_2) = \emptyset,$$

then

$$\begin{aligned} L_1 \cup L_2 = \ & \mathrm{Mi}((\mathrm{Mi}(\#_1\#_2L_1 \cup \#_2\#_1L_2) \longmapsto \Delta_8) \longmapsto \Delta_9), \\ & \text{if } \lambda \notin L_1 \cup L_2, \\ L_1 \cup L_2 = \ & \mathrm{Mi}((\mathrm{Mi}(\#_1\#_2L_1 \cup \#_2\#_1L_2) \longmapsto \Delta_8) \longmapsto \Delta_9) \cup \{\lambda\}, \\ & \text{if } \lambda \in L_1 \cup L_2. \end{aligned}$$

The role of the Δ_8-controlled deletion was to erase the symbol $\#_1$ and that of the Δ_9-controlled deletion to erase $\#_2$. We needed two deletions because only deletion with singletons had to be used. The role of the mirror image operator has been similar as in the previous cases.

We have obtained $L_1 \cup L_2$ starting with the languages L_1, L_2, $\#_1$, $\#_2$, $\emptyset$ in $\mathcal{S}$ and with the set $\{\#_1, \#_2\}$ (which consists of two letters and therefore belongs to $\mathcal{S}$) by applying only insertion, mirror image, union with λ and deletion with singletons. Therefore $L_1 \cup L_2$ is in $\mathcal{S}$, that is, $\mathcal{S}$ is closed under union.

Catenation closure. Let L be a language in $\mathcal{S}$, over the alphabet Σ, and let $\#$ be a letter which does not belong to Σ. If Δ_{10} is the control function defined as:

$$\Delta_{10} : \Sigma \cup \{\#\} \longrightarrow 2^{\Sigma^*}, \Delta_{10}(\#) = L, \Delta_{10}(a) = \emptyset, \forall a \in \Sigma,$$

then

$$\#L^* = \{\#\} \longmapsto\!\!\!\!\!\!\leftarrow {}^*\Delta_{10}.$$

Indeed, the Δ_{10}-controlled insertion inserts words from L only to the right of $\#$, assuring that the insertion amounts to catenation. Defining finally the control function

$$\Delta_{11} : \Sigma \cup \{\#\} \longrightarrow 2^{(\Sigma \cup \{\#\})^*}, \Delta_{11}(a) = \#, \forall a \in \Sigma, \Delta_{11}(\#) = \emptyset,$$

the catenation closure of L will be

$$L^* = \mathrm{Mi}(\mathrm{Mi}(\#L^*) \longmapsto \Delta_{11}) \cup \{\lambda\}.$$

With the help of the mirror image operator, which puts $\#$ to the end of words, the Δ_{11}-controlled deletion erases the letter $\#$ from all the words in $\#L^*$. Finally, Mi restores the form of the words from L.

We have obtained L^* starting from $L, \emptyset$ and $\{\#\}$ in $\mathcal{S}$ and using iterated insertion, mirror image, union with λ and deletion with singletons. Therefore, $\mathcal{S}$ is closed under catenation closure.

As $\mathcal{S}$ contains the singleton letters and is closed under catenation, union and catenation closure, it follows that it contains all the regular languages. According to Proposition 1 the inclusion is proper.

The inclusion $\mathcal{S} \subseteq$ CF follows from the fact that CF is closed under mirror image, insertion, deletion with singletons (see [4]) and iterated insertion (see Proposition 2). □

In the following we will be able to prove that the inclusion $S \subseteq \mathrm{CF}$ is actually strict. For this, an auxiliary result is needed.

Lemma 1 (Pumping lemma) *For each language L in S there exists a natural n_0 such that every word $w \in L$ with $lg(w) > n_0$ possesses a decomposition $w = w_1 w_2 w_3$ satisfying:*

- $w_2 = w_2' w_2''$,
- *there exists a nonempty word u such that $w_1 w_2' u^k w_2'' w_3 \in L$ for all $k \geq 0$.*

Proof. The constant n_0 is defined by structural induction, based on the construction of L using the operations of S.

If L equals a singleton letter, the empty set or the empty word, we define $n_0 = 1$. The operations $\cup\{\lambda\}$, and Mi do not change n_0.

We now consider, in succession, $L \leftarrow\!\!\!| \, \Delta$, $L \leftarrow\!\!\!|^* \Delta$ and $L \mapsto \Delta$ and assume that L and $\Delta(a)$, $a \in \Sigma$, satisfy the lemma with the constants n, respectively n_a, $a \in \Sigma$.

Insertion. Take $n_0 = 2n + \max\{n_a | a \in \Sigma\} + 1$ and let w be a word in $L \leftarrow\!\!\!| \, \Delta$, $w = w_1 w_2 w_3$ with $\lg(w_2) > n_0$.

Being a result of the insertion, the word w is of the form $w = \alpha_1 a w_a \alpha_2$, where $w_a \in \Delta(a)$. One of the following cases holds:

(a) the word w_2 is a subword of α_1 or α_2 or of w_a. In this case the lemma holds as we can directly apply the induction hypothesis.

(b) the word w_2 is a subword of $\alpha_1 a$ and then we can apply the induction hypothesis for the part of w_2 which lies in α_1.

(c) we have that either

- the length of the portion of w_2 in $\alpha_1 a$ is greater than $n + 1$
- the length of the portion of w_2 in w_a is greater than n_a
- the length of the portion of w_2 in α_2 is greater than n.

(If none of these cases happens, the length of w_2 becomes smaller than or equal to n – a contradiction.) In all cases, by applying the induction hypohesis we are able to pump inside the portion of w_2 which lies inside α_1, w_a, respectively α_2.

Iterated insertion. Take $n_0 = 2n + \max\{n_a |\ a \in \Sigma\} + 1$ and a word w in $L \leftarrow\!\!\!|^* \Delta$ be of length greater than n_0. There exists a natural $k \geq 0$ such that $w \in L \leftarrow\!\!\!|^k \Delta$. We shall show that w satisfies the requirements of the lemma by induction on k, the number of iterations.

If $k = 0$ then $w \in L$ and the lemma holds.

Assume now that the lemma holds for words in $L \leftarrow\!\!\!|^k \Delta$ and let $w = w_1 w_2 w_3$ be a word in $L \leftarrow\!\!\!|^{k+1} \Delta$, of length greater then n_0. Then, w is of the form $w = \alpha_1 a w_a \alpha_2$ where $\alpha_1 a \alpha_2$ belongs to $L \leftarrow\!\!\!|^k \Delta$ and w_a to $\Delta(a)$.

If w_2 is a subword of α_1, w_a or α_2 then we can directly apply the induction hypothesis.

If w_2 is a subword of $\alpha_1 a$ we can apply the induction hypothesis for the portion of w_2 which lies in α_1.

If w_2 overlaps with w_a and $\alpha_1 a$ then we can choose $u = w_a$. The words obtained by pumping w_a near a will belong to the $L{\leftarrowtail}^*\Delta$ according to the definition of the iterated insertion.

If w_2 overlaps with w_a and α_2 then again we can choose $u = w_a$ (pump w_a between a and α_2).

Deletion. Take $n_0 = 2n+1$ and take $w = w_1 w_2 w_3$ a word in $L{\rightarrowtail}\Delta$ of length greater than n_0. Then there exists a word $w' \in L$, $w' = \alpha_1 a w_a \alpha_2$, $w_a \in \Delta(a)$ such that $w = \alpha_1 a \alpha_2$.

If w_2 is a subword of α_1 or of α_2, we are through.

If w_2 is a subword of $\alpha_1 a$ the we can apply the induction hypothesis to the portion of w_2 which lies in α_1.

Otherwise, either the overlapping portion of w_2 and α_1 is longer than n or the overlapping portion of w_2 and α_2 is longer than n. In both cases we can apply the induction hypothesis to the overlapping portion.

□

Languages in S are constructed from certain atomic ones by applying some operations. Consequently, to every language L in S, a formula describing the "construction history" of L can be associated. Such a formula is analogous to a regular expression. Observe that the definition of the constant in Lemma 1 is based on this formula. Analogous constants for regular languages are customarily defined in terms of the automaton, rather than in terms of the regular expression.

Proposition 6 *The family S is strictly included in the family of context-free languages.*

Proof. Consider the language $L = \{a^n b^n |\ n \geq 0\}$. L is a context-free language but it does not belong to the family S.

Indeed, if L would belong to S, there would exist a constant n_0 such that every word $w \in L$ with $\lg(w) > n_0$ possesses a decomposition satisfying the requirements of Lemma 1.

The word $u \in \{a, b\}^+$ from Lemma 1 that can be pumped into w, producing words that still belong to L, can be of one of the following forms:

- $u = a^r$, $r > 0$. In this case, by pumping u into w we obtain words in which the number of occurrences of a is arbitrarily large whereas the number of occurrences of b is constant – a contradiction with the form of words in L.
- $u = b^r$, $r > 0$. A similar argument proves that this case also leads to a contradiction.

- u contains both letters a and b. By pumping u into w we are able to produce words where the occurrences of a's and b's alternate arbitrarily many times – a contradiction with the definition of L.

As all the possible ways of choosing u led to contradictions, we conclude that our initial assumption that the language L belongs to S was false. □

Theorem 2 *The family $\mathcal{S}$ is not closed under intersection.*

Proof. We will prove that there exist two languages $L_1, L_2 \in \mathcal{S}$ whose intersection is not a context-free language. As, according to Theorem 1, $\mathcal{S} \subseteq$CF, this will imply that $\mathcal{S}$ is not closed under intersection.

Let L_1 be the language defined by:

$$L_1 = \{\#\} \leftarrowtail^* \Delta_1,$$

where Δ_1 is the control function $\Delta_1 : \{\#, a, b, d\} \longrightarrow 2^{\{\#,a,b,d\}^*}$ defined by:

$$\Delta_1(\#) = \{a\#b\#, b\#a\#, d\#\}, \Delta_1(a) = \Delta_1(b) = \Delta_1(d) = \emptyset.$$

Claim. $L_1 = \{w \in \#(\Sigma\#)^* |\ N_a(w) = N_b(w)\},$ where $\Sigma = \{a, b, d\}$.

" $\subseteq$ " This inclusion is obvious, as we insert an equal number of a's and b's at every iteration step.

" $\supseteq$ " We will show by induction on n that if w is a word in the right member of the equality satisfying $N_a(w) = N_b(w) = n$, then $w \in L_1$.

$n = 0$. Let $w = \#(d\#)^p$, where $p \geq 0$. Then we have:

$$w \in \{\#\} \leftarrowtail^p \Delta_1 \ \subseteq \{\#\} \leftarrowtail^* \Delta_1,$$

as w is obtained by p insertions of $d\#$ next to the first symbol $\#$.

$n \mapsto n + 1$. Assume the statement true for numbers up to n and let w be a word in $\#(\Sigma\#)^*$, containing $n + 1$ letters a and $n + 1$ letters b. The word w is of one of the forms:

$$\begin{aligned} w = &\quad \#\alpha a\#(d\#)^m b\#\beta, m \geq 0, \alpha, \beta \in (\Sigma\#)^*, \\ w = &\quad \#\alpha b\#(d\#)^m a\#\beta, m \geq 0, \alpha, \beta \in (\Sigma\#)^*. \end{aligned}$$

Assume that the first case holds, the other one being similar. Consider the word $w' = \#\alpha\beta$. According to the induction hypothesis, w' is a word in L_1. Therefore we have:

$$w \in \{w'\} \leftarrowtail^{m+1} \Delta_1 \ \subseteq \ \{\#\} \leftarrowtail^* \Delta_1.$$

Indeed, w is obtained from w' by inserting first $a\#b\#$ at the right extremity of α and then inserting m times $d\#$ at the right extremity of $a\#$. We conclude that $w \in L_1$ and the claim is proved.

If we define now the control function $\Delta_2 : \{\#, a, b, d\} \longrightarrow 2^{\{\#,a,b,d\}^*}$ by:

$$\Delta_2(\#) = \{d\#b\#, b\#d\#, a\#\}, \Delta_2(a) = \Delta_2(b) = \Delta_2(d) = \emptyset,$$

one can prove, as before, that:

$$L_2 = \{\#\} \leftarrow\!\!\!\shortmid\ ^* \Delta_2 = \{w \in \#(\Sigma\#)^* |\ N_b(w) = N_d(w)\}.$$

It is easy to show that:

$$L_1 \cap L_2 = \{w \in \#(\Sigma\#)^* |\ N_a(w) = N_b(w) = N_d(w)\},$$

which is not a context-free language. As L_1 and L_2 belong to $\mathcal{S} \subseteq$CF, it follows that $\mathcal{S}$ is not closed under intersection. □

The remaining part of this section will deal with similar questions concerning the generative power, in case all insertions and deletions are replaced with their parallel counterparts. The families thus obtained properly contain REG and are contained in CS. Their relation with the class of context-free languages as well as their relation with $\mathcal{S}$ remains open.

Let $\mathcal{P}$ be the smallest class of languages which contains the empty set, the language $\{\lambda\}$, the singleton letters and is closed under mirror image, union with λ, parallel insertion, iterated parallel insertion and parallel deletion with singletons.

Theorem 3 *$\mathcal{P}$ is contained in the family of context-sensitive languages and properly contains the family of regular languages.*

Proof. The fact that $\mathcal{P}$ contains REG can be shown using the proof of Theorem 1. The control functions that appear in the proof have the value $\emptyset$ for some arguments. However, in the case of parallel insertion (parallel deletion), the control function cannot have the empty set as its value. We will modify the functions as follows. Let \$ be a new symbol which does not occur in any of the alphabets used in Theorem 1. For every insertion and iterated insertion (resp. deletion) the control function gets the value λ (resp. the value \$) for all those letters for which it had previously the value $\emptyset$. After this change, one notices that if we replace everywhere in the proof the insertion, iterated insertion, deletion with singletons with parallel insertion, iterated parallel insertion, parallel deletion with singletons respectively, the same relations hold. This happens because, in all the cases occurring in the proof of Theorem 1, the parallel insertion or deletion will amount in fact to insertion or deletion.

According to Proposition 4, the inclusion REG$\subseteq \mathcal{P}$ is proper.

The inclusion of $\mathcal{P}$ in CS follows from the fact that CS is closed under mirror image, parallel insertion, parallel deletion with singletons (see [4]) and iterated parallel insertion (see Proposition 5). □

Let $\mathcal{P}'$ be the smallest class of languages containing the empty set, $\{\lambda\}$, the singleton letters and closed under mirror image, union with λ, parallel insertion, iterated parallel insertion and parallel deletion. The difference between $\mathcal{P}$ and $\mathcal{P}'$ is that, in the case of $\mathcal{P}$, the parallel deletion is restricted to the case where only singletons are erased.

Theorem 4 *$\mathcal{P}'$ is a Boolean algebra properly containing the family of regular languages.*

Proof. The family $\mathcal{P}$ is included in $\mathcal{P}'$, therefore $\mathcal{P}'$ properly contains the family of regular languages.

It will be showed in the following that $\mathcal{P}'$ is closed under complementation. Let L be a language in $\mathcal{P}'$, over the alphabet Σ, and let $\#, \$$ be letters which do not occur in Σ. Then,

$$\{\#\$\} \cup \#L^c\$\$ = \#\Sigma^*\$\$\# \models \Delta_1,$$

where Δ_1 is the control function:

$$\Delta_1 : \Sigma \cup \{\#, \$\} \longrightarrow 2^{(\Sigma \cup \{\#, \$\})^*}, \Delta_1(\#) = L\$, \Delta_1(a) = \Delta_1(\$) = \#, \forall a \in \Sigma.$$

Indeed, given a word $w = \#u\$\$\# \in \#\Sigma^*\$\$\#$, the Δ_1-controlled parallel deletion:

- if $u \in L$, erases both $u\$$ and the last $\#$, yielding $\#\$$;
- if $u \in \Sigma^* - L$, erases only the last $\#$, yielding $\#u\$\$$.

One can use the control function Δ_2 to erase the marker \$, where

$$\Delta_2 : \Sigma \cup \{\#, \$\} \longrightarrow 2^{(\Sigma \cup \{\$, \#\})^*},$$

$$\Delta_2(\#) = \Delta_2(\$) = \Delta_2(a) = \$\$, \forall a \in \Sigma.$$

Consequently we have:

$$\#L^c = (\{\#\$\} \cup \#L^c\$\$) \models \Delta_2.$$

Using now the control function Δ_3 to erase the marker $\#$:

$$\Delta_3 : \Sigma \cup \{\#\} \longrightarrow 2^{(\Sigma \cup \{\#\})^*}, \Delta_3(\#) = \Delta_3(a) = \#, \forall a \in \Sigma,$$

we obtain,

$$\begin{array}{lll} L^c = & \mathrm{Mi}(\mathrm{Mi}(\#L^c) \models \Delta_3), & \text{if } \lambda \in L, \\ L^c = & \mathrm{Mi}(\mathrm{Mi}(\#L^c) \models \Delta_3) \cup \{\lambda\}, & \text{if } \lambda \notin L. \end{array}$$

The language $\#\Sigma^*\$\$\#$, can be obtained from the singleton letters by using catenation and catenation closure. As, in order to obtain L^c, we have started from Σ, \$, $\#$ and L, and we have used only the operations of $\mathcal{P}'$, we deduce that $L^c \in \mathcal{P}'$. Being closed under complementation and union, $\mathcal{P}'$ is closed also under intersection. Consequently $\mathcal{P}'$ is a Boolean algebra. □

References

[1] R.V.Book, M.Jantzen, C.Wrathall. Monadic Thue systems. *Theoretical Computer Science*, **19**(1982), pp.231-251.

[2] M.Jantzen. Semi-Thue systems and generalized Church-Rosser properties. *Proc.Fete des Mots*, Rouen, France, 1982, pp.60-75.

[3] T.Kimura. Formal description of communication behaviour. *Proc. Johns Hopkins Conf. on Information Sciences and Systems* (1979).

[4] L.Kari. On insertion and deletion in formal languages. *Ph.D. Thesis*, University of Turku, 1991.

[5] L.Kari. Insertion and deletion of words: determinism and reversibility. *Lecture Notes in Computer Science*, vol.629, 1992, pp.315-327.

[6] L.Kari. Generalized derivatives. *Fundamenta Informaticae*, vol.18, nr.1, 1993, pp.27-40.

[7] L.Kari, A.Mateescu, Gh.Paun, A.Salomaa. Deletion sets. To appear in *Fundamenta Informaticae*.

[8] L.Kari, A.Mateescu, Gh.Paun, A.Salomaa. On parallel deletions applied to a word. To appear in *RAIRO*.

[9] A.Salomaa. *Formal Languages*. Academic Press, New York, 1973.

[10] L.Sântean. Six arithmetic-like operations on languages. *Revue Roumaine de Linguistique*, Tome XXXIII, 1988, Cahiers de linguistique theorique et applique, Tome XXV, 1988, No.1, Janvier-Juin, pp.65-73.