K-catenation and applications: k-prefix codes*

Lila Kari and Gabriel Thierrin
Department of Mathematics, University of Western Ontario,
London, Ontario, N6A 5B7, Canada

Abstract

Some generalizations of the usual operations of catenation and quotient of languages are introduced and studied. In particular, language equations involving these operations are considered. Infinite hierarchies of classes of prefix codes and classes of equivalence relations are defined and investigated in connection respectively with the classical notions of prefix codes and right syntactic congruences.

1 Introduction

The catenation and quotient are basic operations in formal language theory. A natural generalization of the catenation uw of the word w to u is obtained if we allow w to be inserted in an arbitrary position in u, instead of its right extremity. This and several other variants of insertion and deletion, together with related topics have recently been studied in [3], [4] [6].

Note that, eventhough insertion generalizes catenation, catenation cannot be obtained as a particular case of insertion. This happens because we cannot force the insertion to take place at the end of the word. The operation defined and studied in this paper provides the control needed to overcome this phenomenon. K-catenation is thus more nondeterministic than catenation, but more restrictive than insertion. When k-catenating w to u we obtain words u_1wu_2, where the length of u_2 is at most k. Remark that now, 0-catenation is exactly the classical catenation.

The paper is organized in the following way. Section 2 studies closure properties of the families of Chomsky hierarchy under k-catenation, and language equations involving the k-catenation operation. The k-quotient and k-deletion are introduced as operations inverse to k-catenation, needed in solving these equations.

*This research was supported by Grant OGP0007877 of the Natural Sciences and Engineering Research Council of Canada

The classical notions of right syntactic congruences and prefix codes in formal languages are related to catenation. Using k-catenation instead of catenation, these notions can be generalized in a natural way. Section 3 and 4 study some of these generalizations, called respectively right k-syntactic congruences and k-prefix codes.

In the following, Σ will denote a finite alphabet and Σ^* the monoid generated by it under the operation of catenation. For a word $u \in \Sigma^*$, $|u|$ denotes its length, and λ stands for the empty word. REG, CF and CS will denote the families of regular, context-free and context-sensitive languages respectively. For further undefined notions in formal language theory and theory of codes the reader is referred to [8], [7], respectively [1], [9].

2 Language equations and closure properties

Definition 1 *Let u, v be words over the alphabet Σ. The k-catenation of v into u is defined by $u[k]v = \{u_1 v u_2 |\ u = u_1 u_2, |u_2| \leq k\}$.*

The usual operation of catenation is the 0-catenation. Clearly, $u[k]v \subseteq u[k+1]v$. For $k > 0$, the k- catenation is not associative and not commmutative.

If $u \leftarrow v$ is the sequential insertion of v into u (see [5] for details), then $u \leftarrow v = \bigcup_{k \geq 0} u[k]v$.

Proposition 1 *The families REG, CF, CS are closed under k-catenation.*

Proof. Let Σ be an alphabet and $\#$ be a letter which does not occur in Σ. Consider the λ-free gsm $g = (\Sigma, \Sigma \cup \{\#\}, \{s_0, s\}, s_0, \{s\}, P)$, where

$$\begin{aligned} P \quad = \quad & \{s_0 a \longrightarrow a s_0 |\ a \in \Sigma\} \cup \{s_0 a \longrightarrow a \# s |\ a \in \Sigma\} \cup \\ & \{s a \longrightarrow a s |\ a \in \Sigma\} \cup \{s_0 a \longrightarrow \# a s |\ a \in \Sigma\}. \end{aligned}$$

Intuitively, the gsm g works as follows: given a nonempty word u as an input, g leaves its letters unchanged, but inserts a marker $\#$ in an arbitrary position in u. It is easy to see that for any language $L \subseteq \Sigma^+$ we have

$$L[k]\{\#\} = g(L) \cap \Sigma^* \{\#\} (\{\lambda\} \cup \Sigma \cup \Sigma^2 \cup \ldots \cup \Sigma^k)$$

Let now $L_1, L_2 \subseteq \Sigma^*$ be two languages belonging to REG, CF or CS. Consider the λ-free substitution $s : (\Sigma \cup \{\#\}^*) \longrightarrow 2^{\Sigma^*}$ defined by $s(\#) = L_2 - \{\lambda\}$, $s(a) = a, \forall a \in \Sigma$. The following relations hold:

$$L_1[k]L_2 = \begin{cases} s(L_1[k]\{\#\}) & \text{if} \quad \lambda \notin L_1 \cup L_2, \\ s(L_1[k]\{\#\}) \cup L_1 & \text{if} \quad \lambda \in L_2 - L_1, \\ s(L_1[k]\{\#\}) \cup L_2 & \text{if} \quad \lambda \in L_1 - L_2, \\ s(L_1[k]\{\#\}) \cup L_1 \cup L_2 & \text{if} \quad \lambda \in L_1 \cap L_2. \end{cases}$$

As the families REG, CF, CS are closed under λ-free substitutions, λ-free gsm and union, they are closed under k- catenation. □

Consider now equations of the type $X[k]L = \Re$, where $L, \Re$ are given languages. We recall the following known result (see, for example, [4]):

Proposition 2 *Let $L, \Re$ be languages over an alphabet Σ and $\diamond, \Box$ be two binary word (language) operations, left-inverses to each other. If the equation $X \diamond L = \Re$ has a solution $X \subseteq \Sigma^*$ then the language $\Re' = (\Re^c \Box L)^c$ is a maximal solution.*

Recall that the operation $\Box$ is said to be the *left-inverse* of the operation $\diamond$ if, for all words u, v, w over Σ, we have: $w \in (u \diamond v)$ *iff* $u \in (w \Box v)$ (see [4]). The relation "is the left-inverse of" is symmetric.

In order to find solutions to our equation, we need to find the left-inverse of the k-catenation.

Definition 2 *For two words $u, v \in \Sigma^*$ the k-quotient of v from u is*

$$u \rhd_k v = \{u_1 u_2 |\ u = u_1 v u_2, |u_2| \leq k\}.$$

For $k = 0$ we obtain the usual left quotient. Note that the k-quotient is the left-inverse of k- catenation.

Proposition 3 *If $\Re$ is a regular language and L an arbitrary one, then $\Re \rhd_k L$ is regular.*

Proof. Let $L, \Re$ be languages over Σ and let $A = (S, \Sigma, s_0, F, P)$ be a finite automaton that accepts $\Re$. For two states $s, s' \in S$ denote:

$$L_{s,s'} = \{w \in \Sigma^* |\ sw \Longrightarrow^* s' \text{ in } A\}.$$

The languages $L_{s,s'}$ are regular. Consider the automaton:

$$A' = (S, \Sigma \cup \{\#\}, s_0, F, P'),\ P' = P \cup \{s\# \longrightarrow s' |\ s, s' \in S \text{ and } L \cap L_{s,s'} \neq \emptyset\},$$

where $\#$ is a letter which does not occur in Σ.

Then we have $\Re \rhd_k L = h(L(A') \cap (\Sigma^* \# \cup \Sigma^* \# \Sigma \ldots \cup \Sigma^* \# \Sigma^k))$, where h is the morphism which erases the marker $\#$. The proposition now follows as REG is closed under intersection with regular languages and morphisms. □

As a consequence of the preceding Proposition, REG is closed under k- quotient.

Moreover, the language $\Re \rhd_k L$ can be effectively constructed if $\Re$ is regular and L is regular or context-free. Indeed, in this case, the emptiness of the intersection occurring in the definition of P' is decidable. Consequently, the automaton A' can be effectively constructed.

The fact that the automaton A is finite implies that there are finitely many possibilities of constructing A'. This leads to the conclusion that, for a given regular language, there are finitely many distinct languages that can be obtained from it by k- quotient.

Proposition 4 *The problem "Does there exist a solution X to the equation $X[k]L = \Re$?" is decidable for regular languages $\Re$ and L.*

Proof. Let $L, \Re$ be languages over the alphabet Σ and consider $\Re' = (\Re^c \rhd_k L)^c$. The language $\Re'$ is regular and can be effectively constructed. As the k-quotient is the left-inverse of k-catenation, if a solution to the equation exists, then $\Re'$ is a maximal solution (see Proposition 2).

The algorithm for deciding our problem will consist of constructing $\Re'$ and deciding whether or not $\Re'[k]L$ equals $\Re$. The last problem is decidable as REG is closed under k-catenation, the closure is effective (see Proposition 1), and the equivalence problem is decidable for regular languages. □

Corollary 1 *Let $\Re$ be a regular language. There exists a finite number $n \geq 1$ of distinct regular languages $\Re''_i$, $1 \leq i \leq n$, such that for any language L the following statements are equivalent:*

(i) There exists a solution X to the equation $X[k]L = \Re$.

(ii) There exists an index i, $1 \leq i \leq n$, such that $\Re''_i[k]L = \Re$.

Moreover, the regular languages $\Re''_i$ can be effectively constructed.

The problem "Does there exist a solution X to the equation $X[k]L = \Re$?" is undecidable for regular languages $\Re$ and context-free languages L. This follows as the same problem is undecidable for catenation (see, for example [3]) and catenation is a particular case of k-catenation.

For the sake of completeness, we investigate the closure properties of CF and CS under k-catenation.

If the language to be deleted is a regular one, the k-catenation can be simulated by a generalized sequential machine with erasing.

Proposition 5 *If L and $\Re$ are languages over Σ, $\Re$ a regular one, there exists a gsm g (with erasing) such that $L \rhd_k \Re = g(L) \cup \{\lambda |\ \lambda \in L \cap \Re\}$.*

Proof. Let $A = (S, \Sigma, s_0, F, P)$ be a finite deterministic automaton which recognizes $\Re$. Construct $g = (\Sigma, \Sigma, S', s'_0, F', P')$ where

$$\begin{array}{rrl} S' = & S\cup & \{s'_0\} \cup \{s_i |\ 1 \leq i \leq k+1\}, \\ P' = & P\cup & \{s'_0 \longrightarrow a s'_0 |\ a \in \Sigma\} \cup \{s'_0 a \longrightarrow s |\ s_0 a \longrightarrow s \in P\} \\ & \cup & \{sa \longrightarrow s_1 |\ sa \longrightarrow s' \in P, s' \in F\} \\ & \cup & \{s_i a \longrightarrow a s_{i+1} |\ 1 \leq i \leq k\} \cup \{s'_0 a \longrightarrow a s_1 |\ a \in \Sigma, \lambda \in \Re\}. \\ F' = & & \{s_i |\ 1 \leq i \leq k+1\}, \end{array}$$

Given a word $u \in L$ as an input, the gsm g works as follows. The rules of the form $s'_0 a \longrightarrow a s'_0$ leave unchanged a prefix of u. The rule $s'_0 a \longrightarrow s$ switches the derivation to P. By using rules of P, we erase a subword v of u.

A final state can be reached only through the application of one of the rules $sa \longrightarrow s_1$ or $s'_0 a \longrightarrow a s_1$. The use of the first one implies that a final state of P

has been reached, that is, the erased word v belongs to $\Re$. The use of the second one implies that the empty word λ (which belongs to $\Re$) has been erased from u.

Finally, the rules of the type $s_i a \longrightarrow a s_{i+1}$ leave unchanged a suffix of u. The fact that we have only $k+1$ terminal states assures us that the length of this suffix is at most k, therefore $g(u)$ will belong to $u \rhd_k v$. In case v is not a subword of u, a final state cannot be reached.

From the above considerations, it is easy to see that the gsm g satisfies the requested equality. □

As CF is closed under gsm mapping and under union, it immediately follows that CF is closed under k-quotient with regular languages.

As they are not closed under right quotient, CF and CS are not closed under k-quotient either.

3 K-deletion and right k-syntactic congruences

In order to study the solutions to the symmetric equation $L[k]Y = \Re$, we will make use of a result analogous to Proposition 2, namely (see [4]):

Proposition 6 *Let $L, \Re$ be languages over an alphabet Σ and $\diamond$, $\Box$ be two binary word (language) operations right-inverses to each other. If the equation $L \diamond Y = \Re$ has a solution Y, then the language $\Re' = (L \Box \Re^c)^c$ is a maximal solution.*

Recall that the operation $\Box$ is said to be the *right-inverse* of the operation $\diamond$ if for all words u, v, w over the alphabet Σ, we have: $w \in (u \diamond v)$ *iff* $v \in (u \Box w)$ (see [4]). The relation "is the right-inverse of" is symmetric.

If $\diamond$ is a binary word operation, the word operation $\diamond^r$ defined by $u \diamond^r w = w \diamond u$ is called *reversed* $\diamond$.

Definition 3 *Let u, v be words in Σ^*. The k-deletion of v from u is defined by:*

$$u \ominus_k v = \{w \in \Sigma^* |\ u = v_1 w v_2,\ v = v_1 v_2,\ |v_2| \leq k\}.$$

If $u \rightleftharpoons v$ is the dipolar deletion (see [3]) of v from u then:

$$u \rightleftharpoons v = \bigcup_{k \geq 0} (u \ominus_k v).$$

The operation of reversed k-deletion is the right inverse of the k-catenation. Indeed, $w \in u[k]v$ iff $w = u_1 v u_2$, $u = u_1 u_2$, $|u_2| \leq k$ iff $v \in w \ominus_k u$ iff $v \in u(\ominus_k)^r w$. Consequently, we can use the k-deletion to solve equations of the form $L[k]Y = \Re$. In order to study the decidability of the existence of solutions to the equation $L[k]Y = \Re$, we need to investigate the closure properties of REG, CF, CS under k- deletion.

Proposition 7 *Let L, $\Re$ be two languages over Σ. If $\Re$ is a regular language then $\Re \ominus_k L$ is regular (regardless of the complexity of L).*

Proof. Let $A = (S, \Sigma, s_0, F, P)$ be a finite automaton that accepts the language $\Re$, in which all states are useful. (A state is useful if there exists a path containing it which starts from the initial state and ends in a final state.) For any two states $s, s' \in S$, consider the language $L_{s,s'}$ defined in Proposition 3. Then we have that

$$\Re \ominus_k L = \bigcup_{(s_1,s_2)\in S'} L_{s_1,s_2}, \text{ where}$$

$$S' = \{(s_1, s_2) \in S \times S | \ \exists s_f \in F : L_{s_0,s_1}(L_{s_2,s_f} \cap (\{\lambda\} \cup \Sigma \cup \Sigma^2 \cup \ldots \cup \Sigma^k)) \cap L \neq \emptyset\}.$$

The proposition now follows as S is finite and therefore $\Re$ will be a finite union of regular languages. □

From the above construction it follows that $\Re \ominus_k L$ can be effectively constructed if $\Re$ is a regular language and L is a regular or context-free language. Indeed, in this case, the operation of intersection used in the definition of S' is effective.

Proposition 8 *The problem "Does there exist a solution Y to the equation $L[k]Y = \Re$?" is decidable for regular languages L and $\Re$.*

Proof. Analogous to that of Proposition 4 and using Proposition 6 and 7 together with the remarks following it. □

As a consequence of the preceding proposition, a result similar to Corollary 1 can also be obtained here.

The fact that the automaton A from Proposition 7 is finite implies that, given k, for any regular $\Re$ there exist finitely many languages that can be obtained from $\Re$ by k-deletion. By using a similar argument we deduce that, for a given regular $\Re$, there are finitely many languages that can be obtained from $\Re$ by k-deletion (regardless of k).

Proposition 9 *The binary relation $R_k(L)$ defined by "$u \equiv v$ $(R_k(L))$ if and only if $L \ominus_k u = L \ominus_k v$" is an equivalence relation.*

Proof. Immediate.

The right syntactic congruence R_L associated with the language L is the equivalence relation $R_0(L)$. For this reason, the equivalence relation $R_k(L)$ will be called the *right k-syntactic congruence* of L, even if $R_k(L)$ is not generally right compatible.

Since $R_{k+1}(L) \subseteq R_k(L)$, then:

$$\cdots \subseteq R_{k+1}(L) \subseteq R_k(L) \subseteq \cdots \subseteq R_1(L) \subseteq R_0(L) = R_L$$

Since the intersection of equivalence relations is an equivalence relation, $R_\infty = \bigcap_{k\geq 0} R_k(L)$ is also an equivalence relation.

As we have seen in Proposition 7, for a given regular language L there exist finitely many languages that can be obtained from it by k-deletion. Consequently, in this case, for any k, the relation $R_k(L)$ is of finite index. Moreover, R_∞ will be a finite intersection and therefore also of finite index. (The index of R_∞ is at most $2^{\mathrm{card}(S\times S)}$ where S is the set of states in a minimal automaton which recognizes L.) Hence:

Proposition 10 *Let L be a regular language. Then there exists a positive integer m such that $R_\infty = \bigcap_{i\geq 0} R_i(L) = R_{m+n}(L) = R_m(L)$ i.e. all the right syntactic congruences $R_i(L)$ and R_∞ have the same finite index for $i \geq m$.*

If the language L is not regular, this is no more the case as it will be shown by Example 1. We give first the following definitions.

Definition 4 *The right k-residue of L is the language defined as $W_k(L) = \{u \in \Sigma^* |\, L \ominus_k u = \emptyset\}$.*

If $k = 0$, then $W_0(L)$ is the usual right residue of L. If not empty, $W_k(L)$ is a right k-ideal and a class of $R_k(L)$.

Clearly $\cdots \subseteq W_k(L) \subseteq W_{k-1}(L) \subseteq \cdots \subseteq W_1(L) \subseteq W_0(L)$.

If the intersection $W_\infty(L) = \bigcap_{k\geq 0} W_k(L) \neq \emptyset$, then $u_1u_2 \in W_\infty(L)$, $x \in \Sigma^*$ imply $u_1xu_2 \in W_\infty(L)$ and by [10], $W_\infty(L)$ is regular.

Example 1.If $\Sigma = \{a, b\}$ and $L = \{a^nb^n | n \geq 1\}$, then it is easy to see that all the right k-residues $W_k(L)$ are different, because $W_{k+1}(L) \subset W_k(L)$. It follows then that all the equivalences $R_k(L)$ are different and we have the strict hierarchy

$$\cdots \subset R_{k+1}(L) \subset R_k(L) \subset \cdots \subset R_1(L) \subset R_0(L) = R_L.$$

We will end this section by investigating the closure of CF and CS under k-deletion. From the fact that CS is not closed under right quotient with regular languages it follows that CS is not closed under k-deletion with regular languages.

On the other hand, CF and CS are closed under $\ominus_k$ with singleton words. Indeed, if L is a language and w is a word over Σ we have:

$$L \ominus_k w = \bigcup_{w_1,w_2}^{w=w_1w_2,|w_2|\leq k} (w_1^{-1}L)w_2^{-1},$$

and both CF and CS are closed under right and left derivative and under union.

Proposition 11 *If $L, \Re$ are languages over X, $\Re$ a regular one, there exists a gsm g with erasing such that:*

$$L \ominus_k \Re = g(L) \cup \{\lambda |\, \lambda \in L \cap \Re\}.$$

Proof. Let $A = (S, \Sigma, s_A, F, P)$ be a finite automaton that recognizes $\Re$. Construct the gsm $g = (\Sigma, \Sigma, S', \{s_A\}, F', P')$, where

$$\begin{array}{lll} S' = & S & \cup\{s_i |\ s \in S, 1 \leq i \leq k+1\} \\ P' = & P & \cup\{sa \longrightarrow as |\ a \in \Sigma, s \in S\} \cup \{sa \longrightarrow s'_1 |\ sa \longrightarrow s' \in P\} \\ & & \cup\{s_i a \longrightarrow s'_{i+1} |\ sa \longrightarrow s' \in P, 1 \leq i \leq k\} \\ & & \cup\{sa \longrightarrow a s_{k+1} |\ a \in \Sigma, s \in S\} \\ F' = & & \{s_i |\ s \in F, 1 \leq i \leq k+1\} \end{array}$$

For each state $s \in S$, we have constructed k indexed copies $s_1, \ldots, s_k$. Given an input of the form $u_1 v u_2 \in L$, where $u_1 u_2 \in \Re$, the gsm g works as follows. First, by using rules from P, the portion u_1 is erased. Rules of the form $sa \longrightarrow as$ jump over the letters of v. (The last rule of this portion switches from normal derivation using states in S to a derivation using only indexed states.) Finally, by using only indexed states, the word u_2 is deleted. As the final states are only indexed states where the index is not bigger than k, the word u_2 to be erased has length at most k. □

As CF is closed under gsm mappings and union it follows that it is closed under k-deletion with regular languages.

4 K-prefix codes

The notion of prefix codes is related to the operation of catenation. Since k-catenation is a generalization of catenation, it is natural to introduce new classes of prefix codes, called k-prefix codes. Their study is the object of this section.

Definition 5 *The relation ρ_k is defined on Σ^* by:*

$$u \rho_k v \ \Leftrightarrow\ v = u_1 x u_2, u = u_1 u_2, |u_2| \leq k$$

The relation ρ_k is a reflexive and antisymmetric binary relation that is left compatible and the transitive closure $\bar{\rho}_k$ of ρ_k is a left compatible partial order.

Remark that if $k = 0$, ρ_0 is the usual prefix order.

A nonempty subset $S \subseteq \Sigma^*$ such that $u, v \in S$ implies $u[k]v \subseteq S$ is called a *k-subsemigroup.* Clearly S is a subsemigroup of Σ^*.

A *right k-ideal* $L \subseteq \Sigma^*$ is a nonempty subset of Σ^* such that $u \in L$ implies $u[k]x \subseteq L$ for all $x \in \Sigma^*$. This is equivalent to $L[k]\Sigma^* \subseteq L$. Every right k-ideal is a right ideal (right 0-ideal) and a k-subsemigroup. If L is a right k-ideal for every $k \geq 0$, then, for all $u = u_1 u_2 \in L$ and $x \in \Sigma^*$, $u_1 x u_2 \in L$.

Definition 6 *If $L \subseteq \Sigma^*$, then:*

$$\rho_k^{[0]}(L) = L, \rho_k^{[1]}(L) = \{v \in \Sigma^* \mid \exists u \in L, u \rho_k v\}, \cdots, \rho_k^{[n]}(L) = \rho_k^{[1]}(\rho_k^{[n-1]}(L)), \cdots$$

$$\rho_k(L) = \bigcup_{n \geq 0} \rho_k^{[n]}(L)$$

Clearly $\rho_k(L) = \{v \in \Sigma^* | \exists u \in L, u\bar{\rho}_k v\}$. The language $\rho_k(L)$ is called the ρ_k-*closure* of L.

Proposition 12 *If L is a nonempty language, $\rho_k(L)$ is a right k-ideal and $\rho_k(L)$ is the intersection of all the right k-ideals containing L.*

Proof. Let $u = u_1u_2 \in \rho_k(L), |u_2| \leq k$ and $x \in \Sigma^*$. Then $u \in \rho_k^{[n]}(L)$ for some $n \geq 0$. From $u\rho_k u_1 x u_2$ follows $u_1 x u_2 \in \rho_k^{[n+1]}(L) \subseteq \rho_k(L)$. Hence $\rho_k(L)$ is a right k-ideal.

Let T be a right k-ideal containing L. Suppose that $\rho_k(L)$ is not contained in T. Then there is an integer n and a word $v \in \rho_k^{[n]}(L) \subseteq \rho_k(L)$ such that $v \notin T$. Suppose n minimal with this property. Then $n \geq 1$ and $\exists\, u \in \rho_k^{[n-1]}(L)$ such that $v = u_1 x u_2$. Because of the minimality of n, then $u \in T$ and, since T is a right k-ideal, $v = u_1 x u_2 \in T$, a contradiction. Therefore $\rho_k(L) \subseteq T$. □

Remark that a language L is a right k-ideal if and only if L is its own ρ_k-closure, i.e. $L = \rho_k(L)$.

Let C be a nonempty subset of Σ^+. Then C is called (i) a *prefix code* if $u, ux \in C$ implies $x = \lambda$, (ii) an *outfix code* if $u_1u_2, u_1xu_2 \in C$ implies $x = \lambda$, (iii) an *infix code* if $u, xuy \in C$ implies $x = y = \lambda$ (See [2] for example).

Definition 7 *A k-prefix code is a nonempty language $P \subseteq \Sigma^+$ such that $u \in P$ and $u[k]x \cap P \neq \emptyset$ implies $x = \lambda$.*

Remark that a k-prefix code is also an m-prefix code for $m \leq k$ and that prefix codes are the 0-prefix codes. Every outfix code is a k-prefix code for all $k \geq 0$. An infix code is not in general a k-prefix code. For example, let $L = ba^+b$ over $\Sigma = \{a, b\}$. Then L is an infix code, but not a k-prefix code for $k \geq 1$. Indeed, $ba^kb = baa^{k-1}b \in L$, $baaa^{k-1}b \in L$, $|a^{k-1}b| \leq k$, but $a \neq \lambda$. The language $\{a^nb^nc^n | n \geq 1\}$ is both an infix code and an outfix code and hence a k-prefix code for all $k \geq 0$. The language $\{a^2b, aba, ba^2, b^2\}$ over $\Sigma = \{a, b\}$ is a finite infix code that is a k-prefix code for $k \geq 0$.

Recall that if ρ is a partial order or a quasiorder (reflexive and antisymmetric relation), then an antichain A is a subset of Σ^* such that $u\rho v, u, v \in A$ implies $u = v$. If $L \subseteq \Sigma^+, L \neq \emptyset$, then L is a prefix code iff L is an antichain for the prefix order, that is ρ_0. In general:

Proposition 13 *A nonempty language $L \subseteq \Sigma^+$ is a k-prefix code if and only if L is a ρ_k-antichain.*

Proof. Immediate from the definitions. □

Let Σ with $|\Sigma| \geq 2$, let $a \in \Sigma$ and $Y = \Sigma \backslash \{a\}$. Then Y^*a^k is a k-prefix code that is not a m-prefix code for all $m > k$. Hence if $P_k(\Sigma)$ denotes the family of the k-prefix codes over Σ, we have the infinite hierarchy:

$$P_0(\Sigma) \supset P_1(\Sigma) \supset \cdots \supset P_k(\Sigma) \supset \cdots$$

With every nonempty language $L \subseteq \Sigma^+$ is associated a k-prefix code $\mathrm{Prf}_k(L)$ defined in the following way: $\mathrm{Prf}_k(L) = \{u \in L | v \in L, v\rho_k u \ \Rightarrow \ u = v\}$, i.e. $\mathrm{Prf}_k(L)$ is the set of words in L that are minimal with respect to the relation ρ_k or $\bar{\rho}_k$. Since $\lambda \notin L$ and $L \neq \emptyset$, then it is clear that $\mathrm{Prf}_k(L)$ is a k-prefix code.

Proposition 14 *(i) If P is a k-prefix code, then $\rho_k(P)$ is a right k-ideal and $Prf_k(\rho_k(P)) = P$.*

(ii) If L is a right k-ideal, $L \neq \Sigma^*$, there exists a unique k-prefix code P, namely $P = Prf_k(L)$, such that $L = \rho_k(P)$.

Proof. (i) The fact that the language $\rho_k(P)$ is a right k-ideal follows from Proposition 12.

"$\mathrm{Prf}_k(\rho_k(P)) \subseteq P$". Let w be a word in $\mathrm{Prf}_k(\rho_k(P))$. If $w \in P$ we are done. Otherwise, $w \in \rho_k(P)$ which implies the existence of $u \in P$ such that $u\overline{\rho}_k w$. This in turn means that there exist words $u_1, u_2, \ldots, u_n$ in $\rho_k(P)$ such that

$$u\rho_k u_1, u_1\rho_k u_2, \ldots, u_n\rho_k w.$$

(The words u_i, $1 \leq i \leq n$, belong to $\rho_k(P)$ because $\rho_k(P)$ is a right k-ideal.)

From $u_n \in \rho_k(P)$, $u_n\rho_k w$, $w \in \mathrm{Prf}_k(\rho_k(P))$ we deduce that $u_n = w$. Going backwards, one finaly reaches the conclusion that $u = w$ which implies that $w \in P$.

"$P \subseteq \mathrm{Prf}_k(\rho_k(P))$". Let w be a word in P. If $w \in \mathrm{Prf}_k(P)$ then, as $P \subseteq \rho_k(P)$ we have that $w \in \mathrm{Prf}_k(\rho_k(P))$.

If $w \notin \mathrm{Prf}_k(P)$ then $w = u_1 x u_2$ where $u_1u_2 \neq w$, $u_1u_2 \in P$, $|u_2| \leq k$. But P is a k-prefix code and therefore this implies $u_1u_2 = w$ – a contradiction.

(ii) Let $P = \mathrm{Prf}_k(L)$. Then P is a k-prefix code, $P \subseteq L$ and $\rho_k(P) \subseteq L$. Conversely, if $v \in L$, there exists $v_1 \in L$ such that $v_1\rho_k v$. Hence $v_1 = r_1 s_1$, $|s_1| \leq k$, $v = r_1 x_1 s_1$ for some $x_1 \in \Sigma^*$. Similarly, since $v_1 \in L$, there exists $v_2 \in L$ such that $v_2\rho_k v_1$ and $v_2 = r_2 s_2, |s_2| \leq k$, $v_1 = r_2 x_2 s_2$ for some $x_2 \in \Sigma^*$. And so on. Finally, by assuming that at each step $x_i \neq \lambda$, we have a descending sequence of v_i: $|v| > |v_1| > \cdots > |v_i| > \cdots$ with $v_i = r_i s_i, |s_i| \leq k$, $v_{i-1} = r_i x_i s_i$ and $x_i \neq \lambda$. Since the descending sequence is finite, we get for some n: $v_n = r_n s_n, |s_n| \leq k$ $v_{n-1} = r_n x_n s_n$ and a further decomposition for v_n is no more possible. This implies that $v_n \in P$. Hence:

$$v_n \in P, v_{n-1} \in \rho_k(v_n), v_{n-2} \in \rho_k(\rho_k(v_{n-1})) \subseteq \rho_k(v_n), \cdots,$$

$$v_1 \in \rho_k(v_n), v \in \rho_k(v_n) \subseteq \rho_k(P).$$

Therefore $L \subseteq \rho_k(P)$ and $L = \rho_k(P)$.

Let now Q be an arbitrary k-prefix code such that $L = \rho_k(Q)$. According to (i), $\rho_k(Q)$ is a right k-ideal and $\mathrm{Prf}_k(\rho_k(Q)) = Q$. Since $L = \rho_k(Q)$, we deduce that $\mathrm{Prf}_k(L) = Q$. By definition, $\mathrm{Prf}_k(L) = P$, therefore we can conclude that $Q = P$. □

A well known property of prefix codes is that their catenation is also a prefix code. The next proposition shows that a similar result holds for k-prefix codes.

Proposition 15 *The catenation of k-prefix codes is a k-prefix code.*

Proof. Let P, Q be two k-prefix codes and let $\alpha \in P$, $\beta \in Q$ such that there is a word in $\alpha\beta[k]v$ which belongs to PQ. We want to show that $v = \lambda$.

We distinguish two cases:

- the word v has been inserted into α or catenated to α. This means $\alpha_1 v \alpha_2 \beta \in PQ$, $|\alpha_2\beta| \leq k$, $\alpha_1, \alpha_2 \in \Sigma^*$, $\alpha = \alpha_1\alpha_2$;
- the word v has been inserted into β.

Consider the first case (the other one can be treated in a similar fashion). As $\alpha_1 v \alpha_2 \beta$ is in PQ, it is a catenation xy, where $x \in P$, $y \in Q$. One of the following situation can occur:

– $\underbrace{\alpha_1'}_{x}\underbrace{\alpha_1'' v \alpha_2 \beta}_{y}$ where $\alpha_1 = \alpha_1'\alpha_1''$. As $\beta \in Q$, $y = \alpha_1'' v \alpha_2 \beta \in Q$, $|\beta| \leq k$ and Q is a k-prefix code, it follows that $\alpha_1'' v \alpha_2 = \lambda$, therefore $v = \lambda$.

– $\underbrace{\alpha_1 v_1}_{x}\underbrace{v_2 \alpha_2 \beta}_{y}$ where $v = v_1 v_2$. From $y = v_2\alpha_2\beta \in Q$, $\beta \in Q$, $|\beta| \leq k$, we deduce that $v_2\alpha_2 = \lambda$. From $\alpha_1 \in P$, $x = \alpha_1 v_1 \in P$, we deduce that $v_1 = \lambda$. Consequently, $v = \lambda$.

– $\underbrace{\alpha_1 v \alpha_2'}_{x}\underbrace{\alpha_2''\beta}_{y}$ where $\alpha_2 = \alpha_2'\alpha_2''$. As $y = \alpha_2''\beta \in Q$, $\beta \in Q$, we have that $\alpha_2'' = \lambda$. This implies that $\alpha_1\alpha_2' \in P$ and as $\alpha_1 v \alpha_2' \in P$ and $|\alpha_2'| \leq |\alpha_2\beta| \leq k$, we conclude that $v = \lambda$.

– $\underbrace{\alpha_1 v \alpha_2 \beta_1}_{x}\underbrace{\beta_2}_{y}$ where $\beta_1\beta_2 = \beta$. As $\beta_1\beta_2$ and β_2 are in Q, $|\beta_2| \leq k$, we have that $\beta_1 = \lambda$. This implies that $\alpha_1 v \alpha_2 \in P$ which, together with the fact that $\alpha_1\alpha_2 \in P$ and $|\alpha_2| \leq k$ implies that $v = \lambda$.

In all cases we obtained that $v = \lambda$, therefore PQ is a k-prefix code. □

The preceding result is to be contrasted with the fact that the insertion of two prefix codes is not anymore a prefix code. Indeed, consider the prefix codes $L_1 = \{a^i b |\ i > 0\}$ and $L_2 = \{b^i a |\ i > 0\}$. The word $a^i b^{k+1} a^{m+1} b$ belongs to $(a^{i+m}b \leftarrow b^{k+1}a)$ and therefore to $L_1 \leftarrow L_2$. However, also $a^i b^{k+1} a$ which is one of its prefixes, belongs to $L_1 \leftarrow L_2$ as it is an element of the set $a^i b \leftarrow b^k a$. This shows that $L_1 \leftarrow L_2$ is not a prefix code.

References

[1] Berstel J. and Perrin D., *Theory of Codes*, Academic Press, Orlando, 1985.

[2] Ito M., Jürgensen H., Shyr H.J. and Thierrin G., Outfix and infix codes and related classes of languages, *J.Comp. and System Sci.*, vol.43(1991), pp.484-508.

[3] Kari L., *On insertion and deletion in formal languages*, PhD thesis, University of Turku, Finland, 1991.

[4] Kari L., On language equations with invertible operations. To appear in *Theoretical Computer Science*.

[5] Kari L., Insertion operations:closure properties. *EATCS Bulletin*, 51(1993), pp.181-191.

[6] Kari L., Mateescu A., Paun G. and Salomaa A., Deletion sets, *Fundamenta Informaticae*, vol.19(1993), pp.355-370.

[7] Lallement G., *Semigroups and Combinatorial Applications*, Wiley, New York, 1979.

[8] Salomaa A., *Formal languages*, Academic Press, New York, 1973.

[9] Shyr H.J., *Free Monoids and Languages* Lecture Notes, National Chung-Hsing University, Taichung, Taiwan, 1991.

[10] Thierrin G., Hypercodes, right convex languages and syntactic monoids, *Proc.Amer.Math.Soc.*, vol.83(1981), pp.255-258.