Block Insertion and Deletion on Trajectories

Bo Cui*, Lila Kari*, Shinnosuke Seki*

Department of Computer Science, University of Western Ontario, London, Ontario, Canada, N6A 5B7

Abstract

In this paper, we introduce block insertion and deletion on trajectories, which provide us with a new framework to study properties of language operations. With the parallel syntactical constraint provided by *trajectories*, these operations properly generalize several sequential as well as parallel binary language operations such as catenation, sequential insertion, k-insertion, parallel insertion, quotient, sequential deletion, k-deletion, etc.

We establish some relationships between the new operations and shuffle and deletion on trajectories, and obtain several closure properties of the families of regular and context-free languages under the new operations. Moreover, we obtain several decidability results of three types of language equation problems which involve the new operations. The first one is to answer, given languages L_1, L_2, L_3 and a trajectory set T, whether the result of an operation between L_1 and L_2 on the trajectory set T is equal to L_3. The second one is to answer, for three given languages L_1, L_2, L_3, whether there exists a set of trajectories such that the block insertion or deletion between L_1 and L_2 on this trajectory set is equal to L_3. The third problem is similar to the second one, but the language L_1 is unknown while languages L_2, L_3 as well as a trajectory set T are given.

Keywords: Block insertion on trajectories, Block deletion on trajectories, Closure properties, Language equations, Decidability

1. Introduction

The study of language operations is a fundamental research area of the theory of computation, and has played an essential role in understanding the mechanisms of generating words and languages. Some basic operations, such as catenation, shuffle, and quotients, have been extensively studied in the literature. As generalizations of these operations, several operations were introduced: sequential and parallel insertion and deletion [1], k-insertion and k-deletion (introduced in [2] under the name of k-catenation and k-quotient, respectively),

*Corresponding author (Tel) +1 519-661-2111 (ext.84024) (Fax) +1 519-661-3515

Email addresses: bcui2@csd.uwo.ca (Bo Cui), lila@csd.uwo.ca (Lila Kari), sseki@csd.uwo.ca (Shinnosuke Seki)

schema for parallel insertion and deletion [3], distributed catenation [4], mix operation [5], and shuffle and deletion on trajectories [6, 7, 8]. The notion of shuffle on trajectories was first introduced by Mateescu, Rozenberg, and Salomaa [7] with an intuitive geometrical interpretation. It provides us with a sequential syntactical control over the operation of insertion: a trajectory describes how to insert the letters of a word into another word. As its left-inverse operation [9], deletion on trajectories was independently introduced by Domaratzki [6], and Kari and Sosík [8].

We introduce two operations here, *block insertion on trajectories* and its *left-language-inverse* operation called *block deletion on trajectories*. Trajectories over the binary alphabet $\{0, 1\}$ enable us to specify selected positions where a language can be inserted. A trajectory corresponds to the spaces at the beginning, between two letters, and at the end of a word. If a digit in a trajectory is 1, this signifies an insertion of the language at that location, and, if it is 0, then no insertion is performed there. Block insertion on trajectories is a proper generalization of several sequential and parallel binary language operations such as catenation, sequential insertion, k-insertion, parallel insertion, etc. For instance, parallel insertion of a language into a word inserts the language between the letters of the word, as well as before the first letter, and after the last letter of the word. Parallel-inserting a language L into a word abc results in $LaLbLcL$. Thus, by using a trajectory consisting of only 1's, parallel insertion of a language into a word can be realized by the block insertion of the language into the word on a trajectory in 1^*. Moreover, different choices of trajectories will provide us with more flexible syntactical control over parallel insertion. Block deletion on trajectories is defined as the left-language-inverse operation of block insertion on trajectories such that if we can obtain a word w by block-inserting a language L into a word u on a trajectory t, then u can be obtained by block-deleting L from w on the same t possibly along with other words. This operation also properly generalizes some operations, such as quotient, sequential deletion, k-deletion, etc.

We notice that a major difference between shuffle on trajectories and block insertion on trajectories is the way of using their trajectories. However, we prove that block insertion on trajectories can be simulated in two steps by using shuffle on trajectories and substitutions, respectively (Lemma 5). Similarly, although deletion on trajectories and block deletion on trajectories use their trajectories differently, we can simulate block deletion on trajectories by using deletion on trajectories and substitutions (Lemma 6). These representation lemmas enable us to make use of the known closure properties of language families under shuffle and deletion on trajectories in order to prove closure properties of these families under block insertion and deletion on trajectories. Some of these closure properties are generalizations of those under the operations which are special cases of block insertion and deletion on trajectories, and among them are several of interest. For instance, deleting an *arbitrary* language from a regular language on a regular set of trajectories results in a regular language (Proposition 6); the corresponding result regarding quotient is well-known [10].

Next, we consider decision problems about language equations of the form

$L_1 \leftarrow_T L_2 = L_3$ (block inserting L_2 into L_1 on T results in L_3) and its block-deletion variant. If all of the four involved languages are given, the problem is the equality test. Once we replace some of these languages with variables $X, Y, \ldots$, the problem becomes finding a solution. In this paper, we consider the equality test as well as finding a solution to $L_1 \leftarrow_X L_2 = L_3$, $X \leftarrow_T L_2 = L_3$, and their block-deletion variants. It is commonly expected that problems are decidable only when the languages involved are all regular, and become undecidable once any of the languages becomes context-free. Indeed, most of the results obtained in this paper agree to this expectation. Exceptions occur when the operation is block deletion with all the involved languages but L_2 being assumed to be regular. Then for both the equality test and the existence of trajectory set, the boundary between decidability and undecidability shifts to between L_2 being context-free and being context-sensitive (Propositions 10, 11 and Propositions 20, 21, respectively).

This paper is organized as follows: the next section contains basic notions and notation used throughout this paper. In Section 3, we provide formal definitions of block insertion and deletion on trajectories and give several of their basic properties as well as the representation lemmas. Section 4 is devoted to the closure properties under these operations. The equality test, existence of trajectory and left operand are discussed in Sections 5, 6, and 7, respectively.

2. Preliminaries and definitions

An alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$ is a nonempty, finite, and totally-ordered set of n-letters. A word over Σ is a sequence of letters in Σ. The length of a word $w \in \Sigma^*$, denoted by $|w|$, is the number of letters in this word. The empty word, denoted by λ, is the word of length 0. The set of all words over Σ is denoted by Σ^*, and $\Sigma^+ = \Sigma^* \setminus \{\lambda\}$ is the set of all nonempty words. A language is a subset of Σ^*. A language consisting of exactly one word is said to be *singleton*. The complement of a language L, denoted by L^c, is defined as $\Sigma^* \setminus L$. The right quotient of a language L by a word u is defined by $Lu^{-1} = \{w \mid wu \in L\}$.

For a letter $a \in \Sigma$, the number of occurrences of a in a word w is denoted by $|w|_a$. The *Parikh image* of a word $w \in \Sigma^*$, denoted by $\Psi(w)$, is $\Psi(w) = \{(|w|_{a_1}, |w|_{a_2}, \ldots, |w|_{a_n})\}$. We can extend this to a language $L \subseteq \Sigma^*$ as $\Psi(L) = \bigcup_{w \in L} \Psi(w)$.

A *(non-deterministic) finite automaton* (NFA) is a tuple $A = (Q, \Sigma, \delta, s, F)$, where Q is a finite set of states, $s \in Q$ is the start state, and $F \subseteq Q$ is the set of final states. $\delta : Q \times \Sigma \to 2^Q$ is called a transition function. If $|\delta(q, a)| \leq 1|$ for any $q \in Q$ and $a \in \Sigma$, then this automaton is called a *deterministic* finite automaton (DFA). We extend δ to $Q \times \Sigma^* \to 2^Q$ in the usual way. Then this automaton accepts a word $w \in \Sigma^*$ if $\delta(s, w) \cap F \neq \emptyset$. It is a well-known fact that a language which is accepted by an NFA can be accepted by a DFA, and such language is said to be *regular*.

The *context-free languages* (CFLs) are produced by *context-free grammars*. If a language is produced by a *linear context-free grammar*, then it is called

a *linear context-free language* (LCFL). For more details about grammars, the reader is referred to [11].

For each letter a of Σ, let $s(a)$ be a language over an alphabet Σ_a. Furthermore, define, $s(\lambda) = \lambda$, $s(au) = s(a)s(u)$ for $a \in \Sigma$ and $u \in \Sigma^*$. Such a mapping s from Σ^* into $2^{\Sigma'^*}$, where Σ' is the union of the alphabets Σ_a, is called a *substitution*. A substitution s is said to be regular (context-free) if each of the languages $s(a)$ is regular (resp. context-free). The family of regular (context-free) languages is closed under regular (resp. context-free) substitution [12]. A substitution h such that each $h(a)$ consists of a single word is called a *homomorphism*. The *inverse substitution* s^{-1} of a substitution s is defined for each $w \in \Sigma^*$ by $s^{-1}(w) = \{u \mid w \in s(u)\}$. Furthermore, for a language $L \subseteq \Sigma^*$, $s^{-1}(L) = \bigcup_{w \in L} s^{-1}(w) = \{u \mid w \in s(u) \text{ for some } w \in L\}$.

Now let us recall the definition of left-inverse operations from [9]. For two binary word operations $\star$ and $\bullet$, the operation $\bullet$ is said to be the *left-inverse* of the operation $\star$ if for all words u, v, w over an alphabet, the equivalence "$w \in (u \star v) \iff u \in (w \bullet v)$" holds.

Lastly, we recall the definitions of shuffle and deletion on trajectories. A *trajectory* is a binary word over an alphabet $\{0, 1\}$. For two words $u, v \in \Sigma^*$, the *shuffle of u with v on a trajectory t*, denoted by $u \sqcup_t v$, is defined as follows:

$$\begin{aligned} u \sqcup_t v \quad = \quad & \{u_1 v_1 \cdots u_k v_k \mid u = u_1 \cdots u_k, v = v_1 \cdots v_k, t = 0^{i_1}1^{j_1} \cdots 0^{i_k}1^{j_k}, \\ & \text{where } |u_m| = i_m \text{ and } |v_m| = j_m \text{ for all } m, 1 \leq m \leq k\}. \end{aligned}$$

As its left-inverse operation, one can define the *deletion of v from a word w on t*, denoted by $w \rightsquigarrow_t v$, as follows:

$$\begin{aligned} w \rightsquigarrow_t v \quad = \quad & \{\, u_1 \cdots u_k \mid w = u_1 v_1 \cdots u_k v_k, v = v_1 \cdots v_k, t = 0^{i_1}1^{j_1} \cdots 0^{i_k}1^{j_k}, \\ & \text{where } |u_m| = i_m \text{ and } |v_m| = j_m \text{ for all } m, 1 \leq m \leq k\}. \end{aligned}$$

Note that, in both of these definitions, it is possible to have $i_1 = 0$ and $j_k = 0$. At any rate, by these definitions, $u \sqcup_t v = w$ if and only if $w \rightsquigarrow_t v = u$.

If T is a set of trajectories, the *shuffle of u with v on the set T of trajectories* and the *deletion of v from w on T* are:

$$u \sqcup_T v = \bigcup_{t \in T} u \sqcup_t v, \qquad w \rightsquigarrow_T v = \bigcup_{t \in T} w \rightsquigarrow_t v.$$

Furthermore, the operations $\sqcup_T$ and $\rightsquigarrow_T$ are extended to languages over Σ, if $L_1, L_2 \subseteq \Sigma^*$, then:

$$L_1 \sqcup_T L_2 = \bigcup_{u \in L_1, v \in L_2} u \sqcup_T v, \qquad L_1 \rightsquigarrow_T L_2 = \bigcup_{w \in L_1, v \in L_2} w \rightsquigarrow_T v.$$

3. Block insertion and deletion on trajectories

In this section, we first introduce the formal definitions of block insertion and block deletion on trajectories. Then, we propose several basic properties of

these operations. Lastly, we compare these operations with shuffle and deletion on trajectories and establish relationships between these four operations.

Let us describe block insertion on trajectories first. Given a word $a_1a_2\cdots a_n$ of length n ($n \geq 0$), one can find $n-1$ spaces between two letters. The operation "*block-inserting a language L_2 into the word $a_1\cdots a_n$ on a trajectory t*" inserts L_2 into some of these spaces, as well as possibly in the space to the left of a_1 or the space to the right of a_n. In order for the operation to be performed (to result in a nonempty set), the trajectory $t \in \{0,1\}^*$ has to be of length $n+1$. Each digit of the trajectory word corresponds to a space and specifies whether L_2 is inserted into the space (if the letter is 1) or not (otherwise). The operation is defined formally as follows:

Definition 1. Let $u = a_1\cdots a_n$ such that $a_1,\ldots,a_n \in \Sigma$, $n \in \mathbb{N}$, $L_2 \subseteq \Sigma^*$, and $t = t_0t_1\cdots t_m$ be a trajectory for some $m \geq 0$ and $t_0, t_1, \ldots, t_m \in \{0,1\}$. The *block insertion of L_2 into u on t* is defined as:

$$u \leftarrow_t L_2 = \begin{cases} \emptyset & \text{if } m \neq n, \\ L_0'a_1L_1'\cdots a_nL_n' & \text{if } m = n, \end{cases}$$

where for $0 \leq k \leq n$, $L_k' = L_2$ if $t_k = 1$ and $L_k' = \{\lambda\}$ if $t_k = 0$.

Example 1. $ab \leftarrow_{110} \{ab, b, bc\} = \{ab, b, bc\}a\{ab, b, bc\}b$ (see the following figure), which is

$$\{abaabb, ababb, ababcb, baabb, babb, babcb, bcaabb, bcabb, bcabcb\}.$$

$$\begin{array}{rccccc} ab \leftarrow_{110} \{ab,b,bc\} = & \{ab,b,bc\} & & \{ab,b,bc\} & & \\ & \downarrow & a & \downarrow & b & \\ t = & 1 & & 1 & & 0 \end{array}$$

Next we define block deletion on trajectories.

Definition 2. Let $w \in \Sigma^*$, $L_2 \subseteq \Sigma^*$, and $t = t_0t_1\cdots t_m$ be a trajectory for some $m \geq 0$ and $t_0, t_1, \ldots, t_m \in \{0,1\}$. The *block deletion of L_2 from w on t* is defined as:

$$\begin{aligned} w \rightarrow_t L_2 \quad = \quad \{a_1\cdots a_m \mid\ & w \text{ can be decomposed as } w = v_0a_1\cdots a_mv_m \\ & \text{with } a_1,\ldots,a_m \in \Sigma, \text{ and for } 0 \leq j \leq m, \\ & v_j \in L_2 \text{ if } t_j = 1, \text{ and } v_j = \lambda \text{ if } t_j = 0\}. \end{aligned}$$

By definition, we can see that λ cannot be a trajectory for block insertion or deletion on trajectories.

Recall the definition of left-inverseness. Since parallel operations are defined as an operation from $\Sigma^* \times 2^{\Sigma^*}$ to 2^{Σ^*} and extended, more appropriate "inverseness" should be defined as follows: for two operations $\circ, \diamond$ thus defined and extended, $w \in (u \circ L) \iff u \in (w \diamond L)$ for any words $u, w \in \Sigma^*$ and a language $L \subseteq \Sigma^*$. If $\circ$ and $\diamond$ satisfies this condition, we say that they are *left-l-inverse* to each other. Block insertion and deletion on the same trajectory set are left-l-inverse to each other. This is confirmed by the following stronger result.

Proposition 1. *For two words $w, u \in \Sigma^*$, a language $L_2 \subseteq \Sigma^*$, and a trajectory t, $w \in u \leftarrow_t L_2$ if and only if $u \in w \rightarrow_t L_2$.*

Example 2. As seen in Example 1, $bcabb \in ab \leftarrow_{110} \{ab, b, bc\}$. We can check that $bcabb \rightarrow_{110} \{ab, b, bc\} = \{ab, cb\}$ (depicted as follows). Note that $bcabb \in cb \leftarrow_{110} \{ab, b, bc\}$.

$$bcabb \rightarrow_{110} \{ab, b, bc\} = \{\ \begin{matrix} bc & b \\ \uparrow & \uparrow \\ a & b \\ 1 & 1 \end{matrix}\ 0\ ,\quad \begin{matrix} b & ab \\ \uparrow & \uparrow \\ c & b \\ 1 & 1 \end{matrix}\ 0\ \}.$$

The new operations are extended so as to take languages as their first operand and trajectories: for $L_1, L_2 \subseteq \Sigma^*$ and a set of trajectories T,

$$L_1 \leftarrow_T L_2 = \bigcup_{u \in L_1, t \in T} u \leftarrow_t L_2, \qquad L_1 \rightarrow_T L_2 = \bigcup_{u \in L_1, t \in T} u \rightarrow_t L_2.$$

Due to these extensions, the next result immediately holds as a corollary of Proposition 1.

Corollary 1. *For two words $w, u \in \Sigma^*$, a language $L_2 \subseteq \Sigma^*$, and a trajectory set T, $w \in u \leftarrow_T L_2$ if and only if $u \in w \rightarrow_T L_2$.*

We now obtain several basic properties of the proposed operations. Let us start with the distributivity with respect to the left operand or trajectory set. Note that distributivity does not hold with respect to the right operand.

Lemma 1. *For languages L_1, L_1', L_2 and trajectory sets T, we have*

1. $(L_1 \cup L_1') \leftarrow_T L_2 = (L_1 \leftarrow_T L_2) \cup (L_1' \leftarrow_T L_2)$;
2. $(L_1 \cup L_1') \rightarrow_T L_2 = (L_1 \rightarrow_T L_2) \cup (L_1' \rightarrow_T L_2)$.

Lemma 2. *For languages L_1, L_2 and trajectory sets T_1, T_2, we have*

1. $L_1 \leftarrow_{(T_1 \cup T_2)} L_2 = (L_1 \leftarrow_{T_1} L_2) \cup (L_1 \leftarrow_{T_2} L_2)$;
2. $L_1 \rightarrow_{(T_1 \cup T_2)} L_2 = (L_1 \rightarrow_{T_1} L_2) \cup (L_1 \rightarrow_{T_2} L_2)$.

The next property is about the 0-trajectory, i.e., a subset of 0^+, which actually does not do anything. Combining the next lemma with Lemma 2 leads us to a corollary (Corollary 2), which shall turn out to be helpful to prove some undecidability results of language equations with block insertion or deletion on trajectories in the later sections.

Lemma 3. *For languages L_1 and L_2, $L_1 \leftarrow_{0^+} L_2 = L_1$ and $L_1 \rightarrow_{0^+} L_2 = L_1$.*

Corollary 2. *Let L_1 be a language and T be a set of trajectories such that $0^+ \subseteq T$. Then $L_1 \leftarrow_T L_2 \supseteq L_1$ and $L_1 \rightarrow_T L_2 \supseteq L_1$.*

As another property of block insertion and deletion, we can see that if $L_2 = \emptyset$, then any trajectory which contains 1 cannot produce any word.

Lemma 4. *Let L_1 be a language and T be a set of trajectories. Then $L_1 \leftarrow_T \emptyset = L_1 \leftarrow_{(T \cap 0^+)} \emptyset$ and $L_1 \rightarrow_T \emptyset = L_1 \rightarrow_{(T \cap 0^+)} \emptyset$.*

As remarked in [6, 7], various operations from formal languages are particular cases of the operations of shuffle on and deletion along trajectories. In a similar manner, the block insertion and deletion enable us to simulate some of the operations.

Remark 1. Here we show that some operations are specific cases of block insertion on trajectories.

1. For $T = 0^*1$, $\leftarrow_T$ is the language catenation.
2. For $T = 0^*10^*$, $\leftarrow_T = \leftarrow$ is the sequential insertion [1], which is defined, for two languages L_1, L_2 over the alphabet Σ, as $L_1 \leftarrow L_2 = \cup_{u \in L_1, v \in L_2}(u \leftarrow v)$, where $u \leftarrow v = \{u_1 v u_2 \mid u = u_1 u_2\}$.
3. For $T = \{0^*10^n \mid 0 \leq n \leq k\}$, $\leftarrow_T = \leftarrow^k$ is the k-catenation [2], which is defined, for two languages L_1 and L_2 over the alphabet Σ, as $L_1 \leftarrow^k L_2 = \cup_{u \in L_1, v \in L_2}(u \leftarrow^k v)$ where $u \leftarrow^k v = \{u_1 v u_2 \mid u = u_1 u_2, |u_2| \leq k\}$.
4. For $T = 1^+$, $\leftarrow_T = \Leftarrow$ is the parallel insertion [1], which is defined, for two languages L_1 and L_2 over the alphabet Σ, as $L_1 \Leftarrow L_2 = \cup_{u \in L_1}(u \Leftarrow L_2)$, where $u \Leftarrow L_2 = \{v_0 a_1 v_1 \cdots a_k v_k \mid k \geq 0, a_j \in \Sigma, 1 \leq j \leq k, v_i \in L_2, 0 \leq i \leq k \text{ and } u = a_1 a_2 \cdots a_k\}$.

Unlike shuffle on trajectories, block insertion on trajectories makes it possible to simulate parallel insertion naturally.

Remark 2. Some operations are specific cases of block deletion on trajectories.

1. For $T = 0^*1$, $\rightarrow_T$ is the right quotient.
2. For $T = 0^*10^*$, $\rightarrow_T = \rightarrow$ is the sequential deletion [1], which is defined, for two languages L_1, L_2 over the alphabet Σ, as $L_1 \rightarrow L_2 = \cup_{u \in L_1, v \in L_2}(u \rightarrow v)$, where $u \rightarrow v = \{w \in \Sigma^* \mid u = w_1 v w_2, w = w_1 w_2\}$.
3. For $T = \{0^*10^n \mid 0 \leq n \leq k\}$, $\rightarrow_T = \rightarrow^k$ is the k-deletion [2], which is defined, for two languages L_1 and L_2 over the alphabet Σ, as $L_1 \rightarrow^k L_2 = \cup_{u \in L_1, v \in L_2}(u \rightarrow^k v)$ where $u \rightarrow^k v = \{u_1 u_2 \mid u = u_1 v u_2, |u_2| \leq k\}$.

In contrast to the case of block insertion on trajectories, parallel deletion [1] is not a particular case of block deletion on trajectories. This is because, unlike parallel deletion, block deletion cannot delete two adjacent words.

Having proposed block insertion and deletion on trajectories, we will establish relationships between these new operations and shuffle and deletion on trajectories. We namely show how to simulate block insertion (deletion) on trajectories by shuffle (resp. deletion) with the help of a homomorphism and a substitution (resp. a homomorphism and an inverse substitution). For a given language L_2, the substitution $s_{L_2} : \Sigma \cup \# \rightarrow \Sigma^*$ is defined as $s_{L_2}(a) = a$ for any $a \in \Sigma$ and $s_{L_2}(\#) = L_2$. When L_2 is clear from the context, the subscript of s_{L_2} is omitted. Note that if L_2 is regular, then s is a regular substitution. The homomorphism required is $\phi : \{0,1\}^* \rightarrow \{0,1\}^*$ defined as $\phi(0) = 0$ and $\phi(1) = 10$.

Lemma 5. *Let L_1, L_2 be languages on Σ and $T \subseteq \{0,1\}^*$ be a set of trajectories. Then*

$$L_1 \leftarrow_T L_2 = s_{L_2}(L_1 \sqcup_{\phi(T)0^{-1}} \#^*)$$

Example 3. Let us recall the example of block insertion considered in Example 1: $ab \leftarrow_{110} \{ab, b, bc\}$. The morphism ϕ maps 110 into 10100: $\phi(110) = \phi(1)\phi(1)\phi(0) = 10100$. Then $ab \sqcup_{\phi(110)0^{-1}} \#^* = \{ab \sqcup_{1010} \#^2\} = \{\#a\#b\}$. Substituting $\{ab, b, bc\}$ into #'s completes the simulation of $ab \leftarrow_{110} \{ab, b, bc\}$.

Block deletion on trajectories is the left-l-inverse operation of block insertion on trajectories, and deletion on trajectories is the left-inverse operation of shuffle on trajectories. Thus, it is likely that we can describe the language of the form $u \rightarrow_t L_2$ by deletion on trajectories. Actually, we can simulate $u \rightarrow_t L_2$ using deletion on trajectories, the homomorphism ϕ, and the inverse substitution s^{-1}. Note that for a language $L \subseteq \Sigma^*$, $s^{-1}(L) = \bigcup_{w \in L} s^{-1}(w)$.

Lemma 6. *Let $L_1, L_2 \subseteq \Sigma^*$ be languages and $T \subseteq \{0,1\}^*$ be a set of trajectories. Then*

$$L_1 \rightarrow_T L_2 = (s_{L_2}^{-1}(L_1) \rightsquigarrow_{\phi(T)0^{-1}} \#^*) \cap \Sigma^*.$$

For a word $w \in L_1$, the inverse substitution s^{-1} guesses which of its infixes in L_2 should be deleted by replacing them with #'s. When the guess was wrong, deleting $\#^*$ along $\phi(T)0^{-1}$ leaves some of the #'s unerased and hence the guess is rejected by taking intersection with Σ^*.

Example 4. In Example 2, we saw that $bcabb \rightarrow_{110} \{ab, b, bc\} = \{ab, cb\}$. Keeping in mind that the length of $\phi(110)0^{-1}$ is 4, if we choose from $s^{-1}(bcabb)$ only the words of length 4, then we obtain the set

$$\{\#abb, bc\#b, \#c\#b, \#a\#b, \#ab\#, bc\#\#, \#c\#\#, \#a\#\#\}.$$

Deleting $\#^*$ along $\phi(110)0^{-1} = 1010$ generates the set $\{cb, ab, c\#, a\#\}$. By taking intersection of this set with Σ^*, we finally obtain $\{ab, cb\}$.

In the next section, we will prove closure properties of language families with respect to block insertion and deletion on trajectories, and these representation lemmas play a significant role there. Closure properties with respect to morphism, substitution, right quotient, or intersection, are known. So we conclude this section with one closure property with respect to the specific homomorphism ϕ.

Lemma 7. *A trajectory set T is regular (context-free) if and only if $\phi(T)0^{-1}$ is regular (resp. context-free).*

Proof. The direct implication follows from the fact that the families of regular languages and context-free languages are closed under homomorphism and the right quotient [13].

In order to prove the converse implication, we first note that $\phi(T) = \phi(T)0^{-1}0$ holds. This is because every word in $\phi(T)$ ends with 0 due to the definition of ϕ.

Hence, $\phi(T)0^{-1}$ being regular (context-free) implies that $\phi(T)$ is regular (resp. context-free). Since ϕ is a mapping that encodes T into $\phi(T)$ with a prefix code $\{0, 10\}$, $\phi(T)$ is uniquely decodable. Thus, $\phi^{-1}(\phi(T)) = T$. Since the family of regular languages (context-free languages) is closed under inverse homomorphism [14, 10], we can conclude that T is regular (resp. context-free). □

4. Closure properties

In this section, we obtain several closure properties of the families of regular languages and context-free languages under block insertion and deletion on regular and context-free trajectory sets, mainly based on the representation lemmas and known closure properties with respect to shuffle and deletion on trajectories.

4.1. Closure properties with respect to block insertion

First of all, we consider the case when all of L_1, L_2, T are regular. The following proposition shows that $L_1 \leftarrow_T L_2$ is regular in such a case.

Proposition 2. *Let L_1, L_2 be regular languages over Σ, and T be a regular set of trajectories. Then $L_1 \leftarrow_T L_2$ is regular.*

Proof. Since T is regular, $\phi(T)0^{-1}$ is regular by Lemma 7. Hence, $L_1 \sqcup_{\phi(T)0^{-1}} \#^*$ is regular due to Theorem 5.1 in [7], which states that, if a trajectory set T is regular, then for any regular languages L_1, L_2, $L_1 \sqcup_T L_2$ is regular. Note that s is a regular substitution because L_2 is regular. The family of regular languages is closed under regular substitution [10] so that $s(L_1 \sqcup_{\phi(T)0^{-1}} \#^*)$ is regular. Lemma 5 concludes that $L_1 \leftarrow_T L_2$ is regular. □

The next proposition proves that if one of L_1, L_2, T is a context-free language and the other two are regular languages, then $L_1 \leftarrow_T L_2$ is context-free.

Proposition 3. *Let L_1, L_2 be languages over Σ, and T be a set of trajectories. If one of L_1, L_2, T is context-free and the other two are regular, then $L_1 \leftarrow_T L_2$ is context-free.*

Proof. We first consider the case when T is context-free and L_1, L_2 are regular. Then, $\phi(T)0^{-1}$ is context-free by Lemma 7. Hence, $L_1 \sqcup_{\phi(T)0^{-1}} \#^*$ is context-free due to Theorem 5.2 in [7], which states that, if a trajectory set T is context-free, then for any regular languages L_1, L_2, $L_1 \sqcup_T L_2$ is context-free. Since the family of context-free languages is closed under context-free substitution, and s is a regular substitution, $s(L_1 \sqcup_{\phi(T)0^{-1}} \#^*)$ is context-free. Lemma 5 concludes that $L_1 \leftarrow_T L_2$ is context-free.

Similarly, we can prove that $L_1 \leftarrow_T L_2$ is context-free in the other two cases due to Theorem 5.3 in [7] which states that, if a trajectory set T is regular, then for any languages L_1, L_2, one of them is regular and the other is context-free, $L_1 \sqcup_T L_2$ is context-free. □

Until now, the difference between L_1 and L_2 in their roles in block insertion and deletion has not shown up. Once we expand the investigation onto the case when two of L_1, L_2, T are context-free, the difference becomes apparent in terms of closure properties as shown in the next two propositions.

Proposition 4. *Among L_1, L_2, T, if either L_1 or T is regular and the other two are context-free, then $L_1 \leftarrow_T L_2$ is context-free.*

Proof. In both cases, $L_1 \sqcup\!\sqcup_{\phi(T)0^{-1}} \#^*$ is context-free. The context-free substitution preserves context-freeness so that $s(L_1 \sqcup\!\sqcup_{\phi(T)0^{-1}} \#^*) = L_1 \leftarrow_T L_2$ is context-free using Lemma 5. □

On the other hand, if L_1 and T are context-free, then even if L_2 is singleton, $L_1 \leftarrow_T L_2$ is not always context-free.

Proposition 5. *There exist context-free languages L_1 and $T \subseteq \{0,1\}^*$, and a regular language L_2 such that $L_1 \leftarrow_T L_2$ is not a context-free language.*

Proof. Consider $L_1 = \{v \in \{a,b\}^* \mid |v|_a = |v|_b\}$, $T = \{t \in \{0,1\}^* \mid |t|_0 = |t|_1 + 1\}$, and $L_2 = \{c\}$. It is clear that

$$L_1 \leftarrow_T L_2 = \{w \in \{a,b,c\}^* \mid |w|_a = |w|_b = |w|_c\}.$$

Hence, $L_1 \leftarrow_T L_2$ is not a context-free language. □

4.2. Closure properties with respect to block deletion

We now proceed to the investigation on the closure properties of the families of regular and context-free languages under block deletion on trajectories. As for block insertion on trajectories, we mainly rely on the representation lemma (Lemma 6) and closure properties with respect to deletion on trajectories [6]. Let us recall some of them here:

1. If L_1, T, L_2 are regular, then $L_1 \rightsquigarrow_T L_2$ is also regular. The author introduced an effective method for constructing NFA accepting $L_1 \rightsquigarrow_T L_2$ based on DFAs for L_1, T, and L_2.
2. If one of L_1, T, and L_2 is context-free and the other two are regular, then $L_1 \rightsquigarrow_T L_2$ is context-free, which can be non-regular.
3. If two languages involved in $L_1 \rightsquigarrow_T L_2$ are context-free, and the other one is regular, then $L_1 \rightsquigarrow_T L_2$ is not necessarily context-free.

Combining the first and second results together, we can see that the regularity of $L_1 \rightsquigarrow_T L_2$, when L_1 and T are regular, depends on the regularity of L_2. In contrast, for block deletion on trajectories, $L_1 \rightarrow_T L_2$ is regular regardless of what L_2 is. The proof of this result requires the following technical lemma.

Lemma 8. *Let $L_2 \subseteq \Sigma^*$ be a language and s be the substitution defined as $s(a) = a$ for any $a \in \Sigma$ and $s(\#) = L_2$. For a regular language L_1, $s^{-1}(L_1)$ is a regular language over $\Sigma \cup \{\#\}$, and if further L_2 is context-free, then $s^{-1}(L_1)$ is effectively constructible.*

Proof. Let $A = (Q, \Sigma, \delta, i, F)$ be a deterministic finite automaton for L_1. For two states $p, q \in Q$, let us define $L_{p,q} = \{w \in \Sigma^* \mid \delta(p, w) = q\}$. Then we build up a finite automaton $A' = (Q, \Sigma \cup \{\#\}, \delta', i, F)$, where

$$\delta' = \delta \cup \{(p, \#, q) \mid L_{p,q} \cap L_2 \neq \emptyset\}. \tag{1}$$

One can easily verify that $L(A') = s^{-1}(L_1)$ and hence $s^{-1}(L_1)$ is regular.

Furthermore, if L_2 is context-free, $L_{p,q} \cap L_2$ is context-free and hence the emptiness check in (1) can be done efficiently. This means that we can effectively construct the finite automaton A'. □

Proposition 6. *Let L_1, L_2 be languages over Σ, and T be a set of trajectories. If L_1 is regular and T is regular (context-free), then $L_1 \to_T L_2$ is regular (resp. context-free).*

Proof. Since L_1 is regular, Lemma 8 implies that $s^{-1}(L_1)$ is regular. The previously-mentioned closure properties with respect to deletion along trajectories implies that $s^{-1}(L_1) \rightsquigarrow_{\phi(T)0^{-1}} \#^*$ is regular (context-free) because $\phi(T)0^{-1}$ is regular (resp. context-free). Lemma 6 concludes that $L_1 \to_T L_2$ is regular (resp. context-free). □

Note that the results of Lemma 8 and Proposition 6 are closely related to the classical result that regular languages are closed under quotient with arbitrary languages [10].

In the case of T being regular in this proof, if a finite automaton for $s^{-1}(L_1)$ is given, the result in [6] mentioned previously implies that we can effectively construct an NFA for $L_1 \to_T L_2$ for a context-free language L_2. As a result, the next proposition follows.

Proposition 7. *For a regular language L_1, a regular set T of trajectories, and a context-free language L_2, $L_1 \to_T L_2$ is not only regular but effectively constructible.*

As expected, analogous results do not hold in the case when either L_1 or T is arbitrary, or even context-free. The case when T is context-free is shown in the following example.

Example 5. Consider $L_1 = a^*b^*$, $T = \{0^n 1 0^n \mid n \geq 0\}$, and $L_2 = \{ab\}$. Then $L_1 \to_T L_2 = \{a^n b^n \mid n \geq 0\}$.

Proposition 6 and this example leave the case where L_1 is context-free and T, L_2 are regular. We will show that in this case $L_1 \to_T L_2$ is context-free. The proof requires one technical lemma about a closure property of the family of context-free languages under inverse regular substitution.

Lemma 9. *The family of context-free languages is closed under inverse regular substitution.*

This lemma holds because we can verify that a regular substitution s can be specified by a finite transduction, and its inverse s^{-1} is defined in the same way as the inverse of a finite transduction was defined in Theorem 2.16 [10], which states that the inverse of a finite transduction is a finite transduction. Thus, s^{-1} is also a finite transduction. Furthermore, we know that the family of context-free languages is closed under finite transduction [14]. It might be worth pointing out that the inverse substitution s^{-1} is defined differently in [14] as follows: for a language L, $s^{-1}(L) = \{w \mid s(w) \subseteq L\}$. Under this definition, the family of context-free languages is not closed under inverse substitution. Examples were provided there.

Proposition 8. *Let T be a set of trajectories, and L_1, L_2 be languages over Σ. If L_1 is context-free and T, L_2 are regular, then $L_1 \rightarrow_T L_2$ is context-free.*

Proof. Lemma 6 states that $L_1 \rightarrow_T L_2 = (s^{-1}(L_1) \rightsquigarrow_{\phi(T)0^{-1}} \#^*) \cap \Sigma^*$. Lemmas 7 and 9 imply that $\phi(T)0^{-1}$ is regular and $s^{-1}(L_1)$ is context-free. Due to the closure properties under deletion on trajectories, $s^{-1}(L_1) \rightsquigarrow_{\phi(T)0^{-1}} \#^*$ is context-free, and hence, $L_1 \rightarrow_T L_2$ is context-free. □

Moreover, in the following example, we can see that there exist a context-free language L_1 and regular languages L_2, T such that $L_1 \rightarrow_T L_2$ is a non-regular context-free language.

Example 6. By swapping the roles of L_1 and T in Example 5 as $L_1 = \{a^n b^n \mid n \geq 1\}$ and $T = 0^*10^*$, we have $L_1 \rightarrow_T \{ab\} = \{a^n b^n \mid n \geq 0\}$.

Finally we consider the three cases when two of L_1, L_2, T are context-free. Note that Proposition 6 has already addressed the case when T and L_2 are context-free. The following proposition gives answers to the other two cases.

Proposition 9. *There exist languages L_1, L_2, and a set of trajectories T satisfying each of the following:*

1. *L_1 and L_2 are context-free, and T is regular, but $L_1 \rightarrow_T L_2$ is not context-free;*
2. *L_1 and T are context-free, and L_2 is regular, but $L_1 \rightarrow_T L_2$ is not context-free.*

Proof. 1. Due to Theorem 3.4 in [15], CFLs are not closed under right quotient. When $T = 0^*1$, $\rightarrow_T$ is the right quotient. Thus, the result is immediate.

2. Consider $L_1 = \{a^n b^n c d^m \mid n, m \geq 0\}$, $T = \{0^{2n}10^n \mid n \geq 0\}$, and $L_2 = cd^*$. We can verify that

$$L_1 \rightarrow_T L_2 = \{a^n b^n c^n \mid n \geq 0\},$$

which is well-known not to be context-free. □

Among the closure properties obtained in this section, the results which guarantee the regularity of the resulting language are of special interest. They enable us to obtain decidability results of language equation problems involving block insertion and deletion, some of which will be considered in the following sections.

5. Decision problems of language equations

Now that we have established closure properties with respect to block insertion and deletion on trajectories, let us shift our attention to decision problems which involve these operations.

We begin our investigation with a simple but essential problem: can we test the equality of a language obtained by block insertion (deletion) on trajectories with another language? These problems are formally described as follows: For given languages L_1, L_2, L_3, and a set T of trajectories,

$Q_{0,i}$: is $L_1 \leftarrow_T L_2 = L_3$?

$Q_{0,d}$: is $L_1 \rightarrow_T L_2 = L_3$?

First of all, we observe positive decidability results for both problems. They are due to the fact that the equality between regular languages is decidable as well as to the closure properties of the family of regular languages established in Section 4. It is noteworthy that the decidability of $Q_{0,d}$ does not require L_2 to be regular as long as L_1 and T are regular. In fact, Proposition 7 implies that, for a context-free language L_2, $Q_{0,d}$ remains decidable.

Proposition 10. *Let T be a set of trajectories, and L_1, L_2, L_3 be languages over Σ. The following statements hold true:*

1. *If all of L_1, L_2, L_3, T are regular, the problem $Q_{0,i}$ is decidable.*
2. *If L_1, L_3, T are regular and L_2 is context-free, the problem $Q_{0,d}$ is decidable.*

Here the question arises of whether $Q_{0,d}$ becomes undecidable if we weaken the assumption on L_2 from being context-free to being context-sensitive. The next proposition answers this question affirmatively.

Proposition 11. *Let L_1, L_3 be regular languages and T be a regular set of trajectories. If L_2 is context-sensitive, then the problem $Q_{0,d}$ is undecidable.*

Proof. We first recall that, for a given context-sensitive language L over Σ, it is undecidable whether $L \neq \emptyset$ [16], and context-sensitive languages are closed under catenation with singleton languages [16]. Note that $L \neq \emptyset$ if and only if $Lb \cap \Sigma^+ \neq \emptyset$, where b is a letter in Σ.

Now, we prove the proposition, and reduce the problem of whether $Lb \cap \Sigma^+ \neq \emptyset$ into $Q_{0,d}$ with $L_1 = \Sigma^+$, $T = \{1\}$, $L_2 = Lb$, and $L_3 = \{\lambda\}$. We claim that

$$\Sigma^+ \rightarrow_1 Lb = \{\lambda\} \Longleftrightarrow Lb \cap \Sigma^+ \neq \emptyset.$$

If $Lb \cap \Sigma^+ \neq \emptyset$, then there exists a word $w \in Lb \cap \Sigma^+$. Since $w \rightarrow_1 w = \{\lambda\}$, the left hand side holds. Conversely, if $Lb \cap \Sigma^+ = \emptyset$, then Lb has to be $\emptyset$. In such a case, $\Sigma^+ \rightarrow_1 Lb = \emptyset$. □

One can reasonably expect that once some of the involved languages become context-free (except the case just considered now), the problems $Q_{0,i}$ and $Q_{0,d}$ turn into undecidable. They actually do, except when L_1, L_2, L_3 are over a unary alphabet. Due to Parikh's theorem [17], context-free languages over a unary alphabet are regular so that assuming L_1, L_2, or L_3 context-free makes no sense. Let us assume that L_1, L_2, L_3 are regular and T is context-free. Then the assumption of L_1, L_2, L_3 being unary implies the existence of a regular trajectory set which is "equivalent" to T in the following sense.

Lemma 10. *Let L_1, L_2 be two languages over a unary alphabet. For any context-free trajectory set T, there exists a regular trajectory set T' such that $L_1 \leftarrow_T L_2 = L_1 \leftarrow_{T'} L_2$ ($L_1 \rightarrow_T L_2 = L_1 \rightarrow_{T'} L_2$).*

Proof. Due to Parikh's theorem, there exists a regular set of trajectories T' such that $\Psi(T) = \Psi(T')$, where Ψ is the Parikh mapping.

We show that $L_1 \leftarrow_T L_2 = L_1 \leftarrow_{T'} L_2$. For that, it suffices to show $L_1 \leftarrow_T L_2 \subseteq L_1 \leftarrow_{T'} L_2$, since the reverse inclusion will hold by symmetry. Suppose that $L_1 \leftarrow_T L_2 \not\subseteq L_1 \leftarrow_{T'} L_2$. Then, there exist a word $u = a^n \in L_1$ for some $n \geq 0$, a trajectory $t = t_0 \cdots t_n \in T$ where $t_i \in \{0, 1\}$ for $0 \leq i \leq n$, and some words in L_2, such that $v_0 a v_1 \cdots a v_n \notin L_1 \leftarrow_{T'} L_2$, where, if $t_i = 0$ $v_i = \lambda$, otherwise, $v_i \in L_2$. Thus, $a^{n+\sum_{0 \leq i \leq n} |v_i|}$ is not in $L_1 \leftarrow_{T'} L_2$. However, this is a contradiction, since there exists $t' \in T'$ such that $\Psi(t') = \Psi(t)$, and it is clear that $a^{n+\sum_{0 \leq i \leq n} |v_i|} \in a^n \leftarrow_{t'} L_2$.

Similarly, we can prove the equality $L_1 \rightarrow_T L_2 = L_1 \rightarrow_{T'} L_2$ holds. □

This lemma implies that, when T is context-free and the operand languages are restricted to be unary languages, we just need to consider a regular set of trajectories T' that is letter equivalent to T. Thus, the problems turn out to be equal to the problems solved in Proposition 10.

Corollary 3. *Let T be a context-free trajectory set, and L_1, L_2, L_3 be regular languages over a unary alphabet. Then both problems $Q_{0,i}$ and $Q_{0,d}$ are decidable.*

In the rest of this section and Sections 6 and 7, we assume that L_1, L_2, L_3 are over a non-unary alphabet. To clarify this assumption, we describe problems by using phrases such as "$Q_{0,i}$ over a binary (ternary) alphabet" if a binary (resp. ternary) alphabet is used for the proof. Note that we will present the proofs of Propositions 27, 29, and 30 using ternary alphabets for the sake of readability. The constructions could be straightforwardly encoded over binary alphabets. In the following, we will prove several undecidability results.

Proposition 12. *Let L_1, L_2, L_3 be languages over a binary alphabet Σ, and T be a set of trajectories. The following statements hold true:*

1. *The problem $Q_{0,i}$ over a binary alphabet is undecidable if one of L_1, L_2, L_3, and T is context-free, and the other three are regular.*
2. *The problem $Q_{0,d}$ over a binary alphabet is undecidable if either L_1 or L_3 is context-free, and the other and T are regular.*

Proof. For $Q_{0,i}$, we consider four cases depending on which of the involved languages is context-free.

Firstly we consider $Q_{0,i}$ with T being context-free. Let L be an arbitrary context-free language over $\Sigma = \{a, b\}$ and let $h : \{a, b\}^* \to \{0, 1\}^*$ be a homomorphism which maps a to 0 and b to 1. Let $T_c = h(L)0$. Recall that the morphism ϕ maps 1 to 10 and 0 to 0. Note that for a trajectory $t \in \{0, 1\}^*$, $0^* \sqcup_t 1^* = \{t\}$ holds. Hence, the representation lemma (Lemma 5) shows that $0^* \leftarrow_{T_c} \{1\} = s_{\{1\}}(0^* \sqcup_{\phi(h(L)0)0^{-1}} \#^*) = 0^* \sqcup_{\phi(h(L))} 1^* = \phi(h(L))$. Now if we could decide $Q_{0,i}$ in this setting, for a regular language L_3, we can decide whether $\phi(h(L)) = \phi(h(L_3))$, which is equivalent to $L = L_3$ because $\phi(h(\cdot))$ is a prefix-coding. However, the equality test between regular and context-free languages is undecidable [13].

For the cases when either L_1 or L_3 is context-free, by letting $T = 0^+$, the problem of whether L_1 is equal to L_3 is reduced to the problem "is $L_1 \leftarrow_T L_2$ equal to L_3?". Due to the reason mentioned above, in these cases $Q_{0,i}$ has to be undecidable. For the case when L_2 is context-free, "is L_2 equal to Σ^*" is reduced to $Q_{0,i}$ by choosing $L_1 = \{\lambda\}$, $T = \{1\}$, and $L_3 = \Sigma^*$.

Now it is clear that the usage of $T = 0^+$ leads us to the undecidability of $Q_{0,d}$ under the given conditions because then $L_1 \to_T L_2 = L_3 \iff L_1 = L_3$. □

Let us try to fill the only one remaining gap about $Q_{0,d}$: when T is context-free. The next proposition shows that $Q_{0,d}$ is undecidable also in this case.

Proposition 13. *The problem $Q_{0,d}$ over a binary alphabet is undecidable if L_1 and L_3 are regular, L_2 is singleton, and T is context-free.*

Proof. Let L be an arbitrary context-free language over $\{a, b\}$, h map a to 01 and b to 10, and f map a to $a\#a$ and b to $\#bb$. Choose $T = h(L)0$, $L_1 = \{a, b\}^*$, $L_2 = \{\#\}$, and $L_3 = \{aa, bb\}^*$. We first observe that, for a word $w \in \{a, b\}^*$ and $t \in T$, $f(w) \to_t L_2 \in \{a, b\}^*$ if and only if $t = h(w)0$. Moreover, if $t = h(w)0$, then $f(w) \to_t L_2$ is the word obtained from w by replacing a with aa and b with bb. Thus, we can conclude that $f(L_1) \to_T L_2 = L_3$ if and only if $L = \{a, b\}^*$. This means that if $Q_{0,d}$ were decidable with L_1, L_3 being regular, L_2 being singleton, and T begin context-free, we could decide whether $L = \{a, b\}^*$. □

We conclude this section with a variant of $Q_{0,i}$ and $Q_{0,d}$ when the left-operand is context-free. For a set of trajectories $T \subseteq \{0, 1\}^*$, the Parikh image of T restricted to 0 is

$$\Psi_0(T) = \{|t|_0 \mid t \in T\}.$$

From the definition of ϕ, the following lemma is clear.

Lemma 11. *For a trajectory set $T \in \{0, 1\}^*$, T is finite if and only if $\Psi_0(\phi(T)0^{-1})$ is finite.*

Considering an alphabet Σ, denote $R_0(T) = \bigcup_{d \in \Psi_0(T)} \Sigma^d$.

Proposition 14. *The problem $Q_{0,i}$ is decidable for a context-free language L_1, regular languages L_2, L_3, and a regular trajectory set T if and only if T is finite.*

Problem	L_1	L_2	L_3	T	Result	Proof
$Q_{0,i}$	Reg	Reg	Reg	Reg	D	Proposition 10
	CFL	Reg	Reg	FIN	D	Proposition 14
	CFL	ANY	Reg	INF	U	Proposition 12
	SIN	CFL	Reg	SIN	U	
	Reg	ANY	CFL	Reg	U	
	Reg	SIN	Reg	CFL	U	
$Q_{0,d}$	Reg	CFL	Reg	Reg	D	Proposition 10
	Reg	CSL	Reg	Reg	U	Proposition 11
	CFL	Reg	Reg	FIN	D	Proposition 15
	CFL	ANY	Reg	INF	U	Proposition 12
	Reg	ANY	CFL	Reg	U	
	Reg	SIN	Reg	CFL	U	Proposition 13

Table 1: Decidability results of the problems $Q_{0,i}$ and $Q_{0,d}$, where L_1, L_2, L_3 are over a non-unary alphabet. SIN, FIN, INF, and CSL stand for a singleton, a finite, an infinite, and a context-sensitive language, respectively. ANY means that not depending on what L_2 is, we can prove the undecidability results.

Proof. We prove here only the direct implication because the other direction is trivial. Assume that T is infinite, i.e., $\Psi_0(\phi(T)0^{-1})$ is infinite due to Lemma 11. Let L be an arbitrary context-free language. Consider the regular language $R = \{0,1\}^* \sqcup_{\phi(T)0^{-1}} \#^* = R_0(\phi(T)0^{-1}) \sqcup_{\phi(T)0^{-1}} \#^*$. Intuitively, this equality implies that a word in $\{a,b\}^*$ is useful for the operation $\sqcup_{\phi(T)0^{-1}}$ only if its length is equal to the number of digit 0 of a trajectory in $\phi(T)0^{-1}$. It was proved in Theorem 6.3 in [18] that $L \sqcup_{\phi(T)0^{-1}} \#^* = R$ if and only if $R_0(\phi(T)0^{-1}) \subseteq L$. Using the representation lemma (Lemma 4), we have $L \leftarrow_T \# = L \sqcup_{\phi(T)0^{-1}} \#^*$. Thus, $L \leftarrow_T \# = R$ if and only if $R_0(\phi(T)0^{-1}) \subseteq L$. The latter problem is known to be undecidable [18] so that $Q_{0,i}$ is also undecidable if T is infinite. □

Using the representation lemma (Lemma 6) and the proof of Theorem 6.4 in [18], we can prove an analogous result for block deletion as follows.

Proposition 15. *The problem $Q_{0,d}$ is decidable for a context-free language L_1, regular languages L_2, L_3, and a regular trajectory set T if and only if T is finite.*

The results proved in this section are summarized in Table 1.

6. Existence of trajectories

We now continue our investigation on language equations involving block insertion and deletion on trajectories. Here language equations with one variable are of interest. In particular, the topic of this section is an equation of the form $L_1 \leftarrow_X L_2 = L_3$ or its block deletion variant, where L_1, L_2, L_3 are given and X is a variable. The questions arise in the following form: For given languages L_1, L_2, and L_3,

$Q_{1,i}$**:** does there exist a trajectory set T such that $L_1 \leftarrow_T L_2 = L_3$?

$Q_{1,d}$**:** does there exist a trajectory set T such that $L_1 \rightarrow_T L_2 = L_3$?

Before investigating these problems under various conditions on L_1, L_2, L_3, we note that when the answer to $Q_{1,i}$ or $Q_{1,d}$ is positive, there also exists a maximum solution $T_{\max}$, which is the union of all the solutions to $L_1 \leftarrow_X L_2 = L_3$ respectively $L_1 \rightarrow_X L_2 = L_3$ (this is due to Lemma 2). Therefore, in order to decide the existence of a solution to $L_1 \leftarrow_X L_2 = L_3$ or $L_1 \rightarrow_X L_2 = L_3$, we can employ a technique proposed in [1, 9] that firstly constructs the maximal solution $T_{\max}$ under the assumption that the equation has a solution, and then checks whether $T_{\max}$ is actually its solution.

For $Q_{1,i}$, this candidate is

$$T_0 = \{t \in \{0,1\}^* \mid L_1 \leftarrow_t L_2 \subseteq L_3\}.$$

Lemma 12. *Let L_1, L_2, L_3 be languages. If $L_1 \leftarrow_X L_2 = L_3$ has a solution, then T_0 is its maximum solution.*

Proof. Since the equation is assumed to have a solution, we can let T be its solution, that is, $L_1 \leftarrow_T L_2 = L_3$. We can also assume the existence of its maximum solution $T_{\max}$ defined as the sum of all the solutions. By the definition of T_0, the two solutions T and $T_{\max}$ are subsets of T_0. Then using Lemma 2, we can easily check that

$$\begin{aligned} L_1 \leftarrow_{T_0} L_2 &= (L_1 \leftarrow_T L_2) \cup (L_1 \leftarrow_{T_0 \setminus T} L_2) \\ &= L_3. \end{aligned}$$

Thus, $T_0 \subseteq T_{\max}$. In conclusion, $T_0 = T_{\max}$. □

Furthermore, we can prove that in the case when L_1, L_2, L_3 are regular, T_0 becomes regular.

Lemma 13. *Let $L_1, L_2, L_3 \subseteq \Sigma^*$ be regular languages. Then T_0 is regular and effectively constructible.*

Proof. Here we prove that T_0^c is regular and effectively constructible. Note that $t \in T_0^c$ if and only if $(L_1 \leftarrow_t L_2) \cap L_3^c \neq \emptyset$.

For a trajectory t, the representation lemma (Lemma 5) enables us to describe $L_1 \leftarrow_t L_2$ as $s(L_1 \sqcup_{\phi(t)0^{-1}} \#^*)$, where s is the substitution that substitutes L_2 for $\#$. By the definition of inverse substitution, we can easily check that

$$s(L_1 \sqcup_{\phi(t)0^{-1}} \#^*) \cap L_3^c \neq \emptyset \iff (L_1 \sqcup_{\phi(t)0^{-1}} \#^*) \cap s^{-1}(L_3^c) \neq \emptyset.$$

Thus, $t \in T_0^c$ is equivalent to that $(L_1 \sqcup_{\phi(t)0^{-1}} \#^*) \cap s^{-1}(L_3^c)$ is non-empty. In [19], Domaratzki and Salomaa prove that this nonemptiness can be effectively checked by constructing a finite automaton. Therefore, T_0^c is regular and effectively constructible. □

Combining these lemmas provides us with a decidability result about $Q_{1,i}$.

Proposition 16. *The problem $Q_{1,i}$ is decidable when L_1, L_2, L_3 are regular.*

Proof. Due to Lemma 12, it suffices to decide whether T_0 is its solution or not. Lemma 13 implies that T_0 is regular, and the closure property shown in Section 4 proves that $L_1 \leftarrow_{T_0} L_2$ is regular. In order to test whether T_0 is a solution of $L_1 \leftarrow_X L_2 = L_3$, we simply compare this regular language with the regular language L_3. □

Now we turn our attention to the case when one of L_1, L_2, L_3 is context-free, and the other two are regular. Only languages over non-unary alphabets will be considered for the reason mentioned previously.

Firstly, we consider $Q_{1,i}$ under the assumption that L_1 is context-free and L_2, L_3 are regular.

Proposition 17. *The problem $Q_{1,i}$ over a binary alphabet is undecidable if L_1 is context-free and L_2, L_3 are regular.*

Proof. We prove this result by reducing the undecidable problem of whether $L_1 = \Sigma^*$ to one instance of our problem with $L_2 = \{\lambda\}$ and $L_3 = \Sigma^*$. We claim that

$$\exists\, T \subseteq \{0,1\}^* \text{ such that } L_1 \leftarrow_T \{\lambda\} = \Sigma^* \iff L_1 = \Sigma^*.$$

Indeed, if $L_1 = \Sigma^*$, then $T = 0^*$ satisfies the equation. Conversely, assume that there exists T such that $L_1 \leftarrow_T \{\lambda\} = \Sigma^*$. Then for all $x \in \Sigma^*$, there exist $y \in L_1$ and $t \in T$ such that $x \in y \leftarrow_t \{\lambda\}$. Note that this happens only if $x = y$ and $|t| = |y| + 1$. Therefore, $x \in L_1$ and $L_1 = \Sigma^*$. □

Due to the asymmetry of the operands of block insertion on trajectories, we next consider $Q_{1,i}$ for a context-free language L_2 and regular languages L_1, L_3. We show that, even if L_2 does not contain the empty word, this question is undecidable. Thus, it is undecidable in general.

Proposition 18. *The problem $Q_{1,i}$ over a binary alphabet is undecidable if L_2 is context-free and L_1, L_3 are regular.*

Proof. We reduce the problem of whether $L_2 = \Sigma^+$ to one instance of our problem with $L_1 = \{\lambda\}$ and $L_3 = \Sigma^+$. Then

$$\exists\, T \subseteq \{0,1\}^* \text{ such that } \{\lambda\} \leftarrow_T L_2 = \Sigma^+ \iff L_2 = \Sigma^+.$$

The rest of this proof is similar to that of Proposition 17; hence, omitted. □

The last case for $Q_{1,i}$ is when the resulting language L_3 is context-free. In order to address this problem, we recall one undecidable result proved in [19]. Let us denote the set of non-negative integers by $\mathbb{N}$, and, for a set $I \subseteq \mathbb{N}$, let $\Sigma^I = \{w \in \Sigma^* \mid |x| \in I\}$. Then, for a given LCFL L, it is undecidable whether there exists $I \subseteq \mathbb{N}$ such that $L = \Sigma^I$.

Proposition 19. *The problem $Q_{1,i}$ over a binary alphabet is undecidable if L_3 is linear context-free and L_1, L_2 are regular.*

Proof. We reduce the problem of whether there exists $I \subseteq \mathbb{N}$ such that $L_3 = \Sigma^I$ to an instance of our problem with $L_1 = \Sigma^*$ and $L_2 = \{\lambda\}$. We claim that

$$\exists\, T \subseteq \{0,1\}^* \text{ such that } L_3 = \Sigma^* \leftarrow_T \{\lambda\} \iff \exists\, I \subseteq \mathbb{N} \text{ such that } L_3 = \Sigma^I.$$

If there exists $I \subseteq \mathbb{N}$ such that $L_3 = \Sigma^I$, then let $T = \{0^{i+1} \mid i \in I\}$. We can verify that $L_3 = \Sigma^* \leftarrow_T \{\lambda\}$. Conversely, if there exists $T \subseteq \{0,1\}^*$ such that $L_3 = \Sigma^* \leftarrow_T \{\lambda\}$, then let $I = \{|t| - 1 \mid t \in T \text{ and } |t| \geq 1\}$. Then $L_3 = \Sigma^I$. □

Having considered $Q_{1,i}$, let us investigate the problem $Q_{1,d}$. Firstly, we prove a decidability result for the case when L_1 and L_3 are regular by taking the same strategy to construct the candidate of maximum solution and check its validity. Let

$$T_d = \{t \in \{0,1\}^* \mid L_1 \rightarrow_t L_2 \subseteq L_3\}.$$

The next lemma is the block deletion variant of Lemma 12, which can be proved in the exactly same way so that we omit its proof.

Lemma 14. *Let L_1, L_2, L_3 be languages. If $L_1 \rightarrow_X L_2 = L_3$ has a solution, then T_d is its maximum solution.*

Lemma 13 has also a block deletion variant as shown below. One significant difference is that this variant does not require L_2 to be regular, but exhibits an algorithmically-good behavior when L_2 is at most context-free.

Lemma 15. *Let $L_1, L_3 \subseteq \Sigma^*$ be regular languages and L_2 be an arbitrary language. Then T_d is regular. Furthermore, if L_2 is context-free, then T_d is effectively constructible.*

Proof. Recall that $L_1 \rightarrow_t L_2 = (s^{-1}(L_1) \rightsquigarrow_{\phi(t)0^{-1}} \#^*) \cap \Sigma^*$ (Lemma 6). Due to Lemma 8, $s^{-1}(L_1)$ is regular because L_1 is regular, and moreover becomes effectively constructible when L_2 is context-free. As done in Lemma 13, $t \in T_d$ if and only if $(s^{-1}(L_1) \rightsquigarrow_{\phi(t)0^{-1}} \#^*) \cap L_3^c \neq \emptyset$. We note that for regular languages R_1, R_2, R_3, Domaratzki and Salomaa demonstrated an effective construction of a finite automaton which accepts a trajectory t satisfying $(R_1 \rightsquigarrow_t R_2) \cap R_3^c \neq \emptyset$ [19]. Now it is clear that T_d is regular. Moreover, if L_2 is context-free, applying their method on the finite automata for $s^{-1}(L_1)$, $\#^*$, and L_3^c makes it possible to effectively construct a finite automaton for T_d. □

Lemmas 14 and 15 lead us to a decidable result for $Q_{1,d}$.

Proposition 20. *The problem $Q_{1,d}$ is decidable if L_2 is context-free and L_1, L_3 are regular.*

It is natural to consider here whether the problem $Q_{1,d}$ remains decidable or not once we change L_2 from being context-free to being context-sensitive in Proposition 20.

Proposition 21. *The problem $Q_{1,d}$ is undecidable if L_2 is context-sensitive and L_1, L_3 are regular.*

Proof. The basic idea used here has been already proposed in the proof of Proposition 11. We claim that $\Sigma^+ \rightarrow_X Lb = \{\lambda\}$ has a solution if and only if $Lb \cap \Sigma^+ \neq \emptyset$. From the proof of that proposition, we know that, if $Lb \cap \Sigma^+ \neq \emptyset$, then $X = \{1\}$ is a solution to the equation on the left hand side. Conversely, if $Lb \cap \Sigma^+ = \emptyset$, then Lb has to be the empty set. Note that, in such a case, the only trajectory sets T such that $\Sigma^+ \rightarrow_T Lb \neq \emptyset$ are subsets of 0^*. However, these sets cannot satisfy $\Sigma^+ \rightarrow_T Lb = \{\lambda\}$. □

Next we consider the problem $Q_{1,d}$ under the conditions that one of L_1 and L_3 is context-free, and the other and L_2 are regular. In these cases $Q_{1,d}$ becomes undecidable. Actually, it is enough for the context-free language to be linear to obtain the undecidability results.

Proposition 22. *The problem $Q_{1,d}$ is undecidable over a binary alphabet if L_1 is linear context-free and L_2, L_3 are regular.*

Proof. We prove the proposition by reducing the problem of whether $L_1 = \Sigma^*$ to one instance of our problem with $L_2 = \{\lambda\}$ and $L_3 = \Sigma^*$. We claim that

$$\exists\, T \subseteq \{0,1\}^* \text{ such that } L_1 \rightarrow_T \{\lambda\} = \Sigma^* \iff L_1 = \Sigma^*.$$

If $L_1 = \Sigma^*$, $T = 0^*$ satisfies the equation. Conversely, assume that there exists T such that $L_1 \rightarrow_T \{\lambda\} = \Sigma^*$. Then for all $x \in \Sigma^*$, there exist $y \in L_1$ and $t \in T$ such that $x \in y \rightarrow_t \{\lambda\}$. Note that this happens only if $x = y$ and $|t| = |y| + 1$. Therefore, $x \in L_1$ and $L_1 = \Sigma^*$. □

Proposition 23. *The problem $Q_{1,d}$ is undecidable over a binary alphabet if L_3 is linear context-free and L_1, L_2 are regular.*

Proof. We prove the proposition by reducing the problem of whether there exists $I \subseteq \mathbb{N}$ such that $L_3 = \Sigma^I$ to one instance of our problem with $L_1 = \Sigma^*$ and $L_2 = \{\lambda\}$. We claim that

$$\exists\, T \subseteq \{0,1\}^* \text{ such that } L_3 = \Sigma^* \rightarrow_T \{\lambda\} \iff \exists\, I \subseteq \mathbb{N} \text{ such that } L_3 = \Sigma^I.$$

If there exists $I \subseteq \mathbb{N}$ such that $L_3 = \Sigma^I$, then let $T = \{0^{i+1} \mid i \in I\}$. We can verify that $L_3 = \Sigma^* \rightarrow_T \{\lambda\}$. Conversely, if there exists $T \subseteq \{0,1\}^*$ such that $L_3 = \Sigma^* \rightarrow_T \{\lambda\}$, then let $I = \{|t| - 1 \mid t \in T \text{ and } |t| \geq 1\}$. Note that we do not consider $\rightarrow_\lambda$, because it is not defined for any language. We can verify that $L_3 = \Sigma^I$. □

We summarize the results on $Q_{1,i}$ and $Q_{1,d}$ proved in this section in Table 2 as follows.

7. Existence of left operands

We consider here two other language equations with one variable of the forms $X \leftarrow_T L_2 = L_3$ and $X \rightarrow_T L_2 = L_3$. The questions are formulated as: for given languages L_2, L_3 and a given trajectory set T,

Problem	L_1	L_2	L_3	Result	Proof
$Q_{1,i}$	Reg	Reg	Reg	D	Proposition 16
	CFL	Reg	Reg	U	Proposition 17
	Reg	CFL	Reg	U	Proposition 18
	Reg	Reg	CFL	U	Proposition 19
$Q_{1,d}$	Reg	CFL	Reg	D	Proposition 20
	Reg	CSL	Reg	U	Proposition 21
	CFL	Reg	Reg	U	Proposition 22
	Reg	Reg	CFL	U	Proposition 23

Table 2: Decidability results of the problems $Q_{1,i}$ and $Q_{1,d}$, where L_1, L_2, L_3 are over a non-unary alphabet, and CSL stands for the family of context-sensitive languages.

$Q_{2,i}$**:** does there exist a solution to $X \leftarrow_T L_2 = L_3$?

$Q_{2,d}$**:** does there exist a solution to $X \rightarrow_T L_2 = L_3$?

By limiting a solution of the language equations considered in $Q_{2,i}$ and $Q_{2,d}$ to a singleton, we can obtain word-variants of these questions as follows: for languages L_2, L_3 and a trajectory set T,

$Q^w_{2,i}$**:** does there exist a word x satisfying $x \leftarrow_T L_2 = L_3$?

$Q^w_{2,d}$**:** does there exist a word x satisfying $x \rightarrow_T L_2 = L_3$?

7.1. Positive decidability results

We first consider questions $Q_{2,i}$ and $Q_{2,d}$. As in the problems to find a trajectory, when the answer to these questions is positive, there exists the maximum solution $X_{\max}$ due to Lemma 1. Therefore, we employ the same technique, which constructs $X_{\max}$ and checks whether this is actually a solution.

Here we propose a theorem of how to construct the $X_{\max}$ candidate for $Q_{2,i}$ and $Q_{2,d}$ in a more general setting where $\leftarrow_T$ and $\rightarrow_T$ are replaced by two binary operations $\circ, \diamond : 2^{\Sigma^*} \times 2^{\Sigma^*} \rightarrow 2^{\Sigma^*}$ which are *left-l-inverse* to each other. This is a generalization of Theorem 4.6 in [9]. We omit its proof because it can be obtained by replacing left-inverse in the proof of their result with left-l-inverse.

Theorem 1. *Let $L_2, L_3 \subseteq \Sigma^*$ be languages and $\circ, \diamond : 2^{\Sigma^*} \times 2^{\Sigma^*} \rightarrow 2^{\Sigma^*}$ be operations which are left-l-inverse to each other. If an equation $X \circ L_2 = L_3$ has a solution, then the language $(L_3^c \diamond L_2)^c$ is its maximum solution.*

As done in Section 6, in order to solve $Q_{2,i}$ ($Q_{2,d}$), it suffices to check whether the candidate of maximum solution $(L_3^c \rightarrow_T L_2)^c$ (resp. $(L_3^c \leftarrow_T L_2)^c$) given in Theorem 1 is actually a solution to $X \leftarrow_T L_2 = L_3$ (resp. $X \rightarrow_T L_2 = L_3$). When all of L_2, T, L_3 are regular, this check can be done efficiently. Thus, we have the following decidability results.

Proposition 24. *Both the problems $Q_{2,i}$ and $Q_{2,d}$ are decidable when L_2, L_3, T are regular.*

Recall that block insertion on trajectories becomes parallel insertion introduced in [1] when $T = 1^*$. Thus, the following is a corollary of Proposition 24 and answers one decidability question that was left open in [1].

Corollary 4. *Let $\diamond$ be the parallel insertion, and R_2, R_3 be regular languages. The problem of whether there exists a solution to $X \diamond R_2 = R_3$ is decidable.*

Now we turn our attention to questions $Q^w_{2,i}$ and $Q^w_{2,d}$. Let us consider a decidability result about the problem $Q^w_{2,i}$ first. By definition, we can easily observe that a word x which satisfies $x \leftarrow_T L_2 = L_3$ is of length at most the length, say ℓ, of a shortest word in L_3, unless L_3 is empty. Thus, $Q^w_{2,i}$ can be solved if we check for all the words of length at most ℓ whether the word becomes a solution to $x \leftarrow_T L_2 = L_3$. This check can be done if L_2 and L_3 are regular, the length of shortest words in L_3 is computable, and we can give a list consisting of all elements of length at most $\ell + 1$ of T.

Proposition 25. *The problem $Q^w_{2,i}$ is decidable if L_2 and L_3 are regular, and one can enumerate a trajectory set T.*

Corollary 5. *The problem $Q^w_{2,i}$ is decidable if L_2 and L_3 are regular and T is recursive.*

In contrast, a solution to $x \rightarrow_T L_2 = L_3$ can be arbitrarily long, but finite. Thus, if L_3 is infinite, clearly there exists no word w such that $w \rightarrow_T L_2 = L_3$. Although the brute-force attack does not work for $Q^w_{2,d}$, we can prove a decidability result for this problem under an interesting condition.

Proposition 26. *The problem $Q^w_{2,d}$ is decidable if*

1. *L_2 is regular,*
2. *one can decide whether L_3 is finite or not, and*
3. *one can enumerate a trajectory set T.*

Proof. Note that the emptiness test can be achieved efficiently for regular languages. With the reason just mentioned, it suffices to consider the case when L_3 is finite. Let ℓ' be the length of longest words in L_3. Then any trajectory in T of length at least $\ell' + 2$ is "useless". Since elements of T can be enumerated, we can effectively construct $T' = \{t \in T \mid |t| \leq \ell' + 1\}$. Due to closure properties of the family of regular languages, the following regular language is effectively constructible:

$$W = (L_3^c \leftarrow_{T'} L_2)^c - \bigcup_{S \subset L_3} (S^c \leftarrow_{T'} L_2)^c,$$

where $\subset$ represents proper inclusion. We claim that, for all $w \in \Sigma^*$, $w \in W$ if and only if $w \rightarrow_{T'} L_2 = L_3$.

Due to Theorem 1, given the equation $X \rightarrow_{T'} L_2 = L_3$, the regular set $R' = (L_3^c \leftarrow_{T'} L_2)^c$ is the maximal set with the property $X \rightarrow_{T'} L_2 \subseteq L_3$. Therefore, w is a solution of $w \rightarrow_{T'} L_2 = L_3$ if and only if

1. $w \in R'$, i.e., $w \rightarrow_{T'} L_2 \subseteq L_3$, and

2. $w \rightarrow_{T'} L_2$ is not a proper subset of L_3, i.e., $w \rightarrow_{T'} L_2 \not\subset L_3$.

Note that Condition 2 is equivalent to the following one: for all $S \subset L_3$, $w \rightarrow_{T'} L_2 \not\subseteq S$, and hence $w \notin (S^c \leftarrow_{T'} L_2)^c$. Thus, we can conclude that all the solutions to the equation $w \rightarrow_{T'} L_2 = L_3$ are in W.

To decide whether there exists a word w such that $w \rightarrow_{T'} L_2 = L_3$, we construct W and test the emptiness of W. □

Corollary 6. *The problem $Q^w_{2,d}$ is decidable if L_2 is regular, L_3 is context-free, and T is recursive.*

7.2. Undecidability results

Next, we obtain undecidability results about $Q_{2,i}$, $Q_{2,d}$, and their word-variants. We exclude the case when L_2 and L_3 are over a unary alphabet.

In the following, we will prove that if one of L_2, L_3, T becomes context-free and the others remain regular, then $Q_{2,i}$ becomes undecidable. This is not always the case for $Q^w_{2,i}$ (cf. Proposition 25), but the unsettled cases are considered, that is when either L_2 or L_3 becomes context-free, and the other one as well as T are regular, then $Q^w_{2,i}$ becomes undecidable.

Remark 3. The problems $Q_{2,i}$ and $Q^w_{2,i}$ are undecidable when L_2 is context-free and L_3, T are regular. This is because these problems with some specific T, say $T = 0^*1$ (catenation), $T = 0^*10^*$ (insertion), or $T = \bigcup_{0 \le n \le k} 0^*10^n$ (k-insertion), are known to be undecidable ([1, 2]).

More generally, we can prove that for any non-empty trajectory set $T \subseteq 0^*10^*$, these problems are undecidable, though we omit its proof here.

The next case is when L_3 is context-free. The following proposition addresses the undecidability of $Q_{2,i}$ and $Q^w_{2,i}$ at the same time. To that end, we employ a technique to reduce an undecidable problem into a language equation $X \leftarrow_T L_2 = L_3$ which can have only a singleton solution.

Proposition 27. *The problems $Q_{2,i}$ and $Q^w_{2,i}$ over a ternary alphabet Σ are undecidable if L_2, T are regular and L_3 is context-free.*

Proof. For a given non-empty context-free language $L \subseteq \Sigma^*$, let $L_3 = \#L$, where $\#$ is a special symbol not included in Σ. Also let $L_2 = \Sigma^*$ and $T = \{01\}$. Due to the definition of T, if X is a solution, then $\{x \in X \mid |x| = 1\}$ is also a solution. We claim that $L = \Sigma^*$ if and only if $X \leftarrow_{01} \Sigma^* = \#L$ has a solution which consists only of a word of length 1. In fact, the only possible solution is $X = \{\#\}$ so that the direct implication is trivial with $X = \{\#\}$. Assume that $L \neq \Sigma^*$, i.e., there exists a word $w \notin L$. Since $\#w \notin \#L$, this equation cannot have the solution $X = \{\#\}$. Consequently, $L = \Sigma^*$ if and only if the equation $X \leftarrow_{01} \Sigma^* = \#L$ has a solution. It is undecidable whether a given non-empty context-free language is equal to Σ^* so that our problem is also undecidable. □

The remaining case is when T is context-free. In this case, $Q^w_{2,i}$ remains decidable as mentioned previously.

Proposition 28. *The problem $Q_{2,i}$ over a binary alphabet is undecidable if L_2 is finite, L_3 is regular, and T is context-free.*

Proof. Let L be an arbitrary CFL over $\{a, b\}$. Let h map a to 01 and b to 10, and choose $T = h(L)0$. Note that $T = \{01, 10\}^*0$ if and only if $L = \{a, b\}^*$.

We claim that $X \leftarrow_T \{c\} = \{\#c\#, c\#\#\}^*$ has a solution if and only if $T = \{01, 10\}^*0$. In order to verify this claim, we firstly observe that for any $t \in T$, $w \leftarrow_t \{c\} \in \{\#c\#, c\#\#\}^*$ if and only if $w = \#^{|t|-1}$ and $w \leftarrow_t \{c\} = f(\phi(t)0^{-1})$, where f substitutes $\#$ for 0 and c for 1. Let $m \geq 0$ such that $t \in \{01, 10\}^m0$. Assume that $w \leftarrow_t \{c\}$ is in $\{\#c\#, c\#\#\}^*$. Note that $|\phi(t)0^{-1}|_1 = m$ and $\phi(t)0^{-1} \in \{010, 100\}^m$. Due to the representation lemma, $w \leftarrow_t \{c\} = w \sqcup_{\phi(t)0^{-1}} c^{|\phi(t)0^{-1}|_1} = w \sqcup_{\phi(t)0^{-1}} c^m$, and the above assumption implies that $w \sqcup_{\phi(t)0^{-1}} c^m \in \{\#c\#, c\#\#\}^m$. By comparing the number of $\#$'s, we can see that $w = \#^{2m}$. Then $w \leftarrow_t \{c\} = f(\phi(t)0^{-1})$. Thus, $X \leftarrow_T \{c\} = \{\#c\#, c\#\#\}^*$ has a solution if and only if $\phi(T)0^{-1} = \{010, 100\}^*$ if and only if $T = \{01, 10\}^*0$. □

Now we change our focus onto $Q_{2,d}$ and its word-variant.

Remark 4. It is known that the problems $Q_{2,d}$ and $Q^w_{2,d}$ with $T = 0^*1$ (right quotient) are undecidable when L_2 is context-free and L_3 is regular [1]. Thus in general the problems $Q_{2,d}$ and $Q^w_{2,d}$ are undecidable for context-free L_2, regular L_3, and regular T.

Proposition 29. *The problem $Q_{2,d}$ over a ternary alphabet is undecidable if L_2 and T are regular, and L_3 is context-free.*

Proof. Note that the inclusion is undecidable for the class of context-free languages which contains neither λ nor a word of length 1. Let $\#$ be a special symbol not included in Σ. Let $L_4, L_5 \subseteq \Sigma^*$ be given context-free languages such that $L_4 \cap (\Sigma \cup \{\lambda\}) = L_5 \cap (\Sigma \cup \{\lambda\}) = \emptyset$. Note that $\#(L_4 \cup L_5) \cup L_4\#$ is context-free. Here we claim that $L_5 \subseteq L_4$ if and only if the following equation has a solution:

$$X \rightarrow_{10^* \cup 0^*1} (\{\#\} \cup \Sigma) = \#(L_4 \cup L_5) \cup L_4\#.$$

If the inclusion holds, then the right-hand side of the equation becomes $\#L_4 \cup L_4\#$ so that the equation has a solution $\#L_4\#$. Next suppose that even when $L_5 \not\subseteq L_4$, the equation found a solution. Then L_5 contains a word w which is not in L_4. Since $\#w$ is in $\#(L_4 \cup L_5)$, X has to contain either $\#w\#$, $\#^2w$, $\#wa$, or $b\#w$ for some $a, b \in \Sigma$. Let $w = w'cd$ for some $w' \in \Sigma^*$ and $c, d \in \Sigma$; note that w is of length at least 2 due to the assumption on L_5. From these four words, this deletion would also generate $w\#$, $\#^2w'c$, wa, and $b\#w'c$, respectively. However, none of them can be a member of $\#(L_4 \cup L_5) \cup L_4\#$. Thus, this claim holds. □

Proposition 30. *The problem $Q_{2,d}$ over a ternary alphabet is undecidable if L_2 is finite, L_3 is regular, and T is context-free.*

Problem	L_2	L_3	T	Result	Proof
$Q_{2,i}$	Reg	Reg	Reg	D	Proposition 24
	CFL	Reg	Reg	U	[1, 2], Remark 3
	Reg	CFL	Reg	U	Proposition 27
	FIN	Reg	CFL	U	Proposition 28
$Q_{2,d}$	Reg	Reg	Reg	D	Proposition 24
	CFL	Reg	Reg	U	[1], Remark 4
	Reg	CFL	Reg	U	Proposition 29
	FIN	Reg	CFL	U	Proposition 30

Table 3: Decidability results of the problems $Q_{2,i}$ and $Q_{2,d}$, where L_2 and L_3 are over a non-unary alphabet.

Problem	L_2	L_3	T	Result	Proof
$Q^w_{2,i}$	Reg	Reg	REC	D	Corollary 5
	CFL	Reg	Reg	U	[1, 2], Remark 3
	Reg	CFL	Reg	U	Proposition 27
$Q^w_{2,d}$	Reg	CFL	REC	D	Corollary 6
	CFL	Reg	Reg	U	[1], Remark 4

Table 4: Decidability results of the problems $Q^w_{2,i}$ and $Q^w_{2,d}$, where L_2 and L_3 are over a non-unary alphabet. CSL and REC stand for the families of context-sensitive languages and of recursive languages, respectively.

Proof. Let L be an arbitrary CFL over $\{a, b\}$, and h be a homomorphism defined as $h(a) = 01$ and $h(b) = 10$. Then we define a trajectory set $T = 0h(L) \cup 0^* \cup 01^+$, and for $F_2 = \{a, b\}$ and $R_3 = \{\#a, \#b\}^+ \cup (\#ab)^*$, we claim the following:

$$h(L) = \{01, 10\}^* \text{ if and only if } X \rightarrow_T F_2 = R_3 \text{ has a solution.}$$

First of all, we note that $(\#ab)^* \rightarrow_{01^+} F_2 = \emptyset$. This is because deleting F_2 from a word according to 01^+ means deleting $2n$-th ($n \geq 1$) letter of the word, but only when all of them are in F_2, and this condition cannot be satisfied as exemplifed that the 4-th letter of $\#ab\#ab$ is $\#$.

If $h(L) = \{01, 10\}^*$, then we can easily check that $X = (\#ab)^*$ is a solution. Conversely, if the equation has a solution X, then X must be a subset of R_3 because T contains 0^*. If X contains a word in $\{\#a, \#b\}^+$, then by deleting F_2 from the word according to 01^+, we would obtain a word in $\#^+$, but this is not in R_3; hence, $X \subseteq (\#ab)^*$. And, this inclusion actually must be equal since we cannot obtain a word in $(\#ab)^*$ by deleting F_2 from another word in the set according to T. Let us define a mapping g as $g(01) = \#b$ and $g(10) = \#a$. If $h(L)$ does not contain t, then $g(t) \notin X \rightarrow_T F_2$. Thus, $h(L)$ must be $\{01, 10\}^*$. □

The results proved in this section are summarized in Tables 3 and 4.

8. Conclusion

In this paper, we introduced the notion of block insertion and deletion on trajectories for the study of properties of language operations under some parallel constraints. These operations are in fact proper generalizations of several known sequential and parallel binary operations in formal language theory such as catenation, sequential insertion, k-insertion, parallel insertion, quotient, sequential deletion, k-deletion, etc.

Mainly based on the representation lemmas, which relate these new operations to shuffle and deletion on trajectories, we examined the closure properties of the families of regular and context-free languages under these operations, and considered three types of language equation problems involving the operations.

In Section 7, the decidability of a solution to the language equation $X \leftarrow_T L_2 = L_3$ and its deletion variant was investigated, but the analogous problem on $L_1 \leftarrow_T X = L_3$ and $L_1 \rightarrow_T X = L_3$ remains open.

Acknowledgements

We wish to express our gratitude to the anonymous referees for their valuable constructive comments on the earlier version of this paper. This research was supported by The Natural Sciences and Engineering Research Council of Canada Discovery Grant R2824A01 and Canada Research Chair Award to L.K.

[1] L. Kari, On Insertion and Deletion in Formal Languages, Ph.D. thesis, University of Turku, Department of Mathematics, SF-20500 Turku, Finland, 1991.

[2] L. Kari, G. Thierrin, K-catenation and applications: k-prefix codes, Journal of Information and Optimization Sciences 16 (1995) 263–276.

[3] L. Kari, S. Seki, Schema for parallel insertion and deletion, in: Y. Gao, H. Lu, S. Seki, S. Yu (Eds.), Developments in Language Theory, volume 6224 of *LNCS*, pp. 267–278.

[4] M. Kudlek, A. Mateescu, On distributed catenation, Theor. Comput. Sci. 180 (1997) 341–352.

[5] M. Kudlek, A. Mateescu, On mix operation, in: G. Păun, A. Salomaa (Eds.), New Trends in Formal Languages, volume 1218 of *LNCS*, pp. 430–439.

[6] M. Domaratzki, Deletion along trajectories, Theor. Comput. Sci. 320 (2004) 293–313.

[7] A. Mateescu, G. Rozenberg, A. Salomaa, Shuffle on trajectories: Syntactic constraints, Theor. Comput. Sci. 197 (1998) 1–56.

[8] L. Kari, P. Sosík, Language Deletion on Trajectories, Technical Report 606, University of Western Ontario, 2003.

[9] L. Kari, On language equations with invertible operations, Theor. Comput. Sci. 132 (1994) 129–150.

[10] S. Yu, Regular languages, in: G. Rozenberg, A. Salomaa (Eds.), Handbook of Formal Languages, volume 1, Springer-Verlag, 1997, pp. 41–110.

[11] J.-M. Autebert, J. Berstel, L. Boasson, Context-free languages and push-down automata, in: G. Rozenberg, A. Salomaa (Eds.), Handbook of Formal Languages, volume 1, Springer-Verlag, 1997, pp. 111–174.

[12] A. Salomaa, Formal Languages, Academic Press, New York, 1973.

[13] J. E. Hopcroft, J. D. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, 1979.

[14] S. Ginsburg, The Mathematical Theory of Context-Free Languages, McGraw-Hill, New York, 1966.

[15] S. Ginsburg, E. H. Spanier, Quotients of context-free languages, Journal of the ACM 10 (1963) 487–492.

[16] A. Mateescu, A. Salomaa, Aspects of classical language theory, in: G. Rozenberg, A. Salomaa (Eds.), Handbook of Formal Languages, volume 1, Springer-Verlag, 1997, pp. 175–252.

[17] R. J. Parikh, On context-free languages, J. ACM 13 (1966) 570–581.

[18] L. Kari, P. Sosík, Aspects of shuffle and deletion on trajectories, Theor. Comput. Sci. 331 (2005) 47–61.

[19] M. Domaratzki, K. Salomaa, Decidability of trajectory-based equations, Theor. Comput. Sci. 345 (2005) 304–330.