Substitution on Trajectories

Lila Kari[1], Stavros Konstantinidis[2], and Petr Sosík[1,3]

[1] Department of Computer Science, The University of Western Ontario, London, ON, Canada, N6A 5B7
{lila,sosik}@csd.uwo.ca
[2] Dept. of Mathematics and Computing Science, Saint Mary's University, Halifax, Nova Scotia, B3H 3C3 Canada
s.konstantinidis@stmarys.ca
[3] Institute of Computer Science, Silesian University, Opava, Czech Republic

Abstract. The word substitutions are binary word operations which can be basically interpreted as a deletion followed by insertion, with some restrictions applied. Besides being itself an interesting topic in formal language theory, they have been naturally applied to modelling noisy channels. We introduce the concept of *substitution on trajectories* which generalizes a class of substitution operations. Within this framework, we study their closure properties and decision questions related to language equations. We also discuss applications of substitution on trajectories in modelling complex channels and a cryptanalysis problem.

1 Introduction

There are two basic forms of the word substitution operation. The *substitution in α by β* means to substitute certain letters of the word α by the letters of β. The *substitution in α of β* means to substitute the letters of β within α by other letters, provided that β is scattered within α. In both cases the overall length of α is not changed. Also, we assume that a letter must not be substituted by the same letter.

These two operations are closely related and, indeed, we prove in Section 4 that they are mutual left inverses. Their motivation comes from coding theory where they have been used to model certain noisy channels [8]. The natural idea is to assume that during a transfer through a noisy channel, some letters of the transferred word can de distorted — replaced by different letters. This can be modelled by a substitution operation extended to sets of words. This approach also allows one to take into account that certain substitutions are more likely than others. Hence the algebraic, closure and other properties of the substitution operation are of interest, to study how a set of messages (=language) can change when transferred through a noisy channel.

In this paper we generalize the idea of substitution using the syntactical constraints — *trajectories*. The *shuffle on trajectories* as a generalization of sequential insertion has been studied since 1996 [16, 17]. Recently also its inverse — the *deletion on trajectories* has been introduced [1, 10]. A *trajectory* acts as a

J. Karhumäki et al. (Eds.): Theory Is Forever (Salomaa Festschrift), LNCS 3113, pp. 145–158, 2004.

syntactical condition, restricting the positions of letters within the word where an operation places its effect. Hence the shuffle and deletion on trajectories can be understood as meta-operations, defining a whole class of insertion/deletion operations due to the set of trajectories at hand. This idea turned out to be fruitful, with several interesting consequences and applications [1–4, 11, 14, 15].

We give a basic description of these operations in Section 3. Then in Section 4 we introduce on a similar basis the *substitution and difference on trajectories*. From the point of view of noisy channels, the application of trajectories allows one to restrict positions of errors within words, their frequency etc. We then study the closure properties of substitution on trajectories in Section 5 and basic decision questions connected with them in Section 6. In Section 7 we discuss a few applications of the substitution on trajectories in modelling complex noisy channels and a cryptanalysis problem. In the former case, the channels involved permit only substitution errors. This restriction allows us to improve the time complexity of the problem of whether a given regular language is error-detecting with respect to a given channel [13].

2 Definitions

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. The *Hamming distance* $H(u, v)$ between two words u and v of the same length is the number of corresponding positions in which u and v differ. For example, $H(abba, aaaa) = 2$.

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 *size* $|A|$ of the automaton A is the number $|S| + |P|$.

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 size $|T|$ of the transducer T (in standard form) is $|S| + |P|$. We refer the reader to [18] 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, we define

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

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. For an operation $\diamondsuit$ we define its left inverse $\diamondsuit^l$ 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 $\diamondsuit$ as

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

Moreover, 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 1. *(i)* $\diamondsuit^{ll} = \diamondsuit^{rr} = \diamondsuit'' = \diamondsuit$,
(ii) $\diamondsuit'^l = \diamondsuit^{r'} = \diamondsuit^{lr}$,
(iii) $\diamondsuit'^r = \diamondsuit^{l'} = \diamondsuit^{rl}$.

Bellow we list several binary word operations together with their left and right inverses [6, 7].

Catenation: [4] $u \cdot v = \{uv\}$, with $\cdot^l = \longrightarrow_{\text{rq}}$ and $\cdot^r = \longrightarrow_{\text{lq}}$.
Left quotient: $u \longrightarrow_{\text{lq}} v = \{w\}$ if $u = vw$, with $\longrightarrow^l_{\text{lq}} = \cdot'$ and $\longrightarrow^r_{\text{lq}} = \cdot$.
Right quotient: $u \longrightarrow_{\text{rq}} v = \{w\}$ if $u = wv$, with $\longrightarrow^l_{\text{rq}} = \cdot$ and $\longrightarrow^r_{\text{rq}} = \longrightarrow'_{\text{lq}}$.
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 $\rightsquigarrow^r = \rightsquigarrow$.

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

3 Shuffle and Deletion on Trajectories

The above insertion and deletion operations can be naturally generalized using the concept of *trajectories.* A trajectory defines an order in which the operation is applied to the letters of its arguments. Notice that this restriction is purely syntactical, as the content of the arguments has no influence on this order. Formally, a trajectory is a string over the *trajectory alphabet* $V = \{0, 1\}$. The following definitions are due to [1, 16, 10].

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

Definition 1. *The shuffle of α with β on the trajectory t, denoted by $\alpha \sqcup\!\sqcup_t \beta$, is defined as follows:*

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

Definition 2. *The deletion of β from α on trajectory t is the following binary word operation:*

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

Observe that due to the above definition, if $|\alpha| \neq |t|$ or $|\beta| \neq |t|_1$, then $\alpha \rightsquigarrow_t \beta = \emptyset$.

A *set of trajectories* is any set $T \subseteq V^*$. We extend the shuffle and deletion to sets of trajectories as follows:

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

The operations $\sqcup\!\sqcup_T$ and $\rightsquigarrow_T$ generalize to languages due to (1).

Example 1. The following binary word 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, and $\rightsquigarrow_T = \longrightarrow_{\mathrm{rq}}$, the right quotient.
2. For $T = 1^*0^*$ we have $\sqcup\!\sqcup_T = \cdot'$, the anti-catenation, and $\rightsquigarrow_T = \longrightarrow_{\mathrm{lq}}$, the left quotient.
3. Let $T = \{0, 1\}^*$, then $\rightsquigarrow_T = \sqcup\!\sqcup$, the shuffle, and $\rightsquigarrow_T = \rightsquigarrow$, the scattered deletion.

We refer to [1, 16, 10] for further elementary results concerning shuffle and deletion on trajectories.

4 Substitution on Trajectories

Based on the previously studied concepts of the insertion and deletion on trajectories, we consider a generalization of three natural binary word operations which are used to model certain noisy channels [8]. Generally, *channel* [13] is a binary relation $\gamma \subseteq \Sigma^* \times \Sigma^*$ such that (u, u) is in γ for every word u in the input domain of γ – this domain is the set $\{u \mid (u, v) \in \gamma \text{ for some word } v\}$. The fact that (u, v) is in γ means that the word v can be received from u via the channel γ. In [8], certain 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. The following definitions allow one to characterize channels with substitution errors.

Definition 3. *If $u, v \in \Sigma^*$ then we define the* substitution in u by v *as*

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

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

Definition 4. *If $u, v \in \Sigma^*$ then we define the* substitution in u of v *as*

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

Definition 5. *Let $u, v \in \Sigma^*$, $|u| = |v|$, let $H(u, v)$ be the Hamming distance of u and v. We define*

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

The above definitions are due to [8], where it is also shown that the left- and the right-inverse of $\bowtie$ are $\triangle$ and $\rhd$, respectively. Given two binary word operations $\diamondsuit_1$, $\diamondsuit_2$, their composition $(\diamondsuit_1 \diamondsuit_2)$ is defined 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 $\{(u, v) \mid v \in u(\triangle \sqcup\!\sqcup)(\Sigma^0 \cup \dots \cup \Sigma^m)\}$.
(i) The channel with at most m substitution and deletion errors is equal to $\{(u, v) \mid v \in u(\rightsquigarrow \bowtie)(\Sigma^0 \cup \dots \cup \Sigma^m)\}$.

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

$$u \bowtie_t v = \{u_1v_1u_2v_2 \dots u_kv_ku_{k+1} \mid k \geq 0,\ u = u_1a_1 \dots u_ka_ku_{k+1},\ v = v_1 \dots v_k,\\ t = 0^{j_1}10^{j_2}1 \dots 0^{j_k}10^{j_{k+1}},\ a_i, v_i \in \Sigma, 1 \leq i \leq k,\ a_i \neq v_i, \forall i, 1 \leq i \leq k,\\ j_i = |u_i|, 1 \leq i \leq k+1\}.$$

Definition 7. *For a trajectory $t \in V^*$ and $u, v \in \Sigma^*$ we define the* substitution in u of v *on trajectory t as*

$$u \triangle_t v = \{u_1a_1u_2a_2 \dots u_ka_ku_{k+1} \mid k \geq 0,\ u = u_1v_1 \dots u_kv_ku_{k+1},\ v = v_1 \dots v_k,\\ t = 0^{j_1}10^{j_2}1 \dots 0^{j_k}10^{j_{k+1}},\ a_i, v_i \in \Sigma, 1 \leq i \leq k,\ a_i \neq v_i, \forall i, 1 \leq i \leq k,\\ j_i = |u_i|, 1 \leq i \leq k+1\}.$$

Definition 8. *For a trajectory $t \in V^*$ and $u, v \in \Sigma^*$ we define the* right difference of u and v *on trajectory t as*

$$u \rhd_t v = \{v_1v_2 \dots v_k \mid k \geq 0,\ u = u_1a_1 \dots u_ka_ku_{k+1},\ v = u_1v_1 \dots u_kv_ku_{k+1},\\ t = 0^{j_1}10^{j_2}1 \dots 0^{j_k}10^{j_{k+1}},\ a_i, v_i \in \Sigma, 1 \leq i \leq k,\ a_i \neq v_i, \forall i, 1 \leq i \leq k,\\ j_i = |u_i|, 1 \leq i \leq k+1\}.$$

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

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

Example 2. Let $T = V^*$, i.e. the set T contains all the possible trajectories. Then $\bowtie_T = \bowtie$, $\triangle_T = \triangle$ and $\rhd_T = \rhd$.

One can observe that similarly as in [8], 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 2. *For a set of trajectories T and words $u, v \in \Sigma^*$, the following holds:*

(i) $\bowtie_T^l = \triangle_T$ *and* $\bowtie_T^r = \rhd_T$,
(ii) $\triangle_T^l = \bowtie_T$ *and* $\triangle_T^r = \rhd_T'$,
(iii) $\rhd_T^l = \triangle_T'$ *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 $\bowtie_t^r = \triangleright_t$, $t \in T$, follow directly by careful reading the definitions of $\bowtie_t$, $\triangle_t$ and $\triangleright_t$. 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 1, $\bowtie_T^l = \triangle_T$ implies $\triangle_T^l = \bowtie_T$ and $\bowtie_T^r = \triangleright_T$ implies $\triangle_T^r = \bowtie_T^{lr} = \bowtie_T^{r'} = \triangleright_T'$.

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

5 Closure Properties

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 3. *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 morphism such that $\phi(a) = 0$, $\phi(b) = 1$. The statements follow easily by definition. □

Similarly as in the case of shuffle and deletion on trajectories [1, 16, 10], the substitution on trajectories can be characterized by simpler language operations.

Lemma 4. *Let $\diamondsuit_T$ be any of the operations $\bowtie_T$, $\triangle_T$, $\triangleright_T$. 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 $T \subseteq V^*$,*

$$L_1 \diamondsuit_T L_2 = g((h_1(L_1) \sqcup\!\sqcup h_2(L_2) \sqcup\!\sqcup T) \cap R). \tag{3}$$

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 (3) 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 1. *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 4 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 3. Consequently, T is regular. The implication (iii) ⇒ (i) can be shown analogously. □

Theorem 2. *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. The same as the proof of Theorem 1, (i) ⇒ (ii). □

Theorem 3. *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 4, and by closure of the class of context-free languages with respect to finite substitution, shuffle, morphisms and intersection with regular languages.

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

1. Let $\diamondsuit_T = \bowtie_T$.

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

2. 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 $(L_1 \triangle_T L_2) \cap a^* b^* a b^* = a^n b^n a b^n$.

3. Let $\diamondsuit_T = \rhd_T$. 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 $T = \{1^{2m}01^m0^* \mid m > 0\}$,
(3) $L_1 = c^+ d c^+ a^*$, $L_2 = \{a^n b^n d a^* \mid n > 0\}$ and $T = \{1^{2m}01^m0^* \mid m > 0\}$,
then in all three cases $(L_1 \rhd_T L_2) \cap \{a, b\}^* = a^n b^n a^n$.

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

□

6 Decision Problems

In this section we study three elementary types of decision problems for language equations of the form $L_1 \diamondsuit_T L_2 = R$, where $\diamondsuit_T$ is one of the operations $\bowtie_T$, $\triangle_T$, $\rhd_T$. These problems, studied already for various binary word operations in [7, 6, 1, 10, 5] and others, are stated as follows. First, given L_1, L_2 and R, one asks whether the above equation holds true. Second, the existence of a solution L_1 to the equation is questioned, when L_1 is unknown (the left operand problem). Third, the same problem is stated for the right operand L_2. All these problems have their variants when one of L_1, L_2 (the unknown language in the case of the operand problems) consists of a single word.

We focus now on the case when L_1, L_2 and T are all regular languages. Then $L_1 \diamondsuit_T L_2$ is also a regular language by Theorems 1, 2, $\diamondsuit_T$ being any of the operations $\bowtie_T$, $\triangle_T$, $\rhd_T$. Immediately we obtain the following result.

Theorem 4. *The following problems are both decidable if the operation $\diamondsuit_T$ is one of $\bowtie_T$, $\triangle_T$, $\rhd_T$, T being a regular set of trajectories:*

(i) For given regular languages L_1, L_2, R, is $L_1 \diamondsuit_T L_2 = R$?
(ii) For given regular languages L_1, R and a word $w \in \Sigma^*$, is $L_1 \diamondsuit_T w = R$?

Also the decidability of the left and the right operand problems for languages are straightforward consequences of the results in Section 5 and some previously known facts about language equations [7].

Theorem 5. *Let $\diamondsuit_T$ be one of the operations $\bowtie_T$, $\triangle_T$, $\rhd_T$. The problem "Does there exist a solution X to the equation $X \diamondsuit_T L = R$?" (left-operand problem) is decidable for regular languages L, R and a regular set of trajectories T.*

Proof. Due to [7], if a solution to the equation $X \diamondsuit_T L = R$ exists, then also $X_{\max} = (R^c \diamondsuit_T^l L)^c$ is also a solution, $\diamondsuit_T$ being an invertible binary word operation. In fact, $X_{\max}$ is the maximum (with respect to the subset relation) of all the sets X such that $X \diamondsuit_T L \subseteq R$. We can conclude that a solution X exists iff

$$(R^c \diamondsuit_T^l L)^c \diamondsuit_T L = R. \tag{4}$$

holds. Observe that if $\diamondsuit_T$ is one of $\bowtie_T$, $\triangle_T$, $\rhd_T$, then $\diamondsuit_T^l$ is $\triangle_T$, $\bowtie_T$ or $\triangle'_T$, respectively, by Lemma 2. Hence the left side of the equation (4) represents an effectively constructible regular language by Theorems 1, 2. Consequently, the validity of (4) is decidable and moreover the maximal solution $X_{\max} = (R^c \diamondsuit_T^l L)^c$ can be effectively found if one exists. □

Theorem 6. *Let $\diamondsuit_T$ be one of the operations $\bowtie_T$, $\triangle_T$, $\rhd_T$. The problem "Does there exist a solution X to the equation $L \diamondsuit_T X = R$?" (right-operand problem) is decidable for regular languages L, R and a regular set of trajectories T.*

Proof. Similarly as in the proof of Theorem 5, a maximal solution to the equation $L \diamondsuit_T X = R$ is $X_{\max} = (L \diamondsuit_T^r R^c)^c$ for a binary word operation $\diamondsuit_T$, see [7]. Hence a solution X exists iff

$$L \diamondsuit_T (L \diamondsuit_T^r R^c)^c = R \tag{5}$$

By Lemma 2, if $\diamondsuit_T$ is one of $\bowtie_T$, $\triangle_T$, $\rhd_T$, then $\diamondsuit_T^r$ is $\rhd_T$, $\rhd'_T$ or $\bowtie_T$, respectively. Again the validity of (5) is effectively decidable by Theorems 1, 2, and, moreover, an eventual maximal solution $X_{\max} = (L \diamondsuit_T^r R^c)^c$ can be effectively found. □

The situation is a bit different in the case when the existence of a singleton solution to the left or the right operand problem is questioned. Another proof technique takes place.

Theorem 7. *Let $\diamondsuit_T$ be one of the operations $\bowtie_T$, $\triangle_T$, $\rhd_T$. The problem "Does there exist a word w such that $w \diamondsuit_T L = R$?" is decidable for regular languages L, R and a regular set of trajectories T.*

Proof. Assume that $\diamondsuit_T$ is one of $\bowtie_T$, $\triangle_T$, $\rhd_T$. Observe first that if $y \in w \diamondsuit_T x$ for some $w, x, y \in \Sigma^*$, then $|y| \leq |w|$. Therefore, if R is infinite, then there cannot exist a solution w of a finite length satisfying $w \diamondsuit_T L = R$. Hence for an infinite R the problem is trivial.

Assume now that R is finite. As shown in [7], the regular set $X_{\max} = (R^c \diamondsuit_T^l L)^c$ is the maximal set with the property $X \diamondsuit_T L \subseteq R$. Hence w is a solution of $w \diamondsuit_T L = R$ iff

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

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

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

Since we assume that R is finite, the set S is regular and effectively constructible by Lemma 2, Theorems 1, 2 and the closure of REG under finite union and complement. Hence it is also decidable whether S is empty or not, and eventually all its elements can be effectively listed. □

Theorem 8. *Let $\diamondsuit_T$ be one of the operations $\bowtie_T$, $\triangle_T$, $\rhd_T$. The problem "Does there exist a word w such that $L \diamondsuit_T w = R$?" is decidable for regular languages L, R and a regular set of trajectories T.*

Proof. Assume first that $\diamondsuit_T$ is one of $\bowtie_T$, $\triangle_T$. Observe that if $y \in x \diamondsuit_T w$ for some $w, x, y \in \Sigma^*$, then $|y| \geq |w|$. Therefore, if a solution w to the equation $L \diamondsuit_T w = R$ exists, then $|w| \leq k$, where $k = \min\{|y| \mid y \in R\}$. Hence, to verify whether a solution exists or not, it suffices to test all the words from $\Sigma^0 \cup \Sigma^1 \cup \cdots \cup \Sigma^k$.

Focus now on the operation $\rhd_T$. Analogously to the case of Theorem 7, we can deduce that there is no word w satisfying $L \rhd_T w = R$, if R is infinite. Furthermore, the set $X_{\max} = (L \rhd_T^r R^c)^c = (L \bowtie_T R^c)^c$ is the maximal set with the property $L \rhd_T X \subseteq R$. The same arguments as in the proof of Theorem 7 allow one to express the set of all singleton solutions as

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

For a finite R, the set S is regular and effectively constructible, hence we can decide whether it contains at least one solution. □

We add that in the above cases of the left and the right operand problems, if there exists a solution, then at least one can be effectively found. Moreover, in the case of their singleton variants, *all* the singleton solutions can be effectively enumerated.

7 Applications

In this section we discuss a few applications of the substitution-on-trajectories operation in modelling certain noisy channels and a cryptanalysis problem. In the former case, we revisit a decidability question involving the property of error-detection.

For positive integers m and l, with $m < l$, consider the SID channel [12] that permits at most m substitution errors in any l (or less) consecutive symbols of any input message. Using the operation $\bowtie_T$, this channel is defined as the

set of pairs of words (u, v) such that u is in $v \bowtie_T \Sigma^*$, where T is the set of all trajectories t such that, for any subword s of t, if $|s| \leq l$ then $|s|_1 \leq m$. In general, following the notation of [8], for any trajectory set T we shall denote by $[\bowtie_T \Sigma^*]$ the channel $\{(u, v) \mid v \in u \bowtie_T \Sigma^*\}$. In the context of noisy channels, the concept of error-detection is fundamental [13]. A language L is called *error-detecting for* a channel γ, if γ cannot transform a word in L_λ to another word in L_λ; that is, if $u, v \in L_\lambda$ and $(u, v) \in \gamma$ then $u = v$. Here L_λ is the language $L \cup \{\lambda\}$. The empty word in this definition is needed in case the channel permits symbols to be inserted into, or deleted from, messages – see [13] for details. In our case, where only substitution errors are permitted, the above definition remains valid if we replace L_λ with L.

In [13] it is shown that, given a rational relation γ and a regular language L, we can decide in polynomial time whether L is error-detecting for γ. Here we take advantage of the fact that the channels $[\bowtie_T \Sigma^*]$ permit only substitution errors and improve the time complexity of the above result.

Theorem 9. *The following problem is decidable in time $O(|A|^2|T|)$.*

Input: NFA A over Σ and NFA T over $\{0, 1\}$.

Output: Y/N, depending on whether $L(A)$ is error-detecting for $[\bowtie_T \Sigma^*]$.

Proof. In [9] it is shown that given an NFA A, one can construct the NFA A^σ, in time $O(|A|^2)$, such that the alphabet of A^σ is $E = \Sigma \times \Sigma$ and the language accepted by A^σ consists of all the words of the form $(x_1, y_1) \cdots (x_n, y_n)$, with each $(x_i, y_i) \in E$, such that $x_1 \cdots x_n \neq y_1 \cdots y_n$ and the words $x_1 \cdots x_n$ and $y_1 \cdots y_n$ are in $L(A)$. Let ϕ be the morphism of E into $\{0, 1\}$ such that $\phi(x, y) = 0$ iff $x = y$. One can verify that $L(A)$ is error-detecting for $[\bowtie_T \Sigma^*]$ iff the language $\phi(L(A^\sigma)) \cap L(T)$ is empty. Using this observation, the required algorithm consists of the following steps: (i) Construct the NFA A^σ from A. (ii) Construct the NFA $\phi(A^\sigma)$ by simply replacing each transition $s(x, y) \to t$ of A^c with $s\phi(x, y) \to t$. (iii) Use a product construction on $\phi(A^\sigma)$ and T to obtain an NFA B accepting $\phi(L(A^\sigma)) \cap L(T)$. (iv) Perform a depth first search algorithm on the graph of B to test whether there is a path from the start state to a final state. □

We close this section with a cryptanalysis application of the operation $\bowtie_T$. Let V be a set of candidate binary messages (words over $\{0, 1\}$) and let K be a set of possible binary keys. An unknown message v in V is encrypted as $v \oplus t$, where t is an unknown key in K, and $\oplus$ is the exclusive-OR logic operation. Let e be an observed encrypted message and let T be a set of possible guesses for t, with $T \subseteq K$. We want to find the subset X of V for which $X \oplus T = e$, that is, the possible original messages that can be encrypted as e using the keys we have guessed in T. In general T can be infinite and given, for instance, by a regular expression describing the possible pattern of the key. We can model this problem using the following observation whose proof is based on the definitions of the operations $\bowtie_T$ and $\oplus$, and is left to the reader.

Lemma 5. *For every word v and trajectory t, $v \bowtie_T \Sigma^* = \{v \oplus t\}$.*

By the above lemma, we have that the equation $X \oplus T = e$ is equivalent to $X \bowtie_T \Sigma^* = e$. By Theorem 5, we can decide whether there is a solution for this equation and, in this case, find the maximal solution $X_{\max}$. In particular, $X_{\max} = (e^c \triangle_T \Sigma^*)^c$. Hence, one needs to compute the set $M \cap X_{\max}$. Most likely, for a general T, this problem is intractable. On the other hand, this method provides an alternate way to approach the problem.

References

1. M. Domaratzki, *Deletion Along Trajectories.* Tech. report 464-2003, School of Computing, Queen's University, 2003, and accepted for publication.
2. M. Domaratzki, *Splicing on Routes versus Shuffle and Deletion Along Trajectories.* Tech. report 2003-471, School of Computing, Queen's University, 2003.
3. M. Domaratzki, *Decidability of Trajectory-Based Equations.* Tech. report 2003-472, School of Computing, Queen's University, 2003.
4. M. Domaratzki, A. Mateescu, K. Salomaa, S. Yu, Deletion on Trajectories and Commutative Closure. In T. Harju and J. Karhumaki, eds., *WORDS'03: 4th International Conference on Combinatorics on Words.* TUCS General Publication No. 27, Aug. 2003, 309–319.
5. M. Ito, L. Kari, G. Thierrin, Shuffle and scattered deletion closure of languages. *Theoretical Computer Science* **245** (2000), 115–133.
6. L. Kari, On insertion and deletion in formal languages, *PhD thesis*, University of Turku, Finland, 1991.
7. L. Kari, On language equations with invertible operations, *Theoretical Computer Science* **132** (1994), 129–150.
8. L. Kari, S. Konstantinidis, Language equations, maximality and error detection. Submitted.
9. L. Kari, S. Konstantinidis, S. Perron, G. Wozniak, J. Xu, *Finite-state error/edit-systems and difference measures for languages and words.* Dept. of Math. and Computing Sci. Tech. Report No. 2003-01, Saint Mary's University, Canada, 2003.
10. L. Kari, P. Sosík, *Language deletion on trajectories.* Dept. of Computer Science technical report No. 606, University of Western Ontario, London, 2003, and submitted for publication.
11. L. Kari, S. Konstantinidis, P. Sosík, *On Properties of Bond-Free DNA Languages.* Dept. of Computer Science Tech. Report No. 609, Univ. of Western Ontario, 2003, and submitted for publication.
12. S. Konstantinidis, An algebra of discrete channels that involve combinations of three basic error types. *Information and Computation* **167** (2001), 120–131.
13. S. Konstantinidis, Transducers and the properties of error detection, error correction and finite-delay decodability. *J. Universal Comp. Science* **8** (2002), 278–291.
14. C. Martin-Víde, A. Mateescu, G. Rozenberg, A. Salomaa, *Contexts on Trajectories*, TUCS Technical Report No. 214, Turku Centre for Computer Science, 1998.
15. A. Mateescu, A. Salomaa, Nondeterministic trajectories. *Formal and Natural Computing: Essays Dedicated to Grzegorz Rozenberg, LNCS* **2300** (2002), 96-106.
16. A. Mateescu, G. Rozenberg, A. Salomaa, Shuffle on trajectories: syntactic constraints, *Theoretical Computer Science* **197** (1998), 1–56.

17. A. Mateescu, G. Rozenberg, A. Salomaa, *Shuffle on Trajectories: Syntactic Constraints*, TUCS Technical Report No. 41, Turku Centre for Computer Science, 1996.
18. G. Rozenberg, A. Salomaa (eds.), *Handbook of Formal Languages,* Springer-Verlag, Berlin, 1997.