Closure Properties of Linear Languages under Operations of Linear Deletion

Tomáš Masopust *
masopust@fit.vutbr.cz

Abstract: In this paper, we give constructive proofs that linear languages are closed under operations of regular deletion and that they are not closed under operations of linear deletion. The operations are called random parallel, parallel, sequential, scattered sequential, and multiple scattered sequential deletion. In addition, we prove that every recursively enumerable language can be obtained from a linear language by linear random parallel, parallel, or sequential deletion. In the conclusion, we formulate two open problems.

Keywords: formal languages, regular languages, linear languages, regular deletion, linear deletion

1 Introduction

The language operations that delete some parts of strings play an important role in modern informatics, as bioinformatics, text algorithms, or cryptography (see [1, 2, 13]). So, it is no surprise that the formal language theory has payed a special attention to their study (see [6, 7, 9, 10, 11]).

This paper studies linear languages and their closure properties under operations of linear deletion. More precisely, it constructively proves that linear languages are closed under operations of regular random parallel, parallel, sequential, scattered sequential, and multiple scattered sequential deletion and that they are not closed under linear types of these operations. It also proves that every recursively enumerable language can be obtained from a linear language by linear random parallel, parallel, or sequential deletion. In the conclusion of this paper, we formulate two open problems.

2 Preliminaries

In this paper, we assume that the reader is familiar with the formal language theory (see [4, 5, 8, 12]).

Let RE, REC, CF, DCF, LIN, and REG denote the families of recursively enumerable, recursive, context-free, deterministic context-free, linear, and regular languages, respectively.

3 Definitions and Examples

Let $L, K \subseteq \Sigma^*$ be two languages.

* Department of Information Systems, Faculty of Information Technology, Brno University of Technology, Bozetechova 2, 612 66 Brno

Definition 1. *Random parallel deletion* of languages L and K is denoted by $[\perp, L, K]$ and defined as the set $[\perp, L, K] = \{u_1u_2 \dots u_nu_{n+1} \in \Sigma^* : u_1x_1u_2 \dots u_nx_nu_{n+1} \in L,\ x_i \in K,\ 1 \le i \le n,\ n \ge 1\}$.

Example 1. Let $L = \{abababa, aababa, abaabaaba\}$ and $K = \{aba\}$.

- $[\perp, \{abababa\}, K] = \{baba, abba, abab, b\}$.
- $[\perp, \{aababa\}, K] = \{aba, aab\}$.
- $[\perp, \{abaabaaba\}, K] = \{abaaba, aba, \varepsilon\}$.
- $[\perp, L, K] = \{baba, abba, abab, b, aba, aab, abaaba, \varepsilon\}$.

Notice that random parallel deletion has to delete at least one substring belonging to K. On the other hand, this operation can also delete all substrings belonging to K. Thus, as a special case, we get two operations—an operation that delete exactly one substring belonging to K (sequential deletion) and an operation that delete all substrings belonging to K (parallel deletion). More precisely, in parallel deletion, no u_i from the definition of random parallel deletion can contain a substring belonging to K (except for ε).

Definition 2. *Parallel deletion* of languages L and K is denoted by $[\perp_a, L, K]$ and defined as the set $[\perp_a, L, K] = \{u_1u_2 \dots u_nu_{n+1} \in \Sigma^* : u_1x_1u_2 \dots u_nx_nu_{n+1} \in L,\ x_j \in K,\ \{u_i\} \cap \Sigma^*(K \setminus \{\varepsilon\})\Sigma^* = \emptyset,\ 1 \le i \le n+1,\ 1 \le j \le n,\ n \ge 1\}$.

Example 2. Let $L = \{abababa, aababa, abaabaaba\}$ and $K = \{aba\}$.

- $[\perp_a, \{abababa\}, K] = \{b, abba\}$.
- $[\perp_a, \{aababa\}, K] = \{aba, aab\}$.
- $[\perp_a, \{abaabaaba\}, K] = \{\varepsilon\}$.
- $[\perp_a, L, K] = \{b, abba, aba, aab, \varepsilon\}$.

Definition 3. *Sequential deletion* of languages L and K is denoted by $[\perp_1, L, K]$ and defined as the set $[\perp_1, L, K] = \{u_1u_2 \in \Sigma^* : u_1xu_2 \in L,\ x \in K\}$.

Example 3. Let $L = \{abababa, ab, aba\}$ and $K = \{aba\}$.

- $[\perp_1, \{abababa\}, K] = \{baba, abba, abab\}$.
- $[\perp_1, \{ab\}, K] = \emptyset$.
- $[\perp_1, \{aba\}, K] = \{\varepsilon\}$.
- $[\perp_1, L, K] = \{baba, abba, abab, \varepsilon\}$.

Definition 4. *Scattered sequential deletion* of languages L and K is denoted by $[\perp_{1s}, L, K]$ and defined as the set $[\perp_{1s}, L, K] = \{u_1u_2 \dots u_nu_{n+1} \in \Sigma^* : u_1x_1u_2 \dots u_nx_nu_{n+1} \in L,\ x_1x_2 \dots x_n \in K,\ n \ge 1\}$.

Example 4. Let $L = \{abacba\}$ and $K = \{ab, ca\}$.

- $[\perp_{1s}, \{abacba\}, \{ab\}] = \{acba, baca, abca\}$.
- $[\perp_{1s}, \{abacba\}, \{ca\}] = \{abab\}$.
- $[\perp_{1s}, \{abacba\}, K] = \{acba, baca, abca, abab\}$.

Definition 5. *Multiple scattered sequential deletion* of L and K is denoted by $[\perp_s, L, K]$ and defined as the set $[\perp_s, L, K] = \{u_1u_2 \dots u_nu_{n+1} \in \Sigma^* : u_1x_1u_2 \dots u_nx_nu_{n+1} \in L,\ x_1x_2 \dots x_n \in K^+,\ n \geq 1\}$.

Example 5. Let $L = \{abacba\}$ and $K = \{ab, ca\}$.

- $[\perp_s, \{abacba\}, \{ab\}] = \{acba, baca, abca, ca\}$.
- $[\perp_s, \{abacba\}, \{ca\}] = \{abab\}$.
- $[\perp_s, \{abacba\}, \{ab, ca\}] = \{acba, baca, abca, ca, abab, ab\}$.

For any two families of languages $\mathcal{X}$ and $\mathcal{Y}$ denote by $\langle x, \mathcal{X}, \mathcal{Y}\rangle$ the set $\langle x, \mathcal{X}, \mathcal{Y}\rangle = \{[x, L, K] : L \in \mathcal{X},\ K \in \mathcal{Y}\}$, where $x \in \{\perp, \perp_a, \perp_1, \perp_{1s}, \perp_s\}$.

4 Results

Linear languages have been proved to be closed under operations of regular deletion (see [9]). The proofs given there are not constructive. Here we describe the constructions.

Theorem 6. $\langle x, LIN, REG\rangle = LIN$, $x \in \{\perp, \perp_1, \perp_{1s}, \perp_s\}$.

Proof. Let $L \in LIN$. Then, $L = [x, L, \{\varepsilon\}] \in \langle x, LIN, REG\rangle$, $x \in \{\perp, \perp_1, \perp_{1s}, \perp_s\}$.

Let $L \in LIN$ and $K \in REG$. Without loss of generality, there is a proper linear grammar $G_L = (N_L, \Sigma_L, P_L, S_L)$ and a regular grammar $G_K = (N_K, \Sigma_K, P_K, S_K)$ such that S_K does not occur on the right-hand side of any rule, $L = \mathcal{L}(G_L)$, and $K = \mathcal{L}(G_K)$. (Notice that $S_K \rightarrow \varepsilon$ is the only possible ε-rule in G_K.) Let us construct new linear grammar $G = (N, \Sigma_L, P, S)$, where $N = \{S\} \cup \{\langle x, B, y, U, V\rangle : x, y \in \Sigma_L^*,\ |x|, |y| \leq \max\{|u|, |v| : A \rightarrow uBv \in P_L\},\ B \in N_L \cup \{\varepsilon\},\ U, V \in N_K \cup \{\varepsilon\}\}$, and P contains rules of the following forms (depending on x):

$x = \perp$:

1) $S \rightarrow \langle \varepsilon, S_L, \varepsilon, S_K, \varepsilon\rangle$
2) $\langle ax, A, yb, U, \varepsilon\rangle \rightarrow \langle ax, A, y, U, \varepsilon\rangle\, b$
3) $\langle ax, A, yb, S_K, V\rangle \rightarrow a\, \langle x, A, yb, S_K, V\rangle$
4) $\langle ax, A, yb, U, Y\rangle \rightarrow \langle x, A, yb, V, Y\rangle$ if $U \rightarrow aV \in P_K,\ V \in N_K \cup \{\varepsilon\}$
5) $\langle ax, A, yb, U, Y\rangle \rightarrow \langle ax, A, y, U, X\rangle$ if $X \rightarrow bY \in P_K,\ X \in N_K$
6) $\langle ax, A, yb, U, S_K\rangle \rightarrow \langle ax, A, y, U, S_K\rangle\, b$
7) $\langle ax, A, yb, U, S_K\rangle \rightarrow \langle ax, A, yb, U, \varepsilon\rangle$
8) $\langle ax, A, yb, \varepsilon, X\rangle \rightarrow a\, \langle x, A, yb, \varepsilon, X\rangle$
9) $\langle ax, A, yb, \varepsilon, X\rangle \rightarrow \langle ax, A, yb, S_K, X\rangle$
10) $\langle \varepsilon, A, \varepsilon, X, Y\rangle \rightarrow \langle x, B, y, X, Y\rangle$ if $A \rightarrow xBy \in P_L$
11) $\langle \varepsilon, \varepsilon, \varepsilon, X, X\rangle \rightarrow \varepsilon$

We prove $w \in \mathcal{L}(G)$ if and only if $w \in [\perp, L, K]$.

Claim 7. *Let* $\langle x, B, y, X, Y\rangle \Rightarrow_G^* w$, $X, Y \in \{S_K, \varepsilon\}$. *Then,*

$$xBy \Rightarrow_{G_L}^* w_1x_1w_2x_2 \dots w_nx_nw_{n+1},$$

$n \geq 0$, $w = w_1w_2 \dots w_{n+1}$, $S_K \Rightarrow_{G_K}^* x_i$, $1 \leq i \leq n$.

Proof. Consider a derivation $\langle x, B, y, X, Y\rangle \Rightarrow_G^* w_1 \dots w_k \langle \varepsilon, \varepsilon, \varepsilon, Z, Z\rangle w_{k+1} \dots w_{n+1} \Rightarrow w = w_1 w_2 \dots w_{n+1}$, where w_i denotes the longest string such that there was no use of any rule of type 4, for $i = 1, \dots, k$, and of type 5, for $i = k+1, \dots, n+1$, in the derivation. We prove the claim by induction on n.

Basis: For $n = 0$, there is no use of any rule of type 4 or 5 in the derivation of w. If $X = Y = S_K$ or $X = Y = \varepsilon$ we have $\langle x, B, y, X, X\rangle$. If $X = \varepsilon$ and $Y = S_K$ we use one of the rules of type 7 or 9 to obtain $\langle x, B, y, X, X\rangle$ or $\langle x, B, y, Y, Y\rangle$. Then, by rewriting B in G_L in the same way as in G, we get $xBy \Rightarrow^* w$. Notice that if $X = S_K$ and $Y = \varepsilon$ we have to use at least one of the rules of type 4 or 5.

Induction hypothesis: Suppose that the claim holds for all $k \leq n$ and consider $n+1$. We show here the case $w_1 x_1$ is derived before w_{n+1}; the other cases would be done analogously.

The derivation must be $\langle x, B, y, X, Y\rangle \Rightarrow^* w_1 \langle x', C, y', S_K, Y'\rangle w'_{n+1}$ and, by the rules of type 4 applied between generating w_1 and w_2, we get

$$w_1 \langle x', C, y', S_K, Y'\rangle w'_{n+1} \Rightarrow^* w_1 \langle x'', D, y'', \varepsilon, Y''\rangle w''_{n+1} \Rightarrow^* w,$$

where $Y' = Y'' = \varepsilon$ if $Y = \varepsilon$, or if $Y = S_K$ then $Y' \in \{S_K, \varepsilon\}$ and if $Y' = \varepsilon$ then $Y'' = Y'$ else $Y'' \in \{S_K, \varepsilon\}$. It means $x'Cy' \Rightarrow^* x_1 x'' D y''$ and $S_K \Rightarrow^* x_1$. It follows from above and the induction hypothesis that

$$xBy \Rightarrow^* w_1 x' C y' w'_{n+1} \Rightarrow^* w_1 x_1 x'' D y'' w''_{n+1} \Rightarrow^* w_1 x_1 w_2 \dots w_n x_n w_{n+1},$$

$S_K \Rightarrow^* x_i$, $i = 1, \dots, n$. □

If $S \Rightarrow \langle \varepsilon, S_L, \varepsilon, S_K, \varepsilon\rangle \Rightarrow^* w$. Then, from the previous claim and the fact that we have to use at least one of the rules of type 4 or 5, $S_L \Rightarrow^* w_1 x_1 w_2 \dots w_n x_n w_{n+1}$, $n \geq 1$, $w = w_1 w_2 \dots w_{n+1}$, $S_K \Rightarrow^* x_i$, $i = 1, \dots, n$. Thus, $w \in [\perp, L, K]$.

Conversely, let $w \in [\perp, L, K]$. There is a derivation $S_L \Rightarrow^* w_1 x_1 w_2 \dots w_n x_n w_{n+1}$, $n \geq 1$, $x_i \in K$, $i = 1, \dots, n$. Then, $S \Rightarrow \langle \varepsilon, S_L, \varepsilon, S_K, \varepsilon\rangle$. Next, while w_i is generated in G_L we generate w_i in G by the rules of type 2, 3, 6, and 8, and while x_j is generated in G_L we delete x_j in G by the rules of type 4 and 5. Thus, $w \in \mathcal{L}(G)$.

$x = \perp_1$: We eliminate the rules allowing to delete more than one substring, i.e. rules of type 7 and 9.

1) $S \rightarrow \langle \varepsilon, S_L, \varepsilon, S_K, \varepsilon\rangle$
2) $\langle ax, A, yb, U, \varepsilon\rangle \rightarrow \langle ax, A, y, U, \varepsilon\rangle\, b$
3) $\langle ax, A, yb, S_K, V\rangle \rightarrow a\, \langle x, A, yb, S_K, V\rangle$
4) $\langle ax, A, yb, U, Y\rangle \rightarrow \langle x, A, yb, V, Y\rangle$ if $U \rightarrow aV \in P_K$, $V \in N_K \cup \{\varepsilon\}$
5) $\langle ax, A, yb, U, Y\rangle \rightarrow \langle ax, A, y, U, X\rangle$ if $X \rightarrow bY \in P_K$, $X \in N_K$
6) $\langle ax, A, yb, U, S_K\rangle \rightarrow \langle ax, A, y, U, S_K\rangle\, b$
8) $\langle ax, A, yb, \varepsilon, X\rangle \rightarrow a\, \langle x, A, yb, \varepsilon, X\rangle$
10) $\langle \varepsilon, A, \varepsilon, X, Y\rangle \rightarrow \langle x, B, y, X, Y\rangle$ if $A \rightarrow xBy \in P_L$
11) $\langle \varepsilon, \varepsilon, \varepsilon, X, X\rangle \rightarrow \varepsilon$

$x = \perp_{1s}$: In each state, we can either generate a next symbol or delete it.

1) $S \rightarrow \langle \varepsilon, S_L, \varepsilon, S_K, \varepsilon \rangle$
2) $\langle ax, A, yb, U, Y \rangle \rightarrow a \langle x, A, yb, U, Y \rangle$
3) $\langle ax, A, yb, U, Y \rangle \rightarrow \langle ax, A, y, U, Y \rangle b$
4) $\langle ax, A, yb, U, Y \rangle \rightarrow \langle x, A, yb, V, Y \rangle$ if $U \rightarrow aV \in P_K, V \in N_K \cup \{\varepsilon\}$
5) $\langle ax, A, yb, U, Y \rangle \rightarrow \langle ax, A, y, U, X \rangle$ if $X \rightarrow bY \in P_K, X \in N_K$
6) $\langle \varepsilon, A, \varepsilon, X, Y \rangle \rightarrow \langle x, B, y, X, Y \rangle$ if $A \rightarrow xBy \in P_L$
7) $\langle \varepsilon, \varepsilon, \varepsilon, X, X \rangle \rightarrow \varepsilon$

$x = \perp_s$: As K^+ is regular, for K regular, the proof is the same as for $x = \perp_{1s}$. We simply use K^+ instead of K.

□

Example 6. Let $G_L = (\{S_L\}, \{a, b\}, \{S_L \rightarrow aS_Lb, S_L \rightarrow ab\}, S_L)$ be a linear grammar. Then, $L = \mathcal{L}(G_L) = \{a^n b^n : n \geq 1\}$. Next, let $G_K = (\{S_K, A\}, \{a, b\}, \{S_K \rightarrow aA, A \rightarrow bA, A \rightarrow b\}, S_K)$ be a regular grammar. Then, $K = \mathcal{L}(G_K) = \{a\} \cdot \{b\}^+$. From the construction described above and by reducing the obtained grammar, we get a grammar, G, generating language $[\perp, L, K] = \{a^m b^n : m \geq n \geq 0\}$. $G = (\{S, \langle a, S_L, b, S_K, \varepsilon \rangle, \langle \varepsilon, S_L, b, S_K, A \rangle\}, \{a, b\}, P, S)$, where P contains following rules:

- $S \rightarrow a \langle a, S_L, b, S_K, \varepsilon \rangle b \mid a \langle \varepsilon, S_L, b, S_K, A \rangle \mid ab \mid a \mid \varepsilon$
- $\langle a, S_L, b, S_K, \varepsilon \rangle \rightarrow a \langle a, S_L, b, S_K, \varepsilon \rangle b \mid a \langle \varepsilon, S_L, b, S_K, A \rangle \mid ab \mid a$
- $\langle \varepsilon, S_L, b, S_K, A \rangle \rightarrow a \langle \varepsilon, S_L, b, S_K, A \rangle \mid a$

While nonterminal $\langle a, S_L, b, S_K, \varepsilon \rangle$ corresponds to a situation none substring belonging to K is deleted from a string generated in G_L, nonterminal $\langle \varepsilon, S_L, b, S_K, A \rangle$ says that some bs have already been deleted and that only as can be generated now on.

Theorem 8. $\langle \perp_a, LIN, REG \rangle = LIN$.

Proof. It is easy to see that $LIN \subseteq \langle \perp_a, LIN, REG \rangle$.

Let $L \in LIN$ and $K \in REG$. Without loss of generality, there is a proper linear grammar $G_L = (N_L, \Sigma_L, P_L, S_L)$ and a regular grammar $G_K = (N_K, \Sigma_K, P_K, S_K)$ such that S_K does not occur on the right-hand side of any rule, $L = \mathcal{L}(G_L)$, and $K = \mathcal{L}(G_K)$. ($S_K \rightarrow \varepsilon$ is the only possible ε-rule in G_K.) Let us construct new linear grammar $G = (N, \Sigma_L, P, S)$, where $N = \{S\} \cup \{\langle x, B, y, U, V, M, N \rangle : x, y \in \Sigma_L^*, |x|, |y| \leq \max\{|u|, |v| : A \rightarrow uBv \in P_L\}, B \in N_L \cup \{\varepsilon\}, U, V \in N_K \cup \{\varepsilon\}, M, N \subseteq N_K \cup \{\varepsilon\}\}$, and P contains rules of the following forms:

1) $S \rightarrow \langle \varepsilon, S_L, \varepsilon, S_K, \varepsilon, \{S_K\}, \{\varepsilon\} \rangle$
2) $\langle ax, A, yb, U, \varepsilon, M, N \rangle \rightarrow \langle ax, A, y, U, \varepsilon, M, N' \rangle b$ if $\varepsilon \notin M, S_K \notin N, b \in \Sigma_L$
3) $\langle ax, A, yb, S_K, V, M, N \rangle \rightarrow a \langle x, A, yb, S_K, V, M', N \rangle$ if $\varepsilon \notin M, S_K \notin N, a \in \Sigma_L$
4) $\langle ax, A, yb, U, Y, M, N \rangle \rightarrow \langle x, A, yb, V, Y, \{S_K\}, N \rangle$ if $\varepsilon \notin M, S_K \notin N$, $U \rightarrow aV \in P_K, V \in N_K \cup \{\varepsilon\}$
5) $\langle ax, A, yb, U, Y, M, N \rangle \rightarrow \langle ax, A, y, U, X, M, \{\varepsilon\} \rangle$ if $\varepsilon \notin M, S_K \notin N$, $X \rightarrow bY \in P_K$
6) $\langle ax, A, yb, U, S_K, M, N \rangle \rightarrow \langle ax, A, y, U, S_K, M, N' \rangle b$ if $\varepsilon \notin M, S_K \notin N, b \in \Sigma_L$
7) $\langle ax, A, yb, U, S_K, M, N \rangle \rightarrow \langle ax, A, yb, U, \varepsilon, M, N \rangle$
8) $\langle ax, A, yb, \varepsilon, X, M, N \rangle \rightarrow a \langle x, A, yb, \varepsilon, X, M', N \rangle$ if $\varepsilon \notin M, S_K \notin N, a \in \Sigma_L$

9) $\langle ax, A, yb, \varepsilon, X, M, N\rangle \to \langle ax, A, yb, S_K, X, M, N\rangle$

10) $\langle \varepsilon, A, \varepsilon, X, Y, M, N\rangle \to \langle x, B, y, X, Y, M, N\rangle$ if $A \to xBy \in P_L$, $\varepsilon \notin M$, $S_K \notin N$

11) $\langle \varepsilon, \varepsilon, \varepsilon, X, X, M, N\rangle \to \varepsilon$ if $\varepsilon \notin M$, $S_K \notin N$

where $M' = \{S_K\} \cup \{D \in N_K \cup \{\varepsilon\} : A \to aD \in P_K, A \in M\}$ and $N' = \{\varepsilon\} \cup \{D \in N_K : D \to bC \in P_K, C \in N\}$.

Notice that in the rules of type 2, 3, 6, 8 it holds that $a, b \in \Sigma_L$, so we never use rule $S_K \to \varepsilon$ here.

The proof of $w \in \mathcal{L}(G)$ if and only if $w \in [\perp_a, L, K]$ is very similar to the one of the previous theorem. Moreover, here we have two sets of nonterminals, M and N, in which we parallelly simulate all derivations in G_K generating the same symbol as the derivation in G does. In M we do the top-down simulation, while in N we do the bottom-up simulation. More precisely, in each derivation step we check, by rewriting all the nonterminals from M and N according to rules in G_K generating the same symbol as G just generates (to the left, for M, and to the right, for N), whether we have generated a substring belonging to K. We also add a new simulation from S_K, i.e. we check whether the substring starting to be generated belongs to K.

Notice also that if $\varepsilon \in K$ then whenever we have a substring belonging to K of length at least 2, we can say that we have deleted ε from it. So, the only substrings belonging to K we do not want to appear in the derived string are of length 1, i.e. the strings from $K \cap \Sigma_L$. For the derivation to the left, the corresponding sequence of derivation steps is 3, 4 ($S_K \to \varepsilon$), and 9. □

Now, we prove that the operations of linear deletion are very powerful—linear languages with the operation of linear deletion characterize recursively enumerable languages.

Theorem 9. $\langle x, LIN, LIN\rangle = RE$, $x \in \{\perp, \perp_a, \perp_1\}$.

Proof. It is not hard to construct a Turing machine accepting $\langle x, LIN, LIN\rangle$, $x \in \{\perp, \perp_a, \perp_1\}$.

Now, suppose $L \in RE$, $L \subseteq \Sigma^*$, $\Sigma = \{a_1, \dots, a_n\}$. *Extended Post correspondence problem*, P, is a tuple $P = (\{(u_1, v_1), \dots, (u_r, v_r)\}, (z_{a_1}, \dots, z_{a_n}))$, where $u_i, v_i, z_a \in \{0,1\}^*$ for $i = 1, \dots, r$, and $a \in \Sigma$. The language represented by P is the set $\mathcal{L}(P) = \{x_1x_2 \dots x_n \in \Sigma^* : \exists s_1, \dots, s_l \in \{1, \dots, r\}, l \geq 1, v_{s_1} \dots v_{s_l} = u_{s_1} \dots u_{s_l} z_{x_1} \dots z_{x_n}\}$. It is known that for each recursively enumerable language, L, there is an extended Post correspondence problem, P, such that $\mathcal{L}(P) = L$ (see [3, Theorem 1]). Thus, $x_1x_2 \dots x_n \in L$ if and only if $x_1x_2 \dots x_n \in \mathcal{L}(P)$. Generate $x_1x_2 \dots x_n$ as follows:

$$\begin{aligned}
S' &\Rightarrow \$S \Rightarrow \$z_{x_n}^R S x_n \Rightarrow \$z_{x_n}^R z_{x_{n-1}}^R S x_{n-1} x_n \\
&\Rightarrow^* \$z_{x_n}^R z_{x_{n-1}}^R \dots z_{x_1}^R S x_1 \dots x_{n-1} x_n \\
&\Rightarrow^* \$z_{x_n}^R z_{x_{n-1}}^R \dots z_{x_1}^R A \$ x_1 \dots x_{n-1} x_n \\
&\Rightarrow^* \$z_{x_n}^R z_{x_{n-1}}^R \dots z_{x_1}^R u_{s_l}^R A v_{s_l} \$ x_1 \dots x_{n-1} x_n \\
&\Rightarrow^* \$z_{x_n}^R z_{x_{n-1}}^R \dots z_{x_1}^R u_{s_l}^R \dots u_{s_1}^R \# v_{s_1} \dots v_{s_l} \$ x_1 \dots x_{n-1} x_n \\
&= \$(u_{s_1} \dots u_{s_l} z_{x_1} \dots z_{x_n})^R \#(v_{s_1} \dots v_{s_l}) \$ x_1 \dots x_n
\end{aligned}$$

$$= \quad \$w_1^R\#w_2\$x_1x_2\ldots x_n$$

and $x_1x_2\ldots x_n \in L$ if and only if there are w_1, w_2 such that $w_1 = w_2$,

where $z_{x_i}, u_{s_j}, v_{s_j} \in \{0,1\}^*$, $\$, \# \notin \Sigma \cup \{0,1\}$.

In addition, there is a linear grammar, G', such that $\mathcal{L}(G') = \{\$w^R\#w\$: w \in \{0,1\}^*\}$. Thus, $L = [x, \mathcal{L}(G), \mathcal{L}(G')]$, $x \in \{\perp, \perp_a, \perp_1\}$. □

Corollary 10. $\langle x, LIN, DCF\rangle = RE$, $x \in \{\perp, \perp_a, \perp_1\}$.

Proof. Language $\mathcal{L}(G')$ from Theorem 9 is deterministic context-free language. □

Theorem 11. $REC \subset \langle x, LIN, LIN\rangle \subseteq RE$, $x \in \{\perp_{1s}, \perp_s\}$.

Proof. Let L be recursively enumerable language, $L \subseteq \Sigma^*$, $\Sigma \cap \{0,1\} = \emptyset$. The proof follows from Theorem 9 since $L = [x, \mathcal{L}(G), \mathcal{L}(G')] \cap \Sigma^*$. If $[x, \mathcal{L}(G), \mathcal{L}(G')]$ is recursive, then so is L. Therefore, for $L \in RE \setminus REC$ the language $[x, \mathcal{L}(G), \mathcal{L}(G')]$ is not recursive language. □

Corollary 12. *If $\langle \perp_{1s}, LIN, LIN\rangle$ is closed under intersection with a regular language, then $\langle \perp_{1s}, LIN, LIN\rangle = RE$. If $\langle \perp_s, LIN, LIN\rangle$ is closed under intersection with a regular language, then $\langle \perp_s, LIN, LIN\rangle = RE$.*

As a corollary we get the following results for context-free languages.

Theorem 13. $\langle x, CF, CF\rangle = RE$, $x \in \{\perp, \perp_a, \perp_1\}$.

Proof. $RE = \langle x, LIN, LIN\rangle \subseteq \langle x, CF, CF\rangle \subseteq RE$. □

Theorem 14. $REC \subset \langle x, CF, CF\rangle \subseteq RE$, $x \in \{\perp_{1s}, \perp_s\}$.

Proof. $REC \subset \langle x, LIN, LIN\rangle \subseteq \langle x, CF, CF\rangle \subseteq RE$. □

5 Open Problems

Here we summarize two open problems:

1. Is it true that $\langle \perp_{1s}, LIN, LIN\rangle = RE$?
2. Is it true that $\langle \perp_s, LIN, LIN\rangle = RE$?

Bibliography

1. M. Amos. *DNA computation*. PhD thesis, University of Warwick, 1997.
2. M. Crochemore and W. Rytter. *Text algorithms*. Oxford University Press, Oxford, 1994.
3. V. Geffert. Context-free-like forms for the phrase-structure grammars. In *Proc. Math. Found. Comput. Sci.*, volume 324 of *Lecture Notes in Comput. Sci.*, pages 309–317, Springer-Verlag, 1988.
4. M. A. Harrison. *Introduction to Formal Language Theory*. Addison-Wesley, 1978.
5. J. E. Hopcroft and J. O. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979.
6. L. Kari. *On insertion and deletion in formal languages*. PhD thesis, University of Turku, Finland, Department of Mathematics, 1991.
7. L. Kari. Deletion operations: closure properties. *Int. J. Comput. Math.*, 52:23–42, 1994.
8. A. Meduna. *Automata and Languages: Theory and Applications*. Springer, London, 2000.

9. A. Meduna and T. Masopust. Closure properties of trios under operations of regular deletion. Submitted.
10. A. Meduna and T. Masopust. Deletion in formal language theory. In preparation.
11. A. Meduna and M. Vítek. New language operations in formal language theory. *Schedae Informaticae*, 13:123–150, 2004.
12. A. Salomaa. *Formal languages*. Academic Press, New York, 1973.
13. J. Seberry and J. Pieprzyk. *Cryptography: an introduction to computer security*. Prentice Hall, New York, 1989.