Language deletion on trajectories

Lila Kari[1] and Petr Sosík[1,2]

[1]Department of Computer Science
The University of Western Ontario, London, ON, Canada, N6A 5B7,
{lila,sosik}@csd.uwo.ca

[2]Institute of Computer Science, Silesian University, Opava, Czech Republic

October 31, 2003

Abstract

We introduce a general framework for the study of properties of sequential word deletion operations over words and languages. A *trajectory* gives a natural syntactical constraint on the scattered deletion of a word from another one, with an intuitive geometrical interpretation. Moreover, deletion on trajectories is the inverse of the shuffle on trajectories which is being studied since 1996. These operations, besides being a classical topic in formal language theory itself, were shown to be useful e.g. in concurrent processes modelling and recently also in biocomputing area.

We study algebraic and closure properties of the deletion on trajectories. Our results in the section devoted to operand problems generalize and extend previously published results concerning solvability of language equations. Finally we define a new operation — *substitution on trajectories*, and study its properties. The operation is applicable among others in coding theory for modelling noisy channels.

1 Introduction

The *binary word operations*, whose simplest examples are catenation and left/right quotient, have been extensively studied in formal language theory. Perhaps the basic topic connected with them is the closure of various classes of formal languages under these operations. The closure properties allow one to construct complex languages as a composition of simpler ones, or, conversely, to decompose a complex objects (automata, grammars) into a structure of simple ones. They are also of crucial importance for forming algebraic structures in the formal language theory, as abstract families of languages (AFL) [10].

Another important topic represent *language equations* with these operations, see e.g. [11]. Various problems connected with them have been studied during the last two decades, see e.g. [2, 3]. The scope of the studied sequential insertion/deletion operations include insertion, shuffle, literal shuffle, deletion, bipolar deletion, scattered deletion, as well as their iterated and regulated versions. Recently, the applications of such language equations was shown in coding theory for modelling noisy channels [4] or in biocomputing for characterization of sets of codewords for DNA computing [5].

Many of the mentioned insertion/deletion operations share the same principle, while they differ in "shape" of the insertion/deletion of a word to/from another one. These similarities give rise to a more general approach, when this "shape" is characterized by a set of special binary strings called the *trajectories.* This approach allows one not only to generalize important results for many frequently used operations into one unified framework, but also to give a broader overview of their nature and mutual relations. Trajectories were introduced in [8, 9] for a class of *insertion* operations, and their closure properties algebraic properties and applications to concurrent processes modelling were studied. Further related problems were addressed in [6, 7]. In this paper we extend their scope also to *deletion* operation which is a natural inverse of insertion.

The paper is organized as follows. In section 2 we define binary insertion and deletion word operations. In the section 3 we resume some of the results from [8, 9], generalizing insertion word operations using trajectories. In the following section we introduce analogous generalization for deletion operations, which are natural inverses of insertion. In sections 5 and 6 we study the closure properties and algebraic properties of deletion on trajectories. While closure properties are similar as in the case of insertion operations, the algebraic properties differ substantially.

We continue in section 7 by investigation of solvability of the regular language equations of the form $L_1 \rightsquigarrow_T L_2 = R$, where $\rightsquigarrow_T$ is the deletion on trajectories. Our results generalize several previous singular theorems from e.g. [2, 3]. In section 8 we extend the study to the case of context free languages and we give an exact characterization of the sets of trajectories for which the validity of the above equation is decidable.

Finally, in section 9 we introduce the new word operations — substitution and difference on trajectories, study their closure properties and mention their possible applications in modelling noisy channels.

2 Definitions and preliminaries

An *alphabet* is a finite and nonempty set of symbols. In the sequel we shall use a fixed alphabet Σ. Σ is assumed to be non-singleton, if not stated otherwise. The set of all words (over Σ) is denoted by Σ^*. This set includes the *empty word* λ. The length of a word w is denoted by $|w|$. $|w|_x$ denotes the number of occurrences of x within u, for $w, x \in \Sigma^*$.

For a nonnegative integer n and a word w, we use w^n to denote the word that consists of n concatenated copies of w.

A language L is a set of words, or equivalently a subset of Σ^*. A language is said to be λ-free if it does not contain the empty word. For a language L, we write L_λ to denote $L \cup \{\lambda\}$. If n is a nonnegative integer, we write L^n for the language consisting of all words of the form $w_1 \cdots w_n$ such that each w_i is in L. We also write L^* for the language $L^0 \cup L^1 \cup L^2 \cup \cdots$ and L^+ for the language $L^* - \{\lambda\}$. The notation L^c represents the complement of the language L, that is, $L^c = \Sigma^* - L$. For the classes of regular, context-free, and context sensitive languages, we use the notations REG, CF and CS, respectively.

A nondeterministic finite automaton with λ productions (or transitions), a λ-*NFA* for short, is a quintuple $A = (S, \Sigma, s_0, F, P)$ such that S is the finite and nonempty set of states, s_0 is the start state, F is the set of final states, and P is the set of productions of the form $sx \to t$, where s and t are states in S, and x is either a symbol in Σ or the empty word. If there is no production with $x = \lambda$, the automaton is called an *NFA*. If for every two productions of the form $sx_1 \to t_1$ and $sx_2 \to t_2$ of an NFA we have that $x_1 \neq x_2$ then the automaton is called a *DFA* (deterministic finite automaton). The language accepted by the automaton A is denoted by $L(A)$. The automaton is called *trim* if every state is reachable from the start state and can reach a final state in F (when $F \neq \emptyset$). The *size* $|A|$

of the automaton A is the number $|S| + |P|$. Note that the number $|S|$ of states of a trim automaton is at most $1 + |P|$; therefore, the size of such an automaton is $|A| = \Theta(|P|)$.

A *finite transducer* (in standard form) is a sextuple $T = (S, \Sigma, \Sigma', s_0, F, P)$ such that Σ' is the output alphabet, the components S, s_0, F are as in the case of λ-NFAs, and the set P consists of productions of the form $sx \to yt$ where s and t are states in S, $x \in \Sigma \cup \{\lambda\}$ and $y \in \Sigma' \cup \{\lambda\}$. If x is nonempty for every production then the transducer is called a *gsm* (generalized sequential machine). If, in addition, y is nonempty for every production then the transducer is called a λ*-free gsm*. The *relation realized by* the transducer T is denoted by $R(T)$. The concept of a trim transducer is the same as that in the case of automata. The size $|T|$ of the transducer T (in standard form) is $|S| + |P|$. Again, when the transducer is trim its size is $|T| = \Theta(|P|)$.

We refer the reader to [12] or [13] for further details on automata and formal languages.

A *binary word operation* is a mapping $\diamondsuit : \Sigma^* \times \Sigma^* \to 2^{\Sigma^*}$, where 2^{Σ^*} is the set of all subsets of Σ^*.

The *characteristic relation* of $\diamondsuit$ is

$$C_\diamondsuit = \{(w, u, v) : w \in u \diamondsuit v\}.$$

For any languages X and Y,

$$X \diamondsuit Y = \bigcup_{u \in X, v \in Y} u \diamondsuit v. \tag{1}$$

It should be noted that every subset B of $\Sigma^* \times \Sigma^* \times \Sigma^*$ defines a unique binary word operation whose characteristic relation is exactly B.

Definition 2.1 *([3]) Let $\diamondsuit$ be a operation. The left inverse $\diamondsuit^l$ of $\diamondsuit$ is defined as*

$$w \in (x \diamondsuit v) \text{ iff } x \in (w \diamondsuit^l v), \text{ for all } v, x, w \in \Sigma^*,$$

and the right inverse $\diamondsuit^r$ of $\diamondsuit$ is defined as

$$w \in (u \diamondsuit y) \text{ iff } y \in (u \diamondsuit^r w), \text{ for all } u, y, w \in \Sigma^*.$$

Definition 2.2 *Let $\diamondsuit$ be a binary word operation. The word operation $\diamondsuit'$ defined by $u \diamondsuit' v = v \diamondsuit u$ is called* reversed $\diamondsuit$.

It should be clear that, for every binary operation $\diamondsuit$, the triple (w, u, v) is in $C_\diamondsuit$ if and only if (u, w, v) is in $C_{\diamondsuit^l}$ if and only if (v, u, w) is in $C_{\diamondsuit^r}$ if and only if (w, v, u) is in $C_{\diamondsuit'}$. If x and y are symbols in $\{l, r, '\}$, the notation $\diamondsuit^{xy}$ represents the operation $(\diamondsuit^x)^y$. Using the above observations, one can establish identities between operations of the form $\diamondsuit^{xy}$.

Lemma 2.3 *(i)* $\diamondsuit^{ll} = \diamondsuit^{rr} = \diamondsuit'' = \diamondsuit$,
(ii) $\diamondsuit^{'l} = \diamondsuit^{r'} = \diamondsuit^{lr}$,
(iii) $\diamondsuit^{'r} = \diamondsuit^{l'} = \diamondsuit^{rl}$.

Also the following technical result will be needed, establishing a relation between composition and left/right inverse:

Lemma 2.4 *Let $u, v \in \Sigma^*$, $R \subseteq \Sigma^*$, and let $\diamondsuit$ be a binary word operation defined as*

$$u \diamondsuit v = (u \diamondsuit_1 v) \diamondsuit_2 R,$$

where $\diamondsuit_1$, $\diamondsuit_2$ are binary word operations. Then

(i) $u \diamondsuit^l v = (u \diamondsuit_2^l R) \diamondsuit_1^l v$,

(ii) $u \diamondsuit^r v = u \diamondsuit_1^r (v \diamondsuit_2^l R)$.

Proof. (i) All the following statements are equivalent:

$$\begin{array}{rcl} x & \in & u \diamondsuit^l v, \\ u & \in & x \diamondsuit v, \\ u & \in & (x \diamondsuit_1 v) \diamondsuit_2 R, \\ \exists w \in (x \diamondsuit_1 v) : u & \in & w \diamondsuit_2 R, \\ \exists w \in (x \diamondsuit_1 v) : w & \in & u \diamondsuit_2^l R, \\ \exists w \in (u \diamondsuit_2^l R) : w & \in & x \diamondsuit_1 v, \\ \exists w \in (u \diamondsuit_2^l R) : x & \in & w \diamondsuit_1^l v, \\ x & \in & (u \diamondsuit_2^l R) \diamondsuit_1^l v. \end{array}$$

(ii) Analogous to (i). □

Next we list several binary word operations together with their left and right inverses [2, 3, 8].

Catenation: [1] $u \cdot v = \{uv\}$, with $\cdot^l = \longrightarrow_{\rm rq}$ and $\cdot^r = \longrightarrow_{\rm lq}$.

Left quotient: $u \longrightarrow_{\rm lq} v = \{w\}$ if $u = vw$, with $\longrightarrow_{\rm lq}^l = \cdot'$ and $\longrightarrow_{\rm lq}^r = \cdot$.

Right quotient: $u \longrightarrow_{\rm rq} v = \{w\}$ if $u = wv$, with $\longrightarrow_{\rm rq}^l = \cdot$ and $\longrightarrow_{\rm rq}^r = \longrightarrow_{\rm lq}'$.

Insertion: $u \longleftarrow v = \{u_1 v u_2 \mid u = u_1 u_2\}$, with $\longleftarrow^l = \longrightarrow$ and $\longleftarrow^r = \rightleftharpoons'$.

Deletion: $u \longrightarrow v = \{u_1 u_2 \mid u = u_1 v u_2\}$, with $\longrightarrow^l = \longleftarrow$ and $\longrightarrow^r = \rightleftharpoons$.

Dipolar deletion: $u \rightleftharpoons v = \{w \mid u = v_1 w v_2, v = v_1 v_2\}$, with $\rightleftharpoons^l = \longleftarrow'$ and $\rightleftharpoons^r = \longrightarrow$.

Shuffle (or scattered insertion): $u \sqcup\!\sqcup v = \{u_1 v_1 \cdots u_k v_k u_{k+1} \mid k \geq 1, u = u_1 \cdots u_k u_{k+1}, v = v_1 \cdots v_k\}$, with $\sqcup\!\sqcup^l = \rightsquigarrow$ and $\sqcup\!\sqcup^r = \rightsquigarrow'$.

Scattered deletion: $u \rightsquigarrow v = \{u_1 \cdots u_k u_{k+1} \mid k \geq 1, u = u_1 v_1 \cdots u_k v_k u_{k+1}, v = v_1 \cdots v_k\}$, with $\rightsquigarrow^l = \sqcup\!\sqcup$ and $\rightsquigarrow^r = \rightsquigarrow$.

Literal shuffle: $a_1 a_2 \ldots a_k \sqcup\!\sqcup_{\rm l}\, b_1 b_2 \ldots b_m = \begin{cases} \{a_1 b_1 a_2 b_2 \ldots a_k b_k b_{k+1} \ldots b_m\} & \text{if } k \leq m, \\ \{a_1 b_1 a_2 b_2 \ldots a_m b_m a_{m+1} \ldots a_k\} & \text{if } k > m, \end{cases}$
where $a_i, b_j \in \Sigma, 1 \leq i \leq k, 1 \leq j \leq m$, with $\sqcup\!\sqcup_{\rm l}^l = \rightsquigarrow_{\rm l}$.

Literal deletion: $u \rightsquigarrow_{\rm l} v = \begin{cases} \{a_1 \ldots a_k\} & \text{if } v = b_1 \ldots b_m \text{ and either} \\ & k \leq m \text{ and } u = a_1 b_1 \ldots a_k b_k b_{k+1} \ldots b_m, \text{ or} \\ & k > m \text{ and } u = a_1 b_1 \cdots a_m b_m a_{m+1} \ldots a_k\}, \\ \emptyset & \text{otherwise,} \end{cases}$
where $a_i, b_j \in \Sigma, 1 \leq i \leq k, 1 \leq j \leq m$, with $\rightsquigarrow_{\rm l}^l = \sqcup\!\sqcup_{\rm l}$.

[1] We shall also write uv for $u \cdot v$.

Balanced literal shuffle: $a_1a_2\ldots a_k \sqcup\!\sqcup_{\rm bl}\, b_1b_2\ldots b_m = \begin{cases} \{a_1b_1a_2b_2\ldots a_kb_k\} & \text{if } k=m, \\ \emptyset & \text{otherwise,} \end{cases}$

where $a_i, b_j \in \Sigma, 1 \le i \le k, 1 \le j \le m$, with $\sqcup\!\sqcup^l_{\rm bl} = \rightsquigarrow_{\rm bl}$.

Balanced literal deletion: $u \rightsquigarrow_{\rm bl} v = \begin{cases} \{a_1\ldots a_k\} & \text{if } v = b_1\ldots b_k,\ u = a_1b_1\ldots a_kb_k, \\ \emptyset & \text{otherwise,} \end{cases}$

where $a_i, b_j \in \Sigma, 1 \le i, j \le k$, with $\rightsquigarrow^l_{\rm bl} = \sqcup\!\sqcup_{\rm bl}$.

3 Shuffle on trajectories

The shuffle on trajectories was introduced in [8, 9] as a concept generalizing the above insertion word operations. *Trajectory* is essentially a syntactical condition specifying how two words α and β are merged together into a resulting word. Formally, a trajectory is a string over the *trajectory alphabet* $V = \{0,1\}$. The following definition and the example are due to [8, 9].

Let Σ be an alphabet and let t be a trajectory, $t \in V^*$. Let α, β be two words over Σ.

Definition 3.1 *The shuffle of α with β on the trajectory t, denoted by $\alpha \sqcup\!\sqcup_t \beta$, is defined as follows:*

$$\alpha \sqcup\!\sqcup_t \beta \;=\; \{\alpha_1\beta_1\ldots\alpha_k\beta_k \mid \alpha = \alpha_1\ldots\alpha_k,\ \beta = \beta_1\ldots\beta_k,\ t = 0^{i_1}1^{j_1}\ldots0^{i_k}1^{j_k},\ \textit{where}\ |\alpha_m| = i_m\ \textit{and}\ |\beta_m| = j_m\ \textit{for all}\ m,\ 1 \le m \le k\}.$$

Observe that in $\alpha \sqcup\!\sqcup_T \beta$, the positions of 0's in t correspond to letters of α, while 1's correspond to letters of β.

A *set of trajectories* is any set $T \subseteq V^*$. We define the *shuffle of α with β on the set T*, denoted by $\alpha \sqcup\!\sqcup_T \beta$, as:

$$\alpha \sqcup\!\sqcup_T \beta = \bigcup_{t\in T} \alpha \sqcup\!\sqcup_t \beta. \tag{2}$$

The operation $\sqcup\!\sqcup_T$ generalizes to languages due to the general principle (1).

Example 3.2 *Let α and β be the words $\alpha = a_1a_2\ldots a_8$, $\beta = b_1b_2\ldots b_5$ and assume that $t = 0^31^20^310101$. The shuffle of α and β on the trajectory t is:*

$$\alpha \sqcup\!\sqcup_t \beta = \{a_1a_2a_3b_1b_2a_4a_5a_6b_3a_7b_4a_8b_5\}.$$

The shuffle operation has a natural two-dimensional geometric representation: the trajectory t defines a line starting in the origin and continuing one unit to the right or up, depending on the symbols in t. The symbol 0 stands for the *right* direction and 1 stands for the *up* direction, see Figure 1. The thinner line corresponds to the above trajectory t, the bolder one to another trajectory $t' = 10^51^3010^2$.

Example 3.3 *The following binary word insertion operations can be expressed via shuffle on trajectories using certain sets of trajectories.*

1. *Let $T = 0^*1^*$, then $\sqcup\!\sqcup_T \;=\; \cdot$, the catenation operation.*

2. *Let $T = 0^*1^*0^*$, then $\sqcup\!\sqcup_T \;= \longleftarrow$, the insertion.*

3. *Let $T = \{0,1\}^*$, then $\rightsquigarrow_T \;=\; \sqcup\!\sqcup$, the shuffle.*

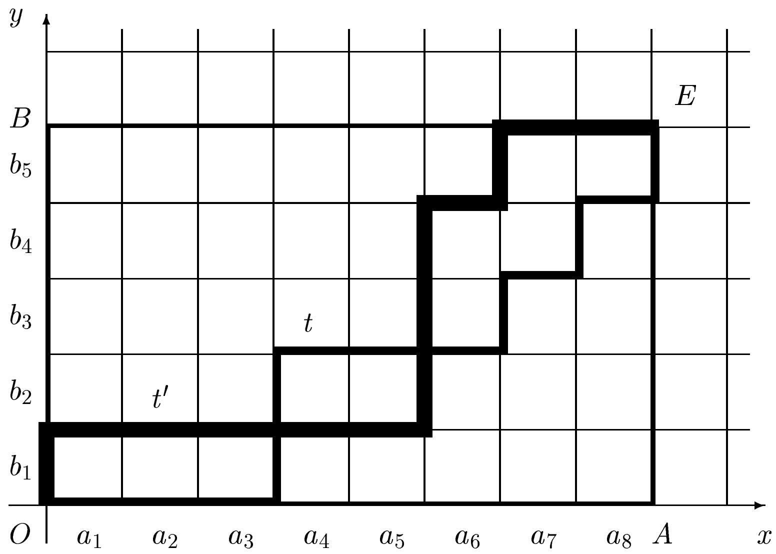

Figure 1: Geometric representation of the shuffle on trajectories. The figure reprinted from [9] with permission of authors.

4. *Let* $T = (01)^*(0^* \cup 1^*)$, *then* $\rightsquigarrow_T \ = \ \sqcup\!\sqcup_{\mathrm{l}}$, *the literal shuffle.*

5. *Let* $T = (01)^*$, *then* $\rightsquigarrow_T \ = \ \sqcup\!\sqcup_{\mathrm{bl}}$, *the balanced literal shuffle.*

The following closure properties of shuffle on trajectories are known to hold.

Lemma 3.4 *Let T be a set of trajectories. The following assertions are equivalent:*

(i) For all regular languages L_1, L_2, $L_1 \sqcup\!\sqcup_T L_2$ is a regular language.

(ii) T is a regular language.

Lemma 3.5 *(i) If any two of the languages L_1, L_2, T are regular and one is context-free, then $L_1 \sqcup\!\sqcup_T L_2$ is a context-free language.*

(ii) If any two of the languages L_1, L_2, T are context-free and one is regular, then $L_1 \sqcup\!\sqcup_T L_2$ is a non-context-free language for some triples (L_1, L_2, T).

We refer to [8, 9] for further examples, results and applications of shuffle on trajectories. The authors show among others an application of the above operations on trajectories to the problem of decomposition of a process to a bundle of concurrent subprocesses.

4 Deletion on trajectories

The above concept of shuffle on trajectories provides an uniform and flexible method for dealing with many natural word and language *insertion* operations, with interesting algebraic properties. We can, however, apply a similar approach also in the *deletion* case.

Definition 4.1 *The deletion of β from α on trajectory t is the following binary word operation:*

$$\alpha \rightsquigarrow_t \beta \;=\; \{\alpha_1 \ldots \alpha_k \mid \alpha = \alpha_1\beta_1 \ldots \alpha_k\beta_k,\ \beta = \beta_1 \ldots \beta_k,\ t = 0^{i_1}1^{j_1} \ldots 0^{i_k}1^{j_k},\ \textit{where}\ |\alpha_m| = i_m\ \textit{and}\ |\beta_m| = j_m\ \textit{for all}\ m,\ 1 \leq m \leq k\}.$$

It follows from the above definition that if $|\alpha| \neq |t|$ or $|\beta| \neq |t|_1$, then $\alpha \rightsquigarrow_t \beta = \emptyset$. We extend the operation $\rightsquigarrow_t$ to languages and sets of trajectories in the same way as for shuffle:

$$\alpha \rightsquigarrow_T \beta \;=\; \bigcup_{t \in T} \alpha \rightsquigarrow_t \beta, \tag{3}$$

$$L_1 \rightsquigarrow_T L_2 \;=\; \bigcup_{\alpha \in L_1,\ \beta \in L_2} \alpha \rightsquigarrow_T \beta, \tag{4}$$

where $\alpha, \beta \in \Sigma^*$, $L_1, L_2 \subseteq \Sigma^*$ and $T \subseteq V^*$.

Example 4.2 *Some basic operations of sequential deletion of words and languages are particular cases of the deletion on trajectories.*

1. *Let $T = 0^*1^*$ or $T = 1^*0^*$, respectively. Then $\rightsquigarrow_T = \longrightarrow_{\mathrm{rq}}$, the right quotient, or $\rightsquigarrow_T = \longrightarrow_{\mathrm{lq}}$, the left quotient, respectively.*
2. *Let $T = 0^*1^*0^*$ or $T = 1^*0^*1^*$, respectively. Then $\rightsquigarrow_T = \longrightarrow$, the deletion operation, or $\rightsquigarrow_T = \rightleftharpoons$, the dipolar deletion, respectively.*
3. *Let $T = \{0,1\}^*$, then $\rightsquigarrow_T = \rightsquigarrow$, the scattered deletion operation.*
4. *Let $T = (01)^*(0^* \cup 1^*)$, then $\rightsquigarrow_T = \rightsquigarrow_{\mathrm{l}}$, the literal deletion.*
5. *Let $T = (01)^*$, then $\rightsquigarrow_T = \rightsquigarrow_{\mathrm{bl}}$, the balanced literal deletion.*

Lemma 4.3 *Let T be a set of trajectories, then*

(i) $\sqcup\!\sqcup^{l}_{T} = \rightsquigarrow_T$ *and* $\sqcup\!\sqcup^{r}_{T} = \rightsquigarrow'_{\widetilde{T}}$,

(ii) $\rightsquigarrow^{l}_{T} = \sqcup\!\sqcup_T$ *and* $\rightsquigarrow^{r}_{T} = \rightsquigarrow_{\widetilde{T}}$,

where $\widetilde{T}$ is the set of trajectories obtained from T by changing all 1's for 0's and vice versa in all the words of T.

Proof. (i) Comparing Definitions 3.1 and 4.1, the statement $\sqcup\!\sqcup^{l}_{t} = \rightsquigarrow_t$ follows immediately, and hence due to (2) and (3), $\sqcup\!\sqcup^{l}_{T} = \rightsquigarrow_T$.

For the right inverse, observe that for any set T, $\sqcup\!\sqcup_{\widetilde{T}} = \sqcup\!\sqcup'_{T}$. Then due to Lemma 2.3, $\sqcup\!\sqcup^{r}_{T} = \sqcup\!\sqcup'^{r}_{\widetilde{T}} = \sqcup\!\sqcup^{l\prime}_{\widetilde{T}} = \rightsquigarrow'_{\widetilde{T}}$.
(ii) Analogous to (i). □

Similarly as with shuffle on trajectories in [8], we can represent the language deletion on trajectories via the scattered deletion and other simple language operations. The following two theorems give examples of such characterizations.

Theorem 4.4 *For all languages L_1 and L_2, $L_1, L_2 \subseteq \Sigma^*$, and for all sets of trajectories $T \subseteq V^*$, there exists a gsm M and a letter-to-letter morphism h such that*

$$L_1 \rightsquigarrow_T L_2 = (M(L_1 \sqcup\!\sqcup T) \rightsquigarrow h(L_2)) \cap \Sigma^*. \tag{5}$$

Proof. Let $\Sigma_1 = \{a_1 \mid a \in \Sigma\}$ be a copy of Σ such that Σ, Σ_1 and V are pairwise disjoint alphabets. Let further $h : \Sigma \longrightarrow \Sigma_1$ be a morphism such that $h(a) = a_1$, $a \in \Sigma$. Consider finally the gsm $M = (Q, \Sigma', \Delta, q_0, \delta, F)$, where $Q = \{q_0, q_1, q_2\}$, $\Sigma' = \Sigma \cup \Sigma_1 \cup V$, $\Delta = \Sigma \cup \Sigma_1$, $F = \{q_0\}$ and

$$\begin{array}{rclcrcl} \delta(q_0, 1) & = & (q_1, \lambda) & \quad & \delta(q_0, 0) & = & (q_2, \lambda) \\ \delta(q_1, a) & = & (q_0, a_1) & \quad & \delta(q_2, a) & = & (q_0, a) \end{array}$$

for all $a \in \Sigma$.

The gsm M accepts only the words from $T \sqcup\!\sqcup_{\mathrm{bl}} L_1$, all other words from $L_1 \sqcup\!\sqcup T$ are rejected. Let $t \sqcup\!\sqcup_{\mathrm{bl}} x$, $x \in L_1$, $t \in T$ be an accepted word, then M rewrites all its substrings $1a$ to a_1, and all its substrings $0a$ to a. Hence within the result $y = M(t \sqcup\!\sqcup_{\mathrm{bl}} x)$, the letters from Σ_1 are exactly those that should be deleted from x via $\rightsquigarrow_t$ (we call them "marked for deletion").

Now, words from $h(L_2)$ can be deleted from y via the operation $y \rightsquigarrow h(L_2)$. Due to the intersection with Σ, the results contributes to the right-hand side of (5) iff the deleted letters were all those marked for deletion, and hence the result is equal to $x \rightsquigarrow_t z$ for some $z \in L_2$. We can conclude that (5) holds. □

Theorem 4.5 *For all languages L_1 and L_2, $L_1, L_2 \subseteq \Sigma^*$, and for all sets T of trajectories, there exists a regular set R and morphisms h_1, h_2 and h_3 such that*

$$L_1 \rightsquigarrow_T L_2 = h_3(((L_1 \sqcup\!\sqcup h_1(T)) \cap R) \rightsquigarrow h_2(L_2)) \cap \Sigma^*. \tag{6}$$

Proof. Consider the following morphisms:

$h_1 : V \longrightarrow V^*$, where $h_1(0) = 0$, $h_1(1) = 11$,
$h_2 : \Sigma \longrightarrow (\Sigma \cup V)^*$, where $h_2(a) = 1a1$, $a \in \Sigma$,
$h_3 : (\Sigma \cup V) \longrightarrow (\Sigma \cup V)^*$, where $h_3(a) = a$, $a \in \Sigma$, $h_3(1) = 1$, $h_3(0) = \lambda$.

Let further $R = (0\Sigma \cup 1\Sigma 1)^*$. The principle of the construction is similar to that of Theorem 4.4 and is left to the reader. □

5 Closure properties

Lemma 5.1 *For an arbitrary language $L \subseteq \{a, b\}^*$ there exists a set of trajectories T such that $L = (ab)^* \rightsquigarrow_T \{a, b\}^*$.*

Proof. Let $T = \phi(L)$, $\phi : \{a, b\} \longrightarrow V^*$ being a morphism such that $\phi(a) = 01$, $\phi(b) = 10$. □

Theorem 5.2 *For all regular languages L_1, L_2, and a regular set of trajectories T, $L_1 \rightsquigarrow_T L_2$ is a regular language.*

Proof. The statement follows by Theorems 4.4, 4.5 and by the fact that the class REG is closed under gsm, morphisms, intersection, shuffle and scattered deletion. Nevertheless, we show a direct construction of the finite automaton A such that $L(A) = L_1 \rightsquigarrow_T L_2$.

Let $A_T = (Q_T, V, \delta_T, s_T, F_T)$ be a nondeterministic finite automaton such that $L(A_T) = T$. Assume further, without loss of generality, that L_1 and L_2 are over the same alphabet Σ. Let $A_i = (Q_i, \Sigma, \delta_i, s_i, F_i)$ be NFA such that $L(A_i) = L_i$, $i = 1, 2$.

We construct a λ-NFA $A = (Q, \Sigma, \delta, s, F)$ combining A_1, A_2 and A_T such that $L(A) = L_1 \rightsquigarrow_T L_2$. Informally, given a word $x \in \Sigma^*$, A chooses nondeterministically a trajectory $t \in T$, words $x_1 \in L_1$ and $x_2 \in L_2$ and tests whether $\{x\} = x_1 \rightsquigarrow_t x_2$. If $a \in \Sigma$ is the recent input symbol of A, and if t_i, the i-th symbol of the trajectory t, is recently read by A_T, then A takes the following action:

(a) if $t_i = 0$, then A reads the input a and simulates one step of A_T and A_1, as in this case the i-th symbol of x_1 comes from x;

(b) if $t_i = 1$, then A does not read the input a (makes a λ-step) and simulates one step of A_T, A_1 and A_2, as in this case the i-th symbol of x_1 comes from x_2.

The λ-NFA A accepts x iff A_1, A_2 and A_T all accept. Formally, $Q = Q_1 \times Q_T \times Q_2$, $s = (s_1, s_T, s_2)$, $F = F_1 \times F_T \times F_2$, and δ is defined as follows:

$$\begin{aligned}
\delta((q_1, d, q_2), a) &= \{(q_1', d', q_2) \mid q_1' \in \delta_1(q_1, a),\ d' \in \delta_T(d, 0)\}, \text{ for each } a \in \Sigma,\\
\delta((q_1, d, q_2), \lambda) &= \{(q_1', d', q_2') \mid q_1' \in \delta_1(q_1, a),\ d' \in \delta_T(d, 1),\ q_2' \in \delta_2(q_2, a),\ a \in \Sigma\},
\end{aligned}$$

for all $q_1 \in Q_1$, $d \in Q_T$, $q_2 \in Q_2$. The reader can easily verify that $L(A) = L_1 \rightsquigarrow_T L_2$. □

Observe further that the size of A is $|A| = |A_1| \cdot |A_T| \cdot |A_2|$.

6 Algebraic properties

For shuffle of trajectories, various algebraic properties have been studied like completeness and determinism of trajectories, commutativity, associativity and distributivity property [8, 9]. Naturally, the set of trajectories T was called commutative (associative, distributive...) iff the operation $\sqcup\!\sqcup_T$ was commutative (associative, distributive...). We respect this notation and hence the reader must keep in mind that the commutativity (associativity, distributivity...) of T generally does not imply the same property of $\rightsquigarrow_T$ and vice versa.

The set of trajectories is called *complete* iff $\alpha \sqcup\!\sqcup_T \beta \neq \emptyset$ for all $\alpha, \beta \in \Sigma^*$. The notion of completeness cannot be implemented in the case of deletion in a meaningful way since for no set of trajectories T we have $\alpha \rightsquigarrow_T \beta \neq \emptyset$, for all $\alpha, \beta \in \Sigma^*$, on one hand. On the other hand, one can easily verify that the determinism property has the same effect in the case of both shuffle and deletion.

Definition 6.1 *A set of trajectories T is deterministic iff* $\mathrm{card}(\alpha \sqcup\!\sqcup_T \beta) \leq 1$, *for all* $\alpha, \beta \in \Sigma^*$.

Theorem 6.2 *A set of trajectories T is deterministic iff* $\mathrm{card}(\alpha \rightsquigarrow_T \beta) \leq 1$, *for all* $\alpha, \beta \in \Sigma^*$.

All the results concerning determinism holds also in the case of deletion on trajectories. We refer to [8, 9] for details.

The *commutativity* and the *associativity* properties are much weaker in the case of deletion on trajectories than in the case of shuffle. Actually, we show that for all but special sets of trajectories T, the operation $\rightsquigarrow_T$ is neither commutative nor associative, on one hand. On the other hand, similarly as $⧢_T$, the operation $\rightsquigarrow_T$ is clearly *distributive* over union, both on the left and on the right side.

Theorem 6.3 *For a set of trajectories T, the following two assertions are equivalent:*

(i) $T \subseteq \{1\}^*$.

(ii) $\rightsquigarrow_T$ *is a commutative operation, i.e.* $L_1 \rightsquigarrow_T L_2 = L_2 \rightsquigarrow_T L_1$ *for all* $L_1, L_2 \subseteq \Sigma^*$.

Proof. (i) Let $T \subseteq \{1\}^*$ be a nonempty set of trajectories, then

$$L_1 \rightsquigarrow_T L_2 = \begin{cases} \{\lambda\} & \text{if } 1^{|w|} \in T \text{ for some } w \in L_1 \cap L_2, \\ \emptyset & \text{otherwise,} \end{cases}$$

for any languages L_1, L_2, and hence $\rightsquigarrow_T$ is commutative.

(ii) Let there be $t \in T$ such that $|t|_0 > 0$. Then for the languages $L_1 = \{a^{|t|}\}$, $L_2 = \{a^{|t|_1}\}$ we have $L_1 \rightsquigarrow_T L_2 = \{a^{|t|_0}\}$, $L_2 \rightsquigarrow_T L_1 = \emptyset$ and $\rightsquigarrow_T$ is not commutative. □

Corollary 6.4 $\rightsquigarrow_T$ *is a commutative operation iff* $L_1 \rightsquigarrow_T L_2 \subseteq \{\lambda\}$ *for all languages* $L_1, L_2 \in \Sigma^*$.

Theorem 6.5 *For a set of trajectories T, the following two assertions are equivalent:*

(i) For all $t_1 \in (1^m ⧢ 0^n)$, $t_2 \in (1^n ⧢ 0^i)$, $t_3 \in (1^j ⧢ 0^{m+n})$, $i, j, m, n \in \mathbb{N}$,

$$\begin{array}{rcll} m > 0 \textit{ and } t_1 \in T & \textit{implies} & t_2 \notin T, & (a) \\ m > 0 \textit{ and } t_1 \in T & \textit{implies} & t_3 \notin T, & (b) \\ t_1 \in T \textit{ and } 0^m \in T & \textit{implies} & 0^n \in T, & (c) \\ t_1 \in T \textit{ and } 0^n \in T & \textit{implies} & 0^m \in T. & (d) \end{array}$$

(ii) $\rightsquigarrow_T$ *is an associative operation, i.e.* $(L_1 \rightsquigarrow_T L_2) \rightsquigarrow_T L_3 = L_1 \rightsquigarrow_T (L_2 \rightsquigarrow_T L_3)$ *for all* $L_1, L_2, L_3 \subseteq \Sigma^*$.

Proof. (i) Consider languages $L_1, L_2, L_3 \subseteq \Sigma^*$ and a set of trajectories T satisfying (i). Let $y \in ((L_1 \rightsquigarrow_T L_2) \rightsquigarrow_T L_3)$, then there are $t_1, t_2 \in T$, $x_1 \in L_1$, $x_2 \in L_2$ and $x_3 \in L_3$ such that $(x_1 \rightsquigarrow_{t_1} x_2) \rightsquigarrow_{t_2} x_3 = \{y\}$.

Let $t_2 \in (1^m ⧢ 0^n)$ for some $m, n \geq 0$, then $t_1 \in (1^k ⧢ 0^{m+n})$ for some $k \geq 0$. We can deduce that $m = 0$ and hence also $x_3 = \lambda$, because for $m > 0$, $t_2 \in T$ would imply $t_1 \not{\!\!\;} nT$ due to (b), a contradiction. As $t_1 \in (1^k ⧢ 0^n)$ and $t_2 = 0^n$, due to (d) also $0^k \in T$ and it follows that $\{y\} = (x_1 \rightsquigarrow_{t_1} x_2) \rightsquigarrow_{t_2} x_3 = x_1 \rightsquigarrow_{t_1} x_2 = x_1 \rightsquigarrow_{t_1} (x_2 \rightsquigarrow_{0^k} x_3) \subseteq L_1 \rightsquigarrow_T (L_2 \rightsquigarrow_T L_3)$.

We have shown that $(L_1 \rightsquigarrow_T L_2) \rightsquigarrow_T L_3 \subseteq L_1 \rightsquigarrow_T (L_2 \rightsquigarrow_T L_3)$. Using analogous arguments for (a) and (c) we can show also that $L_1 \rightsquigarrow_T (L_2 \rightsquigarrow_T L_3) \subseteq (L_1 \rightsquigarrow_T L_2) \rightsquigarrow_T L_3)$ which together assures the associativity of $\rightsquigarrow_T$.

(ii) We show that violation of any of the conditions (a), (b), (c), (d) would make T nonassociative.

(a) Let there be $t_1, t_2 \in T$, $t_1 \in (1^m \sqcup\!\sqcup 0^n)$, $t_2 \in (1^n \sqcup\!\sqcup 0^i)$, for some $m > 0$, $n, i \geq 0$. Consider the word $w \in \{a,b\}^*$, such that $y = \phi(t_1)$, where $\phi(0) = a$, $\phi(1) = b$. Then

$$\begin{aligned} a^{n+i} \rightsquigarrow_T (y \rightsquigarrow_T b^m) &= a^{n+i} \rightsquigarrow_{t_2} (y \rightsquigarrow_{t_1} b^m) = \{a^i\}, \text{ and} \\ (a^{n+i} \rightsquigarrow_T y) \rightsquigarrow_T b^m &= \emptyset \rightsquigarrow_T b^m = \emptyset, \end{aligned}$$

as y contains at least one b. Hence $\rightsquigarrow_T$ is not associative for the languages $\{a^{n+i}\}$, $\{y\}$ and $\{b^m\}$.

(b) The proof is analogous to (a) and is left to the reader.

(c) Let there be $t_1 \in T$, $t_1 \in (1^m \sqcup\!\sqcup 0^n)$, and let further $0^m \in T$ and $0^n \notin T$. Then

$$\begin{aligned} a^{n+m}(\rightsquigarrow_T a^m \rightsquigarrow_T \lambda) &= a^{n+m} \rightsquigarrow_{t_1} (a^m \rightsquigarrow_{0^m} \lambda) = \{a^n\}, \\ (a^{n+m} \rightsquigarrow_T a^m) \rightsquigarrow_T \lambda &= a^n \rightsquigarrow_T \lambda = \emptyset, \end{aligned}$$

and hence $\rightsquigarrow_T$ is nonassociative.

(d) Analogous to (c).

□

Corollary 6.6 *If $\rightsquigarrow_T$ is associative, then $(L_1 \rightsquigarrow_T L_2) \rightsquigarrow_T L_3 = \emptyset = L_1 \rightsquigarrow_T (L_2 \rightsquigarrow_T L_3)$ for all $L_1, L_2 \subseteq \Sigma^*$, $L_3 \subseteq \Sigma^+$.*

Proof. As it is shown in the proof of Theorem 6.5, part (i), if $\rightsquigarrow_T$ is associative and $(x_1 \rightsquigarrow_T x_2) \rightsquigarrow_T x_3 \neq \emptyset$ for some $x_1, x_2, x_3 \in \Sigma^*$, then $x_3 = \lambda$. Hence $(x_1 \rightsquigarrow_T x_2) \rightsquigarrow_T x_3 = \emptyset$ for all $x_1, x_2 \in \Sigma^*$, $x_3 \in \Sigma^+$, and thus also $(L_1 \rightsquigarrow_T L_2) \rightsquigarrow_T L_3 = \emptyset$ for all $L_1, L_2 \subseteq \Sigma^*$, $L_3 \subseteq \Sigma^+$.

Furthermore, if $\rightsquigarrow_T$ is associative and $L_1 \rightsquigarrow_T (L_2 \rightsquigarrow_T L_3) \neq \emptyset$ for some $L_1, L_2 \subseteq \Sigma^*$, $L_3 \subseteq \Sigma^+$, then $(L_1 \rightsquigarrow_T L_2) \rightsquigarrow_T L_3 \neq L_1 \rightsquigarrow_T (L_2 \rightsquigarrow_T L_3)$, a contradiction, since $(L_1 \rightsquigarrow_T L_2) \rightsquigarrow_T L_3 = \emptyset$ as we have just shown. □

Corollary 6.7 *The operations $\longrightarrow_{\mathrm{lq}}$, $\longrightarrow_{\mathrm{rq}}$, $\longrightarrow$, $\rightleftharpoons$, $\rightsquigarrow$, $\rightsquigarrow_{\mathrm{l}}$, $\rightsquigarrow_{\mathrm{bl}}$ are neither commutative, nor associative.*

7 Decidability properties – the regular case

In this and the following section we study three elementary types of decision problems for language equations involving the deletion on trajectories $\rightsquigarrow_T$. They are formulated generally for an arbitrary binary language operation $\diamondsuit$ in accordance with [2].

Q_0 : For given languages L_1, L_2, R, R regular, is $L_1 \diamondsuit L_2 = R$?

Q_1 : For given languages L_2, R, R regular, does there exists $X \subseteq \Sigma^*$ such that $X \diamondsuit L_2 = R$?

Q_2 : For given languages L_1, R, R regular, does there exists $X \subseteq \Sigma^*$ such that $L_1 \diamondsuit X = R$?

The variant of the problem Q_0 for a singleton language $L_2 = \{w\}$ is denoted by Q_0^w.

Similarly, the variants of the problems Q_1 and Q_2 for a singleton language $X = \{w\}$ are denoted by Q_1^w and Q_2^w, respectively.

Let us focus now on our operation $\diamondsuit = \rightsquigarrow_T$. If L_1, L_2 and T are all regular, then $L_1 \rightsquigarrow_T L_2$ is also regular language due to Theorem 5.2. Hence the problems Q_0 and Q_0^w are decidable. The following theorems address the problems Q_1, Q_1^w and Q_2.

Theorem 7.1 *The problem "Does there exists a solution X to the equation $L \rightsquigarrow_T X = R$" (right-operand problem) is decidable for regular languages L, R and a regular set of trajectories T.*

Proof. Due to [3], if a solution to the equation $L \diamondsuit X = R$ exists, then also $X_{\max} = (L \diamondsuit^r R^c)^c$ is also a solution, $\diamondsuit$ being an invertible binary word operation. In fact, $X_{\max}$ is the maximum (with respect to union) of all the sets X such that $L \diamondsuit X \subseteq R$. We can conclude that a solution exists iff

$$L \diamondsuit (L \diamondsuit^r R^c)^c = R \tag{7}$$

holds. Observe that for $\diamondsuit = \rightsquigarrow_T$, $\diamondsuit^r = \rightsquigarrow_{\widetilde{T}}$ (see Lemma 2.3) and hence (7) adopts the form

$$L \rightsquigarrow_T (L \rightsquigarrow_{\widetilde{T}} R^c)^c = R.$$

As the class REG is closed under complement and deletion on trajectories (see Theorem 5.2), validity of the last equality is decidable and moreover the maximal solution $X_{\max} = (L \rightsquigarrow_{\widetilde{T}} R^c)^c$ can be effectively found. $\square$

Theorem 7.2 *The problem "Does there exists a solution X to the equation $X \rightsquigarrow_T L = R$" (left-operand problem) is decidable for regular languages L, R and a regular set of trajectories T.*

Proof. A maximal solution to the equation $X \diamondsuit L = R$ is $X_{\max} = (R^c \diamondsuit^l L)^c$ for an invertible binary word operation $\diamondsuit$, see [3]. Hence analogously as in the proof of Theorem 7.1, a solution to $X \rightsquigarrow_T L = R$ exists iff

$$(R^c \sqcup\!\sqcup_T L)^c \rightsquigarrow_T L = R,$$

as $\sqcup\!\sqcup_T$ is the left inverse of $\rightsquigarrow_T$. Again this equality is decidable due to the closure properties of the class REG and the maximal solution $X_{\max} = (R^c \sqcup\!\sqcup_T L)^c$, if one exists, can be effectively found. $\square$

Theorem 7.3 *The problem "Does there exists a word w such that $w \rightsquigarrow_T L = R$" is decidable for regular languages L, R and a regular set of trajectories T.*

Proof. Observe first that if R is infinite, then there cannot exist a solution w of a finite length satisfying $w \rightsquigarrow_T L = R$. Hence for an infinite R the problem is trivial.

Assume now that R is finite. As shown in [3], given the equation $X \rightsquigarrow_T L = R$, the regular set $X_{\max} = (R^c \sqcup\!\sqcup_T L)^c$ is the maximal set with the property $X \rightsquigarrow_T L \subseteq R$. Hence w is a solution of $w \rightsquigarrow_T L = R$ iff

(i) $w \rightsquigarrow_T L \subseteq R$, i.e. $w \in X_{\max}$, and
(ii) $w \rightsquigarrow_T L \not\subset R$.

Moreover, (ii) is satisfied iff $w \rightsquigarrow_T L \not\subseteq R_1$ for all $R_1 \subset R$, and hence $w \notin (R_1^c ⧢_T L)^c$. Hence we can conclude that the set S of all singleton solutions to the equation $w \rightsquigarrow_T L = R$ can be expressed as

$$S = (R^c ⧢_T L)^c - \bigcup_{R_1 \subset R} (R_1^c ⧢_T L)^c.$$

Since we assume that R is finite, the set S is regular and effectively constructible due to the closure of REG under union, complement and shuffle on trajectories. Hence it is also decidable whether S is empty or not. □

The above proof implies that we can find effectively all the singleton solutions to the equation $w \rightsquigarrow_T L = R$.

8 Decidability properties – the context-free case

In this section we address the decidability problems Q_0, Q_1, Q_2 for both the shuffle and the deletion on trajectories. Generally, for a context-free languages L_1, L_2, a regular language R and a regular set of trajectories T, the problem Q_0 is undecidable. In fact, the problem remains undecidable even if one of the languages L_1, L_2 is regular. Similarly, the problems Q_0^w, Q_1, Q_1^w, Q_2 and Q_2^w are undecidable for a context free language L and regular sets T and R. We refer to [2], where the undecidability of the mentioned problems is shown e.g. for the case of $T = 0^*1^*$, i.e. $⧢_T$ is the catenation and $\rightsquigarrow_T$ is the right quotient operation.

We ask in this section a slightly more intricate question — which properties of T imply (un)decidability of these problems? We consider just one of the operands L_1, L_2 context free and the other one regular. Due to the asymmetry of $\rightsquigarrow_T$ operation we distinguish the left-context-free and the right-context-free case of the problem Q_0. We need the following technical lemma first:

Lemma 8.1 *Let $R_D \subseteq \{0,1\}^*$ be an arbitrary but fixed regular language of the form $R_D = \bigcup_{d \in D}\{0,1\}^d$, for some infinite set $D \subseteq \mathbb{N}$. Then the problem "Does $R_D \subseteq L$ hold true" is undecidable for a context-free language L.*

Proof. Denote by $(U, V) = (u_1, \ldots, u_n; v_1, \ldots, v_n)$, $n \geq 1$, an instance of Post Correspondence Problem (PCP) over $\{0,1\}$, and let $k = \lceil \log_2 n \rceil + 2$. Consider the languages

$$\begin{aligned} L_U &= 1^*\{(i_m)_2 \ldots (i_1)_2 0^k u_{i_1} \ldots u_{i_m} \mid m > 0,\ 1 \leq i_j \leq n \text{ for all } i_j,\ 1 \leq j \leq m\}, \\ L_V &= 1^*\{(i_m)_2 \ldots (i_1)_2 0^k v_{i_1} \ldots v_{i_m} \mid m > 0,\ 1 \leq i_j \leq n \text{ for all } i_j,\ 1 \leq j \leq m\}, \end{aligned}$$

where $(i_j)_2$ denotes the k-digits binary representation of i_j. Let $L = L_U^c \cup L_V^c$. One can easily observe that L is a context-free language and that $L^c = L_U \cap L_V$ represents all the solutions $(i_1, \ldots, i_m)$ of (U, V). We show that $R_D \subseteq L$ holds true iff the instance (U, V) of PCP has no solution.

- If (U, V) has no solution, then $L = \Sigma^*$ and $R_D \subseteq L$ holds true.
- Assume that (U, V) has a solution $(i_1, \ldots, i_m)$ for some $m \geq 1$, and denote by $\ell = |(i_m)_2 \ldots (i_1)_2 0^k u_{i_1} \ldots u_{i_m}|$ the length of the shortest word in L^c corresponding to that solution. Then for each $d \geq \ell$ there exists a word $x \in L^c$ of the length d thanks to the prefix 1^* of the language L^c. Choose a number $d \in D$ such that $d \geq \ell$, then there is a word $x \in R_D$, $|x| = d$, such that $x \in L^c$, hence $x \notin L$ and $R_D \subseteq L$ does not hold.

$\square$

We address the variant of the problem Q_0 when the left operand is context-free first. For a set of trajectories T, denote by $D_0(T) \subseteq \mathbb{N}$ the set

$$D_0(T) = \{|t|_0 \mid t \in T\}.$$

Considering the alphabet $\{0,1\}$, denote further $R_{D_0(T)} = \bigcup_{d \in D_0(T)} \{0,1\}^d$. Obviously, if T is regular, then also $R_{D_0(T)}$ is a regular language as it can be obtained from T by a finite substitution $\phi(0) = \{0,1\}$, $\phi(1) = \lambda$.

Theorem 8.2 *Let T be an arbitrary but fixed regular set of trajectories. The problem "Is $L_1 \sqcup\!\sqcup_T R_2 = R$" is decidable for a context-free language L_1 and regular languages R_2, R if and only if $D_0(T)$ is finite.*

Proof.

(i) Consider a regular set of trajectories T such that $D_0(T)$ is finite. Observe that $L_1 \sqcup\!\sqcup_T R_2 = (L_1 \cap R_{D_0(T)}) \sqcup\!\sqcup_T R_2$. As $R_{D_0(T)}$ is an effectively constructible finite language, the same holds for $L_1 \cap R_{D_0(T)}$. Hence the language $L_1 \sqcup\!\sqcup_T R_2$ is regular and it is decidable whether or not $L_1 \sqcup\!\sqcup_T R_2 = R$ holds for a regular language R.

(ii) Assume that $D_0(T)$ is infinite. Let $L \subseteq \{0,1\}^*$ be an arbitrary context-free language. Consider the regular language

$$R = \{0,1\}^* \sqcup\!\sqcup_T c^* = R_{D_0(T)} \sqcup\!\sqcup_T c^*$$

over the alphabet $\{0,1,c\}$. We show that

$$L \rightsquigarrow_T c^* = R \text{ iff } R_{D_0(T)} \subseteq L. \tag{8}$$

(a) If $R_{D_0(T)} \subseteq L$, then

$$L \sqcup\!\sqcup_T c^* = (L \cap R_{D_0(T)}) \sqcup\!\sqcup_T c^* = R_{D_0(T)} \sqcup\!\sqcup_T c^* = R.$$

(b) If $R_{D_0(T)} \not\subseteq L$, then there is a $w \in R_{D_0(T)}$ such that $w \notin L$. Then the set $w \sqcup\!\sqcup_T c^*$ is nonempty and for all $w' \in w \sqcup\!\sqcup_T c^*$,

$$w' \in R_{D_0(T)} \sqcup\!\sqcup_T c^* \text{ and } w' \notin L \sqcup\!\sqcup_T c^*,$$

hence $w' \in R - (L \sqcup\!\sqcup_T c^*)$ and $L \rightsquigarrow_T c^* \neq R$.

Now, if we could decide whether $L \sqcup\!\sqcup_T c^* = R$ holds or not, then we could due to (8) also decide whether $R_{D_0(T)} \subseteq L$ holds, which contradicts Lemma 8.1.

$\square$

Theorem 8.3 *Let T be an arbitrary but fixed regular set of trajectories. The problem "Is $L_1 \rightsquigarrow_T R_2 = R$" is decidable for a context-free language L_1 and regular languages R_2, R if and only if the set $D_0(T)$ is finite.*

Proof.

(i) For a context-free language L_1 and a regular language R_2, the language $L_1 \rightsquigarrow_T R_2$ is context-free due to the closure of the class CF under deletion on trajectories with regular languages, see section 5. Moreover, we can write

$$L_1 \rightsquigarrow_T R_2 = (L_1 \rightsquigarrow_T R_2) \cap R_{D_0(T)}.$$

Assuming that $D_0(T)$ is finite, the language $L_1 \rightsquigarrow_T R_2$ is also finite and we can effectively enumerate all its words. Hence the problem "Is $L_1 \rightsquigarrow_T R_2 = R$" is decidable.

(ii) Consider a regular set of trajectories T such that $D_0(T)$ is infinite. Let $L \subseteq \{0,1\}^*$ be an arbitrary context-free language. Let

$$\begin{aligned} L_1 &= L \sqcup\!\sqcup_T c^*, \\ R &= (\{0,1\}^* \sqcup\!\sqcup_T c^*) \rightsquigarrow_T c^* \end{aligned}$$

be languages over the alphabet $\{0,1,c\}$. Observe that the definition of $\sqcup\!\sqcup_T$ implies that both

$$L_1 \subseteq R_{D_0(T)} \sqcup\!\sqcup_T c^*, \text{ and} \tag{9}$$

$$\{0,1\}^* \sqcup\!\sqcup_T c^* = R_{D_0(T)} \sqcup\!\sqcup_T c^* \tag{10}$$

hold. We show that

$$L_1 \rightsquigarrow_T c^* = R \text{ iff } R_{D_0(T)} \subseteq L. \tag{11}$$

(a) If $R_{D_0(T)} \subseteq L$, then due to (9) and (10),

$$\begin{aligned} L_1 \rightsquigarrow_T c^* &= (L \sqcup\!\sqcup_T c^*) \rightsquigarrow_T c^* = ((L \cap R_{D_0(T)}) \sqcup\!\sqcup_T c^*) \rightsquigarrow_T c^* = \\ &= (R_{D_0(T)} \sqcup\!\sqcup_T c^*) \rightsquigarrow_T c^* = (\{0,1\}^* \sqcup\!\sqcup_T c^*) \rightsquigarrow_T c^* = R. \end{aligned}$$

(b) If $R_{D_0(T)} \not\subseteq L$, then there is a $w \in R_{D_0(T)}$ such that $w \notin L$. Then the set $W = (w \sqcup\!\sqcup_T c^*) \rightsquigarrow_T c^*$ is nonempty and for all $w' \in W$,

$$w' \in (R_{D_0(T)} \sqcup\!\sqcup_T c^*) \rightsquigarrow_T c^* \text{ and } w' \notin (L \sqcup\!\sqcup_T c^*) \rightsquigarrow_T c^*,$$

which can be due to (10) rewritten as

$$w' \in R - (L_1 \rightsquigarrow_T c^*),$$

and hence $L_1 \rightsquigarrow_T c^* \neq R$.

Hence, as in the previous proof, decidability of "$L_1 \rightsquigarrow_T c^* = R$" would imply decidability of "$R_{D_0(T)} \subseteq L$" which contradicts Lemma 8.1.

□

Corollary 8.4 *Let $\diamondsuit$ be any of the operations $\cdot$, $\longleftarrow$, $\sqcup\!\sqcup$, $\sqcup\!\sqcup_{\mathrm{l}}$, $\sqcup\!\sqcup_{\mathrm{bl}}$, $\longrightarrow$, $\longrightarrow_{\mathrm{lq}}$, $\longrightarrow_{\mathrm{rq}}$, $\rightsquigarrow$, $\rightsquigarrow_{\mathrm{l}}$, $\rightsquigarrow_{\mathrm{bl}}$, $\rightleftharpoons$. Then the problem "Is $L_1 \diamondsuit R_2 = R$" is undecidable for a context-free language L_1 and regular languages R_2, R.*

Now we address the case when the right operand is context-free. We obtain the analogous result as in Theorem 8.2 for the case of $\sqcup\!\sqcup_T$ operation. For $\rightsquigarrow_T$, however, the situation is different.

Theorem 8.5 *Let T be an arbitrary but fixed regular set of trajectories. The problem "Is $R_1 \sqcup\!\sqcup_T L = R$" is decidable for a context-free language L and regular languages R_1, R if and only if $D_1(T) = \{|t|_1 \mid t \in T\}$ is finite.*

Proof. Observe that $\sqcup\!\sqcup_T = \sqcup\!\sqcup'_{\widetilde{T}}$ and $D_1(T) = D_0(\widetilde{T})$. Hence the statement follows by Theorem 8.2. $\square$

Theorem 8.6 *Let T be an arbitrary but fixed regular set of trajectories. If the set $D_{01}(T) = \{|t|_{01} \mid t \in T\}$ is infinite, then the problem "Is $R_1 \rightsquigarrow_T L = R$" is undecidable for a context-free language L and regular languages R_1, R.*

Proof. Let for a regular set of trajectories T, $D_{01}(T)$ is infinite. Denote

$$R_{D_{01}(T)} = \bigcup_{d \in D_{01}(T)} \{0,1\}^d.$$

The reader can easily check that $R_{D_{01}(T)}$ is a regular language. Let L be an arbitrary context-free language over an alphabet $\{0,1\}$. Denote

$$\begin{aligned} R_1 &= M(T \cdot \{c\}), \\ L_1 &= (L \cap R_{D_{01}(T)}) \sqcup\!\sqcup c^*, \\ R &= R_1 \rightsquigarrow_T (R_{D_{01}(T)} \sqcup\!\sqcup c^*), \end{aligned}$$

where $M = (Q, V \cup c, \{0,1,c\}, q_0, \delta, F)$ is a nondeterministic gsm, $Q = F = \{q_0, q_1\}$, and

$$\begin{aligned} \delta(q_0,0) &= (q_1,\{\lambda\}) & \delta(q_1,0) &= (q_1,\{c\}) \\ \delta(q_0,1) &= (q_0,\{c\}) & \delta(q_1,1) &= (q_0,\{a0,b1\}) \\ \delta(q_0,c) &= (q_0,\{c\}) & \delta(q_1,c) &= (q_1,\{cc\}) \end{aligned}$$

One can observe that the language R_1 is produced from T replacing all the substrings 01 by either $a0$ or $b1$, and all the remaining symbols by c in each $t \in T$. We show that

$$R_1 \rightsquigarrow_T L_1 = R \quad \text{iff} \quad R_{D_{01}(T)} \subseteq L. \tag{12}$$

(i) If $R_{D_{01}(T)} \subseteq L$, then we have

$$R_1 \rightsquigarrow_T L_1 = R_1 \rightsquigarrow_T ((L \cap R_{D_{01}(T)}) \sqcup\!\sqcup c^*) = R_1 \rightsquigarrow_T (R_{D_{01}(T)} \sqcup\!\sqcup c^*) = R.$$

(ii) If $R_{D_{01}(T)} \not\subseteq L$, then there is a $w \in R_{D_{01}(T)}$ such that $w \notin L$. Observe that then there is a word $w' \in w \sqcup\!\sqcup c^*$ such that after deleting w' via $\rightsquigarrow_T$ from R_1, the only symbols remaining in the result are a's, b's and c's, and moreover a's and b's are in the same order as 0's and 1's in w.

In other words, $w'' \in R_1 \rightsquigarrow_T w'$ holds for a word $w'' \in \phi(w) \sqcup\!\sqcup c$, where $\phi : \{0,1\}^* \longrightarrow \{a,b\}^*$ is the coding $\phi(0) = a$, $\phi(1) = b$. However, as $w \notin L$, there is no $x \in L_1$ such that $w'' \in R_1 \rightsquigarrow_T x$ would hold. Hence, we can write that

$$w'' \in R_1 \rightsquigarrow_T (R_{D_{01}(T)} \sqcup\!\sqcup_T c^*) \quad \text{and} \quad w'' \notin R_1 \rightsquigarrow_T L_1,$$

which can be rewritten as

$$w'' \in R - (R_1 \rightsquigarrow_T L_1),$$

and hence $R_1 \rightsquigarrow_T L_1 \neq R$.

Now, if we could decide whether $R_1 \rightsquigarrow_T L_1 = R$ holds or not, then we could due to (12) also decide whether $R_{D_{01}(T)} \subseteq L$ holds, which contradicts Lemma 8.1. $\square$

It remains open whether the problem "Is $R_1 \rightsquigarrow_T L = R$" is decidable for a context-free language L and regular languages R_1, R when $D_{01}(T)$ is finite.

Corollary 8.7 *Let $\diamondsuit$ be any of the operations $\sqcup\!\sqcup$, $\sqcup\!\sqcup_{\mathrm{l}}$, $\sqcup\!\sqcup_{\mathrm{bl}}$, $\rightsquigarrow$, $\rightsquigarrow_{\mathrm{l}}$, $\rightsquigarrow_{\mathrm{bl}}$. Then the problem "Is $R_1 \diamondsuit L = R$" is undecidable for a context-free language L and regular languages R_1, R.*

9 Substitution on trajectories

Next we consider generalizations of three natural binary word operations, which are used together with shuffle and deletion to model noisy channels [4]. Generally, channel is a binary relation $\gamma \subseteq \Sigma^* \times \Sigma^*$. The fact that $(u, v) \in \gamma$ means that the word v can be received from u via the channel γ.

In [4], the channels with insertion, deletion and substitution errors are characterized via word operations. For instance, the channel with exactly m insertion errors is the set of all pairs (u, v) such that $v \in u \sqcup\!\sqcup \Sigma^m$, and analogously for deletion errors. We need the following definitions for characterizing channels with substitution errors.

Definition 9.1 *If $u, v \in \Sigma^*$ then we define the* substitution in u by v *as*

$$\begin{aligned} u \bowtie v \;\; = \;\; & \{u_1 v_1 u_2 v_2 \dots u_k v_k u_{k+1} \mid k \geq 0,\ u = u_1 a_1 u_2 a_2 \dots u_k a_k u_{k+1},\ v = v_1 v_2 \dots v_k, \\ & a_i, v_i \in \Sigma, 1 \leq i \leq k,\ a_i \neq v_i, \forall i, 1 \leq i \leq k\}. \end{aligned}$$

The case $k = 0$ corresponds to $v = \lambda$ when no substitution is performed.

Definition 9.2 *If $u, v \in \Sigma^*$ then we define the* substitution in u of v *as*

$$\begin{aligned} u \vartriangle v \;\; = \;\; & \{u_1 a_1 u_2 a_2 \dots u_k a_k u_{k+1} \mid k \geq 0,\ u = u_1 v_1 u_2 v_2 \dots u_k v_k u_{k+1},\ v = v_1 v_2 \dots v_k, \\ & a_i, v_i \in \Sigma, 1 \leq i \leq k,\ a_i \neq v_i, \forall i, 1 \leq i \leq k\}. \end{aligned}$$

Definition 9.3 *Let $u, v \in \Sigma^*$, $|u| = |v|$, let $H(u, v)$ be the Hamming distance of u and v. We define*

$$\begin{aligned} u \rhd v \;\; = \;\; & \{v_1 v_2 \dots v_k \mid k = H(u, v),\ u = u_1 a_1 u_2 a_2 \dots u_k a_k u_{k+1},\ v = u_1 v_1 u_2 v_2 \dots u_k v_k u_{k+1}, \\ & a_i, v_i \in \Sigma, 1 \leq i \leq k,\ a_i \neq v_i, \forall i, 1 \leq i \leq k\}. \end{aligned}$$

The above definitions are due to [4], where it is also shown that the left- and the right-inverse of $\bowtie$ are $\vartriangle$ and $\rhd$, respectively. Given two binary word operations $\diamondsuit_1$, $\diamondsuit_2$, authors define their composition $(\diamondsuit_1 \diamondsuit_2)$ as

$$w \in u(\diamondsuit_1 \diamondsuit_2)v \iff w \in (u \diamondsuit_1 v_1) \diamondsuit_2 v_2, \quad v = v_1 v_2,$$

for all $u, v, w \in \Sigma^*$. Then it is among others shown that:

(i) The channel with at most m substitution and insertion errors is equal to $\{(u, v) \mid v \in u(\vartriangle \sqcup\!\sqcup)(\Sigma^0 \cup \ldots \cup \Sigma^m)\}$.

(i) The channel with at most m substitution and deletion errors is equal to $\{(u,v) \mid v \in u(\rightsquigarrow\bowtie)(\Sigma^0 \cup \ldots \cup \Sigma^m)\}$.

Moreover, further consequences including composition of channels, inversion of channels etc. are derived. The above substitution operations can be generalized using trajectories as follows.

Definition 9.4 *For a trajectory $t \in V^*$ and $u, v \in \Sigma^*$ we define the* substitution in u by v *on trajectory t as*

$$\begin{aligned} u \bowtie_t v \;\; = \;\; & \{u_1v_1u_2v_2\ldots u_kv_ku_{k+1} \mid k \geq 0,\; u = u_1a_1u_2a_2\ldots u_ka_ku_{k+1},\; v = v_1v_2\ldots v_k, \\ & t = 0^{j_1}10^{j_2}1\ldots 0^{j_k}10^{j_{k+1}},\; a_i, v_i \in \Sigma, 1 \leq i \leq k,\; a_i \neq v_i, \forall i, 1 \leq i \leq k, \\ & j_i = |u_i|, 1 \leq i \leq k+1\}. \end{aligned}$$

Definition 9.5 *For a trajectory $t \in V^*$ and $u, v \in \Sigma^*$ we define the* substitution in u of v *on trajectory t as*

$$\begin{aligned} u \vartriangle_t v \;\; = \;\; & \{u_1a_1u_2a_2\ldots u_ka_ku_{k+1} \mid k \geq 0,\; u = u_1v_1u_2v_2\ldots u_kv_ku_{k+1},\; v = v_1v_2\ldots v_k, \\ & t = 0^{j_1}10^{j_2}1\ldots 0^{j_k}10^{j_{k+1}},\; a_i, v_i \in \Sigma, 1 \leq i \leq k,\; a_i \neq v_i, \forall i, 1 \leq i \leq k, \\ & j_i = |u_i|, 1 \leq i \leq k+1\}. \end{aligned}$$

Definition 9.6 *For a trajectory $t \in V^*$ and $u, v \in \Sigma^*$ we define the* right difference of u and v *on trajectory t as*

$$\begin{aligned} u \rhd_t v \;\; = \;\; & \{v_1v_2\ldots v_k \mid k \geq 0,\; u = u_1a_1u_2a_2\ldots u_ka_ku_{k+1},\; v = u_1v_1u_2v_2\ldots u_kv_ku_{k+1}, \\ & t = 0^{j_1}10^{j_2}1\ldots 0^{j_k}10^{j_{k+1}},\; a_i, v_i \in \Sigma, 1 \leq i \leq k,\; a_i \neq v_i, \forall i, 1 \leq i \leq k, \\ & j_i = |u_i|, 1 \leq i \leq k+1\}. \end{aligned}$$

These operations can be generalized to sets of trajectories in the natural way:

$$u \bowtie_T v = \bigcup_{t \in T} u \bowtie_t v, \quad u \vartriangle_T v = \bigcup_{t \in T} u \vartriangle_t v \;\text{ and }\; u \rhd_T v = \bigcup_{t \in T} u \rhd_t v.$$

Example 9.7 *Let $T = V^*$, i.e. the set T contains all the possible trajectories. Then $\bowtie_T = \bowtie$, $\vartriangle_T = \vartriangle$ and $\rhd_T = \rhd$.*

One can observe that similarly as in [4], the above defined substitution on trajectories could be used to characterize channels where errors occur in certain parts of words only, or with a certain frequency and so on. Due to the fact that the trajectory is a *syntactic* restriction, only such channels can be modelled where the occurrence of errors may depend on the *length* of the transferred message, but not on its *content*. In the sequel we study various properties of the above defined substitution operations.

Lemma 9.8 *For a set of trajectories T and words $u, v \in \Sigma^*$, the following holds:*

(i) $\bowtie_T^l \; = \; \vartriangle_T$ *and* $\bowtie_T^r \; = \; \rhd_T$,

(ii) $\vartriangle_T^l \; = \bowtie_T$ *and* $\vartriangle_T^r \; = \; \rhd_T'$,

(iii) $\rhd_T^l \; = \; \vartriangle_T'$ *and* $\rhd_T^r \; = \bowtie_T$.

Proof. (i) Consider the characteristic relation $C_{\bowtie_t}$ of the operation $\bowtie_t$. Observe that $(w,u,v) \in C_{\bowtie_t}$ iff $(u,w,v) \in C_{\bowtie_t^l}$ iff $(v,u,w) \in C_{\bowtie_t^r}$. Then the statements $\bowtie_t^l = \triangle_t$ and $\bowtie_t^r = \rhd_t$, $t \in T$, follow directly by careful reading the definitions of $\bowtie_t$, $\triangle_t$ and $\rhd_t$. Now observe that

$$u \bowtie_T^l v = \bigcup_{t \in T} u \bowtie_t^l v = \bigcup_{t \in T} u \triangle_t v = u \triangle_T v.$$

The proof for $\bowtie_T^r$ is analogous.

(ii) Due to Lemma 2.3, $\bowtie_T^l = \triangle_T$ implies $\triangle_T^l = \bowtie_T$ and $\bowtie_T^r = \rhd_T$ implies $\triangle_T^r = \bowtie_T^{lr} = \bowtie_T^{r'} = \rhd_T'$.

(iii) Similarly, $\bowtie_T^r = \rhd_T$ implies $\rhd_T^r = \bowtie_T$, and consequently $\rhd_T^l = \bowtie_T^{rl} = \bowtie_T^{l'} = \triangle_T'$. □

Before addressing the closure properties of substitution, we show first that any (not necessarily recursively enumerable) language over a two letter alphabet can be obtained as a result of substitution.

Lemma 9.9 *For an arbitrary language $L \subseteq \{a,b\}^*$ there exists a set of trajectories T such that*

(i) $L = a^* \bowtie_T b^*$,

(ii) $L = a^* \triangle_T a^*$.

Proof. Let $T = \phi(L)$, $\phi : \{a,b\}^* \longrightarrow V^*$ being a coding such that $\phi(a) = 0$, $\phi(b) = 1$. □

Similarly as in the case of shuffle and deletion, the substitution on trajectories can be characterized by simpler language operations.

Lemma 9.10 *Let $\diamondsuit_T$ be any of the operations $\bowtie_T$, $\triangle_T$, $\rhd_T$. Then there exists a finite substitution h_1, morphisms h_2, g and a regular language R such that for all languages $L_1, L_2 \subseteq \Sigma^*$, and for all sets of trajectories $T \subseteq V^*$,*

$$L_1 \diamondsuit_T L_2 = g((h_1(L_1) \sqcup\!\sqcup h_2(L_2) \sqcup\!\sqcup T) \cap R). \qquad (13)$$

Proof. Let $\Sigma_i = \{a_i \mid a \in \Sigma\}$, for $i = 1,2,3$, be copies of Σ such that Σ, Σ_1, Σ_2, Σ_3 and V are pairwise disjoint alphabets. For a letter $a \in \Sigma$, we denote by a_i the corresponding letter from Σ_i, $i = 1,2,3$.

Let further $h_1 : \Sigma \longrightarrow (\Sigma_1 \cup \Sigma_3)$ be a finite substitution and let $h_2 : \Sigma \longrightarrow \Sigma^2$ and $g : (\Sigma_1 \cup \Sigma^2 \cup \Sigma^3 \cup V) \longrightarrow \Sigma$ be morphisms.

(i) If $\diamondsuit_T = \bowtie_T$, then define $h_1(a) = \{a_1, a_3\}$, $h_2(a) = a_2$ for each $a \in \Sigma$. Let

$$R = (\Sigma_1 \cdot \{0\} \cup \{a_3 b_2 1 \mid a, b \in \Sigma, a \neq b\})^*.$$

Let further $g(a_1) = a$, $g(a_2) = a$ for all $a_1 \in \Sigma_1$, $a_2 \in \Sigma_2$, and $g(x) = \lambda$ for all $x \in \Sigma_3 \cup V$. Then one can easily verify that (13) holds true.

(ii) If $\diamondsuit_T = \triangle_T$, then let $h_1(a) = \{a_1\} \cup \{a_3\} \cdot \Sigma_1$, $h_2(a) = a_2$ for each $a \in \Sigma$. Let further

$$R = (\Sigma_1 \cdot \{0\} \cup \{a_3 a_2 b_1 1 \mid a, b \in \Sigma, a \neq b\})^*,$$

and $g(a_1) = a$ for all $a_1 \in \Sigma_1$, $g(x) = \lambda$ for all $x \in \Sigma_2 \cup \Sigma_3 \cup V$.

(iii) If $\diamondsuit_T = \rhd_T$, then define $h_1(a) = a_1$, $h_2(a) = \{a_2, a_3\}$ for each $a \in \Sigma$. Let

$$R = (\{a_1 a_2 0 \mid a \in \Sigma\} \cup \{a_1 b_3 1 \mid a, b \in \Sigma, a \neq b\})^*,$$

and $g(a_3) = a$ for all $a_3 \in \Sigma_3$, $g(x) = \lambda$ for all $x \in \Sigma_1 \cup \Sigma_2 \cup V$.

□

The previous lemmata allow us to make statements about closure properties of the substitution operations now.

Theorem 9.11 *For a set of trajectories $T \subseteq V^*$, the following three statements are equivalent.*

(i) T is a regular language.

(ii) $L_1 \bowtie_T L_2$ is a regular language for all $L_1, L_2 \subseteq \Sigma^*$.

(iii) $L_1 \triangle_T L_2$ is a regular language for all $L_1, L_2 \subseteq \Sigma^*$.

Proof. The implications (i) ⇒ (ii) and (i) ⇒ (iii) follow by Lemma 9.10 due to the closure of the class of regular languages with respect to shuffle, finite substitution, morphisms and intersection.

To show the implication (ii) ⇒ (i), assume that $L_1 \bowtie_T L_2$ is a regular language for all $L_1, L_2 \subseteq \Sigma^*$. Let $a, b \in \Sigma$ without loss of generality, then also $L = a^* \bowtie_T b^*$ is a regular language, and $T = \phi^{-1}(L)$, ϕ being the coding defined in the proof of Lemma 9.9. Consequently, T is regular. The implication (iii) ⇒ (i) can be shown analogously. □

Theorem 9.12 *For all regular set of trajectories $T \subseteq V^*$ and regular languages $L_1, L_2 \subseteq \Sigma^*$, $L_1 \rhd_T L_2$ is a regular language.*

Proof. Follows by Lemmata 9.10, 3.4. □

Theorem 9.13 *Let $\diamondsuit_T$ be any of the operations $\bowtie_T$, $\triangle_T$, $\rhd_T$.*

(i) Let any two of the languages L_1, L_2, T be regular and the third one be context-free. Then $L_1 \diamondsuit_T L_2$ is a context-free language.

(ii) Let any two of the languages L_1, L_2, T be context-free and the third one be regular. Then $L_1 \diamondsuit_T L_2$ is a non-context-free language for some triples (L_1, L_2, T).

Proof.

(i) Follows by Lemmata 9.10, 3.5 and by closure of the class of context-free languages with respect to finite substitution, morphisms and intersection with regular languages.

(ii) Consider the alphabet $\Sigma = \{a, b, c, d\}$.

(a) Let $\diamondsuit_T = \bowtie_T$.

(1) Consider $L_1 = \{a^n d b^{2n} \mid n > 0\}$, $L_2 = \{a^m c^m \mid m > 0\}$ and $T = V^*$, then $(L_1 \bowtie_T L_2) \cap a^* d a^* c^* = a^n d a^n c^n$.

(2) Consider $L_1 = \{a^n b^{2n} \mid n > 0\}$, $L_2 = c^+$ and $T = \{0^{2m}1^m \mid m > 0\}$, then $L_1 \bowtie_T L_2 = a^n b^n c^n$.

(3) Consider $L_1 = a^+$, $L_2 = \{b^n c^n \mid n > 0\}$ and $T = \{0^m 1^{2m} \mid m > 0\}$, then $L_1 \bowtie_T L_2 = a^n b^n c^n$.

(b) Let $\diamondsuit_T = \triangle_T$. Consider:

(1) $L_1 = \{a^n b a^k b a^l \mid k + l + 1 = 2n > 0\}$, $L_2 = \{a^m b a^{m+1} \mid m > 0\}$ and $T = 0^+1^+$,

(2) $L_1 = \{a^n b^n a^* \mid n > 0\}$, $L_2 = a^+$ and $T = 0^{2m+1}1^m$,

(3) $L_1 = a^+ b a^+$, $L_2 = \{a^n b a^n \mid n > 0\}$ and $T = \{0^m 1^{2m+1} \mid m > 0\}$,

then in all three cases $(L_1 \triangle_T L_2) \cap a^* b^* a b^* = a^n b^n a b^n$.

(c) Let $\diamondsuit_T = \triangleright_T$. Consider:

(1) $L_1 = \{c^{2m} d c^m a^* \mid m > 0\}$, $L_2 = \{a^n b^n d a^* \mid n > 0\}$ and $T = V^+$,

(2) $L_1 = \{b^n a^n d b^+ a^* \mid n > 0\}$, $L_2 = a^+ b^+ d a^+$ and $T = \{1^{2m} 0 1^m 0^* \mid m > 0\}$,

(3) $L_1 = c^+ d c^+ a^*$, $L_2 = \{a^n b^n d a^* \mid n > 0\}$ and $T = \{1^{2m} 0 1^m 0^* \mid m > 0\}$,

then in all three cases $(L_1 \triangleright_T L_2) \cap \{a, b\}^* = a^n b^n a^n$.

In all the above cases we have shown that $L_1 \diamondsuit_T L_2$ is a non-context-free language.

□

Acknowledgements

This research was supported by the Canada Research Chair Grant and NSERC Discovery Grant to L.K., and also by the Grant Agency of Czech Republic, grant No. 201/02/P079 to P.S.

References

[1] H. Jürgensen, S. Konstantinidis, Codes. In [10], pp. 511–607.

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

[3] L. Kari, On language equations with invertible operations, *Theoretical Computer Science* **132** (1994), 129–150.

[4] L. Kari, S. Konstantinidis, Language equations, maximality and error detection. Submitted.

[5] L. Kari, S. Konstantinidis, P. Sosík, On properties of bond-free DNA languages, *Abstracts of papers presented to the American Mathematical Society*, to appear.

[6] C. Martin-Víde, A. Mateescu, G. Rozenberg, A. Salomaa, *Contexts on Trajectories*, TUCS Technical Report No. 214, Turku Centre for Computer Science, 1998.

[7] A. Mateescu, A. Salomaa, Nondeterministic trajectories. *Formal and Natural Computing: Essays Dedicated to Grzegorz Rozenberg, LNCS* **2300** (2002), 96-106.

[8] A. Mateescu, G. Rozenberg, A. Salomaa, Shuffle on trajectories: syntactic constraints, *Theoretical Computer Science* **197** (1998), 1–56.

[9] A. Mateescu, G. Rozenberg, A. Salomaa, *Shuffle on Trajectories: Syntactic Constraints*, TUCS Technical Report No. 41, Turku Centre for Computer Science, 1996.

[10] G. Rozenberg, A. Salomaa (eds.), *Handbook of Formal Languages,* Springer-Verlag, Berlin, 1997.

[11] A. Salomaa, *Theory of Automata,* Pergamon press, Oxford, 1969.

[12] A. Salomaa, *Formal Languages,* Academic press, London, 1973.

[13] S. Yu, Regular languages. In [10], pp. 41–110.