Deletion operations: closure properties

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

June 3, 2010

KEY WORDS: right/left quotient, sequential deletion, parallel deletion.
C.R. CATEGORIES: F.4.3 [Mathematical Logic and Formal Languages]: Formal Languages: *Operations on languages*; F.4.2 [Mathematical logic and Formal Languages]: Grammars and other rewriting systems: *Grammar types*; D.3.1 [Formal Definitions and Theory]

Abstract

The paper studies some variants of deletion operations which generalize the left/right quotient of languages. The main emphasis is put on how these deletions can be expressed as a combination of other operations, and on closure properties of various language families under deletion. Some results are the expected ones: the sequential (iterated sequential, dipolar) deletion from a regular language produces a regular set regardless of the complexity of the deleted language. On the other hand, it still remains a challenging open problem whether or not the family of regular languages is closed under iterated parallel deletion with singletons.

1 Introduction

The operations of deleting symbols or strings of symbols from a given word (and the natural extensions of such operations to languages) are most fundamental in formal language theory and combinatorics of words. Left and right quotient are special cases of such deletion operations. Examples of the wide range of applications of these operations are bottom-up parsing (a symbol is deleted and replaced by a nonterminal) developmental systems (deletion means death of a cell or a string of cells) and cryptography (decryption may begin by deleting some "garbage" portions in the cryptotext).

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

The most natural extension of the left (right) quotient $w \setminus u$ (u/w) is the *sequential deletion* $u \longrightarrow w$: instead of erasing w from the left (right) extremity of u, we delete any possible occurrence of w in u. The result of the sequential deletion $u \longrightarrow w$ will be thus a set of words instead of a single one, as w may appear more than once as a subword in u. The operation of sequential deletion can be viewed as one rewriting step in a special Thue system (see [1], [2], [5]). Consequently, some of the results concerning iterated sequential deletion (reduction in a special Thue system) have already been obtained in the context of Thue system theory.

As its name suggests, the *parallel deletion* $u \Longrightarrow w$ performs a task similar to the sequential deletion, but in parallel: all the non-overlapping occurrences of w are simultaneously erased from u. An operation similar to the parallel deletion has been defined and studied in [12]. The difference is that in our variant no restriction is imposed on the alphabets of the operands (the two languages can be over the same alphabet).

Deletion operations and various related problems have been investigated in [6], [7], [14], [8], [9], [10], [11]. This paper concentrates on the closure properties of various language families under deletion operations. Closure properties will be studied from two points of view: *(i)* the preservation of certain properties, such as regularity, under deletion operations and *(ii)* the structure of deletion operations expressed in terms of other operations. We believe that such a study will shed light on deletion operations in language theory in general, as well as in the case of the more specific language families investigated.

Section 2 studies the sequential and parallel deletion and their iterated versions. It is shown for example that by sequentially deleting from a regular language an arbitrary one, the result is still regular. The closure of the family of regular languages under iterated parallel deletion remains open.

In Section 3 some more sophisticated versions of deletion are defined. The *permuted deletion* $u \rightsquigarrow w$ consists in erasing from u not only w, but all words which are obtained from w by arbitrarily permuting its letters. The *dipolar deletion* $u \rightleftharpoons w$, the study of which has arised from the necessity of solving certain language equations (see [9]), consists in erasing from u a prefix and a suffix whose catenation equals w.

In the operations mentioned so far, the deletion takes place in an arbitrary position of a word. As a result, the quotient is not a particular case of any of them as we cannot force the place of the deletion. A natural idea of controlling the position of the deletion is that each letter determines what can be deleted after it. The *controlled deletion* is studied in Section 4.

Finally, opposed to the previous variants of deletion, where the word to be erased was treated as a whole, the *scattered deletion* $u \dashrightarrow w$ sparsely erases from u the letters of w, in the same order. Section 5 is concerned with the closure properties of some language families under scattered variants of deletion.

In the following, Σ will denote an alphabet, that is a finite nonempty set, and Σ^* the set of all words over Σ. For a word w, $\lg(w)$ is the length of w, $N_a(w)$ the number of occurrences of the letter a in w and $\mathrm{com}(w)$ the commutative closure of w (all words obtained from w by arbitrarily permuting its letters). REG, CF, CS will denote the family of regular, context-free and context-sensitive languages, respectively. For further formal language notations and notions the reader is referred to [13].

2 Sequential and parallel deletion

The sequential deletion (shortly, SD) is the simplest and most natural generalization of left/right quotient. If u, v are words over Σ, the deletion of v from u consists of erasing v not only from the left/right extremity of u, but from an arbitrary place in u:

$$u \longrightarrow v = \{w \mid u = w_1 v w_2, w = w_1 w_2, w_1, w_2 \in \Sigma^*\}.$$

If v is not a subword of u, the result of the deletion is the empty set. The operation can be extended to languages in the natural fashion.

Example 1 Let $L_1 = \{abababa, ab, ba^2, aba\}$, $L_2 = \{aba\}$. The sequential deletion of L_2 from L_1 is $L_1 \longrightarrow L_2 = \{baba, abba, abab, \lambda\}$. □

The sequential deletion is a partial operation in the sense that when performing $L_1 \longrightarrow L_2$, not all words from L_1 and L_2 contribute to the result. Indeed, the words from L_1 which do not contain any word from L_2 as a subword, as well as the words from L_2 which are not subwords of any word of L_1, do not contribute.

The left and right quotient can be obtained by using the sequential deletion and a marker which forces the position of the deletion. If L_1, L_2 are languages over Σ then,

$$L_2 \backslash L_1 = (\#L_1) \longrightarrow (\#L_2),$$

$$L_1 / L_2 = (L_1\#) \longrightarrow (L_2\#),$$

where # is a new symbol which does not belong to Σ.

The next result is analogous to one obtained for left/right quotient (see for example [4], p.50).

Theorem 1 *If L_1, L_2 are languages over the alphabet Σ, L_2 arbitrary and L_1 a regular one, then $L_1 \longrightarrow L_2$ is a regular language.*

Proof. Let L_1, L_2 be languages over Σ and let $A = (S, \Sigma, s_0, F, P)$ be a finite automaton that accepts L_1. For two states s, s' in S denote:

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

The language $L_{s,s'}$ is regular for each s, s' in S. Consider the automaton:

$$\begin{array}{ll} A' = & (S, \Sigma \cup \{\#\}, s_0, F, P') \\ P' = & P \cup \{s\# \longrightarrow s |\ s, s' \in S \text{ and } L_2 \cap L_{s,s'} \neq \emptyset\}, \end{array}$$

where # is a new symbol which does not occur in Σ.

The theorem follows as we have

$$L_1 \longrightarrow L_2 = h(L(A') \cap \Sigma^* \# \Sigma^*)$$

where h is the morphism $h : (\Sigma \cup \{\#\})^* \longrightarrow \Sigma^*$, $h(\#) = \lambda, h(a) = a,\ \forall a \in \Sigma$.

□

Corollary 1 *The language $L_1 \longrightarrow L_2$ can be effectively constructed if L_1 is a regular language and L_2 is a regular or context-free language.*

Corollary 2 *For any regular language L_1 there exist finitely many languages that can be obtained from L_1 by sequential deletion.*

Proof. It follows from the preceding theorem by the fact that the automaton A is finite. This implies that there are finitely many different possibilities of constructing the automaton A'. □

If the language to be deleted is a regular one, the sequential deletion can be simulated by a generalized sequential machine (gsm) with erasing.

Theorem 2 *Any family of languages which is closed under gsm-mapping is closed under sequential deletion with regular languages.*

Proof. Let $A = (S, \Sigma, s_0, F, P)$ be a finite automaton that recognizes the language R. Construct the gsm with erasing,

$$\begin{array}{ll} g = & (\Sigma, \Sigma, S \cup \{s'_0, s_f\}, s'_0, \{s_f\}, P'), \\ P' = & P \cup \{s'_0 a \longrightarrow a s'_0 |\ a \in \Sigma\} \cup \{s'_0 a \longrightarrow s |\ s_0 a \longrightarrow s \in P\} \cup \\ & \{sa \longrightarrow s_f |\ sa \longrightarrow s' \in P, s' \in F\} \cup \{s_f a \longrightarrow a s_f |\ a \in \Sigma\} \cup \\ & \{s'_0 a \longrightarrow s_f |\ s_0 a \longrightarrow s \in P, s \in F\} \cup \{s'_0 a \longrightarrow a s_f |\ a \in \Sigma, \lambda \in R\}. \end{array}$$

The theorem now holds as we have $L \longrightarrow R = g(L) \cup \{\lambda |\ \lambda \in L \cap R\}$. Indeed, given a word $v \in L$ as an input and a word $w \in R$, the gsm g works as follows: the rules of P erase the word w from v while the ones of the type $s'_0 a \longrightarrow a s'_0$ and $s_f a \longrightarrow a s_f$ cross over the letters which will remain in $v \longrightarrow w$.

□

A parallel variant of deletion will be defined in the sequel. Given words u and v, the *parallel deletion* (shortly PD) of v from u, denoted $u \Longrightarrow v$, consists of the words obtained by simultaneously erasing from u all the non-overlapping occurrences of v. The definition is extended to languages in the natural way. Given a word u and a language L_2, the parallel deletion $u \Longrightarrow L_2$ consists of the words obtained by erasing from u all the non-overlapping occurrences of words in L_2.

Definition 1 *Let L_1, L_2 be languages over the alphabet Σ. The parallel deletion (shortly, PD) of L_2 from L_1 is:*

$$L_1 \Longrightarrow L_2 = \bigcup_{u \in L_1} (u \Longrightarrow L_2), \text{ where}$$

$$\begin{aligned} u \Longrightarrow L_2 = \quad & \{u_1 u_2 \dots u_k u_{k+1} | \ k \geq 1, u_i \in \Sigma^*, 1 \leq i \leq k+1 \text{ and} \\ & \exists v_i \in L_2, 1 \leq i \leq k \text{ such that } u = u_1 v_1 \dots u_k v_k u_{k+1}, \\ & \text{where } \{u_i\} \cap [\Sigma^*(L_2 - \{\lambda\})\Sigma^*] = \emptyset, 1 \leq i \leq k+1\}. \end{aligned}$$

The parallel deletion $u \Longrightarrow L_2$ erases from u the non-overlapping occurrences of words from L_2. Moreover, a supplementary condition has to be fulfilled: between two occurrences of words of L_2 to be erased, no nonempty word from L_2 appears as a subword. This assures that *all* occurrences of words from L_2 have been erased from u, and is taken care of by the last line of the definition. The reason why λ had to be excluded from L_2 is obvious. If this wouldn't be the case and λ would belong to L_2, the condition $\{u_i\} \cap \Sigma^* L_2 \Sigma^* = \emptyset$ would imply $\{u_i\} \cap \Sigma^* = \emptyset$ – a contradiction. Note that words from L_2 can still appear as subwords in $u \Longrightarrow L_2$, as the result of catenating the remaining pieces of u.

Example 2 Let $L_1 = \{abababa, aababa, abaabaaba\}$, $L_2 = \{aba\}$. The parallel deletion of L_2 from L_1 is $L_1 \Longrightarrow L_2 = \{b, abba, aba, aab, \lambda\}$. □

If the language to be deleted is regular, the parallel deletion $L \Longrightarrow R$ can be expressed as a morphic image of the intersection between a regular language and the image of L under a rational transduction.

Theorem 3 *Any family of languages closed under intersection with regular languages, rational transductions and morphisms is closed under parallel deletion with regular languages.*

Proof. We begin by proving the following:

Claim. If L, R are languages over Σ, L a λ-free language and R a regular one, then there exist a rational transducer g, a morphism h and a regular language R' such that:

$$L \Longrightarrow R = h(g(L) \cap R').$$

Indeed, let $A = (S, \Sigma, s_0, F, P)$ be a finite automaton that accepts the language R. Let us consider the rational transducer:

$$\begin{aligned} g = \quad & (\Sigma, \Sigma \cup \{\#\}, S \cup \{s_0'\}, s_0', \{s_0'\}, P'), \\ P' = \quad & P \cup \{s_0' a \longrightarrow a s_0' | \ a \in \Sigma\} \cup \\ & \{s_0' a \longrightarrow s | \ s_0 a \longrightarrow s \in P, a \in \Sigma\} \cup \\ & \{s_0' a \longrightarrow \# s_0' | \ s_0 a \longrightarrow s \in P, a \in \Sigma, s \in F\} \cup \\ & \{s a \longrightarrow \# s_0' | \ s a \longrightarrow s' \in P, a \in \Sigma, s' \in F\} \cup \\ & \{s_0' \longrightarrow \# s_0' | \ \lambda \in R\}. \end{aligned}$$

The rational transducer g performes the following task: given a word of L as an input, it replaces arbitrary many words of R from it with the marker $\#$,

$$\begin{aligned} g(L) = \quad & L \cup \{u_1\#u_2\# \dots u_k\#u_{k+1} |\ k \geq 1, u_i \in \Sigma^*, 1 \leq i \leq k+1 \\ & \text{and } \exists u \in L, v_i \in R, 1 \leq i \leq k : u = u_1 v_1 \dots u_k v_k u_{k+1}\}. \end{aligned}$$

If we consider now the morphism h which erases the marker, $h : (\Sigma \cup \{\#\})^* \longrightarrow \Sigma^*$ defined by $h(\#) = \lambda$, $h(a) = a$, $a \in \Sigma$ and R' the regular set:

$$R' = [(\Sigma \cup \{\#\})^*(R - \{\lambda\})(\Sigma \cup \{\#\})^*]^c \cap (\Sigma^*\#\Sigma^*)^+,$$

then h, g and R' satisfy the equality of the Claim. Indeed, the first set of the intersection in R' takes care that between two erased words (marked with $\#$) no other candidate for erasing occurs. The second set takes care of that the words from L which do not contain any candidate for erasing are not retained in the final result. This is done by retaining only those words in which at least one erasing (marker) occurs.

Let us return now to the proof of the theorem and let L and R be languages over Σ, R a regular one.

If λ is not a subword of L, the theorem follows from the preceding Claim.

If λ belongs to L but not to R, then $L \Longrightarrow R = ((L - \{\lambda\}) \Longrightarrow R)$ and we can use the proof of the Claim for $L - \{\lambda\}$ and R.

If λ belongs to $L \cap R$, then $L \Longrightarrow R = [(L - \{\lambda\}) \Longrightarrow R] \cup \{\lambda\}$ and we can use again the same proof to show that $(L - \{\lambda\}) \Longrightarrow R$ belongs to the family. □

The families CF and CS are closed under neither sequential nor parallel deletion. Moreover, in the context-sensitive case, there exists a language from which the PD of a single word produces a non- context-sensitive language.

Proposition 1 *The family of context-free languages is closed under neither sequential nor parallel deletion.*

Proof. Let L_1, L_2 be the context- free languages:

$$L_1 = \#\{a^i b^{2i} |\ i > 0\}^*,\ L_2 = \#a\{b^i a^i |\ i > 0\}^*.$$

(Similar languages have been used in [3], p.40, to show that CF is not closed under left quotient.)

The language $L_1 \longrightarrow L_2$ is not context-free. Indeed, if it would be context-free, then also the language

$$(L_1 \longrightarrow L_2) \cap b^+ = \ \{b^{2^n} \mid n > 0\}$$

would be context-free, which is a contradiction.

The same example can be used to prove that CF is not closed under parallel deletion, because the presence of the marker assures us that $L_1 \longrightarrow L_2 = L_1 \Longrightarrow L_2$. □

As the family CS is not closed under left quotient with regular languages, it follows that it is not closed under SD with regular languages either. However, as the following theorem implies, CS is closed under sequential deletion with singletons.

Theorem 4 *Any family of languages which is closed under λ-free gsm mappings and linear erasing is closed under sequential deletion with singletons.*

Proof. Let L be a language in such a family and let w be a word over the same alphabet Σ. If $w \in L$ then $L \longrightarrow \{w\} = [(L - \{w\}) \longrightarrow \{w\}] \cup \{\lambda\}$. If $w = \lambda$ then $L \longrightarrow \{\lambda\} = L$. Therefore the theorem will hold if we prove that $L \longrightarrow \{w\}$ belongs to our family for w nonempty and not belonging to L.

We can modify the proof of Theorem 2 such that the constructed gsm is λ-free. Indeed, consider the gsm which instead of erasing the letters of w, replaces them with a new symbol #.

It is easy to see that if $h : (\Sigma \cup \{\#\})^* \longrightarrow \Sigma^*$ is the morphism defined by $h(\#) = \lambda$, $h(a) = a$, $\forall\, a \in \Sigma$ then $h(g(L)) = L \longrightarrow \{w\}$.

If the length of w is $\lg(w) = n$ then, for every word $\alpha \in g(L)$ we have:

$$\lg(\alpha) \leq (n+1)\lg(h(\alpha))$$

which proves that h is an $(n+1)$- linear erasing with respect to $g(L)$. □

The family CS is not closed under parallel deletion.

Proposition 2 *There exist a context-sensitive language L_1 and a word w over an alphabet Σ such that $L_1 \Longrightarrow w$ is not a context-sensitive language.*

Proof. Let L be a recursively enumerable language (which is not context- sensitive) over an alphabet Σ and let a, b be two letters which do not belong to Σ. Then there exists a context-sensitive language L_1 such that (see [13], p.89):

(i) L_1 consists of words of the form $a^i b\alpha$ where $i \geq 0$ and $\alpha \in L$;

(ii) For every $\alpha \in L$, there is an $i \geq 0$ such that $a^i b\alpha \in L_1$.

It is easy to see that $aL_1 \Longrightarrow \{a\} = \; bL$ which is not a context-sensitive language. We have concatenated a to the left of L_1 in order to avoid the case $i = 0$, when the corresponding words from L would have been lost. □

A natural step following the definition of the sequential and parallel deletion is to consider their iterated versions. The *iterated sequential* and *iterated parallel deletion* have somewhat unexpected properties. While the result of iterated SD from a regular language is regular regardless of the complexity of the deleted language (see [5] for a proof of this result in the context of Thue systems theory), the families CF and CS are not closed even under iterated SD with singletons (for the case of CF languages, see [5]). It is an open problem whether REG is closed under iterated PD or iterated PD with singletons.

Definition 2 *Let L_1, L_2 be languages over the alphabet Σ. The iterated sequential deletion of order n, $L_1 \longrightarrow^n L_2$, is defined inductively by the equations:*

$$L_1 \longrightarrow^0 L_2 = L_1,\ \ L_1 \longrightarrow^{i+1} L_2 = (L_1 \longrightarrow^i L_2) \longrightarrow L_2,\ i \geq 0.$$

The iterated sequential deletion (iterated SD) of L_2 from L_1 is then defined as:

$$L_1 \longrightarrow^* L_2 = \bigcup_{n=0}^{\infty} (L_1 \longrightarrow^n L_2).$$

The *iterated parallel deletion (iterated PD)* of L_2 from L_1 is defined by replacing in the preceding definition the sequential deletion " $\longrightarrow$" with the parallel deletion " $\Longrightarrow$".

Example 3 Let $L_1 = \{a^n b^n c^n |\ n \geq 0\}$ and $L_2 = \{ab\}$. Then,

$$L_1 \longrightarrow^* L_2 = \{a^m b^m c^n |\ n, m \geq 0, n \geq m\} = L_1 \Longrightarrow^* L_2.$$

□

However, in general, the results of the iterated SD and iterated PD do not coincide.

Example 4 Let $L_1 = \{w \in \{a,b\}^* |\ N_a(w) = N_b(w)\}$ and $L_2 = \{a, b\}$. Then,

$$L_1 \longrightarrow^* L_2 = \{a,b\}^*,\ \ L_1 \Longrightarrow^* L_2 = L_1,\ \text{whereas}\ L_1 \Longrightarrow L_2 = \{\lambda\}.$$

□

Given two languages L_1 and L_2 over the alphabet Σ, the following inclusions hold $L_1 \Longrightarrow L_2 \subseteq L_1 \Longrightarrow^* L_2 \subseteq L_1 \longrightarrow^* L_2$. Indeed, any parallel deletion can be simulated by a string of sequential deletions. On the other hand, the preceding example shows that the reverse inclusions do not hold.

Proposition 3 *There exist a context-free language L over $\{a, b, \#\}$ and a word w over $\{a, b\}$ such that $L \longrightarrow^* \{w\}$ and $L \Longrightarrow^* \{w\}$ are not context-free languages.*

Proof. Let L be the language $L = \{a^i \# b^{2i} |\ i > 0\}^*$, and $w = ba$.

The theorem follows from the equalities:

$$(L \longrightarrow^* ba) \cap\ a\#^+ b^+ = (L \Longrightarrow^* ba) \cap\ a\#^+ b^+ =\ \{a\#^n b^{2^n} |\ n > 0\}.$$

□

The family of context-sensitive languages is closed under neither iterated SD nor iterated PD, as shown below.

Proposition 4 *Let Σ be an alphabet and a, b letters which do not occur in Σ. There exist a context-sensitive language L_1 over $\Sigma \cup \{a, b\}$ and a word w over $\{a, b\}$ such that $L_1 \longrightarrow^* w$ and $L_1 \Longrightarrow^* w$ are not context-sensitive languages.*

Proof. Let L be the recursively enumerable (which is not context- sensitive) language and L_1 the context-sensitive language defined in Proposition 2.

It is obvious that $(L_1 \longrightarrow^* \{a\}) \cap b\Sigma^* = (L_1 \Longrightarrow^* \{a\}) \cap b\Sigma^* =\ bL$, and bL is not a context-sensitive language. □

3 Exotic variants of deletion

We now investigate a couple of somewhat unsusual types of deletion, namely the permuted and the dipolar deletion. These operations have been defined as a necessary tool for solving some language equations (see [9]).

The *permuted SD* of the word v from the word u, $(u \rightsquigarrow v)$, is the set obtained by erasing from u arbitrary occurrences (but one at a time in the sequential case) of words which are letter-equivalent to v:

$$u \rightsquigarrow v = u \longrightarrow \mathrm{com}(v).$$

The *permuted PD*, $(u \approx\!> v)$, is the set obtained by erasing from u all the non-overlapping occurrences of words which are letter-equivalent to v:

$$u \approx\!> v = u \Longrightarrow \mathrm{com}(v).$$

If none of the words letter-equivalent to v is a subword of u, the result of the permuted SD, as well as of the permuted PD is the empty set.

It follows from the definitions that any family of languages which is closed closed under sequential (parallel) deletion and commutative closure is closed under permuted sequential (parallel) deletion.

Consequently, for example REG is closed under permuted SD, but in this case the result cannot be effectively constructed. Indeed, Theorem 1 states that the result of the sequential deletion from a regular language is always regular. However, Corollary 1 emphasises that the result of the sequential deletion from a regular language can be effectivelly constructed only in case the language to be sequentially deleted is regular or context-free. As there exist regular languages whose commutative closure is not context-free, Corollary 1 cannot be applied in the case of permuted SD. In the particular case where the language to be (permuted sequentially) deleted is a singleton, Corollary 1 is applicable. Therefore the proof of the closure of REG under permuted sequential deletion with singletons is effective.

In the parallel case one obtains the following non-closure result.

Proposition 5 *The family of regular languages is not closed under permuted parallel deletion.*

Proof. Consider the regular languages $L_1 = \$a^*b^*\#\#a^*b^*\$, L_2 = \#\$(ab)^*$. Then the permuted PD of L_2 from L_1 is:

$$L_1 \approx\!> L_2 = \{\$a^n b^m \# \mid m, n \geq 0, m \neq n\} \cup \{\#a^n b^m \$ \mid m, n \geq 0, m \neq n\} \cup \{\lambda\}.$$

Indeed, let $u = \$a^n b^m \#\# a^p b^q \$$ be a word in L_1 and w a word in $com(L_2)$.

Because of the presence of the markers, the result of the permuted PD

$$\$a^n b^m \#\# a^p b^q \$ \approx\!> w,\ w \in com(L_2)$$

is not empty iff either

$$w = \$a^r b^r \#, r \geq 0 \text{ and } m = n = r,$$

or

$$w = \#a^s b^s \$, s \geq 0 \text{ and } p = q = s.$$

The situations being similar, let us assume that the first case holds.

Then, if $p = q$, the result of the operation will be $\{\lambda\}$, as the word $\#a^p b^p \$ \in com(L_2)$ will be deleted in parallel with w from $u \in L_1$. In order to obtain a nonempty word, the condition $p \neq q$ must be satisfied.

In the second case, reasoning similarily, one deduces that the condition $m \neq n$ must be fulfilled in order to get a nonempty word in the result of the deletion.

It has therefore been shown that the words v in the language $L_1 \approx\!> L_2$ have one of the following forms:

$$\begin{aligned} v = \quad & \$a^n b^m \#, n, m \geq 0, n \neq m, \\ v = \quad & \#a^p b^q \$, p, q \geq 0, p \neq q, \\ v = \quad & \lambda. \end{aligned}$$

As words of this form can be obtained in $L_1 \approx\!> L_2$ for any numbers n, m, p, $q \geq 0$, the equality is proved.

The theorem now follows because the language

$$\{\$a^n b^m \# |\ n, m \geq 0, n \neq m\} \cup \{\#a^n b^m \$ |\ n, m \geq 0, n \neq m\} \cup \{\lambda\}$$

is not a regular one. □

The family of context-free languages is not closed even under permuted SD with regular languages as shown below.

Proposition 6 *The family of context-free languages is closed under neither permuted sequential nor permuted parallel deletion with regular languages.*

Proof. Consider the regular language $L_2 = \#\#(a_2 b_2 c_2)^*$ and the context-free language $L_1 = \{a_1^n b_1^m c_1^l \# c_2^l b_2^m a_2^n \# |\ n, m, l \geq 0\}$.

Then the permuted SD of L_2 from L_1 is:

$$L_1 \rightsquigarrow L_2 = L_1 \longrightarrow com(L_2) = \{a_1^n b_1^n c_1^n |\ n \geq 0\}.$$

Indeed, let $u = a_1^n b_1^m c_1^l \# c_2^l b_2^m a_2^n \#$ and $w \in com(L_2)$. The set

$$a_1^n b_1^m c_1^l \# c_2^l b_2^m a_2^n \# \longrightarrow w$$

is not empty iff $w = \#c_2^r b_2^r a_2^r \#$ and $m = n = l = r$. This, in turn, implies that the only word in $u \rightsquigarrow w$ is $a_1^r b_1^r c_1^r$.

As such a word can be obtained in $L_1 \longrightarrow com(L_2)$ for every $r \geq 0$, the requested equality follows.

Because of the presence of the markers, the permuted SD and PD coincide,

$$L_1 \rightsquigarrow L_2 = L_1 \overset{\sim}{\rightsquigarrow} L_2.$$

The theorem now follows as the language $\{a_1^n b_1^n c_1^n |\ n \geq 0\}$ is not a context-free one. □

In the particular case when the language to be deleted is a singleton, the permuted SD and PD preserve the families of regular and context-free languages. This follows as any family of languages which is closed under SD (respectively PD) with singletons and under finite union is closed under permuted SD (permuted PD) with singletons. The same argument can be used to show that CS is closed under permuted SD.

The family of context-sensitive languages will not be closed under permuted SD and permuted PD as it is not closed under permuted SD with regular languages and under permuted PD with singletons.

Proposition 7 *There exists a context-sensitive language L_1 over the alphabet $\Sigma \cup \{a, b\}$ and a regular language R over $\{a, b\}$ such that $L_1 \rightsquigarrow R$ is not context-sensitive.*

Proof. Let L be a recursively enumerable (which is not context-sensitive) language over Σ and L_1 be the context-sensitive language over $\Sigma \cup \{a, b\}$, defined in Proposition 2. It is easy to see that $(L_1 \longrightarrow a^*b) \cap \Sigma^* = L$, which implies that $L_1 \longrightarrow a^*b$ is not context-sensitive. □

As the word to be deleted in Proposition 2 consists of one letter only, the same proof can be used to show that CS is not closed under permuted PD with singletons.

The following operation, the *dipolar deletion*, has been introduced in [6] in the context of solving certain language equations.

The dipolar deletion of the word v from the word u is the set consisting of the words obtained from u by erasing a prefix and a suffix whose catenation equals v:

$$u \rightleftharpoons v = \{w \in \Sigma^* |\ \exists\, v_1, v_2 \in \Sigma^* : u = v_1 w v_2, v = v_1 v_2\}.$$

Example 5 Let $L_1 = \{abaab, aabab, ababa\}$ and $L_2 = \{ab\}$. The dipolar deletion $L_1 \rightleftharpoons L_2 = \{aba, baa, aab\}$. □

Proposition 8 *The family of context-sensitive languages is not closed under dipolar deletion with regular languages.*

Proof. Let L_1, L_2 be languages over an alphabet Σ and $\#, \$$ be letters which do not occur in Σ. The theorem follows from the fact that we have

$$\# L_1 \$ \rightleftharpoons \# L_2 = (L_2 \backslash L_1)\$,$$

and the family of context-sensitive languages is not closed under left quotient with regular languages. □

Theorem 5 *The family of context-sensitive languages is closed under dipolar deletion with singletons.*

Proof. Let L be a language and w be a word over the same alphabet Σ. The theorem follows as we have

$$L \rightleftharpoons \{w\} = \bigcup_{w_1, w_2}^{w_1 w_2 = w} (w_1 \backslash L)/w_2,$$

and CS is closed under left and right quotient with singletons and under finite union. □

Proposition 9 *The family of context-free languages is not closed under dipolar deletion.*

Proof. The proof is similar to that of Proposition 1. Let L_1, L_2 be the languages defined by:

$$\begin{aligned} L_1 = &\quad \#\{a^i b^{2i} |\ i > 0\}^* \$, \\ L_2 = &\quad \#a\{b^i a^i |\ i > 0\}^*. \end{aligned}$$

Then we have

$$(L_1 \rightleftharpoons L_2) \cap b^+ \$ = \{b^{2^n} \$ |\ n > 0\},$$

which is not a context-free language. □

The following result is analogous to Theorem 1: the result of the dipolar deletion from a regular language is regular regardless the complexity of the deleted language.

Theorem 6 *Let L, R be two languages over the alphabet Σ. If R is a regular language then the language $R \rightleftharpoons L$ is regular.*

Proof. Let $A = (S, \Sigma, s_0, F, P)$ be a finite automaton that accepts the language R, in which all the states are useful. A state is called useful if there exists a path containing it which starts from the initial state and ends in a final state. For every two states s_1, s_2 in S define:

$$L_{s_1, s_2} = \{w \in \Sigma^* |\ s_1 w \Longrightarrow^* s_2 \text{ in } A\}.$$

We have that:

$$R \rightleftharpoons L = \bigcup\nolimits_{(s_1, s_2) \in S'} L_{s_1, s_2}, \qquad (*)$$

where

$$S' = \{(s_1, s_2) \in S \times S |\ \exists s_f \in F :\ L_{s_0, s_1} L_{s_2, s_f} \cap L \neq \emptyset\}.$$

The theorem now follows as $R \rightleftharpoons L$ is a regular language, being equal to a finite union of regular languages. □

Corollary 3 *The language $R \rightleftharpoons L$ can be effectively constructed if R is a regular language and L is a regular or context-free language.*

Proof. The claim follows from the proof of the preceding theorem. Indeed, if R is a regular language and L is regular (context-free) then the language $L_{s_0,s_1}L_{s_2,s_f} \cap L$ is regular (context-free) for any states s_1, s_2, s_f. As the emptiness problem is decidable for regular (context-free) languages, the set S' and therefore the language $R \rightleftharpoons L$, can be effectively constructed.

Corollary 4 *For any regular language R there exist finitely many languages that can be obtained from R by dipolar deletion.*

Proof. It follows from the preceding theorem by the fact that the automaton A is finite. This implies that there are finitely many different possibilities of constructing the union from (*).

The languages that can be obtained from R by dipolar deletion will be among the languages:

$$L_{S'} = \bigcup\nolimits_{(s_1,s_2)\in S'} L_{s_1,s_2},$$

where S' is an arbitrary subset of $S \times S$. There exists at most $2^{\mathrm{card}(S\times S)}$ such different languages. □

4 Controlled deletion

We have dealt so far with operations where the deletion took place in arbitrary places of a word. As a consequence, the left/right quotient are not particular cases of any of these operations, because one cannot fix the position where the deletion takes place. A natural idea of controlling the position where the deletion is performed is that every letter determines what can be deleted after it. The left/right quotient will be obtained then as a particular case of *controlled deletion*.

Definition 3 *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 sequential deletion from L (shortly, controlled SD) is defined as:*

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

$$u \mapsto \Delta = \{u_1au_2 \in \Sigma^* | \; u = u_1avu_2 \text{ for some } u_1, u_2 \in \Sigma^*, a \in \Sigma \text{ and } v \in \Delta(a)\}.$$

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

As a language operation, the Δ-controlled SD has the arity $\text{card}(\Sigma) + 1$.

If one imposes the restriction that for a distinguished $b \in \Sigma$, $\Delta(b) = L_2$, and $\Delta(a) = \emptyset$ for any letter $a \neq b$, a special case of controlled SD is obtained: the *sequential deletion next to the letter* b, denoted by $L \stackrel{b}{\longmapsto} L_2$. The SD next to a letter is a binary operation. The words in $L \stackrel{b}{\longmapsto} L_2$ are obtained by erasing from words in L one occurrence of a word of L_2 which appears immediately next to a letter b. The words from L which do not contain the letter b followed by a word from L_2 do not contribute to the result.

Example 6 Let L be the language $L = \{abba, aab, bba, aabb\}$ and Δ the control function $\Delta(a) = b$, $\Delta(b) = a$. Then we have:

$$\begin{aligned} L \longmapsto \Delta &= \{aba, abb, aa, bb, aab\}, \\ L \stackrel{a}{\longmapsto} \{b\} &= \{aba, aa, aab\}, \\ L \stackrel{b}{\longmapsto} \{a\} &= \{abb, bb\}. \end{aligned}$$

□

In general, if L is a language over Σ and $\Delta : \Sigma \longrightarrow 2^{\Sigma^*}$ a control function,

$$L \longmapsto \Delta = \bigcup_{a \in \Sigma} (L \stackrel{a}{\longmapsto} \Delta(a)) = \bigcup_{u \in L} \bigcup_{a \in \Sigma} (u \stackrel{a}{\longmapsto} \Delta(a)).$$

The sequential deletion $L_1 \longrightarrow L_2$ can be expressed in terms of controlled SD by using a control function which has the value L_2 for all letters in Σ and a marker. Indeed,

$$L_1 \longrightarrow L_2 = h(\# L_1 \longmapsto \Delta),$$

where $\Delta(\#) = \Delta(a) = L_2, \forall a \in \Sigma$ and h is the morphism that erases the marker #.

The left quotient can be obtained from the SD next to a letter by using a marker and the morphism h which erases the marker:

$$L_2 \backslash L_1 = h(\# L_1 \stackrel{\#}{\longmapsto} L_2).$$

Notice that if the letter a does not occur in the word u then $u \stackrel{a}{\longmapsto} \Delta(a) = \emptyset$. This happens also if a occurs in u but no word of the form av, $v \in \Delta(a)$ exists in u. In particular, if λ belongs to L, λ does not contribute to the result of the controlled SD:

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

A parallel variant of the controlled 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 $u \Longmapsto \Delta$ 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. 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.

Definition 4 *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, controlled PD) is defined as:*

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

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

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

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

If one imposes the restriction that for a distinguished letter $b \in \Sigma$ we have $\Delta(b) = L_2$, and $\Delta(a) = \lambda$ for any letter $a \neq b$, a special case of controlled PD is obtained: *parallel deletion next to the letter b*. The parallel deletion next to b is denoted by $\overset{b}{\Longmapsto}$. Let us examine the set $u \overset{b}{\Longmapsto} L_2$, where u is a nonempty word and L_2 is a language over an alphabet Σ. If $u = b^k$, $k > 0$, and no word of the form bv, $v \in L_2$ occurs as a subword in u, the set $u \overset{b}{\Longmapsto} L_2$ equals the empty set. If u contains at least one letter different from b, u is retained in the result as we can erase λ near that letter. The other words in $u \overset{b}{\Longmapsto} L_2$ are obtained by finding all the nonoverlapping occurrences of words of the type bv_i, $v_i \in L_2$, in u, and deleting v_i from them. There may exist more than one possibility of finding such a decomposition of u into subwords.

Example 7 Let $L = \{abababa, a^3b^3, abab\}$ and $\Delta(a) = b$, $\Delta(b) = a$. Then:

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

As in the sequential case, if the empty word belongs to L, this does not influence the result of the controlled PD:

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

If the control function has as values regular languages for every letter of the alphabet, the controlled SD can be simulated by a gsm with erasing. Consequently, any family of languages which is closed under erasing gsm mappings will be closed under controlled SD.

Theorem 7 *Let L be a language over Σ and $\Delta : \Sigma \longrightarrow 2^{\Sigma^*}$ a control function whose values are regular languages. There exists a gsm g such that:*

$$L \longmapsto \Delta \ = \ g(L).$$

Proof. According to a previous remark, one can assume that L is λ-free. One can construct now a gsm similar to the one of Theorem 2, with the only modification that every letter a triggers a different derivation (the one corresponding to the automaton which recognizes the language $\Delta(a)$). □

The family of context-free languages is closed under neither controlled SD nor controlled PD as it is not closed under SD and PD next to one letter.

Proposition 10 *There exist two context-free languages L_1, L_2 over an alphabet Σ and a letter $\#$ in Σ such that $L_1 \overset{\#}{\longmapsto} L_2$ and $L_1 \overset{\#}{\Longmapsto} L_2$ are not context-free languages.*

Proof. Let $\Sigma = \{a, b, \#\}$ and L_1, L_2 be the context-free languages:

$$L_1 = \#\{a^i b^{2i} |\ i > 0\}^*,\ \ L_2 = a\{b^i a^i |\ i > 0\}^*.$$

Then, $(L_1 \overset{\#}{\longmapsto} L_2) \cap \#b^+ = (L_1 \overset{\#}{\Longmapsto} L_2) \cap \#b^+ = \{\#b^{2^n} |\ n > 0\}$, which is not a context-free language. □

If the control function has as values only nonempty regular languages then the controlled PD, $L \Longmapsto \Delta$, can be expressed as a morphic image of an intersection between a regular language and the image of L through a gsm with erasing. Consequently, any family of languages which is closed under morphisms, intersection with regular languages and erasing gsm will be closed under controlled PD.

Theorem 8 *Let L be a language over Σ and $\Delta : \Sigma \longrightarrow 2^{\Sigma^*}$ a control function whose values are nonempty regular languages. There exist a gsm g, a morphism h and a regular language R' such that:*

$$L \Longmapsto \Delta \ = \ h(g(L) \cap R').$$

Proof. Similar to the one of Theorem 3. □

The family of context-sensitive languages is not closed under controlled SD and controlled PD. However, in the particular case when the control function has as values only singletons, CS is closed under these operations.

Proposition 11 *Let Σ be an alphabet and $a, b, \#$ symbols which do not belong to Σ. There exists a context-sensitive language L_1' over the alphabet $\Sigma \cup \{a, b, \#\}$ and a regular language R over $\{a, b\}^*$ such that $L_1' \overset{\#}{\longmapsto} R$ and $L_1' \overset{\#}{\Longmapsto} R$ are not context-sensitive languages.*

Proof. Let L be the recursively enumerable language (which is not context-sensitive) over Σ and L_1 the context-sensitive language over $\Sigma \cup \{a, b\}$, defined in Proposition 2.

Because $\#$ is a symbol which does not belong to $\Sigma \cup \{a, b\}$ then

$$(\#L_1) \overset{\#}{\longmapsto} (a^*b) = [(\#L_1) \overset{\#}{\Longmapsto} (a^*b)] \cap \#\Sigma^* = \#L.$$

□

Theorem 9 *The family of context-sensitive languages is closed under controlled sequential and controlled parallel deletion with singletons.*

Proof. We can assume, without loss of generality, that L is a λ-free language. One can modify the construction in Theorem 3 such that g is not a rational transducer (CS is not closed under rational transductions) but a λ-free gsm (CS is closed under λ-free gsm's). Instead of erasing the words, the modified gsm g will replace each of their letters with a marker (in this way the erasing rules are transformed into non-erasing ones). As we are erasing only singletons, the morphism h which deletes the markers is a linear erasing with respect to $g(L)$. Indeed, if $p = \max\{\lg(\Delta(a))|\ a \in \Sigma\}$, then for any word $w \in g(L)$ we have $\lg(w) \leq (p+1)\lg(h(w))$. □

5 Scattered deletion

The variants of deletion dealt with so far have been considered only from the *compact* point of view. A *scattered* variant of the sequential deletion has been defined in [14]. Given two words u and v, if the letters of v can also be found in u, in the same order, the *scattered sequential deletion* erases them from u without taking into account their places; else, the result of the scattered sequential deletion of v from u is the empty set:

$$\begin{aligned} u \dashrightarrow v = \ & \{u_1u_2 \ldots u_{k+1} \in \Sigma^* |\ k \geq 1, u = u_1v_1u_2v_2 \ldots u_kv_ku_{k+1}, \\ & v = v_1v_2 \ldots v_k,\ u_i \in \Sigma^*, 1 \leq i \leq k+1, v_i \in \Sigma^*, 1 \leq i \leq k\}. \end{aligned}$$

The parallel variants of deletion do not have their natural scattered counterparts. Therefore we shall use in the sequel the term *scattered deletion* instead of scattered sequential deletion.

Example 8 Consider the languages $L_1 = \{a^nb^nc^n|\ n \geq 1\}$, $L_2 = \{ab^2c^3\}$. The scattered deletion of L_2 from L_1 is:

$$L_1 \dashrightarrow L_2 = \{a^{n+2}b^{n+1}c^n|\ n \geq 0\},$$

whereas the ordinary sequential deletion is $L_1 \longrightarrow L_2 = \emptyset$. □

Indeed, we notice that the necessary condition for a set $u \rightsquigarrow v$ to be nonempty is much weaker than in the case of sequential deletion. The word v does not need to be a subword of u but u has to contain the letters of v, in the same order.

In general, $L_1 \longrightarrow L_2 \subseteq L_1 \rightsquigarrow L_2$ for any two languages L_1, L_2 over an alphabet Σ.

As expected, the families of regular and context-free languages are closed under scattered deletion with regular languages because we have:

Theorem 10 *If L, R are languages over the alphabet Σ, R a regular one, the scattered deletion $L \rightsquigarrow R$ is the image of L through a gsm mapping.*

Proof. Let $A = (S, \Sigma, s_0, F, P)$ be a finite automaton that recognizes the language R. We construct the gsm with erasing:

$$g = (\Sigma, \Sigma, S, s_0, F, P') \text{ where } P' = P \cup \{sa \longrightarrow as \mid s \in S, a \in \Sigma\}.$$

We have $L \rightsquigarrow P = g(L)$. Indeed, given $u \in L$ as an input and $v \in R$, the gsm works as follows: the rules of P erase the symbols which come from v, in the correct order, whereas those of the form $sa \longrightarrow as$ cross the symbols that will remain in $u \rightsquigarrow v$. □

However, in general, the family of context-free languages is not closed under scattered deletion. In fact a stronger result holds.

Proposition 12 *There exist two linear languages L_1 and L_2 such that the scattered deletion of L_2 from L_1 is not a context-free language.*

Proof. Let L_1, L_2 be the linear languages

$$\begin{aligned} L_1 = &\quad \{a^n(bc)^n(df)^m \mid n, m \geq 1\}, \\ L_2 = &\quad \{c^n d^n \mid n \geq 1\}. \end{aligned}$$

One can easily see that:

$$(L_1 \rightsquigarrow L_2) \cap a^* b^* f^* = \{a^n b^n f^n \mid n \geq 1\}.$$

As CF is closed under intersection with regular sets but $\{a^n b^n f^n | n \geq 1\}$ is not a context-free language, it follows that the language $L_1 \rightsquigarrow L_2$ is not context-free. □

As it is not closed under right and left quotient with regular languages, CS is not closed under scattered deletion either.

Proposition 13 *The family of context-sensitive languages is not closed under scattered deletion with regular languages.*

Proof. Let Σ be an alphabet and denote $\Sigma' = \{a' |\ a \in \Sigma\}$. To every word $w \in \Sigma^*$ corresponds a word $w' \in \Sigma'^*$, obtained from w by changing every letter a into a'.

A λ-free gsm g can be easily constructed to satisfy:

$$g(L) = \{w_1 w_2' |\ w_1, w_2 \in \Sigma^*, w_1 w_2 \in L\}, \qquad (*)$$

where L is an arbitrary λ-free language over Σ.

If we define now the λ-free morphism $h : \Sigma^* \longrightarrow \Sigma'^*$, $h(a) = a'$, $\forall a \in \Sigma$, the following equality holds:

$$L_1/L_2 = \{[g(L_1) \dashrightarrow h(L_2)] \cap \Sigma^*\} \cup \{\lambda |\ \lambda \in L_1 \cap L_2\}$$

for every two languages L_1, L_2 over Σ.

As CS is closed under λ-free gsm mapping, λ-free morphism, union and intersection but it is not closed under right and left quotient with regular languages, it follows that it is not closed under scattered deletion with regular languages either. □

In the particular case when the language to be deleted is a singleton, CS is closed under scattered deletion. This follows because the amount of erasing is limited to the letters of a single word.

The controlled deletion does not have its natural scattered counterpart. However, a scattered variant of the permuted sequential deletion has been defined in [14]. Given two words u and v, if the letters of v can also be found in u, they are erased without taking into account their places or their order; else, the result of the permuted scattered SD of v from u is the empty set:

$$u \rightsquigarrow v = u \dashrightarrow com(v).$$

As we refer all the time to the sequential case, the term *permuted scattered deletion* will be used in the sequel instead of permuted scatterred sequential deletion.

The permuted scattered deletion is a generalization of SD and scattered SD. In general,

$$L_1 \longrightarrow L_2 \subseteq L_1 \dashrightarrow L_2 \subseteq L_1 \rightsquigarrow L_2,$$

for all languages L_1, L_2 over an alphabet Σ.

As $L_1 \rightsquigarrow L_2 = L_1 \dashrightarrow com(L_2)$, if one replaces the language to be deleted with a letter-equivalent language, the result of the permuted scattered SD remains unchanged.

None of the families REG, CF, CS is closed under permuted scattered SD. The operation is still family preserving if the language to be erased is a singleton.

Proposition 14 *The family of regular languages is not closed under permuted scattered deletion.*

Proof. Let L_1, L_2 be the regular languages:

$$L_1 = \{(bc)^m(df)^p \mid m, p \geq 1\},$$
$$L_2 = \{(cd)^n \mid n \geq 0\}.$$

One can prove that $(L_1 \rightsquigarrow L_2) \cap b^*f^* = \{b^m f^m \mid m \geq 1\}$. □

Proposition 15 *The family of context-free languages is not closed under permuted scattered deletion with regular languages.*

Proof. Let L_1, L_2 be the context-free respectively regular languages:

$$L_1 = \{a^n(bc)^n(df)^m \mid n, m \geq 1\},$$
$$L_2 = \{(cd)^n \mid n \geq 1\}.$$

The relation $(L_1 \rightsquigarrow L_2) \cap a^*b^*f^* = \{a^n b^n f^n \mid n \geq 1\}$ is obvious. □

Proposition 16 *The family of context-sensitive languages is not closed under permuted scattered deletion with regular languages.*

Proof. Let L be the recursively enumerable language (which is not context-sensitive) over an alphabet Σ and L_1 the context-sensitive language over $\Sigma \cup \{a, b\}$ defined in Proposition 2. Then, $(L_1 \rightsquigarrow a^*b) \cap \Sigma^* = L$. □

Any family of languages which is closed under scattered deletion with finite sets is closed under permuted scattered deletion with singletons.

6 Conclusions and open problems

Deletion operations and various related problems have recently become of interest and more thoroughly investigated. The notion of *deletion set* and decidability problems connected to it have been considered in [10], [11]. The particular case of sequential deletion where the language to be deleted is a singleton has been studied in [8]. The operation thus obtained is called *derivative* and it generalizes the classical notions of left/right derivative.

The dual insertion operations corresponding to the deletion ones have been defined in [6]. The situation where the insertion and deletion are inverse to each other, which is not generally the case, is considered in [7]. The results have some non-language-theoretical applications like those in cryptography: after inserting in some predefined ways garbage letters into the original message, the decryption is carried out by deleting them. The study of suitable pairs of insertion/deletion having the requested cryptographical properties is currently under progress.

Finally, language equations involving insertion and deletion operations and decidability problems arising from them are studied in [9].

Some specific open problems connected with the operations considered so far are the closure of the family of regular languages under iterated parallel deletion and under iterated parallel deletion with singletons.

7 Acknowledgements

We would like to thank Prof. Arto Salomaa, Prof. Grzegorz Rozenberg, Dr. Gheorghe Paun, Prof. Mathias Jantzen and Dr. Jarkko Kari for extended discussions and valuable suggestions.

References

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

[2] R.Book, M.Jantzen, C,Wrathall. Erasing strings. *Lecture Notes in Computer Science*, 104 (1981), pp.252-260.

[3] S.Ginsburg. *Algebraic and Automata-Theoretic Properties of Formal Languages*. North-Holland, Amsterdam, 1975.

[4] M.A.Harrison. *Introduction to Formal Language Theory*. Addison Wesley, Reading, Massachusetts, 1978.

[5] M.Jantzen. Confluent string rewriting and congruences. *EATCS Bulletin*, 28 (1986), pp.52-72.

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

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

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

[9] L.Kari. On some language equations. Submitted.

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

[11] L.Kari, A.Mateescu, G.Paun, A.Salomaa. On parallel deletions applied to a word. Submitted.

[12] T.Kimura. Formal description of communication behaviour. *Proceedings of John Hopkins Conference on Information Sciences and Systems*, Baltimore, 1979.

[13] A.Salomaa. *Formal Languages*. Academic Press, London, 1973.

[14] 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.