# TRAJECTORY-BASED OPERATIONS

by

MICHAEL DOMARATZKI

A thesis submitted to the
School of Computing
in conformity with the requirements for
the degree of Doctor of Philosophy

Queen's University
Kingston, Ontario, Canada
August 2004

# Abstract

Shuffle on trajectories was introduced by Mateescu *et al.* [147] as a method of generalizing several studied operations on words, such as the shuffle, concatenation and insertion operations. This natural construction has received significant and varied attention in the literature. In this thesis, we consider several unexamined areas related to shuffle on trajectories.

We first examine the state complexity of the shuffle on trajectories. We find that the density of the set of trajectories is an appropriate measure of the complexity of the associated operation, since low density sets of trajectories yield less complex operations.

We introduce the operation of deletion along trajectories, which serves as an inverse to shuffle on trajectories. The operation is also of independent interest, and we examine its closure properties. The study of deletion along trajectories also leads to the study of language equations and systems of language equations with shuffle on trajectories.

The notion of shuffle on trajectories also has applications to the theory of codes. Each shuffle on trajectories operation defines a class of languages. Several of these language classes are important in the theory of codes, including the prefix-, suffix-, biprefix-codes and the hypercodes. We investigate these classes of languages, decidability questions, and related binary relations.

We conclude with results relating to iteration of shuffle and deletion on trajectories. We characterize the smallest language closed under shuffle on trajectories or deletion along trajectories, as well as generalize the notion of primitive words and primitive roots. Further examination of language equations are also possible with the iterated counterparts of shuffle and deletion along trajectories.

# Acknowledgments

I am grateful for everything my supervisor Dr. Salomaa has done for me during the course of our time together. His support has been outstanding and he has dedicated many hours to helping me improve this work, and to our collaborations.

I am grateful to the anonymous referees who have made suggestions for the journal and conference versions of the results presented here.

I would also like to thank the members of my examining committee, Dr. Ilie, Dr. Kashyap, Dr. Skillicorn and Dr. Tennent, for their comments and suggestions which have improved this thesis.

I am in debt to Alexander Okhotin for discussions, collaboration, his answering of my questions, making elegant figures, and not killing me, though I'm sure he would have preferred it.

The following people have also helped me through discussions on the topics in this thesis: Mark Daley, Masami Ito, Alexandru Mateescu, Jeff Shallit, and Petr Sosík.

kristy, amelia and jasper, for everything.

A proof is a proof. What kind of a proof? It's a proof.
A proof is proof, and when you have a good proof,
it's because it's proven.
–Jean Chretien (Sept. 5, 2002).

# Co-Authorship

The work in Chapter 4 is joint work with my supervisor, Dr. K. Salomaa [43, 46]. Portions of Chapter 7, most notably Sections 7.3, 7.5 and 7.7, are also joint work with Dr. Salomaa [44].

# Statement of Originality

I, Michael Domaratzki, certify that all results in this thesis are original, unless otherwise noted. Specifically, those results due to other authors which have appeared in the literature have been cited as necessary.

The work in this thesis has either appeared [43, 36, 37, 39, 40] or is to appear [44, 46] in the literature.

# Contents

# List of Figures

# Chapter 1

# Introduction

## 1.1 Formal Languages and Operations: Introduction and Motivation

Formal language theory, the study of abstract sets of words over a fixed alphabet of symbols, is one of the oldest research areas in the theory of computing. Despite its age, formal language theory also continues to attract new attention, especially its application to various fields, including the theory of codes, bio-informatics and many others.

Arguably, at the core of formal language theory are two central concepts: that generative devices, such as automata and grammars, can be used to define a class of languages, and that languages can be combined to form new languages using language operations. Mateescu and Salomaa state that "a major part of formal language theory can be viewed as the study of finitary devices for generating infinite languages" [148, p. 2]. The importance of language operations is a slightly more subtle point. However, we cannot underestimate their power: for instance, language operations themselves are at the heart of several fundamental language generation systems–for instance, regular expressions, L-systems and context-free grammars. Further, we can mention the study of concepts such as cones and abstract families of languages (AFLs), for which closure properties under language operations are the defining characteristics.

The two concepts of generative devices and language operations form the starting point for any

meaningful study of formal language theory. In particular, closure properties of classes of languages under various language operations give us insight into the both the power of the classes and the power of the operations. Closure properties also help us to meaningfully compare two classes of languages.

However, operations on languages also have crucial importance as emerging areas of research gain importance. In particular, the interpretation of strands of DNA as words in a formal language has been the subject of much research recently, in theoretical areas as well as areas fundamentally linked to the use of DNA as a natural computing method. The language operations under investigation are models of the manner in which DNA strands interact under various settings.

A fundamental development in the area of language equations was the introduction of the notion of shuffle on trajectories, defined by Mateescu *et al.* [147]. Shuffle on trajectories is a framework for defining word operations based on a set of *trajectories*, a language which specifies the way in which the corresponding operation behaves. Thus, shuffle on trajectories is a parameterized class of language operations: each choice of a set of trajectories yields a distinct language operation. The idea of replacing the study of word operations by the study of languages is a major innovation, and leads to very clear, unified results on the applicable language operations.

Operations modeled by shuffle on trajectories include concatenation, the most fundamental language-theoretic operation, the standard shuffle operation, which has a long history in the study of formal languages, as well as many variants of the shuffle operation, including insertion, literal shuffle, perfect shuffle and many others. Each of these operations has been the subject of study in the literature, both for specific applications – including the theory of codes, concurrency theory and other areas – as well as for research into the formal properties of these operations, and their effect on classes of languages.

This thesis examines the concept of trajectories in greater detail. In particular, we seek to unify several different areas of research in theoretical computer science by investigating each of them in the framework of shuffle on trajectories. By formalizing each of these areas, we provide new insight into their fundamental results. Results such as decidability problems and closure properties can be

examined in a uniform way, often leading to much simpler proofs.

One result of this research is a demonstration that the shuffle on trajectories formalism, introduced as a unification for word operations, is also important as a unification for more complex constructs. For instance, we define classes of languages related to codes using the shuffle on trajectories model. By doing so, we can model an entire class of languages by a single set of trajectories. This further re-enforces the value of the trajectory model.

In the following four sections, we present an informal description of the main areas of research in this thesis: descriptional complexity, the theory of codes, language equations and iterated language operations.

## 1.2 Descriptional Complexity of Languages

Descriptional complexity is the study of measures of complexity of languages and operations. It is a very broad area, and includes much work on varied classes of languages. We focus our work on descriptional complexity on the regular languages, a fundamental class of languages. For regular languages, the main focus of work on descriptional complexity is on *state complexity*. The (deterministic, nondeterministic) state complexity of a regular language is the minimal number of states in any (deterministic, nondeterministic) finite automata accepting that language. Given a binary language operation $\alpha$ under which the regular languages are closed, the (worst case) state complexity of $\alpha$ is a function $f$ such that for all regular languages $L_1, L_2$ accepted by automata of size $n_1, n_2$, there exists an automaton of size $f(n_1, n_2)$ accepting $\alpha(L_1, L_2)$. For instance, if we are interested in the union operation, it is known that an automaton of size $n_1 n_2$ can be found accepting the union of two languages which are accepted by automata of size $n_1$ and $n_2$.

Recently, research into the state complexity of regular languages has seen a great deal of activity. This work is motivated by the desire to have reliable estimates of the amount of memory required for automata when certain language operations are applied. This is crucial in applied areas where finite automata are used in practice, for instance, in pattern matching, natural language processing,

and other areas.

We examine the state complexity of shuffle on trajectories in Chapter 4. Being an infinite class of operations, shuffle on trajectories presents some unique challenges; previous results on the state complexity of operations have focused on single operations, rather than an entire family of operations. However, the fact that shuffle on trajectories is a class of language operations parameterized by a set of trajectories–which is itself a language–allows us to make interesting comparisons between the descriptional complexity of a set of trajectories and the state complexity of the resulting shuffle on trajectories operation. We find that another descriptional complexity measure of languages, namely the *density* of a language, gives us interesting insight into the relationship between the complexity of a set of trajectories and the complexity of the resulting language operation. In particular, we find that less dense sets of trajectories correspond to less complex operations, in terms of state complexity.

## 1.3 Codes and Trajectories

A code is a language which has strong decodability properties: given a sequence of words from a code which are concatenated together, there is only one way to recover the original code words from the concatenated sequence. Codes have many applications, including compression, error detection and security [97, pp. 511–512].

Subclasses of the class of codes, such as prefix codes and hypercodes, are often studied for interesting combinatorial and mathematical properties. This is sometimes know as the *theory of variable-length codes*. In Chapter 6, we present our contribution to this area, which we call $T$-codes. Intuitively, $T$-codes represent the natural extension of certain subclasses of codes, including prefix codes and hypercodes, which are defined via shuffle on trajectories. With the definition of $T$-codes, we can present results about classes of $T$-codes by arguing instead about the associated sets of trajectories $T$. This yields general results about properties of interest for $T$-codes.

We note that the idea of a general method for defining several code-like classes of languages

has received some attention in the literature. We briefly note some of this work in Section 6.1. However, we feel that the notion of a $T$-code, in using shuffle on trajectories, has the advantage of being general enough to capture several classes of codes studied in the literature on variable length codes, but at the same time, is specific enough to allow us to obtain interesting results. Specifically, decidability properties are often trivial in the framework of $T$-codes.

## 1.4 Language Equations

The study of language equations, that is, equations or systems of equations consisting of constant languages, language operations, and unknowns, and which are solved in terms of languages, is one of the oldest areas in the theory of computation. Many fundamental areas of computer science are intricately linked to the study of language equations.

As an example, we note the context-free languages, which are crucial in the design of programming languages and compilers. The theory of context-free grammars as a generative device is well developed. However, each context-free grammar is equivalent to a system of language equations, and many deep results in this area have been obtained (see, e.g., Autebert *et al.* [10]).

Our study of language equations will focus on equations whose operations are taken from the class of operations defined by shuffle on trajectories. In studying the decidability of the existence of solutions to such an equation, it will be useful to define the notion of an *inverse* to shuffle on trajectories. This inverse is known as *deletion along trajectories*. We first study the properties of deletion along trajectories in Chapter 5, and apply it to language equations in Chapter 7.

The inverse of a language operation, first defined by Kari [106], allows us to solve language equations much in the same way we solve equations such as $a + x = b$, where $a, b$ are integers and $x$ is unknown, and, as noted by Kari, our inverse works in a similar manner to how subtraction works as an inverse to addition. In particular, given our equation, we can recover the unknown value by applying the inverse operation to the known constants, much in the same way as in solving the equation $a + x = b$.

We further study language equations with two unknowns. In this case, the constructions involved are more complicated. However, we succeed in characterizing a large class of trajectories, including many studied operations, with which we are able to positively solve the decidability problem for language equations in two unknowns. For all of the equation forms we consider, we also obtain complementary undecidability results.

## 1.5 Iteration

In Chapter 8, we investigate iterated versions of trajectory-based operations. Our main motivation is the examination of languages which are closed under a fixed shuffle on trajectories or deletion along trajectories operation. There is a simple relationship between languages which are closed under shuffle on (resp., deletion along) trajectories and the iterated shuffle on (resp., deletion along) trajectories operation.

We also examine other applications of iterated trajectory-based operations. In particular, we examine the concept of primitivity, the property of a word not being able to be expressed as the power of another word, as it is related to shuffle on trajectories. The concept of primitivity, in relation to the concatenation operation, is a natural concept in formal language theory, an interesting intersection between the theory of formal languages and combinatorics, and also the source of one of the most well-known open problems in formal language theory–namely, whether or not the set of all primitive words is a context-free language. Primitivity was extended to both shuffle and insertion by Kari and Thierrin [118] and to a very general class of operations by Hsiao *et al.* [69]. In Chapter 8, we find that with a slightly more natural definition of iterated shuffle on trajectories, we find that the notion of primitivity can be examined without some of the assumptions made in the more general case of Hsiao *et al.* [69]. We also consider extensions to the very well-known results of Lyndon and Schützenberger via shuffle on trajectories. This allows us to further examine conditions of uniqueness and existence of primitive roots of words.

We also revisit language equations in Chapter 8 and characterize the solution to certain explicit

language equations involving shuffle on trajectories using its iterated counterpart. This is in contrast to the implicit language equations examined in Chapter 7, where we examined the question of the decidability of the existence of solutions. Our characterizations of the solutions of explicit language equations is a parallel of classic language equations which have been studied in the literature.

## 1.6 Organization

This thesis is organized as follows. Chapter 2 is devoted to preliminary definitions, and may be referred to as necessary by the reader. Chapter 3 examines related work on shuffle, iteration and shuffle on trajectories.

In Chapter 4, we discuss the complexity of the shuffle on trajectories operation in terms of state complexity, a much studied measure of the complexity of an operation which acts on regular languages.

Chapter 5 develops the concept of *deletion along trajectories*. We show that it serves as an inverse to shuffle on trajectories, in the sense defined by Kari [106]. We also examine the closure properties of deletion on trajectories.

In Chapter 6, we apply the framework of traditional code classes to shuffle on trajectories. This gives a generalization of many classical code classes. We examine many previously studied aspects of codes in this general setting.

Chapter 7 examines the solutions to equations involving shuffle and deletion along trajectories. Several questions for several forms of equations are examined, as well as certain forms of systems of equations involving shuffle on trajectories.

Finally, in Chapter 8, we consider the iteration of shuffle and deletion on trajectories. We are particularly interested in the relationship between iteration and the smallest language closed under shuffle on trajectories.

We conclude in Chapter 9 by examining some open problems raised in this thesis. We also discuss possible future research areas related to the trajectories model.

# Chapter 2

# Preliminary Definitions

We now review some notions that will be used in this thesis, as well as the main definition of shuffle on trajectories, which will be used throughout this thesis. Readers familiar with the concepts below should feel free to consult this chapter only as necessary.

## 2.1 Formal Language Theory

For additional background in formal languages and automata theory, please see Yu [201] or Hopcroft and Ullman [68]. Let $\Sigma$ be a finite set of symbols, called *letters*. The set $\Sigma$ is an *alphabet*. Then $\Sigma^*$ is the set of all finite sequences of letters from $\Sigma$, which are called *words*. The empty word $\epsilon$ is the empty sequence of letters. Given two words $w = w_1 w_2 \cdots w_n$ and $x = x_1 \cdots x_m$ where $x_i, w_j \in \Sigma$ for all $1 \leq i \leq m$ and $1 \leq j \leq n$, their concatenation $wx$ is the word $w_1 w_2 \cdots w_n x_1 x_2 \cdots x_m$. The *length* of a word $w = w_1 w_2 \cdots w_n \in \Sigma^*$, where $w_i \in \Sigma$, is $n$, and is denoted $|w|$. Note that $\epsilon$ is the unique word of length 0. A *language* $L$ is any subset of $\Sigma^*$. If $w \in \Sigma^*$, we will denote the language consisting only of $w$ by $w$ instead of $\{w\}$. For a language $L \subseteq \Sigma^*$, by $|L|$ we denote its cardinality as a set.

Let $L \subseteq \Sigma^*$ be a language. By $\overline{L}$, we mean $\Sigma^* - L$, the complement of $L$. Let $L_1, L_2$ be languages. By $L_1 L_2$ we mean the concatenation of $L_1$ and $L_2$, given by $L_1 L_2 = \{xy : x \in L_1, y \in$

$L_2\}$. If $L$ is a language and $n \geq 0$, then the set $L^n$ is defined recursively as follows: $L^0 = \{\epsilon\}$, $L^{n+1} = L^n L$ for all $n \geq 0$. We denote $L^* = \cup_{n\geq 0} L^n$ and $L^+ = \cup_{n\geq 1} L^n$. If $L_1, \ldots, L_k \subseteq \Sigma^*$ are languages, we use the notation $\prod_{i=1}^k L_i = L_1 L_2 \cdots L_k$. If $L$ is a language and $k$ is a natural number, then we denote $L^{\leq k} = \cup_{i=0}^k L^i$.

Given two languages $L_1, L_2 \subseteq \Sigma^*$, the *left quotient* of $L_2$ by $L_1$ is denoted $L_1 \setminus L_2$ and is given by

$$L_1 \setminus L_2 = \{x \in \Sigma^* \ : \ \exists y \in L_1 \text{ such that } yx \in L_2\}.$$

Similarly, the *right quotient* (or simply *quotient*, if there is no confusion) of $L_2$ by $L_1$ is denoted $L_2/L_1$ and is given by

$$L_2/L_1 = \{x \in \Sigma^* \ : \ \exists y \in L_1 \text{ such that } xy \in L_2\}.$$

The *shuffle* operation is defined as follows: if $x, y \in \Sigma^*$ are words, then the shuffle of $x$ and $y$, denoted $x \sqcup\!\sqcup y$ is defined by

$$x \sqcup\!\sqcup y = \{\prod_{i=1}^n x_i y_i \ : \ x = \prod_{i=1}^n x_i, \ y = \prod_{i=1}^n y_i; \ x_i, y_i \in \Sigma^* \ \forall 1 \leq i \leq n\}.$$

If $L_1, L_2$ are languages, then $L_1 \sqcup\!\sqcup L_2$ is given by $L_1 \sqcup\!\sqcup L_2 = \{x \sqcup\!\sqcup y \ : \ x \in L_1, y \in L_2\}$.

We denote by $\mathbb{N}$ the set of natural numbers: $\mathbb{N} = \{0, 1, 2, \ldots\}$. If we wish to refer to the positive numbers, we will use the notation $\mathbb{N}^+ = \{1, 2, \ldots, \}$. Let $I \subseteq \mathbb{N}$. If there exist $n_0, p \in \mathbb{N}$, $p > 0$, such that for all $x \geq n_0$, $x \in I \iff x + p \in I$, then we say that $I$ is *ultimately periodic* (u.p.). For $n, m \in \mathbb{N}$, we use the notation $m|n$ to denote that $m$ is a divisor of $n$, that is, there exists $k \in \mathbb{N}$ such that $n = km$.

Given a set $X$, we use the notation $2^X = \{Y \ : \ Y \subseteq X\}$. Given alphabets $\Sigma, \Delta$, a *morphism* is a function $h : \Sigma^* \to \Delta^*$ satisfying $h(xy) = h(x)h(y)$ for all $x, y \in \Sigma^*$. Given a morphism $h : \Sigma^* \to \Delta^*$ and a language $L \subseteq \Sigma^*$, then the image of $L$ under $h$ is given by $h(L) = \{h(x) \ : \ x \in L\}$, while if $L' \subseteq \Delta^*$, the inverse image of $L'$ under $h$ is defined by $h^{-1}(L') = \{x \in \Sigma^* \ : \ h(x) \in L'\}$. A *substitution* is a function $h : \Sigma^* \to 2^{\Delta^*}$ satisfying $h(xy) = h(x)h(y)$ for all $x, y \in \Sigma^*$. Given a substitution $h : \Sigma^* \to 2^{\Delta^*}$ and a language $L \subseteq \Sigma^*$, then the image of $L$ under $h$ is given

by $h(L) = \cup_{x \in L} h(x)$. We say that a substitution is *regular* if $h(a) \in$ REG for all $a \in \Sigma$ (see Section 2.2 below for the definitions of the regular languages and REG).

Given a word $w \in \Sigma^*$ and $a \in \Sigma$, $|w|_a$ is the number of occurrences of $a$ in $w$. For instance, if $w = abbaa$, then $|w|_a = 3$ and $|w|_b = 2$. If $w \in \Sigma^*$ is a word, $\text{alph}(w) = \{a \in \Sigma \ : \ |w|_a > 0\}$ . If $L \subseteq \Sigma^*$, $\text{alph}(L) = \cup_{w \in L} \text{alph}(w)$.

For an alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$ with a specified order $a_1 < a_2 < \cdots < a_n$, the *Parikh mapping* is given by $\Psi : \Sigma^* \to \mathbb{N}^n$, as follows:

$$\Psi(w) = (|w|_{a_i})_{i=1}^n.$$

It is extended to $\Psi : 2^{\Sigma^*} \to 2^{\mathbb{N}^n}$ as expected. For instance, if $\Sigma = \{a, b\}$ with $a < b$, and $x = abbaa$, then $\Psi(x) = (2, 3)$. If $L = \{a^n b^n a^n \ : \ n \geq 0\}$, then $\Psi(L) = \{(2n, n) \ : \ n \geq 0\}$. The inverse mapping is given by $\Psi^{-1} : 2^{\mathbb{N}^n} \to 2^{\Sigma^*}$ is given by $\Psi^{-1}(S) = \{u \in \Sigma^* \ : \ \Psi(u) \in S\}$ for all $S \subseteq \mathbb{N}^n$. A language $L \subseteq \Sigma^*$ is said to be *commutative* if $L = \Psi^{-1}(\Psi(L))$. Thus, $L$ is commutative if rearranging the letters from any word in $L$ always yields a word in $L$. For instance, the language $L = \{aab, aba, baa, ab, ba\}$ is commutative. For any language $L$, $com(L)$ is the commutative closure of $L$, i.e., $com(L) = \{v \in \Sigma^* \ : \ \exists u \in L \text{ such that } \Psi(u) = \Psi(v)\}$. For instance, $com(\{abc\}) = \{abc, acb, bac, bca, cab, cba\}$.

We say that a language $L \subseteq \Sigma^*$ is *bounded* if there exist $w_1, w_2, \ldots, w_k \in \Sigma^*$ such that $L \subseteq w_1^* w_2^* \cdots w_k^*$. If $L$ is not bounded we say that it is *unbounded*. The languages $L_1 = \{a^n b^{2n} c^n \ : \ n \geq 0\}$ and $L_2 = (ab)^* + (cd)^*$ are bounded, as $L_1 \subseteq a^* b^* c^*$ and $L_2 \subseteq (ab)^*(cd)^*$. The language $L_3 = \{a, b\}^*$ is known to be unbounded.

## 2.2 Regular Languages

We now describe finite automata and regular languages. A *deterministic finite automaton* (DFA) is a five-tuple $M = (Q, \Sigma, \delta, q_0, F)$ where $Q$ is a finite set of states, $\Sigma$ is an alphabet, $\delta : Q \times \Sigma \to Q$ is a transition function, $q_0 \in Q$ is a distinguished start state, and $F \subseteq Q$ is the set of final states. We

extend $\delta$ to $Q \times \Sigma^*$ in the usual way: if $q \in Q$ and $w \in \Sigma^*$, then define $\delta(q, w) = q$ if $w = \epsilon$ and

$$\delta(q, w) = \delta(\delta(q, w'), a)$$

if $w = w'a$ for some $w' \in \Sigma^*$ and $a \in \Sigma$.

A word $w \in \Sigma^*$ is *accepted by* $M$ if $\delta(q_0, w) \in F$. The *language accepted* by $M$, denoted $L(M)$, is the set of all words accepted by $M$. A language is called *regular* if it is accepted by some DFA.

A *nondeterministic finite automaton* (NFA) is a five-tuple $M = (Q, \Sigma, \delta, q_0, F)$ where $Q$, $\Sigma$, $q_0$ and $F$ are as in the deterministic case, while $\delta : Q \times (\Sigma \cup \{\epsilon\}) \to 2^Q$ is the nondeterministic transition function. Again, $\delta$ is extended to $Q \times \Sigma^*$ in the natural way. To define the action of $\delta$ formally, we require a few notions. First define a binary relation $R_\epsilon \subseteq Q^2$. The relation is given by $q_i R_\epsilon q_j$ if $q_j \in \delta(q_i, \epsilon)$. Let $R_\epsilon^*$ be the reflexive, transitive closure of $R_\epsilon$. Define $cl : Q \to 2^Q$ by

$$cl(q) = \{q' \ : q R_\epsilon^* q'\}.$$

Thus, $cl(q)$ is the set of all states that are reachable from $q$ by following some path of $\epsilon$-transitions in $M$. Further, let $cl(S) = \cup_{q \in S} cl(q)$ for all $S \subseteq Q$. We may now define $\delta$ as a function from $Q \times \Sigma^*$ to $2^Q$: if $q \in Q$ then $\delta(q, \epsilon) = cl(q)$ and for all $a \in \Sigma$ and $w \in \Sigma^*$,

$$\delta(q, wa) = cl\left(\bigcup_{q' \in \delta(q,w)} \delta(q', a)\right).$$

A word $w$ is accepted by $M$ if $\delta(q_0, w) \cap F \neq \emptyset$. It is known that the language accepted by an NFA is regular. We denote the class of regular languages by REG.

For a DFA or NFA $M$, we say that $M$ is *complete* if $\delta$ is a complete function, i.e., if $\delta(q, a)$ is defined for all $q \in Q$ and $a \in \Sigma$.

We can draw a DFA or NFA as a directed graph using the following conventions:

(a) states are drawn as vertices, labelled with their name;

(b) transitions are drawn as directed edges, labelled with the letter of the transition. Thus, if $\delta(q_1, a) = q_2$, there is a directed edge $(q_1, q_2)$ with label $a$;

(c) final states are indicated as vertices with double circles;

(d) the start state is indicated with an unlabelled arrow entering it.

For example, the DFA given in Figure 2.1 has start state 1, final state set $\{2\}$ and transitions $\delta(1, b) = \delta(2, b) = 1$ and $\delta(1, a) = \delta(2, a) = 2$.

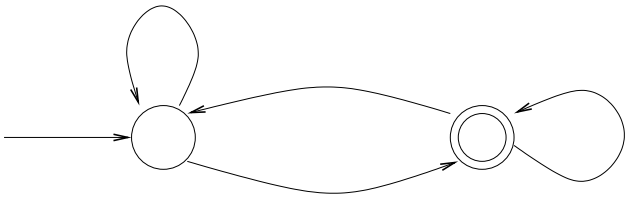

Figure 2.1: A DFA, illustrated.

We also introduce the *Myhill-Nerode congruence* on $\Sigma^*$. Given a language $L \subseteq \Sigma^*$, we denote the Myhill-Nerode congruence with respect to $L$ on $\Sigma^*$ by $\equiv_L$. Given $x, y \in \Sigma^*$, $x \equiv_L y$ if and only if, for all $z \in \Sigma^*$,

$$xz \in L \iff yz \in L.$$

We note that $\equiv_L$ is an equivalence relation and that a language $L$ is regular if and only if $\equiv_L$ has finite index [68, Thm. 3.9].

Finally, we define regular expressions. Let $\Sigma$ be an alphabet. A *regular expression* is a word over the alphabet $\{\emptyset, \epsilon, (, ), *, +\} \cup \Sigma$ defined as follows:

(a) the following are regular expressions: $\epsilon$, $\emptyset$ and $a$ for all $a \in \Sigma$;

(b) if $r_1, r_2$ are regular expressions, so are $(r_1 r_2)$ and $(r_1 + r_2)$;

(c) if $r_1$ is a regular expression, so is $(r_1^*)$.

Given a regular expression $r$, it defines a language $L(r)$ as follows:

(a) $L(\epsilon) = \{\epsilon\}$, $L(\emptyset) = \emptyset$ and $L(a) = \{a\}$;

(b) $L((r_1 + r_2)) = L(r_1) \cup L_(r_2)$;

(c) $L((r_1 r_2)) = L(r_1)L(r_2)$;

(d) $L((r_1^*)) = L(r_1)^*$.

Parentheses in regular expressions may be omitted, subject to the following precedence rules: $*$ has the highest precedence, then concatenation, then $+$. It is known that regular expressions define exactly the regular languages.

## 2.3 Grammars

We now turn to three classes of languages defined by grammars: *context-free languages* (CFLs), *linear context-free languages* (LCFLs) and *context-sensitive languages* (CSLs). These classes are denoted CF, LCF, and CS, respectively. While we describe them formally, it will suffice to note the following well-known inclusions, all of which are proper:

$$\text{REG} \subsetneq \text{LCF} \subsetneq \text{CF} \subsetneq \text{CS}. \tag{2.1}$$

For each of CF, LCF, CS, a grammar is a four-tuple $G = (V, \Sigma, P, S)$, where $V$ is a finite set of non-terminals, $\Sigma$ is a finite alphabet, $P \subseteq ((V \cup \Sigma)^* V (V \cup \Sigma)^*) \times (V \cup \Sigma)^*$ is a finite set of productions, and $S \in V$ is a distinguished start non-terminal. If $(\alpha, \beta) \in P$, we usually denote this by $\alpha \to \beta$.

Such a grammar is a *context-free grammar* (CFG) if $P \subseteq V \times (V \cup \Sigma)^*$, a *linear context-free grammar* (LCFG) if $P \subseteq V \times (\Sigma^* V \Sigma^* \cup \Sigma^*)$, and a *context-sensitive grammar* (CSG) if $(\alpha, \beta) \in P$ implies $\alpha = \eta A \zeta$ and $\beta = \eta \gamma \zeta$ for some $\eta, \zeta, \gamma \in (V \cup \Sigma)^*$, with $\gamma \neq \epsilon$ and $A \in V$.

IF $G = (V, \Sigma, P, S)$ is a grammar (CFG, LCFG or CSG), then given two words $\alpha, \beta \in (V \cup \Sigma)^*$, we denote $\alpha \Rightarrow_G \beta$ if $\alpha = \alpha_1 \alpha_2 \alpha_3$, $\beta = \alpha_1 \beta_2 \alpha_3$ for $\alpha_1, \alpha_2, \alpha_3, \beta_2 \in (V \cup \Sigma)^*$ and $\alpha_2 \to \beta_2 \in P$. Let $\Rightarrow_G^*$ denote the reflexive, transitive closure of $\Rightarrow_G$. Then the language generated by a grammar $G = (V, \Sigma, P, S)$ is given by

$$L(G) = \{x \in \Sigma^* \ : \ S \Rightarrow_G^* x\}.$$

If a language is generated by a CFG (resp., LCFG, CSG), then it is a CFL (resp., LCFL, CSL).

## 2.4 Complexity Theory

We now consider Turing machines. Our presentation is largely based on Hopcroft and Ullman [68, Ch. 7]. A Turing machine (TM) is a seven-tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ where $Q$ is a finite set of states, $\Gamma$ is a finite *tape alphabet*, $B \in \Gamma$ is the blank symbol, $\Sigma \subseteq \Gamma - B$ is the *input alphabet*, $\delta$ is the transition function given by $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$, $q_0 \in Q$ is the start state, and $F \subseteq Q$ is the set of final states. This model of TM is deterministic. A nondeterministic variant is also possible.

Given a TM $M$, an instantaneous description (ID) of $M$ is a word $w_1 q w_2 \in \Gamma^* Q \Gamma^*$. We interpret the ID as meaning that the TM is in state $q$ with tape contents $w_1 w_2$ and the head currently positioned on the first character of $w_2$. We define a relation $\Rightarrow_M$ on the set of IDs as follows: given IDs $w_1 q_1 w_2, u_1 q_2 u_2$,

$$w_1 q_1 w_2 \Rightarrow_M u_1 q_2 u_2 \iff \begin{cases} u_1 = w_1\gamma\,,\ w_2 = \beta u_2, & \text{and } \delta(q_1, \beta) = (q_2, \gamma\,, R), \\ \text{or} \quad w_1 = u_1\gamma\,,\ w_2 = \alpha w_2', u_2 = \gamma\beta w_2' & \text{and } \delta(q_1, \alpha) = (q_2, \beta, L). \end{cases}$$

Let $\Rightarrow_M^*$ be the transitive and reflexive closure of $\Rightarrow_M$. The language accepted by $M$, denoted $L(M)$ is

$$L(M) = \{w \in \Sigma^* \ :\ q_0 w \Rightarrow_M^* \alpha_1 q \alpha_2 \text{ such that } q \in F, \alpha_1, \alpha_2 \in \Gamma^*\}.$$

Given a language $L$, if there exists a TM $M$ such that $L = L(M)$, we say that $L$ is *recursively enumerable* (r.e.). We denote the set of r.e. languages by RE.

Say that a TM $M$ *halts* on input $x$ if it eventually reaches an ID which has no next move, i.e., the current ID has no successors under $\Rightarrow_M$. We may assume without loss of generality that when a word is accepted by $M$, $M$ halts. However, if an input word is not accepted, we note that $M$ may not halt.

If $L$ is accepted by a TM $M$ such that $M$ halts on all inputs, we say that $L$ is *recursive*. The set of all recursive languages is denoted by REC. The inclusions (2.1) may be extended as follows (again, the inclusions below are proper):

$$\text{CS} \subsetneq \text{REC} \subsetneq \text{RE}.$$

Nondeterminism does not affect the classes REC and RE.

We now refine the class of languages computed by a Turing machine. Given a TM $M$, we say that $M$ uses space $c$ on input $w$ if $M$ scans at most $c$ tape cells during the computation on $w$, i.e., $\max\{|v| \ : \ q_0 w \Rightarrow_M^* vqu, w, v, u \in \Gamma^*\} \leq c$. Let $n$ be the length of the input to a TM (i.e., the length of the word $w$ such that $q_0 w$ is the initial ID of the TM). If a deterministic (resp., nondeterministic) TM uses at most $O(s(n))$ space on any input of length $n$, then we say that the language $L(M)$ is in DSPACE$(s)$ (resp., NSPACE$(s)$). It is known that CS $=$ NSPACE$(n)$, i.e., the context-sensitive languages correspond exactly to the class of languages accepted in linear space by a nondeterministic TM. We similarly define the classes DTIME$(f)$ and NTIME$(f)$.

The following classes are also useful to us:

$$\begin{aligned} \text{P} &= \bigcup_{k \geq 1} \text{DTIME}(n^k); \\ \text{NP} &= \bigcup_{k \geq 1} \text{NTIME}(n^k). \end{aligned}$$

Given a function $g : \Sigma^* \to \Sigma^*$, we say that $g$ is *computable* in DSPACE$(s)$ (resp., NSPACE$(s)$, DTIME$(f)$, NTIME$(f)$) if there exists a TM $M$ operating in DSPACE$(s)$ (resp., NSPACE$(s)$, DTIME$(f)$, NTIME$(f)$) such that for all $w \in \Sigma^*$, $q_0 w \Rightarrow_M^* u_1 q u_2$ with $q \in F$ and $u_1 u_2 = g(w)$. Further, $g(w)$ is the only such tape contents which results from halting on input $w$.

A function $f : \mathbb{N} \to \mathbb{N}$ is said to be *space-constructible* if there exists a TM $M$ such that $L(M) \in$ NSPACE$(f)$ and, for all $n \geq 0$, there exists some $x \in \Sigma^n$ such that $M$ uses exactly $f(|x|)$ space on input $x$.

Given two languages $L', L$, we say that $L'$ is *reducible* to $L$ if there exists a function $g : \Sigma^* \to \Sigma^*$ such that $x \in L'$ if and only if $g(x) \in L$. If $g$ is computable in DSPACE(log), then we say that $L'$ is log-space reducible to $L$.

Let $\mathcal{C}$ be a class of languages. The language $L$ is *$\mathcal{C}$-hard* if $L'$ is reducible to $L$ for all $L' \in \mathcal{C}$. The language $L$ is *$\mathcal{C}$-complete* if $L \in \mathcal{C}$ and $L$ is $\mathcal{C}$-hard. For both P and NP, completeness can be defined with respect to log-space reductions.

## 2.5 Decidability

In this section, we briefly describe the concept of decidability and undecidability, and recall the Post correspondence problem (PCP) and several meta-theorems for proving undecidability.

We will often consider problems when discussing undecidability. A *problem* $P$ is simply a predicate, in the following sense: "given an input $x$, does $P(x)$ hold?" For example, if $P$ is the problem of primality, and $x$ is an integer (encoded over our alphabet $\Sigma$), $P(x)$ holds if and only if $x$ is a prime number. Thus, if $x$ is suitably encoded over an alphabet $\Sigma$, $P$ naturally defines a language over $\Sigma^*$, namely, those $x$ such that $P(x)$ holds. Let $L_P$ be this corresponding language (we sometimes simply identify $P$ with the corresponding language, and do not use the notation $L_P$). We say that a problem $P$ is *decidable* if $L_P \in \text{REC}$. Otherwise, $P$ is said to be *undecidable*.

The Post correspondence problem (PCP) is a basic undecidable problem which is often useful in many language-theoretic situations. An *instance* of PCP is

$$M = (u_1, u_2, \ldots, u_n; v_1, v_2, \ldots, v_n)$$

where $n \geq 1$ and $u_i, v_i \in \Sigma^*$ for $1 \leq i \leq n$. A *solution* to $M$ is a list $i_1, i_2, \ldots, i_m$ such that $m \geq 1$, $1 \leq i_j \leq n$ for all $1 \leq j \leq m$ and

$$\prod_{j=1}^{m} u_{i_j} = \prod_{j=1}^{m} v_{i_j}.$$

The following result states that finding solutions to a PCP instance is undecidable [68, Thm. 8.8]:

**Theorem 2.5.1** *Given an alphabet $\Sigma$ and a PCP instance $M = (u_1, \ldots, u_n; v_1, \ldots, v_n)$, where $n \geq 1$ $u_i, v_i \in \Sigma^*$ for $1 \leq i \leq n$, it is undecidable whether there is a solution for $M$.*

We will also use the following undecidability result:

**Theorem 2.5.2** *Let $\Sigma$ be an alphabet with $|\Sigma| \geq 2$ and $G = (V, \Sigma, P, S)$ be an LCFG. It is undecidable whether $L(G) = \Sigma^*$.*

In what follows, a *predicate* on $2^{\Sigma^*}$ is simply a class of languages satisfying some property. By a predicate on a class of languages $\mathcal{C}$, we simply mean the restriction of the predicate from $2^{\Sigma^*}$

to $\mathcal{C}$. If $P$ is a predicate and a language $L \subseteq \Sigma^*$ satisfies $P$, we will denote this fact by $P(L)$. For example, if $P_R$ is the predicate defined by the regular languages, then $P_R(L)$ implies that $L$ is regular. A predicate $P$ on $\mathcal{C}$ is *non-trivial* if $P \notin \{\emptyset, \mathcal{C}\}$.

Meta-theorems are powerful tools for proving undecidability. In this thesis, we will appeal to the following meta-theorem, due to Hunt and Rosenkrantz [70, Thm. 2.10], which will allow us to prove undecidability results for LCF.

**Theorem 2.5.3** *Let $P$ be a predicate on* LCF *over $\Sigma^*$ such that $P(\Sigma^*)$ holds and either of the sets*

$$\{L' \ : \ L' = x \setminus L, x \in \Sigma^+, L \in \text{LCF } \textit{and } P(L)\}$$

*or*

$$\{L' \ : \ L' = L/x, x \in \Sigma^+, L \in \text{LCF } \textit{and } P(L)\}$$

*is a proper subset of* LCF*. Then given an LCFG $G$, it is undecidable whether $P(L(G))$ holds.*

The following is a corollary of Theorem 2.5.3. It is also a particular case of Greibach's Theorem (see, e.g., Hopcroft and Ullman [68, Thm. 8.14]).

**Corollary 2.5.4** *Let $P$ be a non-trivial predicate on* LCF *over $\Sigma^*$ such that $P(\Sigma^*)$ holds and $P$ is preserved under quotient. Then given an LCFG $G$, it is undecidable whether $P(L(G))$ holds.*

## 2.6 Families of Languages

We will require some definitions and notations relating to classes of languages. Let $\mathcal{C}_1, \mathcal{C}_2$ be classes of languages. Then let

$$\begin{aligned} \mathcal{C}_1 \wedge \mathcal{C}_2 &= \{L_1 \cap L_2 \ : \ L_i \in \mathcal{C}_i, i = 1, 2\}; \\ \text{co-}\mathcal{C}_1 &= \{\overline{L} \ : \ L \in \mathcal{C}_1\}. \end{aligned}$$

Our notation $\wedge$ comes from Ginsburg [51], and should not be confused with $\mathcal{C}_1 \cap \mathcal{C}_2 = \{L \ : \ L \in \mathcal{C}_1 \text{ and } L \in \mathcal{C}_2\}$.

Recall that a *cone* (or *full trio*) is a class of languages closed under morphism, inverse morphism and intersection with regular languages [148, Sect. 3].

We will also use the notion of *immune* languages. Let $\mathcal{C}$ be a class of languages. A language $L$ is said to be $\mathcal{C}$-*immune* if $L$ is infinite and for all infinite languages $L' \subseteq L$, $L' \notin \mathcal{C}$. Immunity was introduced for classes of languages by Flajolet and Steyaert [49]; we also refer the interested reader to Balcázar *et al.* [14] for an introduction to immunity as it relates to complexity theory.

## 2.7 Shuffle on Trajectories

The shuffle on trajectories operation is a method for specifying the ways in which two input words may be merged, while preserving the order of symbols in each word, to form a result. Each trajectory $t \in \{0,1\}^*$ with $|t|_0 = n$ and $|t|_1 = m$ specifies the manner in which we can form the shuffle on trajectories of two words of length $n$ (as the left input word) and $m$ (as the right input word). The word resulting from the shuffle along $t$ will have a letter from the left input word in position $i$ if the $i$-th symbol of $t$ is 0, and a letter from the right input word in position $i$ if the $i$-th symbol of $t$ is 1.

We now give the definition of shuffle on trajectories, originally due to Mateescu *et al.* [147]. Shuffle on trajectories is defined by first defining the shuffle of two words $x$ and $y$ over an alphabet $\Sigma$ on a trajectory $t$, a word over $\{0,1\}$. We denote the shuffle of $x$ and $y$ on trajectory $t$ by $x \sqcup\!\sqcup_t y$.

If $x = ax'$, $y = by'$ (with $a, b \in \Sigma$) and $t = et'$ (with $e \in \{0,1\}$), then

$$x \sqcup\!\sqcup_{et'} y = \begin{cases} a(x' \sqcup\!\sqcup_{t'} by') & \text{if } e = 0; \\ b(ax' \sqcup\!\sqcup_{t'} y') & \text{if } e = 1. \end{cases}$$

If $x = ax'$ ($a \in \Sigma$), $y = \epsilon$ and $t = et'$ ($e \in \{0,1\}$), then

$$x \sqcup\!\sqcup_{et'} \epsilon = \begin{cases} a(x' \sqcup\!\sqcup_{t'} \epsilon) & \text{if } e = 0; \\ \emptyset & \text{otherwise.} \end{cases}$$

If $x = \epsilon$, $y = by'$ ($b \in \Sigma$) and $t = et'$ ($e \in \{0,1\}$), then

$$\epsilon \sqcup\!\sqcup_{et'} y = \begin{cases} b(\epsilon \sqcup\!\sqcup_{t'} y') & \text{if } e = 1; \\ \emptyset & \text{otherwise.} \end{cases}$$

We let $x \sqcup\!\sqcup_\epsilon y = \emptyset$ if $\{x, y\} \neq \{\epsilon\}$. Finally, if $x = y = \epsilon$, then $\epsilon \sqcup\!\sqcup_t \epsilon = \epsilon$ if $t = \epsilon$ and $\emptyset$ otherwise.

It is not difficult to see that if $t = \prod_{i=1}^{n} 0^{j_i} 1^{k_i}$ for some $n \geq 0$ and $j_i, k_i \geq 0$ for all $1 \leq i \leq n$, then we have that

$$x \sqcup\!\sqcup_t y = \{\prod_{i=1}^{n} x_i y_i \ : \ x = \prod_{i=1}^{n} x_i, y = \prod_{i=1}^{n} y_i, \text{ with } |x_i| = j_i, |y_i| = k_i \text{ for all } 1 \leq i \leq n\}$$

if $|x| = |t|_0$ and $|y| = |t|_1$ and $x \sqcup\!\sqcup_t y = \emptyset$ if $|x| \neq |t|_0$ or $|y| \neq |t|_1$.

We extend shuffle on trajectories to *sets* $T \subseteq \{0, 1\}^*$ *of trajectories* as follows:

$$x \sqcup\!\sqcup_T y = \bigcup_{t \in T} x \sqcup\!\sqcup_t y.$$

Further, for $L_1, L_2 \subseteq \Sigma^*$, we define

$$L_1 \sqcup\!\sqcup_T L_2 = \bigcup_{\substack{x \in L_1 \\ y \in L_2}} x \sqcup\!\sqcup_T y.$$

### 2.7.1 Examples

We now consider some examples of shuffle on trajectories. Let $x = abc$ and $y = de$. If $t = 00011$, then $x \sqcup\!\sqcup_t y = abcde$. If $t = 00111$, then $x \sqcup\!\sqcup_t y = \emptyset$. Thus, we can see that if $T = 0^*1^*$, we have that

$$L_1 \sqcup\!\sqcup_T L_2 = L_1 L_2,$$

i.e., $T = 0^*1^*$ gives the concatenation operation.

If $x = abc$, $y = de$, and $t = 01001$, then $x \sqcup\!\sqcup_t y = adbce$. If $t = 01010$, then $x \sqcup\!\sqcup_t y = adbec$. Thus, we have that if $T = (0 + 1)^*$, then

$$L_1 \sqcup\!\sqcup_T L_2 = L_1 \sqcup\!\sqcup L_2,$$

i.e., $T = \{0, 1\}^*$ gives the shuffle operation. This is the least restrictive set of trajectories.

If $T = 0^*1^*0^*$, then $\sqcup\!\sqcup_T$ is the insertion operation $\leftarrow$ (see, e.g, Kari [106]) which is defined by $x \leftarrow y = \{x_1 y x_2 \ : \ x_1, x_2 \in \Sigma^*, x_1 x_2 = x\}$ for all $x, y \in \Sigma^*$. Some other examples of operations defined by shuffle on trajectories are given in Figure 2.2 in the following section.

### 2.7.2 Algebraic Properties

We will require some algebraic properties of shuffle on trajectories throughout this thesis. These properties have been studied by Mateescu *et al.* [147].

Let $T \subseteq \{0,1\}^*$. We say that $T$ is *complete* if, for all $x, y \in \Sigma^*$, $x \sqcup\!\sqcup_T y \neq \emptyset$, i.e., there exists some $z \in \Sigma^*$ such that $z \in x \sqcup\!\sqcup_T y$. The set $T$ is said to be *deterministic* if, for all $x, y \in \Sigma^*$, $|x \sqcup\!\sqcup_T y| \leq 1$. Say that $T$ is *associative* (resp., *commutative*) if the corresponding operation $\sqcup\!\sqcup_T$ is associative (resp., commutative), i.e., $x \sqcup\!\sqcup_T (y \sqcup\!\sqcup_T z) = (x \sqcup\!\sqcup_T y) \sqcup\!\sqcup_T z$ for all $x, y, z \in \Sigma^*$ (resp., $x \sqcup\!\sqcup_T y = y \sqcup\!\sqcup_T x$ for all $x, y \in \Sigma^*$). For characterizations and decidability of these properties, we refer the reader to Mateescu *et al.* [147, Sect. 4]. We summarize several examples of shuffle on trajectories and their algebraic properties in Figure 2.2.

| Name | $T$ | Complete? | Determ.? | Assoc.? | Commutative? |
|---|---|---|---|---|---|
| Concatenation | $0^*1^*$ | $\surd$ | $\surd$ | $\surd$ | $\times$ |
| Insertion | $0^*1^*0^*$ | $\surd$ | $\times$ | $\times$ | $\times$ |
| Shuffle | $(0+1)^*$ | $\surd$ | $\times$ | $\surd$ | $\surd$ |
| Perfect Shuffle | $(01)^*$ | $\times$ | $\surd$ | $\times$ | $\times$ |
| Balanced Insertion | $\{0^i 1^{2j} 0^i \ : i, j \geq 0\}$ | $\times$ | $\surd$ | $\surd$ | $\times$ |
| Bi-catenation | $0^*1^* + 1^*0^*$ | $\surd$ | $\times$ | $\times$ | $\surd$ |

Figure 2.2: Some examples of shuffle on trajectories and their algebraic properties.

# Chapter 3

# Related Work

## 3.1 Introduction

In this chapter, we review the literature relevant to this thesis. Our focus is on word operations, such as shuffle, insertion, and quotient, which are specific instances of the formalisms we present in this thesis. We focus primarily on research which is either of theoretical interest, or relates directly to the topics we investigate later in the thesis.

## 3.2 Shuffle

Shuffle is one of the most studied operations on formal languages which is not among the defining operations of regular expressions. Ginsburg and Spanier introduce a definition of shuffle in 1965 [53] in their study of generalized sequential machines. This is the first reference to shuffle as an operation on languages we have been able to find. The natural application of shuffle as a model for interleaving processes yielded much research into shuffle and related operations. In an early paper on shuffle, Ogden *et al.* show that there exist DCFLs $L_1$, $L_2$ such that $L_1 \sqcup\!\sqcup L_2$ is NP-complete [156]. Hausler and Zeiger [63] give an interesting representation theorem for r.e. languages using the homomorphic image of the intersection of a regular language and the shuffle of two fixed Dyck

languages.

We now consider three specific areas of research on arbitrary shuffle: iterated shuffle, shuffle decompositions and grammar formalisms involving shuffle.

### 3.2.1 Iteration

The iteration of shuffle has received much attention in the literature over the last thirty years. This operation is defined much in the same way as Kleene closure: given a language $L$ its shuffle closure is defined as

$$(\sqcup\!\sqcup)^*(L) = \bigcup_{i \geq 0} (\sqcup\!\sqcup)^i(L),$$

where $(\sqcup\!\sqcup)^0(L) = \{\epsilon\}$, $(\sqcup\!\sqcup)^{i+1}(L) = (\sqcup\!\sqcup)^i(L) \sqcup\!\sqcup L$ for all $i \geq 0$. Several notations are used in the literature for denoting $(\sqcup\!\sqcup)^*(L)$, including $L^{\otimes}$ and $L^{\dagger}$.

Much of the interest in shuffle closure comes from the theory of concurrency and formal software engineering research communities. For example, Shaw, in describing the shuffle closure operation in the context of *flow expressions*, notes that shuffle closure is a "concurrent analogue of [the Kleene closure operation]", which "is useful where there may be a variable number of interleaves of some flow [of control], for example in describing systems in which processes or resources may be dynamically created and destroyed. [182, p. 243]". Riddle also performed early research into software engineering using the shuffle closure operation [170]. While the shuffle closure operation is fundamental to this research, various authors (including both Shaw and Riddle) also incorporate synchronization methods for research into software engineering. More recently, Igarashi and Kobayashi [71] cite shuffle expressions as a valid manner in specifying trace sets for use in their formal analysis of resource usage.

Other research into iterated shuffle has proceeded from a purely theoretical standpoint. Warmuth and Haussler [198] show the following elegant result:

**Theorem 3.2.1** *Let* $\Sigma = \{a, b, c\}$. *Given words* $u, v \in \Sigma^*$, *it is* NP*-complete to determine whether* $u \in (\sqcup\!\sqcup)^*(v)$.

Imreh *et al.* [74] have written on the shuffle closure of commutative regular languages. In particular, they give two characterizations of when the shuffle closure of a commutative regular language is again regular.

### 3.2.2 Decomposition

The shuffle decomposition problem has received much attention recently. For shuffle on trajectories, the problem was introduced by Mateescu *et al.* [147], who asked, given a language $L$, is it possible to write $L = L_1 \sqcup\!\sqcup_T L_2$ for some $L_1, L_2, T$, where the complexity of $L_1, L_2, T$ are "somehow smaller [147, p. 38]" than the complexity of $L$ (e.g., each are situated lower in the Chomsky hierarchy than $L$). They called such a simpler expression for $L$ a *parallelization* of $L$, and noted that some languages, such as the non-context free languages $L = \{ww : w \in \Sigma^*\}$ and $L = \{a^n b^{n^2} : n \geq 0\}$ do not have parallelizations into context-free languages.

Câmpeanu *et al.* [21] have studied the problem of deciding whether a regular language $R$ has a parallelization $R = L_1 \sqcup\!\sqcup L_2$, i.e., the case when $T = (0 + 1)^*$. If such a parallelization exists, and $L_1, L_2 \neq \{\epsilon\}$, such an expression is called a (non-trivial) *shuffle decomposition*. Despite much effort, Câmpeanu *et al.* [21] were not able to resolve whether it is decidable, given a regular language $R$, whether $R$ has a non-trivial shuffle decomposition. For certain subclasses of regular languages, Câmpeanu *et al.* were able to positively decide whether a language from that subclass has a non-trivial shuffle decomposition.

Ito [75] has also examined the shuffle decomposition problem for regular languages. Let $\mathcal{I}(n, \Sigma)$ be the class of all regular languages over $\Sigma$ which are accepted by some DFA with at most $n$ states. The main result of Ito [75] is the following:

**Theorem 3.2.2** *Given a regular language $R \subseteq \Sigma^*$ and $n \in \mathbb{N}$, it is decidable whether there exist $L_1, L_2$ with $L_1 \in \mathcal{I}(n, \Sigma)$ and $L_2 \neq \{\epsilon\}$ such that $R = L_1 \sqcup\!\sqcup L_2$.*

The general problem of determining whether a regular language has a non-trivial shuffle decomposition is still open. We will examine the shuffle decomposition problem with respect to a set of

trajectories $T$ (i.e., deciding whether there exists $L_1, L_2$ such that $R = L_1 \sqcup\!\sqcup_T L_2$) in Chapter 7.

Iwama [84] has considered shuffle decomposition in a different sense. Say that languages $(L_1, \ldots, L_n)$ are *uniquely shuffle-decomposable* if each word in $z \in L_1 \sqcup\!\sqcup L_2 \sqcup\!\sqcup \cdots \sqcup\!\sqcup L_n$ can be represented uniquely as $z \in x_1 \sqcup\!\sqcup x_2 \sqcup\!\sqcup \cdots \sqcup\!\sqcup x_n$ with $x_i \in L_i$ for $1 \leq i \leq n$. Given regular languages $(L_1, \ldots, L_n)$, Iwama gives an algorithm to decide whether they are uniquely shuffle-decomposable.

### 3.2.3 Grammar Formalisms

In the theory of concurrency and software engineering, several models have been proposed which adjoin grammars and regular expressions with shuffle and iterated shuffle.

Several papers have considered the class of languages defined by regular expressions adjoined with shuffle and iterated shuffle. This class of languages, under various names, has been extensively studied, and we can only give a list of the work done so far, including that of Gisher [55], Araki *et al.* [8], Araki and Tokura [7], Jędrezejowicz [87, 88, 89, 90, 91, 92], Janzten [86], Jędrzejowicz and Szipietowski [93], and many others.

Guo *et al.* [56] have introduced *synchronization expressions*, which are regular expressions augmented with a restricted form of shuffle. Synchronization expressions were developed as a model for specifying the synchronization which occurs between processes in a parallel system. The notion of synchronization expressions has been further examined by Salomaa and Yu [177, 178] and Clerbout *et al.* [26, 27, 172].

The concept of shuffle-star height (analogous to the usual (Kleene-) star height) has been implicitly studied by Gisher [55] and subsequently by Jędrezejowicz [88, 89, 90], where it was first shown that there exist languages of shuffle-star height $n$ for all $n \geq 0$, over an alphabet of size $3n$ [89]. Jędrezejowicz [90] later extended this to show that there exist languages of shuffle-star height $n$ for all $n \geq 0$ over an alphabet of size seven. Jędrezejowicz leaves open the problem of whether the alphabet size seven is optimal, as well as the problem of characterizing all morphisms which preserve shuffle-star height [90, Rem. 5.2].

Araki and Tokura [7] investigate decision problems for regular expressions augmented with shuffle and shuffle-closure, and show, e.g., that the membership and emptiness problems for these expressions are decidable, while their equivalence and containment problems are undecidable. Further decidability problems are studied by Jędrezojowicz [91].

Shoudai [183] describes a P-complete language using shuffle expressions.

## 3.3 Insertion and Deletion Operations

We now consider results on insertion and deletion operations. The insertion operations we consider are those modelled by shuffle on trajectories, and thus have special relevance to the work in this thesis. We do not survey research on insertion operations which are not modelled by shuffle on trajectories, e.g., the work of Kari [107] on controlled insertion and deletion. The deletion operations we will survey are primarily those which can be modelled by deletion on trajectories, which we introduce in Chapter 5.

### 3.3.1 Insertion Operations

Besides shuffle and concatenation, the (sequential) insertion operation is perhaps the most natural operation which inserts all of the symbols of one word into another. It is defined as follows:

$$u \leftarrow v = \{u_1 v u_2 \ : \ u_1 u_2 = u\}.$$

We noted in Section 2.7.1 that insertion is a particular case of shuffle on trajectories. Kari has studied the properties of insertion [104, 106], including the solutions of language equations involving insertion. We generalize these results in Chapter 7.

The bi-catenation operation is defined as follows: $u \odot v = \{uv, vu\}$. The bi-catenation operation was defined by Shyr and Yu [187], and further studied by Hsiao *et al.* as a particular case of their general study of binary word operations [69]. Shyr and Yu are motivated by considering bi-catenation as a restriction of shuffle, and related code-theoretic properties.

Kari and Thierrin [114, 115] have defined the operation of $k$-insertion as follows: given $k \geq 0$, the $k$-insertion of $u, v \in \Sigma^*$ is defined as

$$u \leftarrow^k v = \{u_1 v u_2 \ : \ u = u_1 u_2, |u_2| \leq k\}.$$

We note that $k$-insertion can be modelled by shuffle on trajectories, and also that

$$u \leftarrow v = \bigcup_{k \geq 0} u \leftarrow^k v.$$

The $k$-insertion operation is motivated by Kari and Thierrin as follows:

> Even though insertion generalizes catenation, catenation cannot be obtained as a particular case of it, as we cannot force the insertion to take place at the end of the word. The $k$-insertion provides the control needed to overcome this drawback. The $k$-insertion is thus more nondeterministic than catenation, but more restrictive than insertion. [115, p. 479]

Kari and Thierrin [114] study the $k$-insertion (and corresponding $k$-deletion) closure of a language. They also define the notion of $k$-prefix codes [114], which are a particular case of $T$-codes introduced in Chapter 6. However, we note that $k$-prefix codes are one of the few cases of research into codes where a novel definition is based primarily on a new language operation, rather than a new binary relation on words.

Berard [16] has introduced both the *literal* and *initial literal shuffle* operations. The motivation is modelling concurrent processes; literal shuffle models synchronized transmission where "each transmitter emits, in turn, one elementary signal [16, p. 51]". Both literal and initial literal shuffle are particular cases of shuffle on trajectories, and are given by $T = (0^* + 1^*)(01)^*(0^* + 1^*)$ and $T = (01)^*(0^* + 1^*)$, respectively. Literal shuffle has been further studied by Tanaka [191] on the closure of the class of prefix codes under literal shuffle, and by Ito and Tanaka [81] who consider the density of initial literal shuffles. Moriya and Yamasaki [154] have studied literal shuffle on $\omega$-words.

### 3.3.2 Deletion Operations

Many deletion operations which are specific instances of the deletion along trajectories model we suggest in Chapter 5 have been considered in the literature. This shows the usefulness of the deletion along trajectories model.

The most studied deletion operations are the left- and right-quotient operations. The first formal study of quotient appears to be by Ginsburg and Spanier [52], who show three fundamental results on right-quotient: that the right-quotient of a CFL by a regular language (or of a regular language by a CFL) is a CFL, that CF is not closed under quotient, and given two CFLs $L_1$, $L_2$, it is undecidable whether $L_1/L_2$ is a CFL. Ginsburg and Spanier attribute the notion of quotient to the "SHARE Theory of Information Handling Committee [52, p.487]".

Latteux *et al.* [130] show that a restricted class of CFLs, called the *one-counter languages*, are closed under quotient, and that every recursively enumerable language can be expressed as the quotient of two LCFLs.

Another well-studied deletion operation is known as *scattered deletion*. Given two words $x, y \in \Sigma^*$, their scattered deletion, denoted $x \rightsquigarrow y$, is given by

$$x \rightsquigarrow y = \left\{ \prod_{i=1}^{n+1} x_i \ : \ x = (\prod_{i=1}^{n} x_i y_i) x_{n+1}, \, y = \prod_{i=1}^{n} y_i \text{ with } x_i, \, y_j \in \Sigma^* \right\}.$$

We extend $\rightsquigarrow$ to languages as expected. The scattered deletion operation, a natural operation on words, has a long history in the literature. For instance, the scattered deletion operation is an implicit operation in the theory of flow expressions (see, e.g., Shaw [182]).

Kari (as Sântean [179]) appears to be the first author to have formally studied the scattered deletion operation (under the name *literal subtraction*) and established several closure properties. This investigation is continued by Kari in a subsequent paper [105].

Also investigated by Kari [105] are several other deletion operations, some of which are modelled by our framework (e.g., sequential deletion), and others which are not (e.g., controlled deletion, parallel deletions and deletion with permuted components). Closure properties of each of these operations are investigated.

The sequential deletion operation is given by $x \rightarrow y = \{x_1x_2 \ : \ x_1yx_2 = x\}$. Kari *et al.* [111] explore results on the cardinality of $w \rightarrow L$, for $w \in \Sigma^*$ and $L \subseteq \Sigma^*$, as well as the decidability of the following problem: given a finite set $F$, do there exist $w \in \Sigma^*$ and $L \subseteq \Sigma^*$ such that $F = w \rightarrow L$?

Language equations involving deletion have been studied by Kari [106]. Recently, Kari and Sosík have continued the investigation of language equations involving scattered deletion, quotient and sequential deletion [113].

Meduna [153] has introduced an interesting deletion operation, called *middle quotient*, defined as follows:

$$L_1|L_2 = \{w \in \Sigma^* \ : \ \exists v \in L_2 \text{ such that } vwv \in L_1\}.$$

The main motivation for introducing this operation is that for any recursively enumerable language $L$, there exist linear CFLs $L_1$, $L_2$ such that $L = L_1|L_2$ [153].

A popular topic in the theory of formal languages is proportional removals. Given a binary relation $r \subseteq \mathbb{N}^2$, the proportional removal of a language $L \subseteq \Sigma^*$ with respect to $r$ is the language

$$P(r, L) = \{x \in \Sigma^* \ : \ \exists y \in \Sigma^* \text{ such that } xy \in L \text{ and } (|x|, |y|) \in r\}.$$

Proportional removals have been studied by Stearns and Hartmanis [189], Amar and Putzolu [4, 5] Seiferas and McNaughton [180], Kosaraju [120, 121, 122], Kozen [123], Zhang [205], the author [35], and others. We study proportional removals extensively in Chapter 5.

Berstel *et al.* [17] consider filtering, which is a deletion operation specified by a sequence of natural numbers $s \subseteq \mathbb{N}$. We will see that filtering is a specific case of deletion along trajectories. Necessary and sufficient conditions on a sequence of natural numbers preserving regularity are given by Berstel *et al.* [17].

### 3.3.3 Interaction

Kari [102] has studied conditions on which the operations of insertion and deletion are reversible and deterministic. In particular, given the inverse operations (intuitively, but also in a sense we will

define in Chapter 5) of (sequential) insertion and deletion, Kari examines under what conditions on words $u, v$ the language $(u \leftarrow v) \rightarrow v$ consists of only one word.

### 3.3.4 Iteration

Iterated insertion and deletion operations have been studied by Ito *et al.* [78, 79], and Kari and Thierrin [117]. The iterated insertion operations considered are sequential insertion, shuffle and $k$-insertion; the corresponding iterated deletion operations are also considered. In each case, the authors consider the *residual* of a language $L$ under the studied operation, and show its relation to the closure of $L$ under the corresponding insertion operation. We generalize these notions for shuffle and deletion along trajectories in Chapter 8.

Ito and Silva [80] have examined closure properties of iterated scattered and sequential deletion. Two open problems proposed by Ito and Silva have been solved by the author and Okhotin [42].

Ito *et al.* [82] have examined shuffle-closed languages, strongly shuffle-closed languages and extended shuffle bases. Characterizations of (strongly) shuffle-closed commutative regular languages are obtained. The notion of extended bases has been developed in the more general setting of binary word operations by Hsiao *et al.* [69].

Kari and Thierrin have generalized the notion of primitivity from Kleene closure to iterated shuffle and insertion [118]. In a broader setting, Hsiao *et al.* [69] have considered iteration and primitivity of arbitrary word operations. However, the setting is so general that obtaining results often requires many assumptions, and results such as closure properties and decidability cannot be obtained.

An interesting application of results on iteration of insertion and deletion operations was noted by Parkes and Thomas [161, 162]. In particular, the word problem for the syntactic monoid of a regular language $R$ can be expressed as the intersection of the insertion- and deletion-closure of $R$, which were introduced by Ito *et al.* [78]. Similar observations were made by Tully [194], but phrased in more group-theoretic terms. Ramesh Kumar and Rajan [169] have further explored the concepts introduced by Tully.

### 3.3.5 Decomposition and Related Language Equations

The problem of decomposition of languages for insertion operations has not been widely studied, except for the case of concatenation. Given a regular language $R$, the problem of determining whether there exist $L_1, L_2$ such that $R = L_1 L_2$ has been considered by Conway [28], Kari [106], and Kari and Thierrin [117]. This problem is decidable. Choffrut and Karhumäki [25] and Polák [167] have considered more general systems of equations and inequalities (see also Baader and Küsters [11] and Baader and Narendran [13], who reduce solving similar systems of equations to solving a single language equation). The equations considered by Choffrut and Karhumäki and Polák include the decomposition equation $R = X_1 X_2$ studied previously by Conway, Kari and Kari and Thierrin, but also include equations of the form $R = r(X_1, \ldots, X_n)$, where $R$ is a regular language and $r(X_1, \ldots, X_n)$ is a regular expression over the variables $X_1, \ldots, X_n$.

Given a language $R$, we say that it is prime if $R = L_1 L_2$ implies that $\{L_1, L_2\} = \{\{\epsilon\}, R\}$. Salomaa and Yu [176] show that the problem of deciding whether a regular language is prime is decidable; see also Mateescu *et al.* [151]. Wood [199] has given conditions on $R$ which ensure that a decomposition $R = L_1 L_2$ is unique.

## 3.4 Shuffle on Trajectories

As already mentioned, shuffle on trajectories was defined by Mateescu *et al.* [147]. Harju *et al.* [61] consider the syntactic monoids recognizing a language constructed from regular languages with shuffle on trajectories. We examine the complementary question for deletion along trajectories in Section 5.3.1. We now describe other areas of research related to shuffle on trajectories.

### 3.4.1 Infinite Words

While we do not deal with infinite words in this thesis, the concept of shuffle on trajectories for infinite words has received attention in the literature. Mateescu *et al.* [147] introduced the notion of shuffle on trajectories for infinite words along with shuffle on trajectories for finite words, and

examined similar algebraic properties for infinite trajectories as for finite trajectories. Trajectories for infinite words are called *ω-trajectories*. Kadrie *et al.* [101] have defined a binary relation defined on $\Sigma^\omega$ and briefly examined its properties (we consider the analog for finite words in Chapter 6).

### 3.4.2 Fairness

Defining a fair operation, that is, one which allows both input languages to have a corresponding letter be "shuffled in" during some reasonable time frame, has been the subject of research related to shuffle on trajectories.

Mateescu *et al.* [147] use the concept of fairness as an example of the usefulness of the model of shuffle on trajectories. They define explicit sets of trajectories and $\omega$-trajectories which have the desired fairness properties. Mateescu *et al.* [152] have extended this to study fairness of multiple languages, which requires defining an extended shuffle on trajectories operation to operation on $n$ languages instead of two. Mateescu and Mateescu [145] have examined the fair and associative trajectories on $\omega$-words.

### 3.4.3 Related Concepts

The notion of shuffle on trajectories has been used in other interesting settings, including grammars, combinatorics and timed automata. We survey these now.

#### Grammar Formalisms

Martin-Vide *et al.* [142] introduce the notion of contextual grammars on trajectories. These are an extension of the notion of a contextual grammar by the addition of a set of trajectories.

In particular a *contextual grammar with contexts shuffled on trajectories* (abbreviated CST) is a four-tuple $G = (\Sigma, B, C, T)$ where $\Sigma$ is an alphabet, $B, C$ are finite languages over $\Sigma$, called the *base* and *contexts*, respectively, and $T = (T_c)_{c \in C}$ is a family of trajectories indexed by elements of $C$, i.e., for each $c \in C$, $T_c \subseteq \{0, 1\}^*$.

The generation of words in $G$ is accomplished as follows: let $x, y \in \Sigma^*$. Then we use the notation $x \Rightarrow_G y$ to denote the fact that there exists $c \in C$ such that $y \in x \sqcup\!\sqcup_{T_c} c$. Let $\Rightarrow_G^*$ be the reflexive and transitive closure of $\Rightarrow_G$. Then the language generated by $G = (\Sigma, B, C, T)$ is denoted $L(G)$ and is given by

$$L(G) = \{w \in \Sigma^* \ : \ \exists x \in B \text{ such that } x \Rightarrow_G^* w\}.$$

Martin-Vide *et al.* give the following example: let $G = (\Sigma, B, C, T)$ be given by $\Sigma = \{a, b\}$, $B = \{\epsilon\}$, $C = \{aa, bb\}$ and $T = (T_{aa}, T_{bb})$, where $T_{aa} = T_{bb} = \{01^n01^n \ : \ n \geq 0\}$. Then $L(G) = \{ww \ : \ w \in \{a, b\}^*\}$.

Martin-Vide *et al.* investigate the relationship between CST and other contextual grammar classes. They also examine the relationship between the complexity of the members of $T$ as languages and the generative capacity of $G$.

Mateescu has also extended the notion of co-operating distributed grammars (CD grammars) to encompass the notion of trajectories [143]. A CD grammar on trajectory $T$ is a six-tuple $\Gamma = (V, \Sigma, S, P_0, P_1, T)$ where $V$ is a finite set of non-terminals, $\Sigma$ is a finite alphabet, $S \in V$ is a distinguished start state, $P_0, P_1 \subseteq V \times (V \cup \Sigma)^*$ are two finite sets of productions, and $T \subseteq \{0, 1\}^*$ is the set of trajectories.

Let $\Rightarrow_i$ denote the relation defined by the CFG $G_i = (V, \Sigma, S, P_i)$, as defined in Section 2.3, for $i = 0, 1$. Then a word $w \in \Sigma^*$ is generated by $\Gamma$ if there exist $t \in T$ of length $n$ and $\alpha_i \in (V \cup \Sigma)^*$ for $1 \leq i \leq n$ such that if $t = t_1t_2 \cdots t_n$ with $t_i \in \{0, 1\}$ then for all $1 \leq i \leq n-1$ $\alpha_i \Rightarrow_{t_i} \alpha_{i+1}$, with $S = \alpha_1$ and $w = \alpha_n$. The language generated by $\Gamma$, denoted $L(\Gamma)$, is the set of all words generated by $\Gamma$. The usual notion of a CD grammar corresponds to $T = 0^*1^*$. Other more complicated notions of acceptance are also considered. The notion of CD grammars on trajectories is also generalized to grammars with $n$ sets of productions $P_0, P_1, \ldots, P_{n-1}$, and a set of trajectories $T \subseteq \{0, \ldots, n-1\}^*$.

#### Timed Automata

Krishnan [124] has utilized the notion of trajectories in the context of discrete event systems and timed automata. The concept of a trajectory is extended to the concept of a scheduler for real-time events.

#### Combinatorics

The notion of shuffle on trajectories has been employed in an interesting combinatorial setting. In particular, Vajnovski [195] has constructed a Gray code for the so-called Motzkin words; the use of shuffle on trajectories in the construction is essential. We do not describe Gray codes or Motzkin words here, the reader may consult [195] for definitions. Baril and Vajnovski [15] also define a Gray code for derangements (permutations with no fixed points), again using shuffle on trajectories in a combinatorial setting.

Vajnovski has also used the concept of shuffle on trajectories as a combinatorial constructor for *multiset permutations* [196] (given $n_0, n_1, n_2, \ldots, n_k \geq 0$, a multiset permutation is a sequence integers in which $i$ appears $n_i$ times for all $0 \leq i \leq k$). A combinatorial constructor enables one to construct complex combinatorial objects (in this case, multiset permutations) out of simpler objects, which is a common theme in combinatorial research. The construction of Vajnovski allows Gray code generation of multiset permutations by a so-called *loopless* method [196], by using shuffle on trajectories.

### 3.4.4 Splicing on Routes

The notion of shuffle on trajectories was extended by Mateescu [144] to encompass certain splicing operations. This extension is called *splicing on routes*. We give the formal definition of splicing on routes in Section 5.7. Splicing on routes is a proper extension of shuffle on trajectories, and also encompasses several unary operations. We discuss the unary operations modelled by splicing on routes in Section 5.7. Bel-Enguix *et al.* use the concept of splicing on routes to model dialog in

natural language [12].

### 3.4.5 Concurrent Work

Independent to this thesis, the concept of deletion on trajectories has been introduced by Kari and Sosík [112]. The authors develop the same framework, and investigate similar closure properties and decidability of solutions to language equations in one variable. Algebraic properties not studied in this thesis are also considered by Kari and Sosík. Unlike the case of shuffle on trajectories, these algebraic characterizations for deletion along trajectories are satisfied only by trivial deletion operations. For example, a deletion operation $\diamond$ modelled by deletion along trajectories is commutative if and only if $L_1 \diamond L_2 \subseteq \{\epsilon\}$ for all languages $L_1, L_2$ [112].

Kari and Sosík [112] also introduce the notion of substitution and right-difference on trajectories. This concept is similar to shuffle and deletion along trajectories, but involves substitution of words rather than interleaving of words. The reader is referred to Kari and Sosík for details. The notion of substitution and right-difference on trajectories is further investigated and applied to the modelling of noisy channels by Kari *et al.* [110].

The use of shuffle and deletion along trajectories has been employed by Kari *et al.* [108] to investigate properties of bonding in DNA strands. The formalism defined by Kari *et al.* is called *bond-free properties*. There are similarities between bond-free properties and the notion of $T$-codes developed in Chapter 6. We discuss these similarities in greater detail in Chapter 6. Kari *et al.* [109] have extended this work on bond-free properties, with particular emphasis on DNA strands satisfy constraints based on the Hamming distance.

Deletion on trajectories has also been used as a tool to characterize when commutative languages are regular by the author and others [41]. We do not examine this application in this thesis.

Work on decidability of language equations involving shuffle on trajectories has been continued by the author and Salomaa [45]. In particular, it is shown that there exists a fixed linear context-free set of trajectories $T$ such that the following problem is undecidable: "given regular languages $R_1, R_2, R_3$, does $R_1 \sqcup_T R_2 = R_3$ hold?" Similar results are given for language equations of the

form $R_1 \sqcup\!\sqcup_T X = R_3$ where $R_1$, $R_3$ are regular and $X$ is unknown.

# Chapter 4

# Descriptional Complexity

## 4.1 Introduction

Descriptional complexity of formal languages deals with the problems of concise descriptions of languages in terms of generative or accepting devices. For instance, the *(deterministic) state complexity* of a regular language $L$ is the minimal number of states in any deterministic finite automaton accepting $L$ [204]. Nondeterministic state complexity of a regular language is similarly defined [48, 65, 66].

There is much interest in descriptional complexity as it relates to the efficiency of implementing operations on languages. For instance, if $f$ is a binary operation which preserves regular languages, then research in state complexity typically seeks to express the *worst-case* state complexity of $f(L_1, L_2)$ as a function of the state complexities of $L_1$ and $L_2$. Informally, we refer to this expression for the complexity of $f(L_1, L_2)$ as *the state complexity* of $f$. For a survey of worst-case state complexity for finite and regular languages, see Yu [202, 203]. We note that research into *average-case* state complexity (instead of worst-case) of $f$ has also been examined by Nicaud [155] and the author [35].

For shuffle on trajectories, Mateescu *et al.* [147] and Harju *et al.* [61] both give proofs that, given a regular set of trajectories $T$ and regular languages $L_1, L_2$, the operation $L_1 \sqcup\!\sqcup_T L_2$ always yields

a regular language. Thus, it is reasonable to consider the state complexity of shuffle on trajectories; this is the goal of this chapter.

It is known that each set $T \subseteq \{0,1\}^*$ defines a unique operation $\sqcup\!\sqcup_T$ (to see this, consider that $0^* \sqcup\!\sqcup_T 1^* = T$). Therefore, the family of shuffle on trajectory operations is very complex, and in this study we only begin to address the many questions which arise from studying the state complexities of these operations. We incorporate other measures of complexity used in formal languages and automata theory, including nondeterministic state complexity and language density (for a definition of density of languages, see Section 4.3).

In particular, we establish a general upper bound, and improve it in the case when the set of trajectories $T$ has constant density. For sets of trajectories with density one, we obtain a lower bound that is of the same order as the upper bound when the state complexity of the set of trajectories grows with respect to the state complexity of the component languages.

We also consider a result of Yu *et al.* [204] on the state complexity of the concatenation operation. We show that the state complexity of $L_1L_2$ can be improved in the case that $L_2$ can be easily accepted by a NFA. However, this is not an improvement in the worst case.

## 4.2 General State Complexity Bounds

Given a regular language $L$, define the *(deterministic) state complexity* of $L$, denoted $sc(L)$, by

$$sc(L) = \min\{|Q| \ : \ M = (Q, \Sigma, \delta, q_0, F) \text{ is a DFA accepting } L\}.$$

It is well known that for a regular language $L$, $sc(L)$ is the index of $\equiv_L$, the Myhill-Nerode congruence with respect to $L$. The *nondeterministic state complexity* of a regular language $L$ is defined similarly by

$$nsc(L) = \min\{|Q| \ : \ M = (Q, \Sigma, \delta, q_0, F) \text{ is an NFA accepting } L\}.$$

Nondeterministic state complexity has recently been studied by Holzer and Kutrib [65, 66] and Ellul [48].

The following theorem [147, Thm. 5.1] states that regular sets of trajectories preserve regularity. It serves as the starting point of this chapter:

**Theorem 4.2.1** *Let $L_1, L_2$ be regular languages over $\Sigma^*$ and let $T \subseteq \{0,1\}^*$ be a regular language. Then $L_1 \sqcup\!\sqcup_T L_2$ is a regular language.*

The construction given by Mateescu *et al.* [147, Thm. 5.1] yields our most general upper bound on the state complexity of shuffle on trajectories. We state our upper bound in terms of nondeterministic state complexity:

**Lemma 4.2.2** *Let $L_1, L_2$ be regular languages over $\Sigma^*$ and $T \subseteq \{0,1\}^*$ be a regular set of trajectories. Then*

$$sc(L_1 \sqcup\!\sqcup_T L_2) \leq 2^{nsc(L_1)nsc(L_2)nsc(T)}.$$

**Proof.** We construct a NFA $M'$ accepting $L_1 \sqcup\!\sqcup_T L_2$. Let $M_i = (Q_i, \Sigma, \delta_i, q_i, F_i)$ be minimal NFAs accepting $L_i$ for $i = 1$ and 2, and let $M_T = (Q_T, \{0,1\}, \delta_T, q_T, F_T)$ be a minimal NFA accepting $T$.

Let $M' = (Q, \Sigma, \delta, q_0, F)$ be an NFA with $Q = Q_1 \times Q_2 \times Q_T$, $q_0 = [q_1, q_2, q_T]$, $F = F_1 \times F_2 \times F_T$ and $\delta$ given by

$$\begin{aligned}\delta([q_i, q_j, q_k], a) \;=\; & \{[q, q_j, q'] \,:\, q \in \delta_1(q_i, a), q' \in \delta_T(q_k, 0)\} \\ & \cup \{[q_i, q, q'] \,:\, q \in \delta_2(q_j, a), q' \in \delta_T(q_k, 1)\}\end{aligned}$$

for all $q_i \in Q_1, q_j \in Q_2, q_k \in Q_T$ and $a \in \Sigma$. Then it is easily verified that $L(M') = L_1 \sqcup\!\sqcup_T L_2$. Since $M'$ is an NFA with $nsc(L_1)nsc(L_2)nsc(T)$ states, the result easily follows, since any NFA with $n$ states can be simulated by a DFA with $2^n$ states. ■

Thus, we have the following interesting corollary:

**Corollary 4.2.3** *Let $L_1, L_2$ be regular languages over $\Sigma^*$ and $T \subseteq \{0,1\}^*$ be a regular set of trajectories. If*

$$sc(L_1 \sqcup\!\sqcup_T L_2) = 2^{sc(L_1)sc(L_2)sc(T)}$$

*then* $sc(L_i) = nsc(L_i)$ *for* $i = 1, 2$ *and* $sc(T) = nsc(T)$.

**Proof.** As $nsc(L) \leq sc(L)$ for all regular languages $L$, the result is evident. ■

Using the idea of Lemma 4.2.2, we may slightly modify a result of Yu *et al.* [204] concerning concatenation:

**Theorem 4.2.4** *Let* $L_1, L_2 \subseteq \Sigma^*$ *be regular languages. Then*

$$sc(L_1L_2) \leq sc(L_1)2^{nsc(L_2)} - k2^{nsc(L_2)-1},$$

*where* $k$ *is the number of final states in the minimal DFA accepting* $L_1$.

This is not an improvement in the worst case, but it again shows that if $L_1, L_2$ are languages with $sc(L_1L_2) = sc(L_1)2^{sc(L_2)} - k2^{sc(L_2)-1}$ then $nsc(L_2) = sc(L_2)$. This applies to the lower bound given by Yu *et al.*: Let $M_B = (\{p_0, p_1, \ldots, p_n\}, \{a, b, c\}, \delta_B, p_0, \{p_{n-1}\})$ be a DFA with $\delta_B$ given by

$$\begin{aligned} \delta_B(p_i, a) &= p_i; \\ \delta_B(p_i, b) &= p_{i+1}; \\ \delta_B(p_i, c) &= p_1; \end{aligned}$$

where the indices are taken modulo $n$. Then if $L = L(M_B)$, $sc(L) = nsc(L)$. Thus, the language given by the above DFA cannot be accepted by an NFA with any less states.

Also note that Theorem 4.2.4 demonstrates that there exist sets of trajectories $T$ for which Lemma 4.2.2 is not optimal. In particular, concatenation is given by the set of trajectories $T = 0^*1^*$, that is, $\sqcup\!\sqcup_T = \cdot$, the concatenation operator. Since $nsc(0^*1^*) = sc(0^*1^*) - 1 = 2$ (see Figure 4.1), Lemma 4.2.2 gives $sc(L_1L_2) \leq 4^{nsc(L_1)nsc(L_2)}$. However, by Theorem 4.2.4, we get that

$$sc(L_1L_2) \leq sc(L_1)2^{nsc(L_2)} \leq 2^{nsc(L_1)+nsc(L_2)}.$$

Thus, we have the following problem:

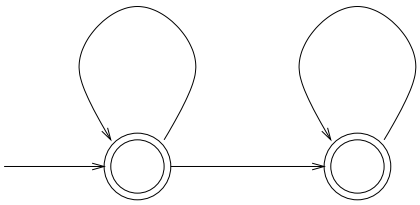

Figure 4.1: A two-state NFA accepting the set $T = 0^*1^*$ of trajectories.

**Open Problem 4.2.5** *For what regular sets of trajectories $T \subseteq \{0,1\}^*$ does the construction given by Lemma 4.2.2 give a construction which is best possible?*

Consider unrestricted shuffle, given by the set of trajectories $T = (0+1)^*$. The bound of Lemma 4.2.2 in this case is $2^{nsc(L_1)nsc(L_2)}$. Câmpeanu *et al.* [22] have shown that there exist languages $L_1$ and $L_2$ accepted by incomplete DFAs having, respectively, $n$ and $m$ states such that any incomplete DFA accepting $L_1 \sqcup\!\sqcup L_2$ has at least $2^{nm} - 1$ states. This bound is optimal for incomplete DFAs, however; for complete DFAs it gives only the lower bound $2^{(sc(L_1)-1)(sc(L_2)-1)}$. However, we regard this as near enough to our goal of Lemma 4.2.2 for our purposes, i.e., we regard $T = (0+1)^*$ as an example of a set of trajectories $T$ satisfying Open Problem 4.2.5.

## 4.3 Slenderness and Trajectories

In this section, we consider the opposite question to Open Problem 4.2.5. That is, we are interested in finding $T \subseteq \{0,1\}^*$ such that Lemma 4.2.2 is not optimal, and in fact, is a very poor bound. To define such $T$, we examine another descriptional complexity measure on languages, that of the *density*. Informally, the density of a language measures the number of words of each length. We find that sets of trajectories $T$ with very small density yield operations $\sqcup\!\sqcup_T$ with small state complexity, compared to Lemma 4.2.2.

We now give the definition of the *density function* of a language $L \subseteq \Sigma^*$. For all $n \geq 0$, define $p_L : \mathbb{N} \to \mathbb{N}$ as

$$p_L(n) = |L \cap \Sigma^n|.$$

That is, $p_L(n)$ gives the number of words of length $n$ in $L$. By the density of a language $L$, we informally mean the asymptotic behaviour of $p_L$. The following important result of Szilard *et al.* [190, Thm. 3] characterizes the density of regular languages:

**Theorem 4.3.1** *A regular language $R$ over $\Sigma$ satisfies $p_R(n) \in O(n^k)$, $k \geq 0$ if and only if $R$ can be represented as a finite union of regular expressions of the following form:*

$$x y_1^* z_1 \cdots y_t^* z_t$$

*where $x, y_1, z_1, \cdots, y_t, z_t \in \Sigma^*$, and $0 \leq t \leq k+1$.*

Call a language $L$ *slender* if $p_L(n) \in O(1)$ [168]. If a regular language $R$ has polynomial density $O(n^k)$, let $t$ be the smallest integer such that $R = \cup_{i=1}^{t} x_i y_{i,1}^* z_{i,1} \cdots y_{i,k_i}^* z_{i,k_i}$, $0 \leq k_i \leq k+1$, $i = 1, \ldots, t$. Then call $t$ the *UkL-index* of $L$. If $k = 0$, we call $t$ the *USL-index* of $L$ (languages with USL index $t$ are called $t$-thin by Păun and Salomaa [168]; slender regular languages were also characterized independently by Shallit [181, Lemma 3, p. 336]).

### 4.3.1 Perfect Shuffle

We first consider a common example of a slender set of trajectories, that of perfect (or balanced literal) shuffle. Recall that perfect shuffle is given by the set of trajectories $T_p = (01)^*$; we denote the perfect shuffle operation by $\sqcup\!\sqcup_p$. Thus, for $x, y \in \Sigma^*$, $x = x_1 x_2 \cdots x_m$, $y = y_1 y_2 \cdots y_n$, where $x_i, y_j \in \Sigma$, the perfect shuffle of $x$ and $y$ is

$$x \sqcup\!\sqcup_p y = \begin{cases} x_1 y_1 x_2 y_2 \cdots x_m y_m & \text{if } m = n; \\ \emptyset & \text{otherwise.} \end{cases}$$

The following result can be obtained directly. However, we will defer the proof by stating that it is an immediate corollary of Lemma 4.3.4, which appears below in Section 4.3.2:

**Lemma 4.3.2** *Let $L_1, L_2$ be regular languages with $sc(L_i) = n_i$ for $i = 1, 2$. Then*

$$sc(L_1 \sqcup\!\sqcup_p L_2) \leq 2n_1 n_2.$$

We can show this to be optimal for all $n_1, n_2$ over a two-letter alphabet.

**Lemma 4.3.3** *Let $\Sigma = \{a, b\}$. Let $n_1, n_2 \geq 0$ be integers. Then there exist regular languages $L_1, L_2 \subseteq \Sigma^*$ with $sc(L_i) = n_i$ for $i = 1, 2$ such that $sc(L_1 \sqcup\!\sqcup_p L_2) = 2n_1n_2$.*

**Proof.** Let $L_1 = \{x \in \{a, b\}^* : |x|_a \equiv 0 \pmod{n_1}\}$ and $L_2 = \{x \in \{a, b\}^* : |x|_b \equiv 0 \pmod{n_2}\}$. It is easily verified that $sc(L_1) = n_1$ and $sc(L_2) = n_2$. We claim that $sc(L_1 \sqcup\!\sqcup_p L_2) \geq 2n_1n_2$.

We consider words of the form $a^{2i}b^j$ for $0 \leq i < n_1$ and $0 \leq j \leq 2n_2 - 1$. For any pairs $[i_1, j_1] \neq [i_2, j_2]$, we have that $a^{2i_1}b^{j_1} \not\equiv_L a^{2i_2}b^{j_2}$ (where $L = L_1 \sqcup\!\sqcup_p L_2$). To show this, we show that any two distinct words $w_1 = a^{2i_1}b^{j_1}$ and $w_2 = a^{2i_2}b^{j_2}$ can be distinguished with the word $u = a^{2(n_1-i_1)}b^{2n_2-j_1}$. We establish now that $w_1 \not\equiv_L w_2$ by showing that $w_1u \in L$ while $w_2u \notin L$.

**Case (i)**: $j_1, j_2$ both odd. Let $0 \leq j_1', j_2' < n_2$ be integers such that $j_1 = 2j_1' + 1$ and $j_2 = 2j_2' + 1$.

Consider $w_1u = a^{2i_1}b^{2j_1'+1}a^{2(n_1-i_1)}b^{2(n_2-j_1')-1}$. Then $w_1u = v_1 \sqcup\!\sqcup_p v_2$ where

$$\begin{aligned} v_1 &= a^{i_1}b^{j_1'+1}a^{n_1-i_1}b^{n_2-j_1'-1}; \\ v_2 &= a^{i_1}b^{j_1'}a^{n_1-i_1}b^{n_2-j_1'}. \end{aligned}$$

Thus $|v_1|_a = n_1$ and $|v_2|_b = n_2$ and so $w_1u \in L$.

As for $w_2u = a^{2i_2}b^{2j_2'+1}a^{2(n_1-i_1)}b^{2(n_2-j_1')-1}$, we have $w_2u = v_3 \sqcup\!\sqcup_p v_4$ where

$$\begin{aligned} v_3 &= a^{i_2}b^{j_2'+1}a^{n_1-i_1}b^{n_2-j_1'-1}; \\ v_4 &= a^{i_2}b^{j_2'}a^{n_1-i_1}b^{n_2-j_1'}. \end{aligned}$$

Then note that $|v_3|_a = n_1 - i_1 + i_2$ and $|v_4|_b = n_2 - j_1' + j_2'$. Since $0 \leq i_1, i_2 < n_1$ and $0 \leq j_1', j_2' < n_2$, and under the assumptions that one of $i_1 \neq i_2$ and $j_1 \neq j_2$ is true, we have either $v_3 \notin L_1$ or $v_4 \notin L_2$. Thus, $w_2u \notin L$.

**Case (ii)**: $j_1, j_2$ both even. Let $0 \leq j_1', j_2' < n_2$ be integers such that $j_1 = 2j_1'$ and $j_2 = 2j_2'$.

Consider $w_1u = a^{2i_1}b^{2j_1'}a^{2(n_1-i_1)}b^{2(n_2-j_1')}$. Again, decomposing $w_1u$ as $w_1u = v_1 \sqcup\!\sqcup_p v_2$ yields

$$v_1 = v_2 = a^{i_1}b^{j_1'}a^{n_1-i_1}b^{n_2-j_1'}.$$

Thus, as $|v_1|_a = n_1$ and $|v_2|_b = n_2$ we have $v_1 \in L_1$, $v_2 \in L_2$ and $w_1u \in L$.

Considering $w_2u = a^{2i_2}b^{2j_2'}a^{2(n_1-i_1)}b^{2(n_2-j_1')}$, we can write $w_2u = v_3 \sqcup\!\sqcup_p v_4$ where

$$v_3 = v_4 = a^{i_2}b^{j_2'}a^{n_1-i_1}b^{n_2-j_1'}$$

and so $|v_3|_a = n_1 - i_1 + i_2$ and $|v_4|_b = n_2 - j_1' + j_2'$. Our assumption that one of $i_1 \neq i_2$ and $j_1 \neq j_2$ is true implies that $v_3 = v_4 \notin L_1 \cap L_2$. Thus, $w_2u \notin L$.

**Case (iii)**: $j_1$ even and $j_2$ odd. Let $0 \leq j_1', j_2' < n_2$ be integers such that $j_1 = 2j_1'$ and $j_2 = 2j_2' + 1$.

Now $w_1u = a^{2i_1}b^{2j_1'}a^{2(n_1-i_1)}b^{2(n_2-j_1')}$. As in case (ii), we have seen that $w_1u \in L$. Consider $w_2u = a^{2i_2}b^{2j_2'+1}a^{2(n_1-i_1)}b^{2(n_2-j_1')}$. Thus $|w_2u| \equiv 1 \pmod 2$ and there do not exist words $v_3, v_4$ such that $v_3 \sqcup\!\sqcup_p v_4 = w_2u$.

**Case (iv)**: $j_1$ odd and $j_2$ even. Let $0 \leq j_1', j_2' < n_2$ be integers such that $j_1 = 2j_1' + 1$ and $j_2 = 2j_2'$. Consider $w_1u = a^{2i_1}b^{2j_1'+1}a^{2(n_1-i_1)}b^{2(n_2-j_1')-1}$. Then as we have seen in case (i), $w_1u \in L$. However, consider $w_2u = a^{2i_1}b^{2j_1'}a^{2(n_1-i_1)}b^{2(n_2-j_1')-1}$. As $|w_2u| \equiv 1 \pmod 2$, there do not exist words $v_3, v_4$ such that $w_2u = v_3 \sqcup\!\sqcup_p v_4$. ■

In the unary case, for any two words $a^i, a^j$, we have

$$a^i \sqcup\!\sqcup_p a^j = \begin{cases} a^{i+j} = a^{2i} & \text{if } i = j; \\ \emptyset & \text{otherwise.} \end{cases}$$

Thus, we see that for unary languages $L_1, L_2 \subseteq a^*$,

$$L_1 \sqcup\!\sqcup_p L_2 = h(L_1 \cap L_2)$$

where $h : a^* \to a^*$ is the morphism defined by $h(a) = a^2$. Thus, we can show that for unary languages

$$sc(L_1 \sqcup\!\sqcup_p L_2) = 2sc(L_1 \cap L_2).$$

The state complexity of intersection on unary languages is well-studied [155, 163, 202]. For instance, if $\gcd(n_1, n_2) = 1$, we can take $L_1 = (a^{n_1})^*$ and $L_2 = (a^{n_2})^*$ [184]. Thus, for these languages $sc(L_1 \sqcup\!\sqcup_p L_2) = 2n_1n_2$. However, if $\gcd(n_1, n_2) > 1$, the situation is more interesting.

For this case, see Pighizzini and Shallit [163]. We also note the work of Nicaud on the average state complexity of intersection [155].

### 4.3.2 Bounds on Slender Trajectories

We may now relate slenderness of sets of trajectories to state complexity. Our first result handles the case where $T = uv^*$.

In what follows, if $u$ is a word of length $n$, then $u(i)$ represents the $(i+1)$-st letter of $u$ for all $0 \leq i \leq n-1$. Further, let $\mathbf{n} = \{0, 1, 2, \ldots, n-1\}$.

**Lemma 4.3.4** *Let $T = uv^*$ where $u, v \in \{0,1\}^*$. Let $L_i$ be regular languages over $\Sigma$, with $sc(L_i) = n_i$, $i = 1, 2$. Let $L = L_1 \sqcup\!\sqcup_T L_2$. Then*

$$sc(L) \leq |uv| n_1 n_2. \tag{4.1}$$

**Proof.** For $i = 1, 2$, let $L_i$ be accepted by a DFA $M_i = (Q_i, \Sigma, \delta_i, q_i, F_i)$ with $|Q_i| = n_i$. We describe $M = (Q, \Sigma, \delta, q_0, F)$ such that $L(M) = L_1 \sqcup\!\sqcup_T L_2$.

Let $n = |uv|$. We let $Q = Q_1 \times Q_2 \times \mathbf{n}$, $q_0 = [q_1, q_2, 0]$, and give $\delta$ by

$$\delta([q_i, q_j, k], a) = \begin{cases} [\delta_1(q_i, a), q_j, k+1] & \text{if} \quad (uv)(k) = 0 \quad \text{and} \quad k < n-1; \\ [\delta_1(q_i, a), q_j, |u|] & \text{if} \quad (uv)(k) = 0 \quad \text{and} \quad k = n-1; \\ [q_i, \delta_2(q_j, a), k+1] & \text{if} \quad (uv)(k) = 1 \quad \text{and} \quad k < n-1; \\ [q_i, \delta_2(q_j, a), |u|] & \text{if} \quad (uv)(k) = 1 \quad \text{and} \quad k = n-1. \end{cases}$$

Finally we let $F = F_1 \times F_2 \times \{|u|\}$. It is easily verified that $L(M)$ accepts the desired language. ■

We now give a bound for sets of trajectories $T = uv^*w$ with $w \neq \epsilon$.

**Lemma 4.3.5** *Let $T = uv^*w$ where $u, v, w \in \{0,1\}^*$ and $w \neq \epsilon$. Let $L_i$ be regular languages over $\Sigma$, with $sc(L_i) = n_i$, $i = 1, 2$. Let $L = L_1 \sqcup\!\sqcup_T L_2$. Then*

$$sc(L) \leq n_1 n_2 \left( |u| + 1 + |v| \frac{(n_1 n_2)^{\left\lceil \frac{|w|}{|v|} \right\rceil + 1} - n_1 n_2}{n_1 n_2 - 1} \right). \tag{4.2}$$

**Proof.** For $i = 1, 2$, let $L_i$ be accepted by a DFA $M_i = (Q_i, \Sigma, \delta_i, q_i, F_i)$ with $|Q_i| = n_i$. We describe $M = (Q, \Sigma, \delta, q_0, F)$ such that $L(M) = L_1 \sqcup_T L_2$.

Let $n = |u|$, $m = |v|$ and $s = |w|$. Let $b \notin \Sigma$ be a fixed new letter. We choose

$$Q = Q_1 \times Q_2 \times \{\mathbf{n} \cup \{b\}\} \cup Q_1 \times Q_2 \times \mathbf{m} \times \bigcup_{i=1}^{\lceil \frac{s}{m} \rceil} (Q_1 \times Q_2)^i. \tag{4.3}$$

Further, we let $q_0 = [q_1, q_2, 0] \in Q_1 \times Q_2 \times \mathbf{n}$.

For notational convenience, we define a set of functions $\gamma_{\alpha,\beta,a} : Q_1 \times Q_2 \to Q_1 \times Q_2$ for all $0 \leq \alpha \leq \lceil \frac{s}{m} \rceil - 1$, $0 \leq \beta < m$, $a \in \Sigma$, as follows

$$\gamma_{\alpha,\beta,a}([p_1, p_2]) = \begin{cases} [\delta_1(p_1, a), p_2)] & \text{if } w(m \cdot \alpha + \beta) = 0; \\ [p_1, \delta_2(p_2, a))] & \text{if } w(m \cdot \alpha + \beta) = 1; \end{cases}$$

for all $[p_1, p_2] \in Q_1 \times Q_2$. Further, we let $\gamma'_{\beta,a} : Q_1 \times Q_2 \to Q_1 \times Q_2$ be defined for all $0 \leq \beta < m$ and $a \in \Sigma$ by

$$\gamma'_{\beta,a}([q_i, q_j]) = \begin{cases} [\delta_1(q_i, a), q_j)] & \text{if } v(\beta) = 0; \\ [q_i, \delta_2(q_j, a))] & \text{if } v(\beta) = 1. \end{cases}$$

The full function $\delta$ is given by the following definitions. First, let $[q_i, q_j, k] \in Q_1 \times Q_2 \times \mathbf{n}$. Then,

$$\delta([q_i, q_j, k], a) = \begin{cases} [\delta_1(q_i, a), q_j, k+1] & \text{if} \quad k < n-1 \quad \text{and} \quad u(k) = 0; \\ [q_i, \delta_2(q_j, a), k+1] & \text{if} \quad k < n-1 \quad \text{and} \quad u(k) = 1; \\ [\delta_1(q_i, a), q_j, b] & \text{if} \quad k = n-1 \quad \text{and} \quad u(k) = 0; \\ [q_i, \delta_2(q_j, a), b] & \text{if} \quad k = n-1 \quad \text{and} \quad u(k) = 1. \end{cases} \tag{4.4}$$

If $[q_i, q_j, b] \in Q_1 \times Q_2 \times \{b\}$,

$$\delta([q_i, q_j, b], a) = [\gamma'_{0,a}(q_i, q_j), 1, \gamma_{0,0,a}(q_i, q_j)] \in Q_1 \times Q_2 \times \mathbf{m} \times Q_1 \times Q_2. \tag{4.5}$$

Now we can define $\delta$ on the set $Q_1 \times Q_2 \times \mathbf{m} \times \bigcup_{i=1}^{\lceil \frac{s}{m} \rceil} (Q_1 \times Q_2)^i$. Let $r \leq \lceil \frac{s}{m} \rceil$.

$$\delta([q_i, q_j, k, p_1^{(1)}, p_2^{(1)}, \ldots, p_1^{(r)}, p_2^{(r)}], a)$$

$$= \begin{cases} [\gamma'_{k,a}(q_i, q_j), & k+1, \gamma_{0,k,a}(p_1^{(1)}, p_2^{(1)}), \ldots, \gamma_{r-1,k,a}(p_1^{(r)}, p_2^{(r)})], \\ & \text{if } 0 < k < m-1; \\ [\gamma'_{k,a}(q_i, q_j), & 0, \gamma_{0,k,a}(p_1^{(1)}, p_2^{(1)}), \ldots, \gamma_{r-1,k,a}(p_1^{(r)}, p_2^{(r)})], \\ & \text{if } k = m-1; \\ [(\gamma'_{0,a}(q_i, q_j), & 1, \gamma_{0,k,a}(q_i, q_j), \gamma_{1,k,a}(p_1^{(1)}, p_2^{(1)}), \ldots, \gamma_{r,k,a}(p_1^{(r)}, p_2^{(r)})], \\ & \text{if } k = 0, r < \lceil s/m \rceil; \\ [(\gamma'_{0,a}(q_i, q_j), & 1, \gamma_{0,k,a}(q_i, q_j), \gamma_{1,k,a}(p_1^{(1)}, p_2^{(1)}), \ldots, \gamma_{r-1,k,a}(p_1^{(r-1)}, p_2^{(r-1)})], \\ & \text{if } k = 0, r = \lceil s/m \rceil. \end{cases}$$

The letter $b$ distinguishes the case when we have not read any copies of $v$ or $w$. We need a special letter to indicate this is the situation.

Let $f \in \mathbf{m}$ be chosen so that $f \equiv s \pmod{m}$. With this, we can define $F$ by

$$F = Q_1 \times Q_2 \times f \times (Q_1 \times Q_2)^{\lceil s/m \rceil - 1} \times F_1 \times F_2.$$

Intuitively, we can explain the construction of $M$ as follows. We note that the above parallel branches $[p_1^{(j)}, p_2^{(j)}]$, simulating a computation along $w$, are always separated by exactly $m$ input letters. Thus in a state

$$[q_i, q_j, i, p_1^{(1)}, p_2^{(1)}, \ldots, p_1^{(r)}, p_2^{(r)}], \quad r \leq \lceil \frac{s}{m} \rceil, \tag{4.6}$$

the index $i$ can keep track of the positions also of the $r$ parallel branches along the suffix $w$ of $T$: the $\ell$-th pair is reading the $((\ell - 1) \cdot m + i)$-th letter of $w$.

When the index $i$ goes from $m-1$ to zero, for each $1 \leq j \leq r-1$ the $j$-th pair of states $[p_1^{(j)}, p_2^{(j)}]$ is shifted into the $(j+1)$-st position (at the same time performing the appropriate state transition simulating $M_1$ or $M_2$). The first pair $[p_1^{(1)}, p_2^{(1)}]$ will then be added (based on the states $[q_i, q_j]$) to simulate the new computation that branches out from the loop $v$ and into the suffix $w$.

The $r$-th computation is terminated when it reaches the end of $w$, that is, after $s$ computation steps. Thus, we can have at most $\lceil s/m \rceil$ active computations on the suffix $w$ of the trajectory.

Note that the transition function of $M$ can implicitly code the word $w$ as follows. When applying the transition function to a pair $[p_1^{(j)}, p_2^{(j)}]$, $1 \leq j \leq r$, and knowing the index $i$ (in the notations of (4.6)), the indices $i$ and $j$ exactly specify the position in the word $w$. Thus $M$ knows whether this position in $w$ is a 0 or a 1 and can simulate a computation step of $M_1$ or $M_2$, respectively. This is implied by the definition of the functions $\gamma_{\alpha,\beta,a}$. ■

The following corollary follows easily by induction, noting that

$$L_1 \sqcup\!\sqcup_{T_1 \cup T_2} L_2 = (L_1 \sqcup\!\sqcup_{T_1} L_2) \cup (L_1 \sqcup\!\sqcup_{T_2} L_2).$$

**Corollary 4.3.6** *Let $T \subseteq \{0,1\}^*$ be a slender regular language with USL-index $t$, and write*

$$T = \bigcup_{i=1}^{t} u_i v_i^* w_i.$$

*Then there exists a function $K$, depending only on the integers $|u_i|, |v_i|, |w_i|$, $1 \leq i \leq t$, such that*

$$sc(L_1 \sqcup\!\sqcup_T L_2) \leq K(sc(L_1)sc(L_2))^{t+s}$$

*where*

$$s = \sum_{i=1}^{t} \left\lceil \frac{|w_i|}{|v_i|} \right\rceil.$$

Our aim is to obtain a lower bound for the shuffle operation on trajectories with USL index 1. It seems likely that the bound (4.2) cannot be reached for any fixed set of trajectories (and for all values of $sc(L_i)$,$i = 1, 2$). In particular, if $|w|$ is fixed and $sc(L_i)$ can grow arbitrarily, then it seems impossible that the $\left\lceil \frac{|w|}{|v|} \right\rceil$ parallel computations on the suffix $w$ could simultaneously reach all combinations of states of the DFAs for $L_1$ and $L_2$. Note that if the computation of $M$ contains parallel branches that simulate the computations of $M_i$ ($1 \leq i \leq 2$), in states $P_i \subseteq Q_i$, then all the states of $P_i$ need to be reachable from a single state of $M_i$ with inputs of length at most $|w|$.

For the above reason, we consider a lower bound for sets of trajectories $uv^*w$ where the length of $v$ and of $w$ can depend on the sizes of the minimal DFAs for the component languages $L_1$ and

$L_2$. Furthermore, to simplify the notations below we give lower bound results for sets of trajectories of the form $v^*w$, i.e., $u = \epsilon$. It would be straightforward to modify the construction for prefixes $u$ of arbitrary length to include the additive term $n_1n_2 \cdot (|u| + 1)$ from (4.2).

**Lemma 4.3.7** *Let $\Sigma = \{a, b, c\}$. For any $n_1, n_2 \in \mathbb{N}$ there exist regular languages $L_i \subseteq \Sigma^*$ with $sc(L_i) = n_i$, $i = 1, 2$, and a set of trajectories $T = v^*w$, where $v, w \in \{0, 1\}^*$, such that*

$$sc(L_1 \sqcup\!\sqcup_T L_2) \geq (n_1n_2)^{\lceil \frac{|w|}{|v|} \rceil + 1}.$$

*The ratio $|w|/|v|$ above can be chosen to be arbitrarily large.*

**Proof.** Let $L_1$, $L_2$ be defined as $L_1 = \{w \in \Sigma^* : |w|_a \equiv 0 \pmod{n_1}\}$ and, $L_2 = \{w \in \Sigma^* : |w|_b \equiv 0 \pmod{n_2}\}$. Clearly $\mathrm{sc}(L_i) = n_i$, $i = 1, 2$. Denote

$$n = \max(n_1, n_2) - 1 \text{ and } m = 2n.$$

For the set of trajectories we choose

$$T = ((01)^m)^*(10)^{mk}, \quad k \geq 1. \tag{4.7}$$

Note that $sc(T) = 2m(k + 1)$. Define $L = L_1 \sqcup\!\sqcup_T L_2$. The set $S \subseteq \Sigma^{(2k+1)m}$ is defined to consist of all words

$$\begin{aligned} S = \{&w_1 \cdots w_{k+1} \ : \\ &w_i \in \{a, c\}^m \sqcup\!\sqcup_p \{b, c\}^m, 1 \leq i \leq k, \ w_{k+1} \in \{a, c\}^n \sqcup\!\sqcup_p \{b, c\}^n\}. \end{aligned} \tag{4.8}$$

If $w \in S$, then we denote by $w_1, w_2, \ldots, w_{k+1}$ the unique components of $w$ as described by (4.8).

For $w \in S$ and $1 \leq i \leq k + 1$, we define the following quantities

$$A(w, a, i) = (\sum_{j=1}^{i} |w_j|_a) \bmod n_1, \qquad A(w, b, i) = (\sum_{j=1}^{i} |w_j|_b) \bmod n_2.$$

**Claim 4.3.8** *Let $w, w' \in S$. If there exists $1 \leq i \leq k + 1$ such that*

$$[A(w, a, i), A(w, b, i)] \neq [A(w', a, i), A(w', b, i)] \tag{4.9}$$

*then $w \not\equiv_L w'$.*

**Proof.** Assume $i$ exists such that (4.9) holds and let $x_i \in \mathbf{n_i}$, $i = 1, 2$, be the integers such that

$$x_1 \equiv -A(w, a, i) \pmod{n_1} \text{ and } x_2 \equiv -A(w, b, i) \pmod{n_2}.$$

Choose

$$u_i = \begin{cases} ((b^{x_2}c^{n-x_2}) \sqcup\!\sqcup_p (a^{x_1}c^{n-x_1}))c^{2m(i-1)} & \text{if } i \leq k, \\ ((a^{x_1}c^{n-x_1}) \sqcup\!\sqcup_p (b^{x_2}c^{n-x_2}))c^{2mk} & \text{if } i = k+1. \end{cases}$$

To establish our claim it is sufficient to show that

$$wu_i \in L \text{ and } w'u_i \notin L. \tag{4.10}$$

Let $w = w_1 \cdots w_{k+1}$, $w' = w'_1 \cdots w'_{k+1} \in S$ be such that (4.9) holds for some index $i$. For each $1 \leq j \leq k$, let $w_j = \Omega_j \sqcup\!\sqcup_p \Pi_j$ and $w'_j = \Omega'_j \sqcup\!\sqcup_p \Pi'_j$ where $\Omega_j, \Omega'_j \in \{a, c\}^m$, $\Pi_j, \Pi'_j \in \{b, c\}^m$, and let $w_{k+1} = \Omega_{k+1} \sqcup\!\sqcup_p \Pi_{k+1}$ and $w'_{k+1} = \Omega'_{k+1} \sqcup\!\sqcup_p \Pi'_{k+1}$ where $\Omega_{k+1}, \Omega'_{k+1} \in \{a, c\}^n$, $\Pi_{k+1}, \Pi'_{k+1} \in \{b, c\}^n$.

(i) First we consider the case where $i \leq k$. Now $|wu_i| = |w'u_i| = 2m(k+i)$, so the only possible trajectory $t \in T$ which could correspond to these words is $t = (01)^{m \cdot i}(10)^{m \cdot k}$. Let $t = t_1 t_2$ where $t_1 = (01)^{m \cdot i}$ and $t_2 = (10)^{m \cdot k}$. Let $\alpha, \alpha', \beta, \beta'$ be the unique words such that $\alpha \sqcup\!\sqcup_t \beta = wu_i$ and $\alpha' \sqcup\!\sqcup_t \beta' = w'u_i$. In particular, let $\alpha = \alpha_1\alpha_2$, $\alpha' = \alpha'_1\alpha'_2$, $\beta = \beta_1\beta_2$ and $\beta' = \beta'_1\beta'_2$ such that

$$\begin{aligned} w_1 \cdots w_i &= \alpha_1 \sqcup\!\sqcup_{t_1} \beta_1; \\ w'_1 \cdots w'_i &= \alpha'_1 \sqcup\!\sqcup_{t_1} \beta'_1; \\ w_{i+1} \cdots w_{k+1}u_i &= \alpha_2 \sqcup\!\sqcup_{t_2} \beta_2; \\ w'_{i+1} \cdots w'_{k+1}u_i &= \alpha'_2 \sqcup\!\sqcup_{t_2} \beta'_2. \end{aligned}$$

Then note that necessarily

$$\begin{aligned} \alpha_1 &= \Omega_1\Omega_2 \cdots \Omega_i; \\ \beta_1 &= \Pi_1\Pi_2 \cdots \Pi_i; \\ \alpha'_1 &= \Omega'_1\Omega'_2 \cdots \Omega'_i; \\ \beta'_1 &= \Pi'_1\Pi'_2 \cdots \Pi'_i. \end{aligned}$$

and

$$\begin{aligned}
\beta_2 &= \Omega_{i+1}\Omega_{i+2}\cdots\Omega_{k+1}b^{x_2}c^{n-x_2}c^{m(i-1)};\\
\alpha_2 &= \Pi_{i+1}\Pi_{i+2}\cdots\Pi_{k+1}a^{x_1}c^{n-x_1}c^{m(i-1)};\\
\beta_2' &= \Omega'_{i+1}\Omega'_{i+2}\cdots\Omega'_{k+1}b^{x_2}c^{n-x_2}c^{m(i-1)};\\
\alpha_2' &= \Pi'_{i+1}\Pi'_{i+2}\cdots\Pi'_{k+1}a^{x_1}c^{n-x_1}c^{m(i-1)}.
\end{aligned}$$

Thus, we can now easily compute $|\alpha|_a$, $|\alpha'|_a$, $|\beta|_b$, $|\beta'|_b$.

$$\begin{aligned}
|\alpha|_a &= |\alpha_1|_a + |\alpha_2|_a\\
&= A(w,a,i) + |\alpha_2|_a\\
&= A(w,a,i) + |\Pi_{i+1}\Pi_{i+2}\cdots\Pi_{k+1}|_a + x_1\\
&= A(w,a,i) + x_1 \equiv 0 \pmod{n_1}.
\end{aligned}$$

as $\Pi_j \in \{b,c\}^*$ and $x_1 \equiv -A(w,a,i) \pmod{n_1}$. An identical analysis yields that $|\alpha'|_a \equiv A(w',a,i) - A(w,a,i) \pmod{n_1}$. We can similarly examine $\beta$ and $\beta'$, to give

$$\begin{aligned}
|\beta|_b &\equiv 0 \pmod{n_2}\\
|\beta'|_b &\equiv A(w',b,i) - A(w,b,i) \pmod{n_2}
\end{aligned}$$

The congruences $|\alpha|_a \equiv 0 \pmod{n_1}$, $|\beta|_b \equiv 0 \pmod{n_2}$ give $wu_i \in L$. By (4.9), we conclude that one of $\alpha' \notin L_1$, $\beta' \notin L_2$ holds, and thus $w'u_i \notin L$.

(ii) Second we consider the case $i = k+1$. Now $|wu_i| = |w'u_i| = 2m(2k+1)$, so the corresponding trajectory is $t = t_1t_2$ where $t_1 = (01)^{m(k+1)}$ and $t_2 = (10)^{m\cdot k}$. In this case recall that $u_{k+1} = ((a^{x_1}c^{n-x_1}) \sqcup\!\sqcup_p (b^{x_2}c^{n-x_2}))c^{2mk}$, and the suffix $c^{2mk}$ of $u_{k+1}$ corresponds exactly to the suffix $t_2$ of the trajectory. Thus when the word $wu_i$ (respectively, $w'u_i$) is written in the form $\alpha \sqcup\!\sqcup_t \beta$ (respectively, $\alpha' \sqcup\!\sqcup_t \beta'$) all letters $a$ in the word correspond to ("come from") the component in $L_1$ and all letters $b$ correspond to the component in $L_2$. By (4.9), we conclude that $\alpha \in L_1$ and $\beta \in L_2$ but necessarily one of $\alpha' \notin L_1$ or $\beta' \notin L_2$ holds. The completes the proof that (4.10) holds. ■

We now continue with the proof of Lemma 4.3.7. We claim that the map $\varphi : S \to (\mathbf{n_1} \times \mathbf{n_2})^{k+1}$, given by

$$w \mapsto [A(w, a, i), A(w, b, i)]_{i=1}^{k+1}, \tag{4.11}$$

is surjective. To see this, note that if $w \in S$ then $A(w, a, i)$ and $A(w, b, i)$ depend only on the subwords $w_1, \ldots, w_i$. Thus after $w_1, \ldots, w_i$ are chosen we can always select an arbitrary value for $[A(w, a, i+1), A(w, b, i+1)]$ since $[|w_{i+1}|_a, |w_{i+1}|_b]$ can have any value in $\mathbf{n_1} \times \mathbf{n_2}$. (This holds also in the case $i = k$.) Thus, $\varphi$ is surjective, and by Claim 4.3.8, for distinct $z, z' \in (\mathbf{n_1} \times \mathbf{n_2})^{k+1}$, the sets $\varphi^{-1}(z)$ and $\varphi^{-1}(z')$ lie in different equivalence classes of $\equiv_L$. Thus, $sc(L) \geq (n_1 n_2)^{k+1}$. ■

In the notations of Lemma 4.3.7, the upper bound (4.2) is of the order $|v| \cdot (n_1 n_2)^{\lceil \frac{|w|}{|v|} \rceil + 1}$ where $|v|$ can be chosen as a constant times $\max(n_1, n_2)$. In the proof of Lemma 4.3.7 we counted only equivalence classes of $\equiv_L$ that had representatives of length $(2k+1)m$. Using the same construction we can get an improved lower bound by taking into account also equivalence classes with representatives of different lengths. This bound approaches the upper bound when $|v|$ grows compared to $sc(L_i), i = 1, 2$.

**Lemma 4.3.9** *Let $\Sigma = \{a, b, c\}$. Let $n_1, n_2 \in \mathbb{N}$ be arbitrary and $n = \max(n_1, n_2) - 1$. There exist regular languages $L_i \subseteq \Sigma^*$ with $sc(L_i) = n_i$, $i = 1, 2$, and a set of trajectories $T = v^* w$, where $v, w \in \{0, 1\}^*$, $|v| \geq 4n$, such that*

$$sc(L_1 \sqcup\!\sqcup_T L_2) \geq (|v| - 4n + 1)(n_1 n_2)^{\lceil \frac{|w|}{|v|} \rceil + 1} + |v| \frac{(n_1 n_2)^{\lceil \frac{|w|}{|v|} \rceil - 1} - 1}{n_1 n_2 - 1}. \tag{4.12}$$

*The quantity $|v|$ and the ratio $|w|/|v|$ above can be chosen to be arbitrarily large compared to $sc(L_1)$ and $sc(L_2)$.*

**Proof.** We use the notations from the proof of Lemma 4.3.7 with the only change that $m \geq 2n$ can be arbitrary (instead of $m = 2n$).

For a word $w$ with $|w| \leq 2m(k+1)$ and $w = w_1 \cdots w_{i-1} w_i$ where $|w_j| = 2m$, $j = 1, \ldots, i-1$, $0 \leq |w_i| \leq 2m$, we say that the $j$th component of $w$ is $w_j$, $j = 1, \ldots, i$.

Since $T$ consists of only words with length a multiple of $2m$, it follows that for any $w, w' \in \Sigma^*$ if $|w| \not\equiv |w'| \pmod{2m}$ then $w \not\equiv_L w'$. Note that any word in $\Sigma^*$ can be completed to a word in $L$ by adding a suitable suffix. On the other hand, if $w, w' \in S$ and $w \not\equiv_L w'$ then

$$wc^j \not\equiv_L w'c^j \text{ for all } 1 \leq j \leq 2m - 4n. \tag{4.13}$$

Note that $\varphi(w) \neq \varphi(w')$ and the suffix $c^j$ does not change the numbers of occurrences of $a$'s and $b$'s in the $(k+1)$-st component. Furthermore, we can always find a word $u$ of length $2m-2n-j\,(\geq 2n)$ such that $wc^ju \in L$ (or $w'c^ju \in L$) which may be needed to establish the inequivalence of $wc^j$ and $w'c^j$ if $\varphi(w)$ and $\varphi(w')$ differ only in their first component.

Since we know that $S$ contains $(n_1n_2)^{\lceil \frac{|w|}{|v|} \rceil + 1}$ pairwise inequivalent words, the above observations give us $(2m - 4n + 1)(n_1n_2)^{\lceil \frac{|w|}{|v|} \rceil + 1}$ equivalence classes which is the first term of (4.12).

Let $S_i$, $1 \leq i \leq k$, denote the set of prefixes of $S$ having length $2mi$. Similarly as in the proof of Lemma 4.3.7, we see that $S_i$ contains representatives of $(n_1n_2)^i$ distinct equivalence classes of $\equiv_L$. Using a similar argument as above for (4.13) we see that if $w, w' \in S_i$, $i < k$, and $w \not\equiv_L w'$ then $wc^j \not\equiv_L w'c^j$ for all $1 \leq j < 2m$. Note that since $i < k$ the suffix $c^j$ does not belong to the $(k+1)$-st component and it can have any length up to $2m$. Furthermore, each word

$$wc^j, \quad w \in S_i,\ i \leq k-2,\ 0 \leq j < 2m \tag{4.14}$$

can be completed to a word in $L$ using a suffix of length $2m(k-i) - j$ and not by any suffix of shorter length. Thus any two words of different length as in (4.14) cannot be equivalent, and any word as in (4.14) cannot be equivalent to any word as in (4.13). This yields

$$2m \sum_{i=0}^{k-2} (n_1n_2)^i$$

equivalence classes which is the last term of (4.12). ■

As a consequence of Lemma 4.3.9 we have:

**Theorem 4.3.10** *The upper bound (4.2) is asymptotically optimal if $sc(T)$ (that is, $|v|$) can be arbitrarily large compared to $sc(L_i)$, $i = 1, 2$.*

In comparing Theorem 4.3.10 and Lemma 4.3.3, we note that Lemma 4.3.3 is a tighter bound, and is better than Theorem 4.3.10 in the restricted sense that Lemma 4.3.3 takes a set of trajectories (albeit a very specific and fixed set of trajectories) and defines languages for which we have matching upper and lower bounds. This is subtly different from Theorem 4.3.10, which takes languages and defines a set of trajectories for which the upper bound is obtained. The reasoning for this, we recall, is discussed prior to the statement of Lemma 4.3.7.

## 4.4 Future Directions

### 4.4.1 Polynomial Density Trajectories

We may also consider the case of polynomial-density sets of trajectories, i.e., sets of trajectories $T$ with $p_T(n) \in O(n^k)$ for $k \geq 1$, by extending the ideas of Lemma 4.3.5. We can employ nondeterministic state complexity when it is to our advantage. However, the upper bound which we obtain is not much better than the bound of Lemma 4.2.2. We note that an extension to linear density sets of trajectories would encompass the case of $T = 0^*1^*$. By Theorem 4.2.4, we know that this linear density bound would not be as good an improvement over Lemma 4.2.2 compared to, e.g., Corollary 4.3.6.

### 4.4.2 Exponential Density Trajectories

Recall that the example of arbitrary shuffle, shown by Câmpeanu *et al.* to have state complexity no better than our construction in Lemma 4.2.2, uses the set of trajectories $T = (0 + 1)^*$ of density $2^n$. We also note that, by Szilard *et al.* [190], the density of a regular language over $\Sigma$ is either $O(p(n))$, where $p$ is a polynomial, or $\Omega(|\Sigma|^n)$.

Thus, we may conjecture that a set of trajectories $T$ yields an operation which is, in the worst case, no better than Lemma 4.2.2 only in the case when $p_T(n) \in \Omega(2^n)$, i.e. $T$ has exponential density.

### 4.4.3 Other Open Problems

Our constructions in Lemmas 4.3.7 and 4.3.9 use three-letter alphabets. Can these constructions be improved to two-letter alphabets? The problem of restricting the alphabet size to be as small as possible is often challenging. For example, in the case of concatenation, the state complexity problem was solved for a three-letter alphabet by Yu *et al.* [204], but the case of a two-letter alphabet was open until very recently [95, 94].

## 4.5 Conclusions

In this chapter we have examined the state complexity of shuffle on trajectories. This area has been previously examined by Câmpeanu *et al.* [22] for the case of $T = \{0, 1\}^*$, and by Yu *et al.* [204] for the case of $T = 0^*1^*$. In this chapter, we have considered state complexity of arbitrary shuffle on trajectories.

We have also considered the specific case where the set $T$ of trajectories is slender, i.e., contains only a constant number of words of each length. In this case, we have shown that shuffle on the set $T$ of trajectories has a considerably lower state complexity than in the case of a general $T \subseteq \{0, 1\}^*$.

# Chapter 5

# Deletion along Trajectories

## 5.1 Introduction

As we have seen, shuffle on trajectories is a powerful method for unifying operations which insert all the letters of one word into another. Concurrent to this research, Kari and others [106, 117] have done research into the inverses of insertion- and shuffle-like operations, which have yielded decidability results for language equations such as $XL = R$ where $L$, $R$ are regular languages and $X$ is unknown. The inverses of insertion- and shuffle-like operations are deletion-like operations such as *deletion*, *quotient*, *scattered deletion* and *bi-polar deletion* [106].

In this chapter, we introduce the notion of *deletion along trajectories*, which is the analogous notion to shuffle on trajectories for deletion-like operations. We show how it unifies operations such as deletion, quotient, scattered deletion and others. We investigate the closure properties of deletion along trajectories. We also show how each shuffle operation based on a set of trajectories $T$ has an inverse operation (both right and left inverse, see Section 5.8), defined by a deletion along a renaming of $T$. This yields the result that it is decidable whether language equations of the form $L \sqcup\!\sqcup_T X = R$ for regular languages $L$ and $R$ have a solution $X$, for any regular set $T$ of trajectories.

We also investigate those $T$ which are not regular but for which the deletion along the set of trajectories $T$ preserves regularity. Theorems 5.4.1 and 5.4.2 explicitly define classes of sets of

trajectories, which include non-regular sets, which preserve regularity.

## 5.2 Definitions

We now give the main definition of this chapter, called *deletion along trajectories*, which models deletion operations controlled by a set of trajectories. Let $x, y \in \Sigma^*$ be words with $x = ax'$, $y = by'$ $(a, b \in \Sigma)$. Let $t$ be a word over $\{i, d\}$ such that $t = et'$ with $e \in \{i, d\}$. Then we define $x \rightsquigarrow_t y$, the deletion of $y$ from $x$ along trajectory $t$, as follows:

$$x \rightsquigarrow_t y = \begin{cases} a(x' \rightsquigarrow_{t'} by') & \text{if } e = i; \\ x' \rightsquigarrow_{t'} y' & \text{if } e = d \text{ and } a = b; \\ \emptyset & \text{otherwise.} \end{cases}$$

Also, if $x = ax'$ $(a \in \Sigma)$ and $t = et'$ $(e \in \{i, d\})$, then

$$x \rightsquigarrow_t \epsilon = \begin{cases} a(x' \rightsquigarrow_{t'} \epsilon) & \text{if } e = i; \\ \emptyset & \text{otherwise.} \end{cases}$$

If $x \neq \epsilon$, then $x \rightsquigarrow_\epsilon y = \emptyset$. Further, $\epsilon \rightsquigarrow_t y = \epsilon$ if $t = y = \epsilon$. Otherwise, $\epsilon \rightsquigarrow_t y = \emptyset$.

**Example 5.2.1:** Let $x = abcabc$, $y = bac$ and $t = (id)^3$. Then we have that $x \rightsquigarrow_t y = acb$. If $t = i^2d^3i$ then $x \rightsquigarrow_t y = \emptyset$. □

Let $T \subseteq \{i, d\}^*$. Then

$$x \rightsquigarrow_T y = \bigcup_{t \in T} x \rightsquigarrow_t y.$$

We extend this to languages as expected: Let $L_1, L_2 \subseteq \Sigma^*$ and $T \subseteq \{i, d\}^*$. Then

$$L_1 \rightsquigarrow_T L_2 = \bigcup_{\substack{x \in L_1 \\ y \in L_2}} x \rightsquigarrow_T y.$$

Note that $\rightsquigarrow_T$ is neither an associative nor a commutative operation on languages, in general. We consider the following examples of deletion along trajectories (for any operations not defined, we refer the reader to the appropriate paper cited below):

(a) if $T = i^*d^*$, then $\rightsquigarrow_T = /$, the right-quotient operation;

(b) if $T = d^*i^*$, then $\rightsquigarrow_T = \backslash$, the left-quotient operation;

(c) if $T = i^*d^*i^*$, then $\rightsquigarrow_T = \rightarrow$, the deletion operation (see, e.g., Kari [103, 106]);

(d) if $T = (i + d)^*$, then $\rightsquigarrow_T = \rightsquigarrow$, the scattered deletion operation (see, e.g., Ito *et al.* [79]);

(e) if $T = d^*i^*d^*$, then $\rightsquigarrow_T = \rightleftharpoons$, the bi-polar deletion operation (see, e.g., Kari [106]);

(f) let $k \geq 0$ and $T_k = i^*d^*i^{\leq k}$. Then $\rightsquigarrow_{T_k} = \rightarrow^k$, the $k$-deletion operation (see, e.g., Kari and Thierrin [114]).

Also, we note the difference between deletion along trajectories from the operation *splicing on routes* defined by Mateescu [144], which is a generalization of shuffle on trajectories which allows discarding letters from either input word. Splicing on routes serves to generalize the *crossover operation* used in DNA computing by restricting the manner in which it may combine letters, in a manner similar to how shuffle on trajectories restricts the way in which the shuffle operator may combine letters (see Mateescu [144] for details and a definition of the crossover operation).

## 5.3 Closure and Characterization Results

The following lemma is proven by a direct construction:

**Lemma 5.3.1** *If $T, L_1, L_2$ are regular, then $L_1 \rightsquigarrow_T L_2$ is also regular.*

**Proof.** Let $M_1, M_2, M_T$ be DFAs for $L_1, L_2, T$, respectively, with

$$\begin{aligned} M_j &= (Q_j, \Sigma, \delta_j, q_j, F_j), \quad \text{for } j = 1, 2, \text{ and} \\ M_T &= (Q_T, \{i, d\}, \delta_T, q_T, F_T). \end{aligned}$$

Let $M = (Q_1 \times Q_2 \times Q_T, \Sigma, \delta, [q_1, q_2, q_T], F_1 \times F_2 \times F_T)$ be an NFA with $\delta$ given by

$$\delta([q_j, q_k, q_\ell], a) = \{[\delta_1(q_j, a), q_k, \delta_T(q_\ell, i)]\}$$

for all $[q_j, q_k, q_\ell] \in Q_1 \times Q_2 \times Q_T$ and $a \in \Sigma$. Further,

$$\delta([q_j, q_k, q_\ell], \epsilon) = \{[\delta_1(q_j, a), \delta_2(q_k, a), \delta_T(q_\ell, d)] \ : \ a \in \Sigma\}$$

for all $[q_j, q_k, q_\ell] \in Q_1 \times Q_2 \times Q_T$. We can verify that $M$ accepts $L_1 \rightsquigarrow_T L_2$. ■

We now show that if any one of $L_1$, $L_2$ or $T$ is non-regular, then $L_1 \rightsquigarrow_T L_2$ may not be regular:

**Theorem 5.3.2** *There exist languages $L_1, L_2$ and a set of trajectories $T \subseteq \{i, d\}^*$ satisfying each of the following:*

*(a) $L_1$ is a CFL, $L_2$ is a singleton and $T$ is regular, but $L_1 \rightsquigarrow_T L_2$ is not regular;*

*(b) $L_1, T$ are regular, and $L_2$ is a CFL, but $L_1 \rightsquigarrow_T L_2$ is not regular;*

*(c) $L_1$ is regular, $L_2$ is a singleton, and $T$ is a CFL, but $L_1 \rightsquigarrow_T L_2$ is not regular.*

*In each case, the CFL may be chosen to be an LCFL.*

**Proof.** We first note the following identity:

$$L \rightsquigarrow_{i^*} \{\epsilon\} = L.$$

Thus, if we take any non-regular (linear) CFL $L$, we can establish (a).

For (b), we take the following languages:

$$\begin{aligned} L_1 &= (a^2)^*(b^2)^*, \\ T &= (di)^*, \\ L_2 &= \{a^n b^n \ : \ n \geq 0\}. \end{aligned}$$

Note that $L_2$ is a non-regular (linear) CFL. With these languages, we have that $L_1 \rightsquigarrow_T L_2 = L_2$. Finally, to establish part (c), we take

$$\begin{aligned} L_1 &= a^*\#b^*, \\ T &= \{i^n d i^n \ : \ n \geq 0\}, \\ L_2 &= \{\#\}. \end{aligned}$$

We note that $T$ is a non-regular linear CFL, and that

$$L_1 \rightsquigarrow_T L_2 = \{a^n b^n \ : \ n \geq 0\}.$$

This establishes the theorem. ■

In Section 5.4, we discuss non-regular sets of trajectories which preserve regularity. Recall that a *weak coding* is a morphism $\pi : \Sigma^* \to \Delta^*$ such that $\pi(a) \in \Delta \cup \{\epsilon\}$ for all $a \in \Sigma$. We have the following characterization of deletion along trajectories:

**Theorem 5.3.3** *Let $\Sigma$ be an alphabet. There exist weak codings $\rho_1, \rho_2, \tau, \varphi$ and a regular language $R$ such that for all $L_1, L_2 \subseteq \Sigma^*$ and all $T \subseteq \{i, d\}^*$,*

$$L_1 \rightsquigarrow_T L_2 = \varphi\left(\rho_1^{-1}(L_1) \cap \rho_2^{-1}(L_2) \cap \tau^{-1}(T) \cap R\right).$$

**Proof.** Let $\hat{\Sigma} = \{\hat{a} \ : \ a \in \Sigma\}$ be a copy of $\Sigma$. Define the morphism $\rho_1 : (\hat{\Sigma} \cup \Sigma \cup \{i, d\})^* \to \Sigma^*$ as follows: $\rho_1(\hat{a}) = \rho_1(a) = a$ for all $a \in \Sigma$ and $\rho_1(i) = \rho_1(d) = \epsilon$. Define $\rho_2 : (\hat{\Sigma} \cup \Sigma \cup \{i, d\})^* \to \Sigma^*$ as follows: $\rho_2(\hat{a}) = a$ for all $a \in \Sigma$, $\rho_2(a) = \epsilon$ for all $a \in \Sigma$ and $\rho_2(d) = \rho_2(i) = \epsilon$.

Define $\tau : (\hat{\Sigma} \cup \Sigma \cup \{i, d\})^* \to \{i, d\}^*$ as follows: $\tau(\hat{a}) = \tau(a) = \epsilon$ for all $a \in \Sigma$, $\tau(i) = i$ and $\tau(d) = d$. We define $\varphi : (\hat{\Sigma} \cup \Sigma \cup \{i, d\})^* \to \Sigma^*$ as $\varphi(\hat{a}) = \epsilon$ for all $a \in \Sigma$, $\varphi(a) = a$ for all $a \in \Sigma$, and $\varphi(i) = \varphi(d) = \epsilon$. Finally, we note that the result can be proven by letting $R = (i\,\Sigma + d\,\hat{\Sigma})^*$. ■

Thus, we have the following corollary:

**Corollary 5.3.4** *Let $\mathcal{C}$ be a cone. Let $L_1, L_2, T$ be languages such that two are regular and the third is in $\mathcal{C}$. Then $L_1 \rightsquigarrow_T L_2 \in \mathcal{C}$.*

Note that the closure of cones under quotient with regular sets [68, Thm. 11.3] is a specific instance of Corollary 5.3.4. Lemma 5.3.1 can also be proven by appealing to Theorem 5.3.3. We also note that the CFLs are a cone, thus we have the following corollary (a direct construction is also possible):

**Corollary 5.3.5** *Let $T, L_1, L_2$ be languages such that one is a CFL and the other two are regular languages. Then $L_1 \rightsquigarrow_T L_2$ is a CFL.*

The following result shows that if any of the conditions of Corollary 5.3.5 are not met, the result might not hold:

**Theorem 5.3.6** *There exist languages $L_1, L_2$ and a set of trajectories $T \subseteq \{i, d\}^*$ satisfying each of the following:*

*(a) $L_1, L_2$ are (linear) CFLs and $T$ is regular, but $L_1 \rightsquigarrow_T L_2$ is not a CFL;*

*(b) $L_1, T$ are (linear) CFLs, and $L_2$ is a singleton, but $L_1 \rightsquigarrow_T L_2$ is not a CFL;*

*(c) $L_1$ is regular, $L_2, T$ are (linear) CFLs, but $L_1 \rightsquigarrow_T L_2$ is not a CFL.*

**Proof.** (a) The result is immediate, since it is known (see, e.g., Ginsburg and Spanier [52, Thm. 3.4]) that the CFLs are not closed under right quotient (given by the set of trajectories $T = i^*d^*$). The languages described by Ginsburg and Spanier which witness this non-closure are linear CFLs.

(b) Let $\Sigma = \{a, b, c, \#\}$ and define $L_1, L_2 \subseteq \Sigma^*$ and $T \subseteq \{i, d\}^*$ by

$$\begin{aligned} L_1 &= \{a^n b^n \# c^m \ : \ n, m \geq 0\}; \\ L_2 &= \{\#\}; \\ T &= \{i^{2n} d i^n \ : \ n \geq 0\}. \end{aligned}$$

Note that $L_1, T$ are indeed linear CFLs. Then we can verify that

$$L_1 \rightsquigarrow_T L_2 = \{a^n b^n c^n \ : \ n \geq 0\},$$

which is not a CFL.

(c) Let $\Sigma = \{a, b, c, \#\}$. Then let

$$\begin{aligned} L_1 &= (a^2)^*(b^2)^* \# c^*; \\ L_2 &= \{a^n b^n \# \ : \ n \geq 0\}; \\ T &= \{(di)^{2n} d i^n \ : \ n \geq 0\}. \end{aligned}$$

Then we can verify that $L_1 \rightsquigarrow_T L_2 = \{a^n b^n c^n \ : \ n \geq 0\}$, which is not a CFL. This completes the proof. ■

Note that the CSLs are not a cone, since it is known that they are not closed under arbitrary morphism (see, e.g., Mateescu and Salomaa [148, Thm. 2.12] for the closure properties of the CSLs). Thus, Corollary 5.3.4 does not apply to the CSLs. In fact, it is also known that the CSLs are not closed under (left or right) quotient with regular languages.

### 5.3.1 Recognizing Deletion Along Trajectories

We now consider the problem of giving a monoid recognizing deletion along trajectories, when the languages and set of trajectories under consideration are regular. Harju *et al.* [61] give a monoid which recognizes $L_1 \sqcup\!\sqcup_T L_2$ when $L_1$, $L_2$ and $T$ are regular.

For a background on recognition of formal languages by monoids, please consult Pin [164]. A *monoid* is a semigroup with unit element. Let $L \subseteq \Sigma^*$ be a language. We say that a monoid $M$ recognizes $L$ if there exists a morphism $\varphi : \Sigma^* \to M$ and a subset $F \subseteq M$ such that $L = \varphi^{-1}(F)$.

The following is a characterization of the regular languages due to Kleene (see, e.g., Pin [164, p. 17]):

**Theorem 5.3.7** *A language is regular if and only if it is recognized by a finite monoid.*

Consider arbitrary regular languages $L_1, L_2 \subseteq \Sigma^*$ and $T \subseteq \{i, d\}^*$. Then our goal is to construct a monoid recognizing $L_1 \rightsquigarrow_T L_2$.

Let $M_1, M_2, M_T$ be finite monoids recognizing $L_1, L_2, L_T$, with morphisms $\varphi_j : \Sigma^* \to M_j$ for $j = 1, 2$, $\varphi_T : \{i, d\}^* \to M_T$ and subsets $F_1, F_2, F_T$, respectively.

As in Harju *et al.* [61], we consider the monoid $\mathcal{P}(M_1 \times M_2 \times M_T)$ consisting of all subsets of $M_1 \times M_2 \times M_T$. The monoid operation is given by $AB = \{xy \ : \ x \in A, y \in B\}$ for all $A, B \in \mathcal{P}(M_1 \times M_2 \times M_T)$, and the product of elements of $M_1 \times M_2 \times M_T$ is defined componentwise.

We can now establish that $\mathcal{P}(M_1 \times M_2 \times M_T)$ recognizes $L_1 \rightsquigarrow_T L_2$. We first define a subset $D \subseteq M_1 \times M_2 \times M_T$ which will be useful:

$$D = \{[\varphi_1(x), \varphi_2(x), \varphi_T(d^{|x|})] \ : \ x \in \Sigma^*\}.$$

Then we define $\varphi : \Sigma^* \to \mathcal{P}(M_1 \times M_2 \times M_T)$ by giving its action on each element $a \in \Sigma$:

$$\varphi(a) = \{[\varphi_1(xa), \varphi_2(x), \varphi_T(d^{|x|}i)] \ : \ x \in \Sigma^*\}.$$

Then, we note that for all $y \in \Sigma^*$,

$$\varphi(y)D = \{[\varphi_1(\alpha), \varphi_2(\beta), \varphi_T(t)] \ : \ y \in \alpha \rightsquigarrow_t \beta, \alpha, \beta \in \Sigma^*, t \in \{i, d\}^*\}. \tag{5.1}$$

Thus, it suffices to take

$$F = \{K \in \mathcal{P}(M_1 \times M_2 \times M_T) \ : \ KD \cap (F_1 \times F_2 \times F_T) \neq \emptyset\}.$$

Thus, considering (5.1), we have that

$$L_1 \rightsquigarrow_T L_2 = \varphi^{-1}(F).$$

This establishes the following result:

**Lemma 5.3.8** *Let $L_j$ be a regular language recognized by $M_j$ for $j = 1, 2$ and $T \subseteq \{i, d\}^*$ be a regular set of trajectories recognized by the monoid $M_T$. Then $\mathcal{P}(M_1 \times M_2 \times M_T)$ recognizes $L_1 \rightsquigarrow_T L_2$.*

Thus, Lemma 5.3.8 gives another proof of Lemma 5.3.1.

### 5.3.2 Equivalence of Trajectories

We briefly note that two sets of trajectories over $\{i, d\}$ define the same deletion operation if and only if they are equal. More precisely, if $T_1, T_2 \subseteq \{i, d\}^*$, say that $T_1$ and $T_2$ are *equivalent* if $L_1 \rightsquigarrow_{T_1} L_2 = L_1 \rightsquigarrow_{T_2} L_2$ for all languages $L_1, L_2$.

**Lemma 5.3.9** *Let $T_1, T_2 \subseteq \{i, d\}^*$. Then $T_1$ and $T_2$ are equivalent if and only if $T_1 = T_2$.*

**Proof.** If $T_1 = T_2$ then clearly $T_1$ and $T_2$ are equivalent. If $T_1$ and $T_2$ are not equal, then without loss of generality, let $t \in T_1 - T_2$. Let $n = |t|_i$ and $m = |t|_d$. Then it is not hard to see that $i^n \in \{t\} \rightsquigarrow_{T_1} \{d^m\}$, but that $i^n \notin \{t\} \rightsquigarrow_{T_2} \{d^m\}$, i.e., $T_1$ and $T_2$ are not equivalent. ■

Thus, the decidability of the equivalence problem for $T_1, T_2 \subseteq \{i, d\}^*$ is well known. For instance, it is decidable whether $T_1, T_2$ are equivalent if, e.g., $T_1, T_2$ are DCFLs, but undecidable if $T_1$ is regular and $T_2$ is an arbitrary CFL.

## 5.4 Regularity-Preserving Sets of Trajectories

Consider the following result of Mateescu *et al.* [147, Thm. 5.1]: if $L_1 \sqcup\!\sqcup_T L_2$ is regular for all regular languages $L_1, L_2$, then $T$ is regular. This result is clear upon noting that for all $T$, $0^* \sqcup\!\sqcup_T 1^* = T$.

However, in this section, we note that the same result does not hold if we replace "shuffle on trajectories" by "deletion along trajectories". In particular, we demonstrate a class of sets of trajectories $\mathcal{H}$, which contains non-regular languages, such that for all regular languages $R_1, R_2$, and for all $H \in \mathcal{H}$, $R_1 \rightsquigarrow_H R_2$ is regular. We also characterize all $H \subseteq i^*d^*$ which preserve regularity (i.e., such that $R_1 \rightsquigarrow_H R_2$ is regular for all regular languages $R_1, R_2$), and give some examples of non-CF trajectories which preserve regularity.

As motivation, we begin with a basic example. Let $\Sigma$ be an alphabet. Let $H = \{i^n d^n : n \geq 0\}$. Note that

$$R_1 \rightsquigarrow_H R_2 = \{x \in \Sigma^* : \exists y \in R_2 \text{ such that } xy \in R_1 \text{ and } |x| = |y|\}.$$

We can establish directly (by constructing an NFA) that for all regular languages $R_1, R_2 \subseteq \Sigma^*$, the language $R_1 \rightsquigarrow_H R_2$ is regular. However, $H$ is a non-regular CFL.

We remark that $R_1 \rightsquigarrow_H R_2$ is similar to proportional removals studied by Stearns and Hartmanis [189], Amar and Putzolu [4, 5] Seiferas and McNaughton [180], Kosaraju [120, 121, 122], Kozen [123], Zhang [205], the author [35], Berstel *et al.* [17], and others. In particular, we note the case of $\frac{1}{2}(L)$, given by

$$\frac{1}{2}(L) = \{x \in \Sigma^* : \exists y \in \Sigma^* \text{ such that } xy \in L \text{ and } |x| = |y|\}.$$

Thus, $\frac{1}{2}(L) = L \rightsquigarrow_H \Sigma^*$. The operation $\frac{1}{2}(L)$ is one of a class of operations which preserve regularity. Seiferas and McNaughton completely characterize those binary relations $r \subseteq \mathbb{N}^2$ such

that the operation

$$P(L, r) = \{x \in \Sigma^* : \exists y \in \Sigma^* \text{ such that } xy \in L \text{ and } r(|x|, |y|)\}$$

preserves regularity.

Recall that a *binary relation* $r$ on a set $S$ is any subset of $S^2$. Call a binary relation $r \subseteq \mathbb{N}^2$ *u.p.-preserving* if $A$ u.p. implies

$$r^{-1}(A) = \{i \ : \ \exists j \in A \text{ such that } r(i, j)\}$$

is also u.p.[1]. Then, the binary relations $r$ such that $P(\cdot, r)$ preserves regularity are precisely the u.p.-preserving relations [180].

We note the inclusion

$$L_1 \rightsquigarrow_H L_2 \subseteq \frac{1}{2}(L_1) \cap L_1/L_2$$

holds for $H = \{i^n d^n \ : \ n \geq 0\}$. However, equality does not hold in general. Consider the languages $L_1 = \{0^2, 0^4\}$, $L_2 = \{0^3\}$. Then note that $0 \in \frac{1}{2}(L_1) \cap L_1/L_2$. However, $0 \notin L_1 \rightsquigarrow_H L_2$. Thus, we note that $L_1 \rightsquigarrow_H L_2 \neq \frac{1}{2}(L_1) \cap L_1/L_2$ in general.

We now consider arbitrary relations $r \subseteq \mathbb{N}^2$ for which

$$H_r = \{i^n d^m \ : \ r(n, m)\} \subseteq i^* d^*$$

preserves regularity. By modifying the construction of Seiferas and McNaughton, we obtain the following result:

**Theorem 5.4.1** *Let $r \subseteq \mathbb{N}^2$ be a binary relation and $H_r = \{i^n d^m \ : \ r(n, m)\}$. The operation $\rightsquigarrow_{H_r}$ is regularity-preserving if and only if $r$ is u.p.-preserving.*

**Proof.** Assume that $\rightsquigarrow_{H_r}$ preserves regularity. Then $L \rightsquigarrow_{H_r} \Sigma^*$ is regular for all regular languages $L$. But $L \rightsquigarrow_{H_r} \Sigma^* = P(L, r)$. Thus, $r$ must be u.p.-preserving [180].

[1] Recall that u.p. (ultimately periodic) was defined in Section 2.1.

For the reverse implication, we modify the construction of Seiferas and McNaughton [180, Thm. 1]. Let $L_1, L_2$ be regular languages. Let $M_1 = (Q_1, \Sigma, \delta_1, q_0, F_1)$ be the minimal complete DFA for $L_1$. Then, for each $q \in Q_1$, we let $L_1^{(q)}$ be the language accepted by the DFA $M_1^{(q)} = (Q_1, \Sigma, \delta_1, q_0, \{q\})$. Let $R_q$ be the language accepted by the DFA $N_1^{(q)} = (Q_1, \Sigma, \delta_1, q, F_1)$. Note that $L_1^{(q)} = \{w \in \Sigma^* : \delta(q_0, w) = q\}$ and $R_q = \{w \in \Sigma^* : \delta(q, w) \in F_1\}$.

As $M_1$ is complete, $\Sigma^* = \bigcup_{q \in Q_1} L_1^{(q)}$. Thus,

$$L_1 \rightsquigarrow_{H_r} L_2 = \bigcup_{q \in Q_1} (L_1 \rightsquigarrow_{H_r} L_2) \cap L_1^{(q)}.$$

It suffices to demonstrate that $(L_1 \rightsquigarrow_{H_r} L_2) \cap L_1^{(q)}$ is regular. But we note that

$$\begin{aligned}
(L_1 \rightsquigarrow_{H_r} L_2) \cap L_1^{(q)} &= \{x \in L_1^{(q)} : \exists y \in L_2 \text{ such that } xy \in L_1 \text{ and } r(|x|, |y|)\} \\
&= \{x \in L_1^{(q)} : \exists y \in (R_q \cap L_2) \text{ such that } r(|x|, |y|)\} \\
&= \{x \in \Sigma^* : \exists y \in (R_q \cap L_2) \text{ such that } r(|x|, |y|)\} \cap L_1^{(q)} \\
&= \{x \in \Sigma^* : |x| \in r^{-1}(\{|y| : y \in (R_q \cap L_2)\})\} \cap L_1^{(q)}.
\end{aligned}$$

It is easy to see that if $L$ is regular, $\{|y| : y \in L\}$ is a u.p. set. As $r$ is u.p.-preserving, $r^{-1}(\{|y| : y \in R_q \cap L_2)\})$ is also u.p. ■

Note that, in general, $L_1 \rightsquigarrow_{H_r} L_2 \neq P(L_1, r) \cap L_1/L_2$. Consider the following particular examples of regularity-preserving trajectories:

(a) Consider the relation $e = \{(n, 2^n) : n \geq 0\}$. Then $H_e$ preserves regularity (see, e.g., Zhang [205, Sect. 3]). However, $H_e$ is not CF. The set $H_e$ is, however, a linear conjunctive language (see Okhotin [160] for the definition of conjunctive and linear conjunctive languages, and for the proof that $H_e$ is linear conjunctive).

(b) Consider the relation $f = \{(n, n!) : n \geq 0\}$. Then $H_f$ preserves regularity (see again Zhang [205, Thm. 5.1]). However, $H_f$ is not a CFL, nor a linear conjunctive language [160].

Thus, there are non-CF trajectories which preserve regularity. Kozen states that there are even $H_r$ which preserve regularity but are "highly noncomputable" [123, p. 3].

We can extend the class of non-regular sets of trajectories $T$ such that $L_1 \rightsquigarrow_T L_2$ is regular for all regular languages $L_1, L_2$ by considering $T$ such that $T \subseteq (d^*i^*)^m d^*$ for some $m \geq 1$ (The choice of $T \subseteq (d^*i^*)^m d^*$ rather than, e.g., $T \subseteq (i^*d^*)^m$ or $T \subseteq (d^*i^*)^m$ is arbitrary. The same type of formulation and arguments can be applied to these similar types of sets of trajectories). To consider such non-regular $T$, it will be advantageous to adopt the notations of Zhang [205] on *boolean matrices*. We summarize these notions below; for a full review, the reader may consult the original paper.

For any finite set $Q$, let $\mathcal{M}(Q)$ denote the set of square Boolean matrices indexed by $Q$. Let $\mathcal{V}(Q)$ denote the set of Boolean vectors indexed by $Q$. For an automaton over a set of states $Q$, we will associate with it matrices from $\mathcal{M}(Q)$ and vectors from $\mathcal{V}(Q)$.

In particular, let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA. Then for each $a \in \Sigma$, let $\nabla_a \in \mathcal{M}(Q)$ be the matrix defined by transitions on $a$, that is,$\nabla_a(q_1, q_2) = 1$ if and only if $\delta(q_1, a) = q_2$. Let $\nabla = \sum_{a \in \Sigma} \nabla_a$ (where addition is taken to be Boolean addition, i.e., $0+0 = 0, 0+1 = 1+0 = 1+1 = 1$). Thus, $\nabla(q_1, q_2) = 1$ if and only if there is some $a \in \Sigma$ such that $\delta(q_1, a) = q_2$. Note that taking powers of $\nabla$ yields information on paths of different lengths: for all $i \geq 0$, $\nabla^i(q_1, q_2) = 1$ if and only if there is a path of length $i$ from $q_1$ to $q_2$.

For any $Q' \subseteq Q$, let $I_{Q'} \in \mathcal{V}(Q)$ be the characteristic vector of $Q'$, given by $I_{Q'}(q) = 1$ if and only if $q \in Q'$. If $Q'$ is a singleton $q$, we denote $I_{\{q\}}$ by $I_q$. Note that if $Q_1, Q_2 \subseteq Q$ and $i \geq 0$, then $I_{Q_1} \cdot \nabla^i \cdot I_{Q_2}^t = 1$ if and only if there is a path of length $i$ from some state in $Q_1$ to some state in $Q_2$ (here, $I^t$ denotes the transpose of $I$).

Call a function $f : \mathbb{N} \to \mathbb{N}$ *ultimately periodic with respect to powers of Boolean matrices* [205], abbreviated m.u.p. (for "matrix ultimately periodic"), if, for all square Boolean matrices $\nabla$, there exist natural numbers $e$, $p$ ($p > 0$) such that for all $n \geq e$,

$$\nabla^{f(n)} = \nabla^{f(n+p)}.$$

The functions $n!$ and $2^n$ are known to be m.u.p. [205].

Let $m \geq 1$. We will define a class of $T \subseteq (d^*i^*)^m d^*$ such that for all regular languages $R_1$, $R_2$,

$R_1 \rightsquigarrow_T R_2$ is regular. In particular, let $m \geq 1$, and let $f_\ell^{(j)} : \mathbb{N} \to \mathbb{N}$ be a m.u.p. function for each $1 \leq \ell \leq m+1$ and $1 \leq j \leq m$. Define $X_\ell : \mathbb{N}^m \to \mathbb{N}$ for $1 \leq \ell \leq m+1$ by

$$X_\ell(n_1, n_2, \ldots, n_m) = \sum_{j=1}^{m} f_\ell^{(j)}(n_j). \tag{5.2}$$

We will use the abbreviation $\vec{n} = (n_1, n_2, \ldots, n_m)$. Finally, we define

$$T = \{\prod_{j=1}^{m} (d^{X_j(\vec{n})} i^{n_j}) d^{X_{m+1}(\vec{n})} \ : \ \vec{n} = (n_1, \ldots, n_m) \in \mathbb{N}^m\}. \tag{5.3}$$

The set $T$ satisfies our intuition that the '$i$-portions' may not interact with each other, but may interact with any '$d$-portion' they wish to. Our claim that these $T$ preserve regularity is proven in the following theorem.

**Theorem 5.4.2** *Let $m \geq 1$, and $f_\ell^{(j)}$ be m.u.p. for $1 \leq \ell \leq m+1$ and $1 \leq j \leq m$. Let $T \subseteq (d^*i^*)^m d^*$ be defined by (5.2) and (5.3). Then for all regular languages $R_1$, $R_2$, the language $R_1 \rightsquigarrow_T R_2$ is regular.*

In this section only, let $\mathbf{m} = \{0, 1, 2, 3, \ldots, m\}$ for any $m \geq 1$.

**Proof.** Let $M_i = (Q_i, \Sigma, \delta_i, s_i, F_i)$ be a DFA accepting $R_i$ for $i = 1, 2$. Let $M_{1,2} = (Q_1 \times Q_2, \Sigma, \delta_0, [s_1, s_2], F_1 \times F_2)$ where $\delta$ is given by $\delta_0([q_1, q_2], a) = (\delta_1[q_1, a], \delta_2[q_2, a])$ for all $[q_1, q_2] \in Q_1 \times Q_2$ and all $a \in \Sigma$. Note that $M_{1,2}$ accepts $R_1 \cap R_2$. Let $\nabla$ be the adjacency matrix for $M_{1,2}$. For each $1 \leq j \leq m$ and $1 \leq \ell \leq m+1$, let $e_\ell^{(j)}$ and $p_\ell^{(j)}$ be natural numbers such that $\nabla^{f_\ell^{(j)}(n)} = \nabla^{f_\ell^{(j)}(n+p_\ell^{(j)})}$ for all $n \geq e_\ell^{(j)}$.

For all $1 \leq j \leq m$ and $1 \leq \ell \leq m+1$, let $g_\ell^{(j)} = e_\ell^{(j)} + p_\ell^{(j)}$ and define the set

$$\mathfrak{M}(j, \ell) = \{\nabla^{f_\ell^{(j)}(i)} \ : \ 0 \leq i \leq g_\ell^{(j)}\} \times \mathbf{g}_\ell^{(\mathbf{j})}.$$

We will define an NFA $M = (Q, \Sigma, \delta, S, F)$ which we claim accepts $R_1 \rightsquigarrow_T R_2$. The NFA will be nondeterministic, and will also have multiple start states. It is well known that multiple start states do not affect the regularity of the language accepted (see, e.g., Yu [201, p. 54]); our presentation is chosen for ease of description.

We now proceed with defining $M$. Our state set $Q$ is given by

$$Q = \mathbf{m} \times (\prod_{\ell=1}^{m}(\prod_{j=1}^{m} \mathfrak{M}(j,\ell)) \times Q_1^3 \times Q_2) \times \prod_{j=1}^{m} \mathfrak{M}(j, m+1).$$

Let $\mu_{j,\ell} = [\nabla^{f_\ell^{(j)}(0)}, 0] \in \mathfrak{M}(j,\ell)$. Our set $S$ of initial states is given by

$$S = \{1\} \times (\prod_{\ell=1}^{m}\prod_{j=1}^{m} \mu_{j,\ell} \times \{[q,q] \,:\, q \in Q_1\} \times Q_1 \times Q_2) \times \prod_{j=1}^{m} \mu_{j,m+1}.$$

To partially motivate this definition, the elements of the form $Q_1^3$ will represent one path through $M_1$: the first element will represent our nondeterministic "guess" of where the path starts, the second state will actually trace the path through $M_1$ (along a portion of our input word) and the third state represents our guess of where the path will end. Thus, during the course of our computation, the first and third elements are never changed; only the second is affected by the input word. The first and third elements are used to verify (once the computation has completed) that our guesses for the start and finish are correct, and that they correspond ("match up") with the guessed paths for the adjacent components. The elements of $Q_2$ will represent our guesses of the intermediate points of the path through $M_2$; similarly to our guesses in $Q_1$, they will not change through the course of the computation.

Our set of final states $F$ is given by those states of the form

$$\{m\} \times \left[\left[[A_\ell^{(j)}, c_\ell^{(j)}]_{j=1}^{m} q_\ell^{(1)}, q_\ell^{(2)}, q_\ell^{(3)}, r_\ell\right]_{\ell=1}^{m}, (A_{m+1}^{(j)}, c_\ell^{(j)})_{j=1}^{m}\right],$$

where the following conditions are met:

(F-i) for all $1 \le \ell \le m$, $I_{(q_{\ell-1}^{(3)}, r_{\ell-1})} \cdot (\prod_{j=1}^{m} A_\ell^{(j)}) \cdot I^t_{(q_\ell^{(1)}, r_\ell)} = 1$ (we let $q_0^{(3)} = s_1$, the start state of $M_1$ and $r_0 = s_2$ the start state of $M_2$);

(F-ii) $I_{(q_m^{(3)}, r_m)} \cdot (\prod_{j=1}^{m} A_{m+1}^{(j)}) \cdot I^t_{F_1 \times F_2} = 1$;

(F-iii) for all $1 \le \ell \le m$, we have $q_\ell^{(2)} = q_\ell^{(3)}$.

We will see that the matrix $A_\ell^{(j)}$ will ensure there is a path of length $f_\ell^{(j)}(n_j)$ through $M_1 \times M_2$. Thus, condition (F-i) will ensure that we have a path from our guessed end state of the previous $i$-portion

through to the guessed start state of the next $i$-portion. This will correspond to the presence of some word $w$ of length $\sum_{j=1}^{m} f_\ell^{(j)}(n_j)$ which takes us $M$ from the end state of the previous $i$-portion to the start of the next $i$-portion. The condition (F-ii) will ensure that the final $d$-portion ends in a final state in both $M_1$ and $M_2$.

Condition (F-iii) verifies that the nondeterministic "guesses" for the end of each $i$-portion path is correct.

Finally, we may define the action of $\delta$. We will adopt the convention of Zhang [205] and denote by $\langle c \rangle_a^b$ the quantity

$$\langle c \rangle_a^b = \begin{cases} c & \text{if } c \leq a; \\ a + ((c-a) \bmod b\,) & \text{otherwise.} \end{cases}$$

Further, to describe the action of $\delta$ more easily, we introduce auxiliary functions $\Upsilon_{\ell,\alpha}$ for all $1 \leq \ell \leq m+1$ and $1 \leq \alpha \leq m$. In particular

$$\Upsilon_{\ell,\alpha} : \prod_{j=1}^{m} \mathfrak{M}(j,\ell) \to \prod_{j=1}^{m} \mathfrak{M}(j,\ell)$$

is given by

$$\begin{aligned} &\Upsilon_{\ell,\alpha}([\nabla^{f_\ell^{(j)}(c_\ell^{(j)})}, c_\ell^{(j)}]_{j=1}^{m}) \\ &\quad = \left[ [\nabla^{f_\ell^{(j)}(c_\ell^{(j)})}, c_\ell^{(j)}]_{j=1}^{\alpha-1}, \nabla^{f_\ell^{(\alpha)}(\langle c_\ell^{(\alpha)}+1 \rangle_{e_\ell^{(\alpha)}}^{p_\ell^{(\alpha)}})}, \langle c_\ell^{(\alpha)}+1 \rangle_{e_\ell^{(\alpha)}}^{p_\ell^{(\alpha)}}, [\nabla^{f_\ell^{(j)}(c_\ell^{(j)})}, c_\ell^{(j)}]_{j=\alpha+1}^{m} \right]. \end{aligned}$$

Note that $\Upsilon_{\ell,\alpha}$ updates the $\alpha$-th component, while leaving all other components unchanged.

Then we define $\delta$ by

$$\begin{aligned} &\delta\left(\left[\alpha, \left[[\nabla^{f_\ell^{(j)}(c_\ell^{(j)})}, c_\ell^{(j)}]_{j=1}^{m}, p_\ell^{(1)}, p_\ell^{(2)}, p_\ell^{(3)}, r_\ell\right]_{\ell=1}^{m}, [\nabla^{f_{m+1}^{(j)}(c_{m+1}^{(j)})}, c_{m+1}^{(j)}]_{j=1}^{m}\right], a\right) \\ = \;& \Big\{\Big[\alpha+\beta, [\Upsilon_{\ell,\alpha+\beta}([\nabla^{f_\ell^{(j)}(c_\ell^{(j)})}, c_\ell^{(j)}]_{j=1}^{m}), p_\ell^{(1)}, p_\ell^{(2)}, p_\ell^{(3)}, r_\ell]_{\ell=1}^{\alpha+\beta-1}, \\ &\quad \Upsilon_{\alpha+\beta,\alpha+\beta}([\nabla^{f_{\alpha+\beta}^{(j)}(c_{\alpha+\beta}^{(j)})}, c_{\alpha+\beta}^{(j)}]_{j=1}^{m}), p_{\alpha+\beta}^{(1)}, \delta_1(p_{\alpha+\beta}^{(2)}, a), p_{\alpha+\beta}^{(3)}, r_{\alpha+\beta} \\ &\quad [\Upsilon_{\ell,\alpha+\beta}([\nabla^{f_\ell^{(j)}(c_\ell^{(j)})}, c_\ell^{(j)}]_{j=1}^{m}), p_\ell^{(1)}, p_\ell^{(2)}, p_\ell^{(3)}, r_\ell]_{\ell=\alpha+\beta+1}^{m}, \\ &\quad \Upsilon_{m+1,\alpha+\beta}([\nabla^{f_{m+1}^{(j)}(c_{m+1}^{(j)})}, c_{m+1}^{(j)}]_{j=1}^{m})\Big] : 0 \leq \beta \leq m-\alpha\Big\}. \end{aligned}$$

Note that, though the definition of $\delta$ is complicated, its action is straight-forward. The index $\alpha$ indicates the '$i$-portion' which is currently receiving the input. Given that we are currently in the $\alpha$-th $i$-portion, we may nondeterministically choose to move to any of the subsequent portions. The action of the function $\Upsilon_{\ell,\alpha}$ is to simulate the corresponding function $f_\ell^\alpha$.

We show that $L(M) \subseteq R_1 \rightsquigarrow_T R_2$. If we arrive at a final state, by (F-i), for each $1 \leq \ell \leq m$ there is a word $x_\ell$ of length $X_\ell(\vec{n})$ which takes us from state $q_{\ell-1}^{(3)}$ to $q_\ell^{(1)}$ in $M_1$ and also takes us from $r_{\ell-1}$ to $r_\ell$ in $M_2$. By the choice of $S$, $\delta$ and condition (F-iii), for each $1 \leq \ell \leq m$, there is a word $w_i$ of length $n_i$ which takes us from state $q_\ell^{(1)}$ to $q_\ell^{(3)}$. Further, the input word is of the form $w = w_1 w_2 \cdots w_m$. Finally, by (F-ii), there is a word $x_{m+1}$ of length $X_{m+1}(\vec{n})$ which takes us from state $q_m$ to a final state in $M_1$ and from $r_m$ to a final state in $M_2$. The situation is illustrated in Figure 5.1.

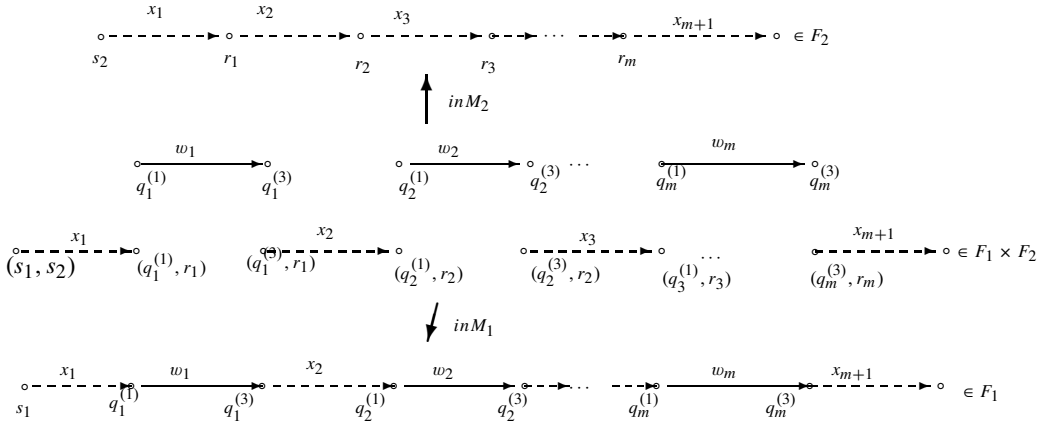


Figure 5.1: Construction of the words in $M_1$ and $M_2$ from the action of $M$.

Thus, we conclude that $x_1 w_1 \cdots x_m w_m x_{m+1} \in R_1$, $x_1 \cdots x_{m+1} \in R_2$ and $|x_\ell| = X_\ell(\vec{n})$ for all $1 \leq \ell \leq m+1$. Thus, $w_1 \cdots w_m \in R_1 \rightsquigarrow_T R_2$. A similar argument, which is left to the reader, shows the reverse inclusion. ■

As an example, consider $m = 1$ and let $f_1^{(1)}$, $f_2^{(2)}$ both be the identity function. Then the

conditions of Theorem 5.4.2 are met and $T = \{d^n i^n d^n \ : \ n \geq 0\}$. Consider then that

$$R_1 \rightsquigarrow_T \Sigma^* = \{x \ : \ \exists y, z \in \Sigma^{|x|} \text{ such that } yxz \in R_1\}.$$

This is the 'middle-thirds' operation, which is sometimes used as a challenge problem for undergraduates in formal language theory (see, e.g., Hopcroft and Ullman [68, Ex. 3.17]). We may immediately conclude that the regular languages are preserved under the middle-thirds operation.

We note that the condition that $(n_1, n_2, \ldots, n_m) \in \mathbb{N}^m$ in (5.3) can be replaced by the conditions that, for all $1 \leq j \leq m$, $n_j \in I_j$ for an arbitrary u.p. set $I_j \subseteq \mathbb{N}$. The construction adds considerable detail to the proof of Theorem 5.4.2, and is omitted. With this extension, we can also consider a class of examples given by Amar and Putzolu [5], which are equivalent to trajectories of the form

$$AP(k_1, k_2, \alpha) = \{i^{mk_1} d^{mk_2+\alpha} \ : \ m \geq 0\},$$

for fixed $k_1, k_2, \alpha \geq 0$ with $\alpha < k_1 + k_2$. For any $k_1, k_2, \alpha \geq 0$, we can conclude that the operation $\rightsquigarrow_A$ preserves regularity, where $A = AP(k_1, k_2, \alpha)$. This was established by Amar and Putzolu [5] by means of *even linear grammars*.

Pin and Sakarovitch use a very general and elegant method to prove that certain operations preserve regularity [165]. This method can be used to prove that certain operations which can be modeled by trajectories preserve regularity; it is not known whether the methods developed here can be extended to cover theses cases. For example, the let $T_\zeta$ be given by

$$T_\zeta = \{d^{nk} i^k d^{nk} \ : \ k, n \geq 0, 2n + 1 \text{ is prime}\}.$$

Then Pin and Sakarovitch prove that $L \rightsquigarrow_{T_\zeta} \Sigma^*$ is regular for all regular languages $L$ [165, p. 292][2].

[2]Note that the definition of $T_\zeta$ given here matches that given by Pin and Sakarovitch [165]. In a preliminary version [166], a different deletion operation is defined which *can* be modeled by a set of trajectories to which Theorem 5.5.1 below can be applied.

## 5.5 $i$-Regularity

Recall that a language $L \subseteq \Sigma^*$ is *bounded* if there exist $w_1, w_2, \ldots, w_n \in \Sigma^*$ such that $L \subseteq w_1^* w_2^* \cdots w_n^*$. We say that $L$ is *letter-bounded*[3] if $w_i \in \Sigma$ for all $1 \leq i \leq n$.

We now define a class of letter-bounded sets of trajectories, called *$i$-regular* sets of trajectories, which will have strong closure properties. In particular, we can delete, along an $i$-regular set of letter-bounded trajectories, any language from a regular language and the resulting language will be regular. This will allow us in Section 7.3 to give positive decidability results for the related shuffle decomposition problem.

Let $\Delta_m$ be the alphabet $\Delta_m = \{\#_1, \#_2, \ldots, \#_m\}$ for any $m \geq 1$. We define a class of regular substitutions from $(d + \Delta_m)^*$ to $2^{(i+d)^*}$, denoted $\mathfrak{S}_m$, as follows: a regular substitution $\varphi : (d + \Delta_m)^* \to 2^{(i+d)^*}$ is in $\mathfrak{S}_m$ if both

(a) $\varphi(d) = \{d\}$; and

(b) for all $1 \leq j \leq m$, there exist $a_j, b_j \in \mathbb{N}$ such that $\varphi(\#_j) = i^{a_j}(i^{b_j})^*$.

For all $m \geq 1$, we also define a class of languages over the alphabet $d + \Delta_m$, denoted $\mathfrak{T}_m$, as the set of all languages $T \subseteq \#_1 d^* \#_2 d^* \cdots \#_{m-1} d^* \#_m$. Define the class of trajectories $\mathfrak{I}$ as follows:

$$\mathfrak{I} = \{T \subseteq \{i, d\}^* \ : \ \exists m \geq 1, T_m \in \mathfrak{T}_m, \varphi \in \mathfrak{S}_m \text{ such that } T = \varphi(T_m)\}.$$

If $T \in \mathfrak{I}$, we say that $T$ is *$i$-regular*. As we shall see, the condition that $T$ be $i$-regular is sufficient for showing that $R \rightsquigarrow_T L$ is regular for all regular languages $R$ and all languages $L$.

**Theorem 5.5.1** *Let $T \in \mathfrak{I}$. Then for all regular languages $R$ and all languages $L$, $R \rightsquigarrow_T L$ is a regular language.*

**Proof.** Let $T \in \mathfrak{I}$. Let $m \geq 1$, $T' \in \mathfrak{T}_m$ and $\varphi \in \mathfrak{S}_m$ be such that $T = \varphi(T')$. Then we define $K(T) \subseteq \mathbb{N}^{m-1}$ as

$$K(T) = \{(j_1, \ldots, j_{m-1}) \ : \ \#_1 d^{j_1} \#_2 d^{j_2} \cdots \#_{m-1} d^{j_{m-1}} \#_m \in T'\}.$$

[3]The term *strictly bounded* is sometimes used for this situation, e.g, Dassow *et al.* [32]. However, other sources, e.g., Harju and Karhumäki [60] and Mateescu *et al.* [150] use the same term differently.

Let $a_j, b_j$ be defined so that $\varphi(\#_j) = i^{a_j}(i^{b_j})^*$ for all $1 \leq j \leq m$. Let $I_j = \{a_j + nb_j \ : \ n \geq 0\}$ for all $1 \leq j \leq m$.

Let $R$ be regular and $L$ be arbitrary. Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA accepting $R$. For all $q_j, q_k \in Q$, let $R(q_j, q_k) = L((Q, \Sigma, \delta, q_j, \{q_k\}))$. Note that

$$R(q_j, q_k) = \{w \in \Sigma^* \ : \ q_k \in \delta(q_j, w)\}.$$

For $I \subseteq \mathbb{N}$, let $R'_I(q_j, q_k) = R(q_j, q_k) \cap \{x \ : \ |x| \in I\}$.

We now define the set $Q_R(T, L) \subseteq Q^{2m-2}$:

$$\begin{aligned} Q_R(T, L) = \{(q_1, q_2, \ldots, q_{2m-2}) \in Q^{2m-2} \\ : \ \exists (k_j)_{j=1}^{m-1} \in K(T) \text{ such that } L \cap \prod_{\ell=1}^{m-1} R'_{\{k_\ell\}}(q_{2\ell-1}, q_{2\ell}) \neq \emptyset\}. \end{aligned} \tag{5.4}$$

We claim that

$$R \rightsquigarrow_T L = \bigcup_{\substack{(q_j)_{j=1}^{2m-2} \in Q_R(T,L) \\ q_f \in F}} \left( \prod_{\ell=1}^{m-1} R'_{I_\ell}(q_{2(\ell-1)}, q_{2\ell-1}) \right) \cdot R'_{I_m}(q_{2m-2}, q_f). \tag{5.5}$$

Let $x \in R \rightsquigarrow_T L$. Then we can write $x = x_1 x_2 \cdots x_m$ such that there exists some $z = z_1 z_2 \cdots z_{m-1} \in L$ such that $y = x_1 z_1 x_2 z_2 \cdots x_{m-1} z_{m-1} x_m \in R$. Further, by the conditions on $T$, $(|z_j|)_{j=1}^{m-1} \in K(T)$ and $|x_j| \in I_j$ for all $1 \leq j \leq m$. We let $q \overset{x}{\vdash} q'$ denote the fact that $\delta(q, x) = q'$ in $M$. As $y \in R$, there are some $q_1, q_2, \ldots, q_{2m-2}, q_f \in Q$ such that

$$q_0 \overset{x_1}{\vdash} q_1 \overset{z_1}{\vdash} q_2 \overset{x_2}{\vdash} \cdots \overset{x_{m-1}}{\vdash} q_{2m-3} \overset{z_{m-1}}{\vdash} q_{2m-2} \overset{x_m}{\vdash} q_f$$

and $q_f \in F$. Then $z_j \in R'_{\{|z_j|\}}(q_{2j-1}, q_{2j})$ for all $1 \leq j \leq m-1$, $x_j \in R'_{I_j}(q_{2(j-1)}, q_{2j-1})$ for all $1 \leq j \leq m-1$ and $x_m \in R'_{I_m}(q_{2m-2}, q_f)$. Further, note that

$$z \in L \cap \prod_{\ell=1}^{m-1} R'_{\{|z_\ell|\}}(q_{2\ell-1}, q_{2\ell}).$$

We conclude that $(q_1, q_2, \ldots, q_{2m-2}) \in Q_R(T, L)$, as $(|z_j|)_{j=1}^{m-1} \in K(T)$, and thus $x$ is contained in the right-hand side of (5.5).

For the reverse inclusion, let $(q_1, \ldots, q_{2m-2}) \in Q_R(T, L)$ and $q_f \in F$. Let $(k_1, \ldots, k_{m-1}) \in K(T)$ be a $(m-1)$-tuple which witnesses $(q_1, q_2, \ldots, q_{2m-2})$'s membership in $Q_R(T, L)$. Then we show that $(\prod_{\ell=1}^{m-1} R'_{I_\ell}(q_{2(\ell-1)}, q_{2\ell}))R'_{I_m}(q_{2m-2}, q_f) \subseteq R \rightsquigarrow_T L$.

Let $z_j \in R'_{\{k_j\}}(q_{2j-1}, q_{2j})$ for all $1 \leq j \leq m-1$ be such that $z = z_1 \cdots z_{m-1} \in L$. Such $z_j$ exist by definition of $Q_R(T, L)$. Let $x_j \in R'_{I_j}(q_{2(j-1)}, q_{2j-1})$ for all $1 \leq j \leq m-1$, and $x_m \in R'_{I_m}(q_{2m-2}, q_f)$ be arbitrary. Then

$$q_0 \overset{x_1}{\vdash} q_1 \overset{z_1}{\vdash} q_2 \overset{x_2}{\vdash} \cdots \overset{x_{m-1}}{\vdash} q_{2m-3} \overset{z_{m-1}}{\vdash} q_{2m-2} \overset{x_m}{\vdash} q_f.$$

Thus, $y = x_1 z_1 \cdots x_{m-1} z_{m-1} x_m \in R$. Further, the length considerations are met by definition of $I_j$ and $(k_1, k_2, \ldots, k_{m-1}) \in K(T)$. Thus $x \in y \rightsquigarrow_T z \subseteq R \rightsquigarrow_T L$.

Thus, since $Q_R(T, L)$ is finite, $R \rightsquigarrow_T L$ is a finite union of regular languages, and thus is regular. ■

**Corollary 5.5.2** *Let $T \subseteq \{i, d\}^*$ be a finite union of $i$-regular sets of trajectories. Then for all regular languages $R$ and all languages $L$, the language $R \rightsquigarrow_T L$ is regular.*

We note that if $T$ is not $i$-regular, it may define an operation which does not preserve regularity in the sense of Theorem 5.5.1. In particular, from the proof of Theorem 5.3.2, we have that if $T = (di)^*$,

$$(a^2)^*(b^2)^* \rightsquigarrow_T \{a^n b^n \ : \ n \geq 0\} = \{a^n b^n \ : \ n \geq 0\},$$

a non-regular CFL. For $T = (i + d)^*$, we have that

$$\left((ab)^*\#(ab)^* \rightsquigarrow_T \{a^n \# b^n \ : \ n \geq 0\}\right) \cap b^*a^* = \{b^n a^n \ : \ n \geq 0\}.$$

Further, if $T$ is letter-bounded but not $i$-regular, then $T$ may not preserve regularity. Again, from the proof of Theorem 5.3.2, we have that if $T = \{i^n d i^n \ : \ n \geq 0\}$. Then $a^*\#b^* \rightsquigarrow_T \{\#\} = \{a^n b^n \ : \ n \geq 0\}$. Note that in this case, the language $\{\#\}$ is a singleton. We also have that there is a non-$i$-regular set of trajectories,

$$T = \{i^n d^n i^n \ : \ n \geq 0\},$$

and a regular language $R$ such that $R \rightsquigarrow_T \Sigma^*$ is not a regular language. In particular, we have the following example. Let $\Sigma = \{a, b, c\}$ and $R = a^*bc^*$. Then $R \rightsquigarrow_T \Sigma^*$ is not a regular language, as $(R \rightsquigarrow_T \Sigma^*) \cap a^*c^* = \{a^n c^n \ : \ n \geq 0\}$.

As an example of Theorem 5.5.1, consider $T = \{d^n i^m d^n \ : \ n, m \geq 0\}$. It is easily verified that $T \in \mathfrak{I}$ (consider $T' = \{\#_1 d^n \#_2 d^n \#_3 \ : \ n \geq 0\}$, and $\varphi$ defined by $\varphi(\#_1) = \varphi(\#_3) = \{\epsilon\}$ and $\varphi(\#_2) = i^*$). Thus, the language $R \rightsquigarrow_T L$ is regular for all regular languages $R$ and all languages $L$. For any language $L \subseteq \Sigma^*$, define $sq(L) = \{x^2 \ : \ x \in L\}$. Consider then that

$$R \rightsquigarrow_T sq(L) = \{w \ : vwv \in R, v \in L\}.$$

This precisely defines the *middle-quotient* operation, which has been investigated by Meduna [153] for linear CFLs. Let $R \mid L$ denote the middle quotient of $R$ by $L$, i.e., $R \mid L = R \rightsquigarrow_T sq(L)$. Thus, we can immediately conclude the following result, which was not considered by Meduna:

**Theorem 5.5.3** *Given a regular language $R$ and arbitrary language $L$, the language $R|L$ is regular.*

## 5.6 Filtering and Deletion along Trajectories

Recently, Berstel *et al.* [17] introduced the concept of filtering. Here we examine the notion of filtering, and show that it is a particular case of deletion along trajectories.

Given a sequence $s \subseteq \mathbb{N}$, and a word $w \in \Sigma^*$ with $w = w_1 \cdots w_n$, $w_i \in \Sigma$, the filtering of $w$ by $s$ is given by $w[s] = w_{s_0} w_{s_1} \cdots w_{s_k}$ where $k$ is such that $s_k \leq n < s_{k+1}$. For example, if $s = (1, 2, 4, 7)$, then $abcacb[s] = aba$. Filtering is extended monotonically to languages.

For every $s \subseteq \mathbb{N}$, let $\omega_s \ : \ \mathbb{N} \to \{i, d\}$ be given by $\omega_s(j) = i$ if $j \in s$ and $\omega_s(j) = d$ otherwise[4]. Let $T_s \subseteq \{0, 1\}^*$ be defined by

$$T_s = \{\prod_{j=0}^{n} \omega_s(j) \ : \ n \geq 0\}.$$

Then we clearly have that

$$L[s] = L \rightsquigarrow_{T_s} \Sigma^*,$$

[4]That is, $\omega_s$ is the characteristic $\omega$-word of $s$ over $\{i, d\}$.

for all sequences $s \subseteq \mathbb{N}$. Note that for all $s \subseteq \mathbb{N}$, $T_s$ is *prefix-closed* (i.e., if $t_1 \in T_s$ and $t_2$ is a prefix of $t_1$, then $t_2 \in T_s$).

For all sequences $s = (s_j)_{j \geq 1} \subseteq \mathbb{N}$, let $\partial s = ((\partial s)_j)_{j \geq 1}$ be defined by $(\partial s)_j = s_{j+1} - s_j$ for $j \geq 1$. The sequence $\partial s$ is called the differential sequence of $s$. A sequence $s \subseteq \mathbb{N}$ is said to be *residually ultimately periodic* if for each finite monoid $F$ and each monoid morphism $\varphi : \mathbb{N} \to F$, $\varphi(s)$ is ultimately periodic.

Berstel *et al.* [17] characterize those sequences $s \subseteq \mathbb{N}$ which preserve regularity. In particular, a sequence $s$ preserves regularity if and only if it is *differentially residually ultimately periodic*, i.e., the sequence $\partial s$ is residually ultimately periodic.

## 5.7 Splicing on Routes

Splicing on routes was introduced by Mateescu [144] to model generalizations of the crossover splicing operation (see Mateescu [144] for a definition of the crossover splicing operation). Crossover splicing simulates the manner in which two DNA strands may be spliced together at multiple locations to form several new strands, see Mateescu for a discussion [144]. Splicing on routes has also been used to model dialogue in natural languages [12].

Splicing on routes generalizes the crossover splicing operation by specifying a set $T$ of routes which restricts the way in which splicing can occur. The result is that specific sets of routes can simulate not only the crossover operation, but also such operations on DNA such as the *simple splicing* and the *equal-length crossover* operations (see Mateescu for details and definitions of these operations [144]). Splicing on routes is also a generalization of the shuffle on trajectories operation.

In this section, we consider the simulation of splicing on a route by shuffle and deletion along trajectories. We show that there exist three fixed weak codings $\pi_1, \pi_2, \pi_3$ such that for all routes $t$, we can simulate the splicing on $t$ of two words $w_1, w_2$ by a fixed combination of the shuffle and deletion of the same languages $w_1, w_2$ along the trajectories $\pi_1(t), \pi_2(t), \pi_3(t)$. As a corollary, it is shown that every unary operation defined by splicing on routes can also be performed by a deletion

along trajectories.

We define the concept of *splicing on routes*, and note the difference between deletion along trajectories from splicing on routes, which allows discarding letters from either input word. In particular, a *route* is a word $t$ specified over the alphabet $\{0, \overline{0}, 1, \overline{1}\}$, where, informally, 0, 1 means insert the letter from the appropriate word, and $\overline{0}$, $\overline{1}$ means discard that letter and continue.

Formally, let $x, y \in \Sigma^*$ and $t \in \{0, \overline{0}, 1, \overline{1}\}^*$. We define the splicing of $x$ and $y$, denoted $x \bowtie_t y$, recursively as follows: if $x = ax'$, $y = by'$ ($a, b \in \Sigma$) and $t = ct'$ ($c \in \{0, \overline{0}, 1, \overline{1}\}$), then

$$x \bowtie_{ct'} y = \begin{cases} a(x' \bowtie_{t'} y) & \text{if } c = 0; \\ (x' \bowtie_{t'} y) & \text{if } c = \overline{0}; \\ b(x \bowtie_{t'} y') & \text{if } c = 1; \\ (x \bowtie_{t'} y') & \text{if } c = \overline{1}. \end{cases}$$

If $x = ax'$ and $t = ct'$, where $a \in \Sigma$ and $c \in \{0, \overline{0}, 1, \overline{1}\}$, then

$$x \bowtie_{ct'} \epsilon = \begin{cases} a(x' \bowtie_{t'} \epsilon) & \text{if } c = 0; \\ (x' \bowtie_{t'} \epsilon) & \text{if } c = \overline{0}; \\ \emptyset & \text{otherwise.} \end{cases}$$

If $y = by'$ and $t = ct'$, where $a \in \Sigma$ and $c \in \{0, \overline{0}, 1, \overline{1}\}$, then

$$\epsilon \bowtie_{ct'} y = \begin{cases} b(\epsilon \bowtie_{t'} y') & \text{if } c = 1; \\ (\epsilon \bowtie_{t'} y') & \text{if } c = \overline{1}; \\ \emptyset & \text{otherwise.} \end{cases}$$

We have $x \bowtie_\epsilon y = \epsilon$ if $\{x, y\} \neq \{\epsilon\}$. Finally, we set $\epsilon \bowtie_t \epsilon = \epsilon$ if $t = \epsilon$ and $\emptyset$ otherwise. We extend $\bowtie_t$ to sets of trajectories and languages as expected:

$$\begin{aligned} x \bowtie_T y &= \bigcup_{t \in T} x \bowtie_t y \quad \forall T \subseteq \{0, \overline{0}, 1, \overline{1}\}^*, x, y \in \Sigma^*; \\ L_1 \bowtie_T L_2 &= \bigcup_{\substack{x \in L_1 \\ y \in L_2}} x \bowtie_T y. \end{aligned}$$

For example, if $x = abc$, $y = cbc$ and $T = \{010011, 0\overline{1}0\overline{0}11\}$, then $x \bowtie_T y = \{acbcbc, abbc\}$.

We now demonstrate that splicing on routes can be simulated by a combination of shuffle on trajectories and deletion along trajectories.

**Theorem 5.7.1** *There exist weak codings $\pi_1, \pi_2 : \{0, 1, \overline{0}, \overline{1}\}^* \to \{i, d\}^*$ and a weak coding $\pi_3 : \{0, 1, \overline{0}, \overline{1}\}^* \to \{0, 1\}^*$ such that for all $t \in \{0, \overline{0}, 1, \overline{1}\}^*$, and for all $x, y \in \Sigma^*$, we have*

$$x \bowtie_t y = (x \rightsquigarrow_{\pi_1(t)} \Sigma^*) \sqcup\!\sqcup_{\pi_3(t)} (y \rightsquigarrow_{\pi_2(t)} \Sigma^*).$$

**Proof.** Let $\pi_1, \pi_2 : \{0, \overline{0}, 1, \overline{1}\}^* \to \{i, d\}^*$ and $\pi_3 : \{0, \overline{0}, 1, \overline{1}\} \to \{0, 1\}^*$ be given by

$$\begin{array}{cccccccccccc} \pi_1(0) & = & i; & \pi_1(\overline{0}) & = & d; & \pi_1(1) & = & \epsilon; & \pi_1(\overline{1}) & = & \epsilon; \\ \pi_2(0) & = & \epsilon; & \pi_2(\overline{0}) & = & \epsilon; & \pi_2(1) & = & i; & \pi_2(\overline{1}) & = & d; \\ \pi_3(0) & = & 0; & \pi_3(\overline{0}) & = & \epsilon; & \pi_3(1) & = & 1; & \pi_3(\overline{1}) & = & \epsilon. \end{array}$$

We first show the left-to-right inclusion. Let $z \in x \bowtie_t y$. The result is by induction on $|t|$. If $|t| = 0$, then $x = y = z = \epsilon$. Thus, we can easily verify that $z \in (\epsilon \rightsquigarrow_\epsilon \epsilon) \sqcup\!\sqcup_\epsilon (\epsilon \rightsquigarrow_\epsilon \epsilon)$.

Let $|t| > 0$. Then $t = ct'$ for $c \in \{0, \overline{0}, 1, \overline{1}\}$. We prove only the case where $c = 0$ and $c = \overline{0}$. The other two cases are similar and are left to the reader.

(a) $c = 0$. Then $x = ax'$ and $z \in a(x' \bowtie_{t'} y)$ for some $x' \in \Sigma^*$. Thus, $z = az'$ for some $z' \in (x' \bowtie_{t'} y)$. By induction, $z' \in (x' \rightsquigarrow_{\pi_1(t')} \Sigma^*) \sqcup\!\sqcup_{\pi_3(t')} (y \rightsquigarrow_{\pi_2(t')} \Sigma^*)$. Let $u \in x' \rightsquigarrow_{\pi_1(t')} \Sigma^*$ and $v \in y \rightsquigarrow_{\pi_2(t')} \Sigma^*$ be such that $z' \in u \sqcup\!\sqcup_{\pi_3(t')} v$.

Note that $\pi_1(t) = i\pi_1(t')$. Thus by definition of $\rightsquigarrow_T$, $au \in x \rightsquigarrow_{\pi_1(t)} \Sigma^*$. Similarly, as $\pi_2(t) = \pi_2(t')$, $v \in y \rightsquigarrow_{\pi_2(t)} \Sigma^*$. Finally $\pi_3(t) = 0\pi_3(t')$. Thus, $au \sqcup\!\sqcup_{\pi_3(t)} v = a(u \sqcup\!\sqcup_{\pi_3(t')} v) \ni az' = z$. Thus, the result holds for $c = 0$.

(b) $c = \overline{0}$. Then $x = ax'$ and $z \in (x' \bowtie_{t'} y)$ for some $x' \in \Sigma^*$. Thus, by induction $z \in (x' \rightsquigarrow_{\pi_1(t')} \Sigma^*) \sqcup\!\sqcup_{\pi_3(t')} (y \rightsquigarrow_{\pi_2(t')} \Sigma^*)$. Let $u \in x' \rightsquigarrow_{\pi_1(t')} \Sigma^*$ and $v \in y \rightsquigarrow_{\pi_2(t')} \Sigma^*$ be such that $z \in u \sqcup\!\sqcup_{\pi_3(t')} v$.

Note in this case that $\pi_1(t) = d\pi_1(t')$. Thus, $u \in x \rightsquigarrow_{\pi_1(t)} \Sigma^*$. Similarly, as $\pi_2(t) = \pi_2(t')$, $v \in y \rightsquigarrow_{\pi_2(t)} \Sigma^*$. Finally, $\pi_3(t) = \pi_3(t')$. Thus, $u \sqcup\!\sqcup_{\pi_3(t)} v = u \sqcup\!\sqcup_{\pi_3(t')} v \ni z$. Thus, the result holds for $c = \overline{0}$.

We now prove the reverse inclusion. Let $z \in (x \rightsquigarrow_{\pi_1(t)} \Sigma^*) \sqcup\!\sqcup_{\pi_3(t)} (y \rightsquigarrow_{\pi_2(t)} \Sigma^*)$. We show the result by induction on $t$. For $|t| = 0$, $t = \epsilon$. Thus $\pi_1(t) = \pi_2(t) = \pi_3(t) = \epsilon$. By definition of $\sqcup\!\sqcup_t$, $L_1 \sqcup\!\sqcup_\epsilon L_2$ is non-empty if and only if $\epsilon \in L_1 \cap L_2$, which implies $z = \epsilon$. Thus, $\epsilon \in (x \rightsquigarrow_\epsilon \Sigma^*)$, and similarly for $y$ in place of $x$. By definition of $\rightsquigarrow_t$, this implies that $x = y = \epsilon$. Thus, $z \in x \bowtie_t y$, by definition. The inclusion is proven for $|t| = 0$.

Let $|t| > 0$. Thus, there is some $c \in \{0, \overline{0}, 1, \overline{1}\}$, and $t' \in \{0, \overline{0}, 1, \overline{1}\}^*$ such that $t = ct'$. We distinguish between four cases, for each choice of $c$ in $\{0, \overline{0}, 1, \overline{1}\}$, however, we only prove the cases $c = 0$ and $c = \overline{0}$. The other two cases are very similar, and are left to the reader.

(a) $c = 0$. Note that $\pi_1(t) = i\pi_1(t')$, $\pi_2(t) = \pi_2(t')$ and $\pi_3(t) = 0\pi_3(t')$. Let $u, v \in \Sigma^*$ be words such that $u \in x \rightsquigarrow_{\pi_1(t)} \Sigma^*$, $v \in y \rightsquigarrow_{\pi_2(t)} \Sigma^*$ and $z \in u \sqcup\!\sqcup_{\pi_3(t)} v$.

As $\pi_3(t) = 0\pi_3(t')$, we have, by definition of $\sqcup\!\sqcup_t$, that $u = au'$, $z = az'$ and $z' \in u' \sqcup\!\sqcup_{\pi_3(t')} v$ for some $a \in \Sigma$ and $u', z' \in \Sigma^*$. Now, as $au' \in x \rightsquigarrow_{i\pi_1(t')} \Sigma^*$, there exists $x' \in \Sigma^*$ such that $x = ax'$ and $u' \in x' \rightsquigarrow_{\pi_1(t')} \Sigma^*$. Also, note that $v \in y \rightsquigarrow_{\pi_2(t')} \Sigma^*$. Thus, combining these yields that

$$z' \in (x' \rightsquigarrow_{\pi_1(t')} \Sigma^*) \sqcup\!\sqcup_{\pi_3(t')} (y \rightsquigarrow_{\pi_2(t')} \Sigma^*).$$

By induction, $z' \in x' \bowtie_{t'} y$. Thus,

$$z = az' \in a(x' \bowtie_{t'} y) = ax' \bowtie_{0t'} y = x \bowtie_t y.$$

Thus, the inclusion is proven.

(b) $c = \overline{0}$. Then $\pi_1(t) = d\pi_1(t')$, $\pi_2(t) = \pi_2(t')$ and $\pi_3(t) = \pi_3(t')$. Let $u, v \in \Sigma^*$ be such that $u \in x \rightsquigarrow_{\pi_1(t)} \Sigma^*$, $v \in y \rightsquigarrow_{\pi_2(t)} \Sigma^*$ and $z \in u \sqcup\!\sqcup_{\pi_3(t)} v$.

As $u \in x \rightsquigarrow_{\pi_1(t)} \Sigma^*$, let $u_0 \in \Sigma^*$ be such that $u \in x \rightsquigarrow_{\pi_1(t)} u_0$. As $\pi_1(t) = d\pi_1(t')$, there are some $b \in \Sigma$, $x', u_0' \in \Sigma^*$ such that $x = bx'$, $u_0 = bu_0'$ and $u \in x' \rightsquigarrow_{\pi_1(t')} u_0'$. Thus, $u \in x' \rightsquigarrow_{\pi_1(t')} \Sigma^*$. Note that $v \in y \rightsquigarrow_{\pi_2(t')} \Sigma^*$. Thus, $z \in u \sqcup\!\sqcup_{\pi_3(t')} v \subseteq (x' \rightsquigarrow_{\pi_1(t')} \Sigma^*) \sqcup\!\sqcup_{\pi_3(t')} (y \rightsquigarrow_{\pi_2(t')} \Sigma^*)$. By induction, $z \in x' \bowtie_{t'} y$. Thus, we can see that $(bx' \bowtie_t y) = x' \bowtie_{t'} y \ni z$. This proves the inclusion.

The result is now proven. ■

**Corollary 5.7.2** *There exist weak codings* $\pi_1, \pi_2 : \{0, 1, \overline{0}, \overline{1}\}^* \to \{i, d\}^*$ *and* $\pi_3 : \{0, 1, \overline{0}, \overline{1}\}^* \to \{0, 1\}^*$ *such that for all* $T \subseteq \{0, \overline{0}, 1, \overline{1}\}^*$ *and* $L_1, L_2 \subseteq \Sigma^*$,

$$L_1 \bowtie_T L_2 = \bigcup_{t \in T} (L_1 \rightsquigarrow_{\pi_1(t)} \Sigma^*) \sqcup\!\sqcup_{\pi_3(t)} (L_2 \rightsquigarrow_{\pi_2(t)} \Sigma^*).$$

Unfortunately, the identity

$$L_1 \bowtie_T L_2 = (L_1 \rightsquigarrow_{\pi_1(T)} \Sigma^*) \sqcup\!\sqcup_{\pi_3(T)} (L_2 \rightsquigarrow_{\pi_2(T)} \Sigma^*)$$

does not hold in general, even if $L_1, L_2$ are singletons and $|T| = 2$. For example, if $L_1 = \{ab\}$, $L_2 = \{cd\}$ and $T = \{\overline{0}01\overline{1}, 0\overline{01}1\}$, then

$$\begin{aligned} L_1 \bowtie_T L_2 &= \{bc, ad\}; \\ (L_1 \rightsquigarrow_{\pi_1(T)} \Sigma^*) \sqcup\!\sqcup_{\pi_3(T)} (L_2 \rightsquigarrow_{\pi_2(T)} \Sigma^*) &= \{ac, ad, bc, bd\}. \end{aligned}$$

However, if $T$ is a *unary* set of routes, by which we mean that $T \subseteq \{0, \overline{0}\}^*\overline{1}^*$, then we have the following result, which is easily established:

**Corollary 5.7.3** *Let* $T \subseteq \{0, \overline{0}\}^*\overline{1}^*$. *Then for all* $L \subseteq \Sigma^*$,

$$L \bowtie_T \Sigma^* = L \rightsquigarrow_{\pi_1(T)} \Sigma^*.$$

We refer the reader to Mateescu [144] for a discussion of unary operations defined by splicing on routes. As an example, consider that with $T = \{0^n\overline{0}^n : n \geq 0\}\overline{1}^*$, $L \bowtie_T \Sigma^* = \frac{1}{2}(L)$, where $\frac{1}{2}(L)$ was given in Section 5.4.

## 5.8 Inverse Word Operations

In this section, we show that deletion along trajectories constitutes the inverse of shuffle on trajectories, in the sense introduced by Kari [106].

We now define a word operation for our purposes. Given an alphabet $\Sigma^*$, a *word operation* is any binary function $\diamond : (\Sigma^*)^2 \to 2^{\Sigma^*}$. We usually denote a word operation as an infix operator. A word operation is extended to languages in a monotone way, as we have already seen for shuffle and deletion along trajectories: given $L_1, L_2 \subseteq \Sigma^*$,

$$L_1 \diamond L_2 = \bigcup_{\substack{x \in L_1 \\ y \in L_2}} x \diamond y.$$

Note that unlike Hsiao *et al.* [69], we do not make any assumptions about the action of $\diamond$ on $\epsilon$ as an argument.

### 5.8.1 Left Inverse

Given two binary word operations $\diamond, \star : (\Sigma^*)^2 \to 2^{\Sigma^*}$, we say that $\diamond$ is a *left-inverse* of $\star$ [106, Defn. 4.1] if, for all $u, v, w \in \Sigma^*$,

$$w \in u \star v \iff u \in w \diamond v.$$

For instance, the operations of concatenation and right-quotient are left-inverses of each other, as $w = uv$ iff $u \in w/v$.

Let $\tau : \{0,1\}^* \to \{i,d\}^*$ be the morphism given by $\tau(0) = i$ and $\tau(1) = d$. Then we have the following characterization of left-inverses:

**Theorem 5.8.1** *Let $T \subseteq \{0,1\}^*$ be a set of trajectories. Then $\sqcup\!\sqcup_T$ and $\rightsquigarrow_{\tau(T)}$ are left-inverses of each other.*

**Proof.** We show that for all $t \in \{0,1\}^*$, $w \in u \sqcup\!\sqcup_t v \iff u \in w \rightsquigarrow_{\tau(t)} v$. The proof is by induction on $|w|$. For $|w| = 0$, we have $w = \epsilon$. Thus, by definition of $\sqcup\!\sqcup_t$ and $\rightsquigarrow_t$, we have that

$$\epsilon \in u \sqcup\!\sqcup_t v \iff u = v = t = \epsilon \iff u \in (\epsilon \rightsquigarrow_{\tau(t)} v).$$

Let $w \in \Sigma^*$ with $|w| > 0$ and assume that the result is true for all words shorter than $w$. Let $w = aw'$ for $a \in \Sigma$.

First, assume that $aw' \in u \sqcup\!\sqcup_t v$. As $|t| = |w|$, we have that $t \neq \epsilon$. Let $t = et'$ for some $e \in \{0, 1\}$. There are two cases:

(a) If $e = 0$, then we have that $u = au'$ and that $w' \in u' \sqcup\!\sqcup_{t'} v$. By induction, $u' \in w' \rightsquigarrow_{\tau(t')} v$. Thus,

$$\begin{aligned} w \rightsquigarrow_{\tau(t)} v &= (aw' \rightsquigarrow_{i\tau(t')} v) \\ &= a(w' \rightsquigarrow_{\tau(t')} v) \ni au' = u. \end{aligned}$$

(b) If $e = 1$, then we have that $v = av'$ and $w' \in u \sqcup\!\sqcup_{t'} v'$. By induction, $u \in w' \rightsquigarrow_{\tau(t')} v'$. Thus,

$$\begin{aligned} w \rightsquigarrow_{\tau(t)} v &= (aw' \rightsquigarrow_{d\tau(t')} av') \\ &= (w' \rightsquigarrow_{\tau(t')} v') \ni u. \end{aligned}$$

Thus, we have that in both cases $u \in w \rightsquigarrow_{\tau(t)} v$.

Now, let us assume that $u \in w \rightsquigarrow_{\tau(t)} v$. As $|t| = |\tau(t)| = |w| \geq 1$, let $t = et'$ for some $e \in \{0, 1\}$. We again have two cases:

(a) If $e = 0$, then $\tau(e) = i$. Then necessarily $u = au'$, and $u' \in w' \rightsquigarrow_{\tau(t')} v$. By induction $w' \in u' \sqcup\!\sqcup_{t'} v$. Thus,

$$\begin{aligned} u \sqcup\!\sqcup_t v &= (au' \sqcup\!\sqcup_{0t'} v) \\ &= a(u' \sqcup\!\sqcup_{t'} v) \ni aw' = w. \end{aligned}$$

(b) If $e = 1$, then $\tau(e) = d$. Then necessarily $v = av'$, and $u \in (w' \rightsquigarrow_{\tau(t')} v')$. By induction, $w' \in u \sqcup\!\sqcup_{t'} v'$. Thus,

$$\begin{aligned} u \sqcup\!\sqcup_t v &= (u \sqcup\!\sqcup_{1t'} av') \\ &= a(u \sqcup\!\sqcup_{t'} v') \ni aw' = w. \end{aligned}$$

Thus $w \in u \sqcup\!\sqcup_t v$. This completes the proof. ■

We note that Theorem 5.8.1 agrees with the observations of Kari [106, Obs. 4.7].

### 5.8.2 Right Inverse

Given two binary word operations $\diamond, \star : (\Sigma^*)^2 \to 2^{\Sigma^*}$, we say that $\diamond$ is a *right-inverse* [106, Defn. 4.1] of $\star$ if, for all $u, v, w \in \Sigma^*$,

$$w \in u \star v \iff v \in u \diamond w.$$

Let $\diamond$ be a binary word operation. The word operation $\diamond^r$ given by $u \diamond^r v = v \diamond u$ is called *reversed* $\diamond$ [106].

Let $\pi : \{0,1\}^* \to \{i,d\}^*$ be the morphism given by $\pi(0) = d$ and $\pi(1) = i$. We can repeat the above arguments for right-inverses instead of left-inverses:

**Theorem 5.8.2** *Let $T \subseteq \{0,1\}^*$ be a set of trajectories. Then $\sqcup\!\sqcup_T$ and $(\rightsquigarrow_{\pi(T)})^r$ are right-inverses of each other.*

**Proof.** Let $sym_s : \{0,1\}^* \to \{0,1\}^*$ be the morphism defined by $sym_s(0) = 1$ and $sym_s(1) = 0$. Then it is easy to note (cf., Mateescu *et al.* [147, Rem. 4.9(i)]) that

$$x \in u \sqcup\!\sqcup_t v \iff x \in v \sqcup\!\sqcup_{sym_s(t)} u.$$

Thus, using Theorem 5.8.1, we note that

$$\begin{aligned} x \in u \sqcup\!\sqcup_t v &\iff x \in v \sqcup\!\sqcup_{sym_s(t)} u \\ &\iff v \in x \rightsquigarrow_{\tau(sym_s(t))} u \\ &\iff v \in u (\rightsquigarrow_{\tau(sym_s(t))})^r x. \end{aligned}$$

Thus, the result follows on noting that $\pi \equiv \tau \circ sym_s$. ■

This again agrees with the observations of Kari [106, Obs. 4.4].

We also consider the right-inverse of $\rightsquigarrow_T$ for all $T \subseteq \{i,d\}^*$. However, unlike the left-inverse of $\rightsquigarrow_T$, the right-inverse of $\rightsquigarrow_T$ is again a deletion operation. Let $sym_d : \{i,d\}^* \to \{i,d\}^*$ be the morphism given by $sym_d(i) = d$ and $sym_d(d) = i$.

**Theorem 5.8.3** *Let $T \subseteq \{i, d\}^*$ be a set of trajectories. The operation $\rightsquigarrow_T$ has right-inverse $\rightsquigarrow_{sym_d(T)}$.*

**Proof.** By Theorems 5.8.2 and 5.8.1, we note that

$$\begin{aligned} x \in y \rightsquigarrow_t z \quad &\iff \quad y \in x \sqcup\!\sqcup_{\tau^{-1}(t)} z \\ &\iff \quad z \in y \rightsquigarrow_{\pi(\tau^{-1}(t))} x \end{aligned}$$

The result follows on noting that $\pi \circ \tau^{-1} \equiv sym_d$. ■

We note that Theorem 5.8.3 agrees with the observations of Kari [106, Obs. 4.4].

## 5.9 Conclusions

We have defined deletion along trajectories, and examined its closure properties. Deletion along trajectories is shown to be a useful generalization of the many deletion-like operations which have been studied in the literature. The closure properties of deletion along trajectories differ from that of shuffle on trajectories in that there exist non-regular and non-CF sets of trajectories which define deletion operations which preserve regularity.

We have also demonstrated that shuffle on trajectories and deletion along trajectories form mutual inverses of each other in the sense of Kari [106]. In Chapter 7, we will use the fact that shuffle and deletion along trajectories are mutual inverses of each other to solve language equations involving these operations. In Chapter 6, we will use the inverse characterizations to allow us to prove positive decidability results.

# Chapter 6

# Trajectory-Based Codes

## 6.1 Introduction

The theory of codes is a fundamental area of formal language theory, with many important applications. The class of prefix codes is a particularly important subclass of codes, and is fundamentally linked to the nature of concatenation as the underlying operation. Further research in codes has considered the subclasses of codes which arise from replacing catenation with other, related operations, most notably shuffle (the hypercodes) and insertion (the outfix codes).

In this chapter, we generalize these results by considering $T$-codes. A $T$-code is any language $L$ satisfying the equation $(L \sqcup\!\sqcup_T \Sigma^+) \cap L = \emptyset$. Thus, we consider the natural extension of prefix codes to all operations defined by shuffle on trajectories, and examine the properties of these classes of languages.

The idea of studying general classes of codes has received much attention in the literature (see, e.g., Shyr and Thierrin [186], Jürgensen *et al.* [99] and Jürgensen and Yu [100]). Further, the definition of a $T$-code which we present can also be formulated in dependency theoretic terms (see, e.g., Jürgensen and Konstantinidis [97] for a survey of dependency theory). Some of the results we have obtained can be proven by appealing to dependency theory, however, our proofs are simpler in our restricted situation.

In addition, there are works in the literature which consider the problem of defining codes based on arbitrary binary relations, see, e.g., the work of Jürgensen *et al.* [99] on codes defined by binary relations and Shyr and Thierrin [186] for work on so-called *strict* binary relations. We will see that we can also view $T$-codes as anti-chains under the natural binary relation defined by $T$.

With this research in mind, we nonetheless feel the framework of $T$-codes is useful in that it helps us to see results relating to codes defined by shuffle on trajectories in a new way. The restriction of considering only those codes defined by shuffle on trajectories gives us new insight into these classes, including prefix-, suffix-, bi(pre)fix-, infix-, outfix-, shuffle- and hyper-codes, by focusing our attention to classes of codes which are specific enough to allow reasoning on the associated sets of trajectories, but general enough to encompass all of the above interesting and well-studied classes of codes.

We also feel that introducing the idea of $T$-code will allow more unified results to be obtained on the various classes of codes, since specific conditions on sets of trajectories (i.e., languages) will be easier to obtain than more general conditions on arbitrary relations. In particular, we have obtained results which do not appear to have been considered before in the more general framework of dependency theory or binary relations.

Further, we note that the notion of $T$-codes is useful elsewhere in the study of iterated shuffle and deletion along trajectories, for instance, in analyzing the *shuffle-base* of certain languages. We examine this relationship in Section 8.9. Finally, the study of $T$-codes, much like the study of shuffle on trajectories in general, allows us to examine what assumptions must be made on an operation in order for certain results to follow. We find that even when these assumptions have been studied in the literature, the proofs obtained for the specific cases of shuffle on trajectories are often simpler.

We obtain several interesting results on $T$-codes. We generalize a result relating outfix and hyper-codes and the notion of (embedding-) convexity to all $T$-codes. Further, the known closure properties of shuffle on trajectories allow us to easily conclude positive decidability results for the problem of determining membership in classes of $T$-codes (including maximal $T$-codes), which were previously determined by ad-hoc constructions in the literature.

We note that recently, a more general concept than $T$-codes has been independently introduced by Kari *et al.* [108], motivated by the bonding of strands of DNA and DNA computing. Their framework, called *bond-free properties*, is also a general setting which involves shuffle on trajectories. Generally, the motivations for our work and those of the work by Kari *et al.* are different, and the decidability results which are similar are noted below.

## 6.2 Definitions

Recall that a non-empty language $L$ is a *code* if $u_1u_2\cdots u_m = v_1v_2\cdots v_n$ where $u_i, v_j \in L$ for $1 \leq i \leq m$ and $1 \leq j \leq n$ implies that $n = m$ and $u_i = v_i$ for $1 \leq i \leq n$. For background on codes, we refer the reader to Berstel and Perrin [18], Jürgensen and Konstantinidis [97] or Shyr [184].

We now come to the main definition of this chapter. Let $L \subseteq \Sigma^+$ be a language. Then, for any $T \subseteq \{0,1\}^*$, we say that $L$ is a *$T$-code* if $L$ is non-empty and $(L \sqcup\!\sqcup_T \Sigma^+) \cap L = \emptyset$. If $\Sigma$ is an alphabet and $T \subseteq \{0,1\}^*$, let $\mathcal{P}_T(\Sigma)$ denote the set of all $T$-codes over $\Sigma$. If $\Sigma$ is understood, we will denote the set of $T$-codes over $\Sigma$ by $\mathcal{P}_T$.

There has been much research into the idea of $T$-codes for particular $T \subseteq \{0,1\}^*$, including

(a) prefix codes, corresponding to $T = 0^*1^*$ (concatenation);

(b) suffix codes, corresponding to $T = 1^*0^*$ (anti-catenation);

(c) biprefix (or bifix) codes, corresponding to $T = 0^*1^* + 0^*1^*$ (bi-catenation);

(d) outfix and infix codes, corresponding to $T = 0^*1^*0^*$ (insertion) and $T = 1^*0^*1^*$, (bi-polar insertion) respectively;

(e) shuffle-codes, corresponding to bounded sets of trajectories such as

(e-i) $T = (0^*1^*)^n$ for fixed $n \geq 1$ (prefix codes of index $n$);

(e-ii) $T = (1^*0^*)^n$ for fixed $n \geq 1$ (suffix codes of index $n$);

(e-iii) $T = 1^*(0^*1^*)^n$ for fixed $n \geq 1$ (infix codes of index $n$);

(e-iv) $T = (0^*1^*)^n0^*$ for fixed $n \geq 1$ (outfix codes of index $n$);

(f) hypercodes, corresponding to $T = (0+1)^*$ (arbitrary shuffle);

(g) $k$-codes, corresponding to $T = 0^*1^*0^{\leq k}$ ($k$-catenation, see Kari and Thierrin [114]) for fixed $k \geq 0$; and

(h) for arbitrary $k \geq 1$, codes defined by the sets of trajectories $PP_k = 0^* + (0^*1^*)^{k-1}0^*1^+$, $PS_k = 0^* + 1^+0^*(1^*0^*)^{k-1}$, $PI_k = 0^* + (1^*0^*)^k1^+$, $SI_k = 0^* + 1^+(0^*1^*)^k$, $PB_k = PP_k \cup PS_k$ and $BI_k = PI_k \cup SI_k$, see Long [135], or Ito *et al.* [77] for $PI_1$, $SI_1$.

For a list of references related to (a)–(f), see Jürgensen and Konstantinidis [97, pp. 549–553]. In this chapter, we let

$$H = (0+1)^*, \tag{6.1}$$

$$P = 0^*1^*, \tag{6.2}$$

$$S = 1^*0^*, \tag{6.3}$$

$$I = 1^*0^*1^*, \tag{6.4}$$

$$O = 0^*1^*0^*, \text{ and} \tag{6.5}$$

$$B = P \cup S. \tag{6.6}$$

## 6.3 General Properties of $T$-codes

We can give two alternate characterizations of $T$-codes in terms of the left and right inverses of shuffle on trajectories. These are given via the morphisms $\tau, \pi : \{0,1\}^* \to \{i,d\}^*$ defined by $\tau(0) = i$, $\tau(1) = d$, $\pi(0) = d$ and $\pi(1) = i$. We can easily prove the following two equalities by appealing to Theorems 5.8.1 and 5.8.2. In particular, we have for all $T \subseteq \{0,1\}^*$, and all $\Sigma$,

$$\mathcal{P}_T(\Sigma) = \{L \subseteq \Sigma^+ : (L \rightsquigarrow_{\tau(T)} \Sigma^+) \cap L = \emptyset\}, \tag{6.7}$$

$$\mathcal{P}_T(\Sigma) = \{L \subseteq \Sigma^+ : L \rightsquigarrow_{\pi(T)} L \subseteq \{\epsilon\}\}. \tag{6.8}$$

For some particular $T$, these characterizations are well-known, e.g., (6.7) for $T = 0^*1^*$ is given by Berstel and Perrin [18, Prop. II.1.1.(ii)].

We now note that the term $T$-code is somewhat of a misnomer: some $T$-codes are not codes. However, we feel that as $T$-codes are the natural analogues of prefix codes when catenation is replaced by $\sqcup\!\sqcup_T$, the term $T$-code is appropriate. The following example shows how $T$-codes can fail to be codes:

**Example 6.3.1:** Let $T = (01)^*$. Then $\sqcup\!\sqcup_T$ corresponds to perfect shuffle (also known as balanced literal shuffle). Then note that $L = \{aa, bb, aabb\}$ is a $T$-code: there is no way to perfectly shuffle $aa$ (resp., $bb$) and any other word of length 2 to get $aabb$. However, $L$ is not a code: $aa \cdot bb = aabb$. □

The following states that more restrictive sets of trajectories (potentially) result in more languages being $T$-codes; the proof is immediate:

**Lemma 6.3.2** *Let $T_1 \subseteq T_2 \subseteq \{0,1\}^*$. Then for all $\Sigma$, $\mathcal{P}_{T_1}(\Sigma) \supseteq \mathcal{P}_{T_2}(\Sigma)$.*

By the fact that all prefix codes are codes, we conclude the following, which complements Example 6.3.1:

**Corollary 6.3.3** *Let $T \supseteq 0^*1^*$. Then every $T$-code is a code.*

Let $\mathcal{P}_{\text{CODE}}$ denote the set of all codes. We now show that for all $T \subseteq \{0,1\}^*$, $\mathcal{P}_T \neq \mathcal{P}_{\text{CODE}}$. We will require the following well-known characterization of two element codes (see, e.g., Berstel and Perrin [18, Cor. 2.9]):

**Theorem 6.3.4** *Let $L = \{x_1, x_2\} \subseteq \Sigma^+$. Then $L$ is not a code if and only if there exist $z \in \Sigma^+$, $i, j \in \mathbb{N}^+$ such that $x_1 = z^i$ and $x_2 = z^j$.*

**Lemma 6.3.5** *Let $T \subseteq \{0,1\}^*$. Then $\mathcal{P}_T(\Sigma) \neq \mathcal{P}_{\text{CODE}}(\Sigma)$ for all $\Sigma$ with $|\Sigma| > 1$.*

**Proof.** Let $T \subseteq \{0,1\}^*$. If $T \subseteq 0^* + 1^*$, then $\mathcal{P}_T = \mathcal{P}_\emptyset = 2^{\Sigma^+} - \{\emptyset\}$ (the first equality will become clear after Theorem 6.3.7 below), which is clearly not the set of codes.

Thus, we can assume that there is some $t \in T$ with $|t|_1, |t|_0 > 0$. Let $n = |t|_0$. Consider that $t \in 0^n \sqcup\!\sqcup_t \{0,1\}^+$. Thus $L = \{t, 0^n\} \subseteq \{0,1\}^+$ is not a $T$-code.

If $L$ is not a code, then $t$ and $0^n$ are powers of the same word, i.e., $t \in 0^*$. This contradicts our choice of $t$. Thus, $L$ is a code. ■

We also observe that $\mathcal{P}_{T_1} \cap \mathcal{P}_{T_2} = \mathcal{P}_{T_1 \cup T_2}$. We note that the dual case does not hold. In the case of $\mathcal{P}_{T_1 \cap T_2}$, we have the inclusion $\mathcal{P}_{T_1} \cap \mathcal{P}_{T_2} \subseteq \mathcal{P}_{T_1 \cap T_2}$. But of course equality does not hold in general. For example, with $T_1 = 0^*1^*$ and $T_2 = 1^*0^*$, $\mathcal{P}_{T_1 \cap T_2} = \mathcal{P}_{0^*+1^*} = \mathcal{P}_\emptyset = 2^{\Sigma^+} - \{\emptyset\}$ (the second equality will be established in Theorem 6.3.7 below). However, $\mathcal{P}_{T_1} \cap \mathcal{P}_{T_2} = \mathcal{P}_{T_1 \cup T_2}$, the set of biprefix codes.

We can also ask if $T_1 \subset T_2$ ($\subset$ denotes proper inclusion) implies that $\mathcal{P}_{T_1} \supset \mathcal{P}_{T_2}$. The answer is yes, as long as the difference between $T_2$ and $T_1$ contains non-unary words.

**Theorem 6.3.6** *Let $T_1 \subset T_2$ be such that $(T_2 - T_1) \cap \overline{0^* + 1^*} \neq \emptyset$. Then for all $\Sigma$ with $|\Sigma| \geq 2$, $\mathcal{P}_{T_1}(\Sigma) \supset \mathcal{P}_{T_2}(\Sigma)$.*

**Proof.** Let $t \in (T_2 - T_1) \cap \overline{0^* + 1^*}$. Let $t_0, t_1$ be defined by $t_0 = 0^{|t|_0}$ and $t_1 = 1^{|t|_1}$. Then note that $t_0, t_1 \neq \epsilon$, by our choice of $t$. Thus, we have that $\{t, t_0\} \subseteq \{0,1\}^+$. We claim that $L_t = \{t, t_0\} \in \mathcal{P}_{T_1} - \mathcal{P}_{T_2}$.

To see that $L_t \notin \mathcal{P}_{T_2}$, note that $t \in t_0 \sqcup\!\sqcup_t t_1$. As $t \in T_2$ and $t_1 \neq \epsilon$, $L_t$ is not a $T_2$-code. Assume that $L_t$ is not a $T_1$-code. As $|t| > |t_0|$, the only way that $L_t$ can fail to be a $T_1$-code is if there exists $x \in \{0,1\}^+$ such that $t \in t_0 \sqcup\!\sqcup_{T_1} x$. By definition, as $|t|_0 = |t_0|$, we must have that $x = t_1 = 1^{|t|_1}$. But $t \in t_0 \sqcup\!\sqcup_{T_1} t_1$ only if $t \in T_1$, which is not the case. ■

**Theorem 6.3.7** *Let $T_1 \subset T_2$ and $T_2 - T_1 \subseteq 1^* + 0^*$. Then for all $\Sigma$ with $|\Sigma| > 1$, $\mathcal{P}_{T_1}(\Sigma) = \mathcal{P}_{T_2}(\Sigma)$.*

**Proof.** Assume, contrary to what we want to prove, that $L \subseteq \Sigma^+$ is a $T_1$-code which is not a $T_2$-code. As $L$ is not a $T_2$-code, there exist $x, z \in L$, $y \in \Sigma^+$ and $t \in T_2$ such that $z \in x \sqcup\!\sqcup_t y$. As $L$ is a $T_1$-code, $z \notin x \sqcup\!\sqcup_{T_1} y$. Thus $t \notin T_1$. By assumption, this implies that $t \in 1^* + 0^*$.

If $t \in 1^*$, then by definition of $\sqcup\!\sqcup_T$, $z \in x \sqcup\!\sqcup_t y$ implies that $x = \epsilon$, contrary to our choice of $L$. If $t \in 0^*$, then by definition, $y = \epsilon$, contrary to our choice of $y$. In either case, we have arrived at a contradiction. ■

Thus, we have completely characterized when reducing a set of trajectories corresponds to an increase in the languages which are $T$-codes. In particular, we note the following corollary:

**Corollary 6.3.8** *Let $T_1, T_2 \subseteq \{0,1\}^*$ be regular sets of trajectories. Then it is decidable whether $\mathcal{P}_{T_1} = \mathcal{P}_{T_2}$.*

**Proof.** We note that $\mathcal{P}_{T_1} = \mathcal{P}_{T_2}$ if and only if $(T_1 - T_2) \cup (T_2 - T_1) \subseteq 0^* + 1^*$. Since $T_1, T_2$ are regular, so is $(T_1 - T_2) \cup (T_2 - T_1)$, and the inclusion is decidable. ■

We now examine further questions of decidability.

**Lemma 6.3.9** *Let $T \subseteq \{0,1\}^*$ be a fixed CF set of trajectories. Then given a regular language $L$, it is decidable whether $L$ is a $T$-code.*

**Proof.** Since $L$ is regular and $T$ is a CFL, $L \sqcup\!\sqcup_T \Sigma^+$, and $(L \sqcup\!\sqcup_T \Sigma^+) \cap L$ are CFLs. Thus, we can test whether $(L \sqcup\!\sqcup_T \Sigma^+) \cap L = \emptyset$, which precisely defines $L$ being a $T$-code. ■

This result can also be proved using dependency theory. As every $T \subseteq \{0,1\}^*$ defines a 3-dependence system, and every context-free $T$ defines a dependence system whose associated support can be accepted by a 3-tape PDA, the problem of determining membership in $\mathcal{P}_T$ is decidable; see Jürgensen and Konstantinidis [97, Sect. 9] for details. Further, Kari *et al.* [108, Thm. 4.7] establish a similar decidability result in their framework of *bond-free properties*. When translated to our setting, it states that given $T$, $R$ regular, we can decide if $R \in \mathcal{P}_T$.

A class of languages $\mathcal{C}$ is said to have *decidable membership problem* if, given $L \subseteq \Sigma^*$ with $L \in \mathcal{C}$, it is decidable whether $x \in L$ for an arbitrary $x \in \Sigma^*$. We have the following positive decidability result:

**Lemma 6.3.10** *Let $\mathcal{C}$ be a class of languages with decidable membership. Let $T \subseteq \{0,1\}^*$ be a set of trajectories such that $T \in \mathcal{C}$. Then given a finite language $F$, it is decidable whether $F \in \mathcal{P}_T$.*

**Proof.** Let $F \subseteq \Sigma^+$ be a finite set. Let $n = \max\{|x| : x \in F\}$. Since membership in $T$ is decidable, we can test all $t \in \{0,1\}^{\leq n}$ for membership in $T$. Thus, we can effectively compute $T^{\leq n} = T \cap \{0,1\}^{\leq n}$. It is easily observed that $F \cap (F \sqcup\!\sqcup_{T^{\leq n}} L) = F \cap (F \sqcup\!\sqcup_T L)$ for all $L$.

Since $F, T^{\leq n}, \Sigma^+$ are regular, we can test $F \cap (F \sqcup\!\sqcup_{T^{\leq n}} \Sigma^+) = \emptyset$. Thus, the result follows. ■

We conclude with the following method of constructing a $T$-code from an arbitrary language.

**Lemma 6.3.11** *Let $T \subseteq \{0,1\}^*$. Let $L \subseteq \Sigma^+$ be a non-empty language. Then $L_0 = L - (L \sqcup\!\sqcup_T \Sigma^+) \in \mathcal{P}_T(\Sigma)$.*

**Proof.** As $L_0 \subseteq L$ and $\sqcup\!\sqcup_T$ is a monotone operation, $(L_0 \sqcup\!\sqcup_T \Sigma^+) \subseteq (L \sqcup\!\sqcup_T \Sigma^+)$. Thus, $L_0 \cap (L_0 \sqcup\!\sqcup_T \Sigma^+) \subseteq L_0 \cap (L \sqcup\!\sqcup_T \Sigma^+)$ and $L_0 \cap (L \sqcup\!\sqcup_T \Sigma^+) = \emptyset$ by definition of $L_0$. ■

The following is proven in exactly the same manner as Lemma 6.3.11:

**Lemma 6.3.12** *Let $T \subseteq \{0,1\}^*$. Let $L \subseteq \Sigma^+$ be a non-empty language. Then $L_0 = L - (L \rightsquigarrow_{\tau(T)} \Sigma^+) \in \mathcal{P}_T(\Sigma)$.*

## 6.4 The Binary Relation defined by Trajectories

We can also define $T$-codes by appealing to a definition based on binary relations. In particular, for $T \subseteq \{0,1\}^*$, define $\omega_T$ as follows: for all $x, y \in \Sigma^*$,

$$x \; \omega_T \; y \iff y \in x \sqcup\!\sqcup_T \Sigma^*.$$

Then it is clear that $L \subseteq \Sigma^+$ is a $T$-code if and only if $L$ is an anti-chain under $\omega_T$ (i.e, $x, y \in L$ and $x \; \omega_T \; y$ implies $x = y$).

We note that the relation analogous to $\omega_T$ for infinite words and $\omega$-trajectories was defined by Kadrie *et al.* [101], and its properties were briefly investigated. Kadrie *et al.* do not investigate the

analogous relation with the same amount of detail as below and do not appear to be motivated by the theory of codes.

We immediately note that if $T_1, T_2 \subseteq \{0,1\}^*$ are sets of trajectories, there is not necessarily a set of trajectories $T$ such that $\omega_T = \omega_{T_1} \cap \omega_{T_2}$, i.e., such that $x\ \omega_T\ y \iff (x\ \omega_{T_1}\ y) \wedge (x\ \omega_{T_2}\ y)$. For instance, for $P = 0^*1^*$ and $S = 1^*0^*$, the relation $\omega_P \cap \omega_S$ is given by $\leq_d$, where $x \leq_d y$ if and only if there exist $u, v \in \Sigma^*$ such that $y = xu = vx$. This relation cannot be represented by a set of trajectories:

**Lemma 6.4.1** *For all $T \subseteq \{0,1\}^*$, $\omega_T \neq \leq_d$.*

**Proof.** Assume that there exists $T \subseteq \{0,1\}^*$ such that $\omega_T = \leq_d$. Consider $L_0 = \{0, 00\}$. As $0 \leq_d 00$, we must have that $0\ \omega_T\ 00$. Thus, $00 \in 0 \sqcup\!\sqcup_T 0$ and $\{01, 10\} \cap T \neq \emptyset$. Thus, without loss of generality assume that $01 \in T$. The case $10 \in T$ is similar.

Consider now $L_1 = \{0, 01\}$. We observe that $L_1$ is an anti-chain under $\leq_d$, i.e., $0 \leq_d 01$ does not hold. However, $01 \in 0 \sqcup\!\sqcup_T 1$. Thus, $0\ \omega_T\ 01$, and $\omega_T \neq \leq_d$. ■

For a discussion of $\leq_d$, see Shyr [184, Ch. 8]. We now recall some of the properties of the binary relations $\omega_T$ that will be useful. In what follows, we will refer to $T$ having a property $P$ if and only if $\omega_T$ has property $P$.

### 6.4.1 Anti-symmetry

Recall that a binary relation $\rho$ is *anti-symmetric* if $x\ \rho\ y$ and $y\ \rho\ x$ implies $x = y$. We note that $\omega_T$ always gives an anti-symmetric binary relation:

**Lemma 6.4.2** *Let $T \subseteq \{0,1\}^*$. The relation $\omega_T$ is anti-symmetric.*

**Proof.** Let $x, y \in \Sigma^*$ be such that $x\ \omega_T\ y\ \omega_T\ x$. Then let $t_1, t_2 \in T$ and $\alpha, \beta \in \Sigma^*$ be such that $x \in y \sqcup\!\sqcup_{t_1} \alpha$ and $y \in x \sqcup\!\sqcup_{t_2} \beta$. By definition of shuffle on trajectories, $|x| = |y| + |\alpha|$ and $|y| = |x| + |\beta|$. Thus, $|\alpha| = |\beta| = 0$, i.e., $\alpha = \beta = \epsilon$. But now $x \in y \sqcup\!\sqcup_{t_1} \epsilon$, which implies that $x = y$, again by definition of shuffle on trajectories. ■

### 6.4.2 Reflexivity

Recall that a binary relation $\rho$ on $\Sigma^*$ is *reflexive* if $x \rho x$ for all $x \in \Sigma^*$.

**Lemma 6.4.3** *Let $T \subseteq \{0,1\}^*$. Then $T$ is reflexive if and only if $0^* \subseteq T$.*

**Proof.** Let $0^* \subseteq T$. Then $x \in x \sqcup\!\sqcup_{0^{|x|}} \epsilon$, i.e., $x \; \omega_T \; x$. Thus $\omega_T$ is reflexive. For the converse, let $x \in x \sqcup\!\sqcup_T \Sigma^*$ for all $x \in \Sigma^*$. Then clearly $0^{|x|} \in T$ for all $x \in \Sigma^*$, which implies $0^* \subseteq T$. ■

**Corollary 6.4.4** *Given a CF set $T \subseteq \{0,1\}^*$ of trajectories, it is decidable whether $T$ is reflexive.*

**Proof.** Let $T' = T \cap 0^*$, which is a unary CFL, and thus regular. In fact, if $T$ is effectively context-free, then $T'$ is effectively regular. We can then test the equality $0^* = T'$. ■

### 6.4.3 Positivity

A binary relation $\rho$ on $\Sigma^*$ is said to be *positive* if $\epsilon \rho x$ for all $x \in \Sigma^*$.

**Lemma 6.4.5** *Let $T \subseteq \{0,1\}^*$. Then $T$ is positive if and only if $1^* \subseteq T$.*

**Proof.** Let $1^* \subseteq T$. Then $u \in \epsilon \sqcup\!\sqcup_{1^{|u|}} u$ for all $u \in \Sigma^*$, whereby $\epsilon \; \omega_T \; u$, as $1^{|u|} \in T$. The reverse implication is similarly established. ■

**Corollary 6.4.6** *Given a CF set $T \subseteq \{0,1\}^*$ of trajectories, it is decidable whether $T$ is positive.*

### 6.4.4 ST-Strictness

Shyr and Thierrin [186] define the concept of a *strict* binary relation. To avoid confusion with the concept of a *strict ordering* (see, e.g., Choffrut and Karhumäki [24, Sect. 7.1]), we will call a binary relation $\rho$ on $\Sigma^*$ *ST-strict* if it satisfies the following four properties:

(a) $\rho$ is reflexive;

(b) $\rho$ is positive;

(c) for all $u, v \in \Sigma^*$, $u \rho v$ implies $|u| \leq |v|$;

(d) for all $u, v \in \Sigma^*$, $u \rho v$ and $|u| = |v|$ implies $u = v$.

We now consider $T$ such that $\omega_T$ is ST-strict. We first note that conditions (c) and (d) are satisfied by all $T$. Indeed, if $u \, \omega_T \, v$, then $v \in u \sqcup\!\sqcup_T \Sigma^*$, which implies that $|v| \geq |u|$. Further, if $|u| = |v|$, then $u \, \omega_T \, v$ implies that $v \in u \sqcup\!\sqcup_T \epsilon$, which implies that $u = v$.

Thus, as we already have necessary and sufficient conditions on $T$ being reflexive and positive, the following results are immediate:

**Corollary 6.4.7** *Let $T \subseteq \{0,1\}^*$. Then $T$ is ST-strict if and only if $0^* + 1^* \subseteq T$.*

**Corollary 6.4.8** *Given a CF set $T \subseteq \{0,1\}^*$ of trajectories, it is decidable whether $T$ is ST-strict.*

**Corollary 6.4.9** *Let $T_1, T_2 \subseteq \{0,1\}^*$ be ST-strict. Then $\mathcal{P}_{T_1} = \mathcal{P}_{T_2}$ if and only if $T_1 = T_2$.*

### 6.4.5 Cancellativity

A binary relation $\rho$ on $\Sigma^*$ is said to be *left-cancellative* (resp., *right-cancellative*) if $uv \, \rho \, ux$ implies $v \, \rho \, x$ (resp., $vu \, \rho \, xu$ implies $v \, \rho \, x$) for all $u, v, x \in \Sigma^*$. The relation $\rho$ is *cancellative* if it is both left- and right-cancellative.

Given $T \subseteq \{0,1\}^*$, we define two sets of trajectories, $s(T), p(T) \subseteq \{0,1\}^*$, as follows:

$$\begin{aligned} p(T) &= \{t_1 1^j \ : \ t_1 t_2 \in T, 0 \leq j \leq |t_2|\}, \\ s(T) &= \{1^j t_2 \ : \ t_1 t_2 \in T, 0 \leq j \leq |t_1|\}. \end{aligned}$$

**Lemma 6.4.10** *Let $T \subseteq \{0,1\}^*$. Then $T$ is left-cancellative (resp., right-cancellative) if $s(T) \subseteq T$ (resp., $p(T) \subseteq T$).*

**Proof.** We establish the result for left-cancellativity only; the other case is symmetric. Let $s(T) \subseteq T$. Then let $u, v, x \in \Sigma^*$ be such that $uv \, \omega_T \, ux$. Let $t \in T$ and $\alpha \in \Sigma^*$ chosen so that $ux \in uv \sqcup\!\sqcup_t \alpha$. Write $t = t_1 t_2$ and $\alpha = \alpha_1 \alpha_2$ so that $ux \in (u \sqcup\!\sqcup_{t_1} \alpha_1)(v \sqcup\!\sqcup_{t_2} \alpha_2)$. Let $\beta_1, \beta_2 \in \Sigma^*$ be chosen so that

$\beta_1 \in u \sqcup\!\sqcup_{t_1} \alpha_1$, $\beta_2 \in v \sqcup\!\sqcup_{t_2} \alpha_2$ and $ux = \beta_1\beta_2$. As $|\beta_1| = |u| + |\alpha_1| \geq |u|$, there exists $\gamma \in \Sigma^*$ such that $u\gamma = \beta_1$ and $x = \gamma\beta_2$. Note that $|\gamma| \leq |\beta_1| = |t_1|$. Thus,

$$x \in \gamma\,(v \sqcup\!\sqcup_{t_2} \alpha_2).$$

Let $t_3 = 1^{|\gamma|}t_2 \in s(T)$. By assumption, $t_3 \in T$. Further,

$$x \in v \sqcup\!\sqcup_{t_3} \gamma\,\alpha_2.$$

We conclude that $v\ \omega_T\ x$. ■

**Corollary 6.4.11** *Let $T \subseteq \{0,1\}^*$. If $s(T) \cup p(T) \subseteq T$, then $T$ is cancellative.*

We now consider a condition of Jürgensen *et al.* [99]. Say that a binary relation $\rho$ on $\Sigma^*$ is *leviesque* if $uv\ \rho\ xy$ implies that $u\ \rho\ x$ or $v\ \rho\ y$, for all $u, v, x, y \in \Sigma^*$.

**Lemma 6.4.12** *Let $T \subseteq \{0,1\}^*$. If $s(T) \cup p(T) \subseteq T$, then $T$ is leviesque.*

**Proof.** Let $rs\ \omega_T\ xy$. Then there exist $t \in T$ and $\alpha \in \Sigma^*$ such that $xy \in rs \sqcup\!\sqcup_t \alpha$. Then there exist factorizations $t = t_1t_2$, $\alpha = \alpha_1\alpha_2$ such that $xy \in (r \sqcup\!\sqcup_{t_1} \alpha_1)(s \sqcup\!\sqcup_{t_2} \alpha_2)$. Let $\beta_1, \beta_2 \in \Sigma^*$ be such that $\beta_1 \in r \sqcup\!\sqcup_{t_1} \alpha_1$, $\beta_2 \in s \sqcup\!\sqcup_{t_2} \alpha_2$ and $xy = \beta_1\beta_2$. There are two cases:

(i) If $|x| \geq |\beta_1|$, then there exists $\gamma \in \Sigma^*$ such that $x = \beta_1\gamma$ and $\gamma\, y = \beta_2$. Note that $|\gamma| \leq |\beta_2| = |t_2|$. Consider that $x = \beta_1\gamma \in (r \sqcup\!\sqcup_{t_1 1^{|\gamma|}} \alpha_1\gamma)$. As $t_1 1^{|\gamma|} \in p(T) \subseteq T$, $r\ \omega_T\ x$.

(ii) If $|x| \leq |\beta_1|$, there exists $\gamma \in \Sigma^*$ such that $x\gamma = \beta_1$ and $y = \gamma\beta_2$. Note that $|\gamma| \leq |\beta_1| = |t_1|$. In this case, $y = \gamma\beta_2 \in (s \sqcup\!\sqcup_{1^{|\gamma|}t_2} \gamma\,\alpha_2)$. Thus, as $1^{|\gamma|}t_2 \in s(T) \subseteq T$, we have $s\ \omega_T\ y$.

Thus, $rs\ \omega_T\ xy$ implies $(r\ \omega_T\ x)$ or $(s\ \omega_T\ y)$. ■

### 6.4.6 Compatibility

Let $\rho$ be a binary relation on $\Sigma^*$. Then we say that $\rho$ is *left-compatible* (resp., *right-compatible*) if, for all $u, v, w \in \Sigma^*$, $u\ \rho\ v$ implies that $wu\ \rho\ wv$ (resp., $uw\ \rho\ vw$). If $\rho$ is both left- and right-compatible, we say it is *compatible*.

**Lemma 6.4.13** *Let $T \subseteq \{0,1\}^*$. Then $T$ is right-compatible (resp., left-compatible) if and only if $T0^* \subseteq T$ (resp., $0^*T \subseteq T$).*

**Proof.** We establish the result for right-compatibility. The result for left-compatibility is symmetrical.

Let $T0^* \subseteq T$. Let $u, v, w \in \Sigma^*$ with $u\ \omega_T\ v$. Then there exist $t \in T$ and $\alpha \in \Sigma^*$ such that $v \in u \sqcup\!\sqcup_t \alpha$. As $t' = t0^{|w|} \in T$, $vw \in uw \sqcup\!\sqcup_{t'} \alpha$. Thus $uw\ \omega_T\ vw$.

Assume that $T0^*$ is not a subset of $T$. Then there exist $t \in T$ and $i \in \mathbb{N}$ such that $t0^i \notin T$. Let $j = |t|_0$ and $k = |t|_1$. Consider that $0^j\ \omega_T\ t$, as $t \in 0^j \sqcup\!\sqcup_t 1^k$. However, $0^j \cdot 0^i\ \omega_T\ t \cdot 0^i$ does not hold, as $t0^i \in 0^{j+i} \sqcup\!\sqcup_T 1^k$ would imply that $t0^i \in T$. Thus, $T$ is not right-compatible. ■

The following corollary is immediate; it is identical to the condition that $T0^* \cup 0^*T \subseteq T$:

**Corollary 6.4.14** *Let $T \subseteq \{0,1\}^*$. Then $T$ is compatible if and only if $0^*T0^* \subseteq T$.*

**Corollary 6.4.15** *Given a regular set $T \subseteq \{0,1\}^*$ of trajectories, it is decidable whether $T$ is (left- or right-) compatible.*

**Lemma 6.4.16** *Given an LCF set $T \subseteq \{0,1\}^*$ of trajectories, it is undecidable whether $T$ is (left- or right-) compatible.*

**Proof.** We prove only the case for left-compatibility; the other cases are similar and are left to the reader. We apply a meta-theorem of Hunt and Rosenkrantz (Theorem 2.5.3). First, we note that $T = \{0,1\}^*$ is left-compatible.

Let $T = \{0^n1^n : n \geq 0\}$. We claim that there is no LCF set $T' \subseteq \{0,1\}^*$ of trajectories and trajectory $t \in \{0,1\}^+$ such that $T = T'/t$. Assume that there were such $T', t$. Then as $\epsilon \in T = T'/t$, we must have $t \in T'$. As $T'$ is left-compatible, we have that $0t \in T'$. Thus $0 \in T'/t = T$, a contradiction. Thus, the set

$$\{T : \exists \text{ left-compatible LCF } T' \subseteq \{0,1\}^*, t \in \{0,1\}^+ \text{ such that } T = T'/t\}$$

is a proper subset of the LCFLs. Therefore, we may apply Theorem 2.5.3, and it is undecidable whether a given LCF set of trajectories is left-compatible. ■

Recall the definitions of $P$, $S$ and $O$ given by (6.2), (6.3) and (6.5). Let $\mathcal{P}_P, \mathcal{P}_S, \mathcal{P}_O$ be the class of prefix, suffix and outfix codes. We can conclude the following corollary about positive $T$ which satisfy compatibility conditions. Parts (a) and (b) of the following result have been established for all partial orders by Jürgensen *et al.* [99]; the proofs are immediate in our case:

**Corollary 6.4.17** *Let $T \subseteq \{0,1\}^*$ be positive. Then the following hold:*

*(a) if $T$ is left-compatible, then $\mathcal{P}_T \subseteq \mathcal{P}_P$;*

*(b) if $T$ is right-compatible, then $\mathcal{P}_T \subseteq \mathcal{P}_S$;*

*(c) if $T$ is compatible, then $\mathcal{P}_T \subseteq \mathcal{P}_O$.*

*Furthermore, in each case equality of the classes holds if and only if it holds for the sets of trajectories involved.*

**Proof.** We prove (b); the rest are similar. If $T$ is positive then $1^* \subseteq T$. If $T$ is right compatible, then $T0^* \subseteq T$. Thus, $S = 1^*0^* \subseteq T$. The inclusions thus hold by Lemma 6.3.2; for the equalities, we note that $P$, $S$, $O$ are ST-strict and for each of (a),(b) and (c), $T$ is also ST-strict. ■

### 6.4.7 Transitivity

Recall that a binary relation $\rho$ on $\Sigma^*$ is said to be *transitive* if $x \rho y$ and $y \rho z$ imply that $x \rho z$ for all $x, y, z \in \Sigma^*$. We now consider conditions on $T$ which will ensure that $\omega_T$ is a transitive relation. Transitivity is often, but not always, a property of the binary relations defining the classic code classes. For instance, both bi-prefix and outfix codes are defined by binary relations which are not transitive, and hence not a partial order. We now give necessary and sufficient conditions on a set $T$ of trajectories defining a transitive binary relation.

First, we define three morphisms we will need. Let $D = \{x, y, z\}$ and $\varphi, \sigma, \psi : D^* \to \{0,1\}^*$

be the morphisms given by

$$\begin{array}{rclrclrcl} \varphi(x) & = & 0, & \sigma(x) & = & 0, & \psi(x) & = & 0, \\ \varphi(y) & = & 0, & \sigma(y) & = & 1, & \psi(y) & = & 1, \\ \varphi(z) & = & 1, & \sigma(z) & = & \epsilon, & \psi(z) & = & 1. \end{array}$$

Note that these morphisms are similar to the substitutions defined by Mateescu *et al.* [147], whose purpose is to give necessary and sufficient conditions on a set $T$ of trajectories defining an associative operation. Indeed, our condition is a weakening of their conditions, which, intuitively, reflects the fact that any associative operation $\sqcup\!\sqcup_T$ defines a transitive binary relation $\omega_T$ (note, however, that $T = 1^*0^*1^*$ is transitive but not associative).

**Theorem 6.4.18** *Let $T \subseteq \{0,1\}^*$. Then $T$ is transitive if and only if*

$$\psi(\varphi^{-1}(T) \cap \sigma^{-1}(T)) \subseteq T. \tag{6.9}$$

**Proof.** ($\Leftarrow$): Let $T$ define a transitive binary relation. Let $w \in \psi(\varphi^{-1}(T) \cap \sigma^{-1}(T))$. Then there exist $t_1, t_2 \in T$ such that $w \in \psi(\varphi^{-1}(t_1) \cap \sigma^{-1}(t_2))$. Let $t \in \varphi^{-1}(t_1) \cap \sigma^{-1}(t_2)$ be chosen so that $w \in \psi(t)$.

Consider $t_1$. Let $n \in \mathbb{N}$ and $\alpha_i, \beta_i \in \mathbb{N}$ be chosen for $1 \leq i \leq n$ so that

$$t_1 = \prod_{i=1}^{n} 0^{\alpha_i} 1^{\beta_i}.$$

Note that

$$\varphi^{-1}(t_1) = \prod_{i=1}^{n} (x+y)^{\alpha_i} z^{\beta_i}.$$

As $t \in \varphi^{-1}(t_1)$ and $t \in \sigma^{-1}(t_2)$, by definition of $\sigma$, we must have that $t_2 = \prod_{i=1}^{n} s_i$ for $s_i \in \{0,1\}^*$ satisfying $|s_i| = \alpha_i$. Thus, we have that $|t_2| = |t_1|_0$. Furthermore, $t \in (x^{p_1} \sqcup\!\sqcup_{t_2} y^{p_2}) \sqcup\!\sqcup_{t_1} z^{p_3}$, where $p_1 = |t_2|_0$, $p_2 = |t_2|_1$ and $p_3 = |t_1|_1$. Consider now that $w \in \psi(t)$, so that

$$w \in (0^{p_1} \sqcup\!\sqcup_{t_2} 1^{p_2}) \sqcup\!\sqcup_{t_1} 1^{p_3}.$$

Clearly, $0^{p_1} \sqcup\!\sqcup_{t_2} 1^{p_2} = t_2$. Thus, $w \in t_2 \sqcup\!\sqcup_{t_1} 1^{p_3}$, as well. By definition, we then have that $0^{p_1}\ \omega_T\ t_2\ \omega_T\ w$. By the transitivity of $T$, $0^{p_1}\ \omega_T\ w$, i.e.,

$$w \in 0^{p_1} \sqcup\!\sqcup_T \{0, 1\}^*.$$

Note that $|w|_1 = p_2 + p_3$ and $|w|_0 = p_1$. The only word $v$ over $\{0, 1\}$ such that $w \in 0^{p_1} \sqcup\!\sqcup_T v$ is $v = 1^{p_2+p_3}$ (regardless of $T$). That is, $w \in 0^{p_1} \sqcup\!\sqcup_T 1^{p_2+p_3}$. But from this, we must have that $w \in T$. Thus, we have that $\psi(\varphi^{-1}(T) \cap \sigma^{-1}(T)) \subseteq T$.

($\Rightarrow$): Assume that $\psi(\varphi^{-1}(T) \cap \sigma^{-1}(T)) \subseteq T$. Let $u, v, w \in \Sigma^*$ be such that $u\,\omega_T\,v$ and $v\,\omega_T\,w$. We wish to show that $u\ \omega_T\ w$. Let $t_1, t_2 \in T$ and $\theta_1, \theta_2 \in \Sigma^*$ be such that $w \in v \sqcup\!\sqcup_{t_1} \theta_1$ and $v \in u \sqcup\!\sqcup_{t_2} \theta_2$. Thus, $w \in (u \sqcup\!\sqcup_{t_2} \theta_2) \sqcup\!\sqcup_{t_1} \theta_1$. Note then that $|t_1|_0 = |t_2|$. Let $n \in \mathbb{N}$ and $\alpha_i, \beta_i \in \mathbb{N}$ be chosen for $1 \leq i \leq n$ so that

$$t_1 = \prod_{i=1}^{n} 0^{\alpha_i} 1^{\beta_i}.$$

Furthermore, let $t_2 = \prod_{i=1}^{n} s_i$ be so that $|s_i| = \alpha_i$ for all $1 \leq i \leq n$. For all $1 \leq i \leq n$, let $\eta_i$ be the word obtained from $s_i$ by replacing 0 with $x$ and 1 with $y$, i.e., $\{\eta_i\} = \sigma^{-1}(s_i) \cap \{x, y\}^*$. Then let

$$t = \prod_{i=1}^{n} \eta_i z^{\beta_i}.$$

We can verify that $\varphi(t) = t_1$ and $\sigma(t) = t_2$. Thus, $t \in \varphi^{-1}(t_1) \cap \sigma^{-1}(t_2)$. Let $t' = \psi(t)$. By assumption, $t' \in T$, and we further note that

$$t' = \prod_{i=1}^{n} s_i 1^{\beta_i}.$$

We now define a morphism $h : D^* \to \{0, 1\}^*$ given by $h(x) = \epsilon$, $h(y) = 0$ and $h(z) = 1$. Let $\theta \in \theta_2 \sqcup\!\sqcup_{h(t)} \theta_1$. Then we can verify that $w \in (u \sqcup\!\sqcup_{t_2} \theta_2) \sqcup\!\sqcup_{t_1} \theta_1 \subseteq u \sqcup\!\sqcup_{t'} \theta \subseteq u \sqcup\!\sqcup_T \Sigma^*$. Thus, $u\ \omega_T\ w$ as required. ■

**Remark 6.4.19** *As an alternate formulation for Theorem 6.4.18, we note that, for all $T \subseteq \{0, 1\}^*$, $T$ is transitive if and only if $T \sqcup\!\sqcup_T 1^* \subseteq T$. The reader can verify this by establishing that $T \sqcup\!\sqcup_T 1^* = \psi(\varphi^{-1}(T) \cap \sigma^{-1}(T))$ holds for all $T \subseteq \{0, 1\}^*$.*

**Corollary 6.4.20** *Given a regular set $T \subseteq \{0,1\}^*$ of trajectories, it is decidable whether $T$ is transitive.*

**Proof.** Since the regular languages are closed under morphism and inverse morphism, and inclusion of regular languages is decidable, we can determine whether the inclusion (6.9) holds. ■

The following decidability result also holds, since we can determine whether $T \supseteq 0^*$ and (6.9) hold if $T$ is regular:

**Corollary 6.4.21** *Given a regular set $T \subseteq \{0,1\}^*$ of trajectories, it is decidable whether $\omega_T$ is a partial order.*

We now turn to undecidability. We will use PCP, which we defined in Section 2.5.

**Theorem 6.4.22** *Given a CF set $T \subseteq \{0,1\}^*$ of trajectories, it is undecidable whether $T$ is transitive.*

**Proof.** Let $P = (u_1, u_2, \ldots, u_n; v_1, v_2, \ldots, v_n)$ be a PCP instance. Define

$$\begin{aligned} L_1 &= \{01^{i_1}01^{i_2}\cdots 01^{i_m}0^{n+1}1^{n+1}u_{i_m}u_{i_{m-1}}\cdots u_{i_1} \ : \ m \geq 1, 1 \leq i_p \leq n, 1 \leq p \leq m\}; \\ L_2 &= \{01^{i_1}01^{i_2}\cdots 01^{i_m}0^{n+1}1^{n+1}v_{i_m}v_{i_{m-1}}\cdots v_{i_1} \ : \ m \geq 1, 1 \leq i_p \leq n, 1 \leq p \leq m\}. \end{aligned}$$

Let $K = L_1 \cap L_2$. It is easy to see that $P$ has a solution if and only if $K \neq \emptyset$. Let $T = \{0,1\}^* - K$. Thus, $P$ has no solutions if and only if $T = \{0,1\}^*$. It is easily verified that that $T$ is a CFL.

We now show that $P$ has no solutions if and only if $T$ is transitive. If $P$ has no solutions, then clearly $T = \{0,1\}^*$ is transitive.

Assume that $P$ has a solution. Then there is some word

$$t = 01^{i_1}01^{i_2}\cdots 01^{i_m}0^{n+1}1^{n+1}u_{i_m}u_{i_{m-1}}\cdots u_{i_1} \notin T.$$

Note that $(L_1 \cup L_2) \cap 0^2(0+1)^* = \emptyset$, since $m \geq 1$, and $i_p \geq 1$ for all $1 \leq p \leq m$. Thus, we have that

$$t_1 = 001^{i_1-1}01^{i_2}\cdots 01^{i_m}0^{n+1}1^{n+1}u_{i_m}u_{i_{m-1}}\cdots u_{i_1} \notin K \subseteq L_1 \cup L_2.$$

Thus $t_1 \in T$. Let $\alpha = |t_1|_0$. Note that as $n \geq 1$, certainly $|x|_1 \geq 2$ for all $x \in L_1 \cup L_2$. Thus, we have $t_2 = 010^{\alpha-2} \in T$.

Assume now that $T$ is transitive, contrary to what we want to prove. By (6.9), as $t_1, t_2 \in T$, we must have that $\psi(\varphi^{-1}(t_1) \cap \sigma^{-1}(t_2)) \subseteq T$. But it is easy to verify that $t \in \psi(\varphi^{-1}(t_1) \cap \sigma^{-1}(t_2))$, which is a contradiction. Thus, $T$ is not transitive.

Therefore $P$ has a solution if and only if $T$ is not transitive, and we conclude that it is undecidable whether $T$ is transitive. ■

Consider, by (6.9), or by direct observation, that if $\{T_i\}_{i\in I}$ is a family of transitive sets of trajectories, then the set $\cap_{i\in I} T_i$ is also transitive. Thus, we can define the transitive closure of a set $T$ of trajectories as follows: for all $T \subseteq \{0,1\}^*$, let $tr(T) = \{T' \subseteq \{0,1\}^* \ : \ T \subseteq T', T' \text{ transitive}\}$. Note that $tr(T) \neq \emptyset$, as $\{0,1\}^* \in tr(T)$ for all $T \subseteq \{0,1\}^*$. Define $\widehat{T}$ as

$$\widehat{T} = \bigcap_{T' \in tr(T)} T'. \tag{6.10}$$

Then note that $\widehat{T}$ is transitive and is the smallest transitive set of trajectories containing $T$. The operation $\widehat{\cdot} : 2^{\{0,1\}^*} \to 2^{\{0,1\}^*}$ is indeed a closure operator (much like the closure operators on sets of trajectories constructed by Mateescu *et al.* [147] for, e.g., associativity and commutativity) in the algebraic sense, since $T \subseteq \widehat{T}$, and $\widehat{\cdot}$ preserves inclusion and is idempotent. Thus, we can, for instance, note the following result:

**Lemma 6.4.23** *If $T \supseteq O$ $(= 0^*1^*0^*)$, then $\widehat{T} = H$ $(= \{0,1\}^*)$.*

**Proof.** The result follows, since it is known (and easily observed) that $\widehat{O} = H$ (see, e.g., Ito *et al.* [77, Rem. 3.2]). ■

For particular instances of Lemma 6.4.23, see Thierrin and Yu [193, Prop. 2.3] or Long [136, Thm. 2.1].

A partial order is said to be a *division ordering* [24] if it is positive and compatible.

**Lemma 6.4.24** *Let $T \subseteq \{0,1\}^*$ be a partial order. If $T$ is a division ordering, then $T = (0+1)^*$.*

**Proof.** As $T$ is positive and compatible, then $T \supseteq 1^*$ and $T \supseteq 0^*T0^*$. Thus, $T \supseteq O$. As $T$ is a partial order, then $T$ is transitive. Thus, $T = \widehat{T} \supseteq \widehat{O} = H$. The result follows. ■

Consider the operator $\Omega_T : 2^{\{0,1\}^*} \to 2^{\{0,1\}^*}$ given by

$$\Omega_T(T') = T \cup T' \cup \psi(\sigma^{-1}(T') \cap \varphi^{-1}(T')). \tag{6.11}$$

By definition of $\Omega_T$, any fixed point $\Omega_T(T_0) = T_0$ contains $T$ and is transitive. Then we have

$$\emptyset \subseteq \Omega_T(\emptyset) = T \subseteq \Omega_T^2(T) \subseteq \Omega_T^3(T) \subseteq \cdots \subseteq \widehat{T}.$$

Since the operations of $\epsilon$-free morphism, inverse morphism, union and intersection are monotone and continuous [158], $\Omega_T$ is monotone and continuous and thus $\widehat{T}$ is the least upper bound of $\{\Omega_T^i(\emptyset)\}_{i\geq 0}$. Thus, given $T$, we can find $\widehat{T}$ by iteratively applying $\Omega_T$ to $T$, and in fact

$$\widehat{T} = \bigcup_{i\geq 0} \Omega_T^i(T). \tag{6.12}$$

This observation allows us to construct $\widehat{T}$, and, for instance, gives us the following result (a similar result for $\omega$-trajectories is given by Kadrie *et al.* [101]):

**Lemma 6.4.25** *There exists a regular set of trajectories $T \subseteq \{0,1\}^*$ such that $\widehat{T}$ is not a CFL.*

**Proof.** Consider $T = (01)^*$, corresponding to perfect or balanced literal shuffle. Then we note that $\widehat{T} \cap 01^* = \{01^{2^n-1} \ : \ n \geq 1\}$. ■

**Open Problem 6.4.26** *Given $T \in$ REG (or $T \in$ CF), is it decidable whether $\widehat{T} \in$ CF?*

### 6.4.8 Monotonicity

A binary relation $\rho$ on $\Sigma^*$ is said to be *monotone* (see, e.g, Ehrenfeucht *et al.* [47, p. 315]) if $x \rho y$ and $u \rho v$ implies $xu \rho yv$ for all $x, y, u, v \in \Sigma^*$. Occasionally, the concept of monotonicity is included as a requirement in compatibility, but we separate the two concepts here for clarity. We note that monotone here is a condition on $T$, rather than the monotonicity of the operation $\sqcup\!\sqcup_T$

(i.e., that $L_1 \subseteq L_2, L_3 \subseteq L_4$, and $T_1 \subseteq T_2$ imply that $L_1 \sqcup\!\sqcup_{T_1} L_3 \subseteq L_2 \sqcup\!\sqcup_{T_2} L_4$), which holds for all $T$.

**Lemma 6.4.27** *Let $T \subseteq \{0,1\}^*$. Then $T$ is monotone if and only if $T^2 \subseteq T$ if and only if $T = T^+$.*

**Proof.** The fact that $T^2 \subseteq T$ if and only if $T = T^+$ is obvious. Thus, we establish that $T$ is monotone if and only if $T^2 \subseteq T$.

Assume that $T^2 \subseteq T$. Let $x_i \ \omega_T \ y_i$ for $i = 1, 2$. Let $t_i \in T$ and $\alpha_i \in \Sigma^*$ be chosen so that $y_i \in x_i \sqcup\!\sqcup_{t_i} \alpha_i$ for $i = 1, 2$. Then as $t_1 t_2 \in T$, we have the fact that $y_1 y_2 \in x_1 x_2 \sqcup\!\sqcup_{t_1 t_2} \alpha_1 \alpha_2$ implies that $x_1 x_2 \ \omega_T \ y_1 y_2$. Thus $T$ is monotone.

Assume that $T$ is monotone. Let $t_1, t_2 \in T$ be arbitrary. Let $n_i = |t_i|_0$ and $m_i = |t_i|_1$ for $i = 1, 2$. Thus, we have that $0^{n_i} \ \omega_T \ t_i$ for $i = 1, 2$. By the monotonicity of $T$, $0^{n_1+n_2} \ \omega_T \ t_1 t_2$. Thus, there exist $t \in T$ and $\alpha \in \{0,1\}^*$ such that $t_1 t_2 \in 0^{n_1+n_2} \sqcup\!\sqcup_t \alpha$. But it is now clear that $\alpha = 1^{m_1+m_2}$ and $t = t_1 t_2$. Thus $t_1 t_2 \in T$ and $T^2 \subseteq T$. ■

The following corollary is immediate, since it is decidable whether $T^+ = T$ for regular languages.

**Corollary 6.4.28** *Given a regular set $T$ of trajectories, it is decidable whether $T$ is monotone.*

We also have the following undecidability result:

**Lemma 6.4.29** *Given an LCF set $T \subseteq \{0,1\}^*$ of trajectories, it is undecidable whether $T$ is monotone.*

**Proof.** We apply Theorem 2.5.3. First, we note that $T = \{0,1\}^*$ is monotone. Further, we note that the LCF set of trajectories $T = \{0^n 1^n \ : \ n \geq 0\}$ is not expressible as $T = T'/t$ for any monotone $T' \subseteq \{0,1\}^*$ and $t \in \{0,1\}^+$ (Indeed, if this were the case, then as $\epsilon \in T$, $t \in T'$. As $T' = (T')^+$, we have that $t^2, t^3 \in T'$ and $t, t^2 \in T$. But the only way this can happen is if $t = \epsilon$). Thus, we may apply Theorem 2.5.3. and it is undecidable whether a given LCF set of trajectories is monotone. ■

We can now consider the monotone closure of a set $T$ of trajectories, much in the same way we considered the transitive closure in Section 6.4.7. However, we do not need the same level of detail, since it is clear that the monotone closure of $T$ is $T^+$. Thus, we have the following result:

**Lemma 6.4.30** *Let $T \subseteq \{0,1\}^*$ be a regular (resp., CF, CS, recursive) set of trajectories. Then the monotone closure of $T$ is also a regular (resp., CF, CS, recursive) set of trajectories.*

Recall that $H = \{0,1\}^*$ and $\mathcal{P}_H$ corresponds to the set of biprefix codes.

**Lemma 6.4.31** *Let $T \subseteq \{0,1\}^*$. If $T$ is ST-strict and monotone, then $\mathcal{P}_T = \mathcal{P}_H$.*

**Proof.** Let $T$ be ST-strict and monotone. As $T$ is ST-strict, $\epsilon, 0, 1 \in T$. As $T$ is monotone, $\{\epsilon, 0, 1\}^+ = H \subseteq T$. Thus, $T = H$ and the result follows. ■

### 6.4.9 Well-Foundedness

A partial order $\rho$ is said to be *well-founded* (see, e.g., Choffrut and Karhumäki [24, Sect. 7.1]) if every strictly descending chain under $\rho$ is finite. We note that for relations defined by sets of trajectories, well-foundedness is implied by partial orders (and even by reflexive binary relations):

**Theorem 6.4.32** *Let $T \subseteq \{0,1\}^*$ be a partial order. Then $\omega_T$ is a well-founded partial order.*

**Proof.** Let $T$ be a partial order. Then $T$ is reflexive.

Let $\{w_i\}_{i \geq 1}$ be a descending chain, i.e., $w_{i+1}\ \omega_T\ w_i$ for all $i \geq 1$. Then $|w_{i+1}| \leq |w_i|$ for all $i \geq 1$. Let $K = |w_1|$. Thus, $|w_i| \leq K$ for all $i \geq 1$. Thus, there must exist some $j \geq 1$ such that $|w_j| = |w_{j+1}|$. In particular, this implies that $w_j = w_{j+1}$, and so by the reflexivity of $T$, $w_j\ \omega_T\ w_{j+1}$. Thus, $\{w_i\}_{i \geq 1}$ is not an infinite strictly descending chain. ■

## 6.5 Transitivity and Bases

Given a closure operator $\widehat{\cdot}$ and a closed set $S = \widehat{S}$, a base $B$ is a subset $B \subseteq S$ such that $\widehat{B} = S$ and $B$ is minimal with this property with respect to inclusion. Mateescu *et al.* note that in general, problems relating to the existence and effective constructibility of bases are "very challenging mathematically [147, p. 30]." Mateescu *et al.* list several problems relating to bases and associativity for shuffle on trajectories which, to our knowledge, are still open [147, Prob. 3–6, p. 29]. In this section, we investigate the problems of bases with respect to transitivity closure, which we studied in Section 6.4.7.

We will require the following notation. For all $n \geq 1$, let $\vee_n : (\{0,1\}^n)^2 \to \{0,1\}^n$ be pointwise 'OR'. For instance, $0101 \vee_4 1100 = 1101$. Let $\leq_\vee^{(n)}$ be the ordering on the associated poset, i.e., for all $x, y \in \{0,1\}^n$, $x \leq_\vee^{(n)} y$ if and only if $x \vee_n y = y$.

We now consider the notion of a transitivity-base. Given $T \subseteq \{0,1\}^*$ such that $T$ is transitive, a set $B \subseteq \{0,1\}^*$ of trajectories is said to be a *transitivity-base* for $T$ if $B \subseteq T$, $\widehat{B} = T$ and $B$ is minimal with respect to inclusion for the above properties (recall that $\widehat{\cdot}$ is the transitive closure operator defined in Section 6.4.7, cf., (6.10) and (6.12)).

Let $\Pi : 2^{\{0,1\}^*} \to 2^{\{0,1\}^*}$ be defined by

$$\Pi(T) = \psi(\varphi^{-1}(T - 0^*) \cap \sigma^{-1}(T - 0^*)).$$

Note that by Remark 6.4.19, $\Pi(T) = (T - 0^*) \sqcup\!\sqcup_{T-0^*} 1^*$. Further, let $\alpha : 2^{\{0,1\}^*} \to 2^{\{0,1\}^*}$ be given by

$$\alpha(T) = T - \Pi(T).$$

We now establish that every language has a transitivity-base.

**Theorem 6.5.1** *Let $T \subseteq \{0,1\}^*$ be a transitive set of trajectories. Then $\alpha(T)$ is a transitivity-base for $T$.*

**Proof.** Clearly, $\alpha(T) \subseteq T$. Thus, we first demonstrate that $\widehat{\alpha(T)} = T$. Let $t \in T$ be arbitrary. We establish by induction on the length of $t$ that $t \in \widehat{\alpha(T)}$.

For the base case, suppose that $t$ is a trajectory of minimal length in $T$. Suppose, contrary to what we want to prove, that $t \notin \widehat{\alpha(T)} \supseteq \alpha(T)$. Then as $t \notin \alpha(T)$, $t \in \Pi(T)$. Let $t_1, t_2 \in T - 0^*$ be such that $t \in \psi(\varphi^{-1}(t_1) \cap \sigma^{-1}(t_2))$. By definition of $\psi, \varphi, \sigma$, there exist $n \geq 1$, $i_j, k_j \geq 0$ for $1 \leq j \leq n$ and $s_j \in \{0,1\}^{i_j}$ for $1 \leq j \leq n$ such that

$$\begin{aligned} t_1 &= \prod_{j=1}^{n} 0^{i_j} 1^{k_j} \\ t_2 &= \prod_{j=1}^{n} s_j \\ t &= \prod_{j=1}^{n} s_j 1^{k_j}. \end{aligned}$$

As $\sum_{j=1}^{n} k_j \neq 0$, $t \neq t_2$ and $|t_2| < |t|$. As $t_2 \notin 0^*$, $t \neq t_1$. Now $t_2 \in T$ and $|t_2| < |t|$ contradicts our choice of $t$.

Assume that for all $t \in T$ with $|t| \leq n$, $t \in \widehat{\alpha(T)}$. Let $m = \min\{n' > n \ : \ T \cap \{0,1\}^{n'} \neq \emptyset\}$. We now establish that for all $t \in T \cap \{0,1\}^m$, $t \in \widehat{\alpha(T)}$. Let $p \geq 1$ and $t_1, t_2, \ldots, t_p$ be the trajectories of length $m$ in $T$, ordered by $\leq_{\vee}^{(m)}$. We establish the result by induction.

Let $t_1$ be any trajectory in $T$ of length $m$ which is minimal under $\leq_{\vee}^{(m)}$. Assume that $t_1 \notin \widehat{\alpha(T)}$. Then $t_1 \notin \alpha(T)$ as well. Let $s_1, s_2 \in T - 0^*$ be such that $t_1 \in \psi(\varphi^{-1}(s_1) \cap \sigma^{-1}(s_2))$. Note that $s_1 \neq t_1$, $|s_1| = |t_1|$ and $s_1 \leq_{\vee}^{(m)} t_1$, contradicting our choice of $t_1$. Thus, $t_1 \in \widehat{\alpha(T)}$.

Now, let $t \in T$ be any word in $T$ of length $m$, and assume that for all $s \in T$ of length $m$ such that $s \leq_{\vee}^{(m)} t$, $s \in \widehat{\alpha(T)}$ Assume, contrary to what we want to prove, that $t \notin \widehat{\alpha(T)}$. Then again, $t \notin \alpha(T)$ and thus there exist $s_1, s_2 \in T - 0^*$ such that $t \in \psi(\varphi^{-1}(s_1) \cap \sigma^{-1}(s_2))$. Note that $|s_2| < |s_1| = |t|$, and $s_1 \neq t$. Thus, by induction on $|t|$, $s_2 \in \widehat{\alpha(T)}$. By induction on the partial ordering induced by $\leq_{\vee}^{(m)}$, $s_1 \in \widehat{\alpha(T)}$. By Theorem 6.4.18, as $\widehat{\alpha(T)}$ is transitive, $t \in \widehat{\alpha(T)}$. Thus, $T \cap \{0,1\}^m \subseteq \widehat{\alpha(T)}$. Therefore, by induction $T \subseteq \widehat{\alpha(T)}$ and as $\widehat{\cdot}$ preserves inclusion and $T = \widehat{T}$ ($T$ is transitive), the reverse inclusion also holds.

Thus, it remains to establish that $\alpha(T)$ is minimal with respect to inclusions among all $T'$ with $\widehat{T'} = T$. Assume, contrary to what we want to prove, that there exists $T' \subseteq \{0,1\}^*$ such that

$T' \subset \alpha(T)$, where the inclusion noted is proper, and $\widehat{T'} = T$.

Recall the operator $\Omega_T$ defined by (6.11). Let $j = \min\{i : \Omega^i_{T'}(T') \cap (\alpha(T) - T') \neq \emptyset\}$. Clearly $j$ exists, as $\emptyset \neq (\alpha(T) - T') \subseteq T = \widehat{T'} = \cup_{i \geq 0} \Omega^i_{T'}(T')$. Let $t \in \Omega^j_{T'}(T')$ be arbitrary. We show that $t \notin \alpha(T) - T'$, contrary to our choice of $j$. This will give us our contradiction.

Consider that $t \in \Omega^j_{T'}(T') = \Omega^{j-1}_{T'}(T') \cup T' \cup \psi(\varphi^{-1}(\Omega^{j-1}_{T'}(T')) \cap \sigma^{-1}(\Omega^{j-1}_{T'}(T')))$. If $t \in \Omega^{j-1}_{T'}(T')$, then by choice of $j$, $t \notin \alpha(T) - T'$. Also, if $t \in T'$, then $t \notin \alpha(T) - T'$. Thus, assume that there exist $t_1, t_2 \in \Omega^{j-1}_{T'}(T') \subseteq T$ such that $t \in \psi(\varphi^{-1}(t_1) \cap \sigma^{-1}(t_2))$. Note that if $t_1, t_2 \notin 0^*$, then $t \in \Pi(T)$. In this case, $t \notin \alpha(T)$. Thus, we may assume that $t_1 \in 0^*$ or $t_2 \in 0^*$.

If $t_1 \in 0^*$, then we can see that $t = t_2 \notin \alpha(T) - T'$. Further, if $t_2 \in 0^*$, then we see that $t = t_1 \notin \alpha(T) - T'$. Thus, we have established our contradiction, and $\alpha(T)$ is a transitivity-base for $T$. ■

We have the following corollary:

**Corollary 6.5.2** *Let $T$ be a finite (resp., regular, context-free, recursive) transitive set of trajectories. Then $T$ has a finite (resp., regular, co-*NP*, recursive) transitivity-base.*

**Proof.** The cases when $T$ is finite, regular or recursive are immediate, based on the closure properties of these classes of languages. We turn to the case when $T$ is a CF set of trajectories. Consider that

$$\Pi(T) = \psi(\varphi^{-1}(T - 0^*) \cap \sigma^{-1}(T - 0^*)).$$

and that $\alpha(T) = T - \Pi(T)$. Note that $T - 0^*$, $\varphi^{-1}(T - 0^*)$ and $\sigma^{-1}(T - 0^*)$ are all CFLs. We claim now that $\Pi(T)$ is in NP. To see this, note that $\psi$ is a letter-to-letter morphism (i.e., $\psi(a) \in \{0, 1\}$ for all $a \in \{x, y, z\}$). Thus, to determine if a trajectory $t$ is a member of $\Pi(T)$, we nondeterministically guess a trajectory $t_1$ of the same length as $t$. We then test whether $t \in \psi(t_1)$, and whether $t_1 \in \varphi^{-1}(T - 0^*) \cap \sigma^{-1}(T - 0^*)$. As testing membership in CFLs can be done in polynomial time, and as $t_1$ is the same length as $t$, the above is a nondeterministic polynomial-time algorithm for determining membership in $\Pi(T)$. It follows that $\alpha(T) = T - \Pi(T)$ is in co-NP. ■

**Example 6.5.3:** We give some examples:

(a) let $T = 0^*1^*$. We can compute that $\alpha(T) = 0^*(\epsilon + 1)$.

(b) If $T = (0 + 1)^*$, then $\alpha(T) = 0^*(\epsilon + 1)0^*$.

(c) Let $T = \{0^i 1^{2j} 0^i \ : \ i, j \geq 0\}$. Then $\alpha(T) = (00)^* + \{0^i 110^i \ : \ i \geq 0\}$. Note that $T$ and $\alpha(T)$ are both context-free.

(d) If $T = 1^*0^*1^*$, then $\alpha(T) = 0^* + 10^* + 0^*1$.

□

We now show that the context-free languages are not closed under $\alpha$. This requires a slightly more complex construction, which we give now:

**Theorem 6.5.4** *The context-free languages are not closed under $\alpha$.*

**Proof.** Let $T = \{0^i 1^j \ : \ 1 \leq i \leq j\}^*$. We leave it to the reader to verify that $T \in \text{CF}$ and $T$ is transitive. Note that $T \cap 0^* = \emptyset$.

**Claim 6.5.5** *If $t \in \Pi(T)$, then $3|t|_0 \leq |t|_1$.*

Let $t \in \Pi(T)$, and let $t_1, t_2 \in T$ be such that $t \in t_1 \sqcup\!\sqcup_{t_2} 1^*$. As $t_1, t_2 \in T$, we have $|t_i|_0 \leq |t_i|_1$ for $i = 1, 2$. Further, we note that $|t_1|_0 + |t_1|_1 = |t_1| = |t_2|_0$, $|t|_0 = |t_1|_0$, and $|t|_1 = |t_1|_1 + |t_2|_1$. Thus, we have that

$$3|t|_0 = 3|t_1|_0 \leq |t_1|_1 + 2|t_1|_0 \leq |t_1|_1 + (|t_1|_0 + |t_1|_1) \leq |t_1|_1 + |t_2|_0 \leq |t_1|_1 + |t_2|_1 = |t|_1.$$

Thus, the claim is proven. □

We now return to the proof that $\alpha(T)$ is not a CFL. Assume, contrary to what what we want to prove, that $\alpha(T)$ is a CFL. Then $\alpha(T) \cap 0^+1^+0^+1^+$ is also a CFL.

We employ Ogden's Lemma [68, Lemma 6.2]. Let $n$ be the constant associated with Ogden's Lemma. Assume without loss of generality that $n \geq 1$. Let $t = 0^n 1^n 0^n 1^{5n-1} \in T$. Note that $|t|_0 = 2n$ and $|t|_1 = 6n - 1$, thus $t \notin \Pi(T)$. Therefore, $t \in \alpha(T) \cap 0^+1^+0^+1^+$. Let us consider

the first $n$ occurrences of zero as marked. Let $t = uvwxy$. Then we note that both $v, x$ must occur within a block of letters of one type, otherwise, we can consider $t = uv^2wx^2y \notin 0^+1^+0^+1^+$. Now, $v$ or $x$ must contain at least one of the marked letters. Note that if either (a) $vwx$ is entirely contained in the first block of zeroes or (b) $v$ is contained in the first block of zeroes and $x$ is contained in the second block of zeroes or the second block of ones, then $uv^2wx^2y$ has the form $0^{n+k}1^nz$ for some word $z$ starting with zero. This word is clearly not in $T$, thus not in $\alpha(T)$.

Thus, we must have that $v$ is contained in the first block of zeroes, and $x$ is contained in the first block of ones. Let $v = 0^i$ and $x = 1^j$ for some $i, j \geq 0$ with $i > 0$ and $j \geq 0$. We have two cases:

(a) $i \neq j$. Let $k = 0$ if $i > j$ and $k = 2$ if $i < j$. Then note that $n + (k-1)i > n + (k-1)j$ for this choice of $k$. Further, $uv^kwx^ky = 0^{n+(k-1)i}1^{n+(k-1)j}0^n1^{5n-1}$. Clearly, $uv^kwx^ky \notin T$.

(b) $i = j$. Note that $n \geq i = j \neq 0$. Thus, consider $uwy = 0^{n-i}1^{n-i}0^n1^{5n-1}$. We claim that $uwy \in \Pi(T)$. Consider $t_1 = 0^{2n-i}1^{2n-i}$ and $t_2 = 0^{n-i}1^{n-i}0^n1^n0^{2n-i}1^{2n+(i-1)}$. Then $uwy \in t_1 \sqcup\!\sqcup_{t_2} 1^*$. It remains to show that $t_1, t_2 \in T - 0^*$. That $t_1, t_2 \notin 0^*$ is easily observed, as $n \neq 0$ and $i \leq n$. Clearly, $t_1 \in T$. Further, as $2n - i < 2n + (i-1)$ for $i > 0$, $t_2 \in T$ as well.

Thus, $\alpha(T)$ is not a CFL, as required. ■

We briefly discuss the problem of bases for monotone sets of trajectories. Recall that the closure operator for monotonicity is $T^+$. The problem of finding a base for a monotone set of trajectories is therefore a classical problem; we refer the reader to Brzozowski [20]. In particular, we note that the construction $\mu(T) = T - T^2$ gives a base for a monotone set of trajectories $T$.

## 6.6 Convexity and Transitivity

Let $\widehat{T}$ again represent the transitive closure of $T$. We now examine the relationship between $T$-codes and $\widehat{T}$-codes for arbitrary $T \subseteq \{0,1\}^*$. We call a language $L \subseteq \Sigma^*$ *$T$-convex* if, for all $y \in \Sigma^*$ and $x, z \in L$, $x\ \omega_T\ y$ and $y\ \omega_T\ z$ implies $y \in L$.

We now characterize when a language is $T$-convex using shuffle and deletion along trajectories.

**Lemma 6.6.1** *Let $T \subseteq \{0,1\}^*$. Then $L \subseteq \Sigma^*$ is $T$-convex if and only if $(L \sqcup\!\sqcup_T \Sigma^*) \cap (L \rightsquigarrow_{\tau(T)} \Sigma^*) \subseteq L$.*

**Proof.** Let $L$ be a $T$-convex language. Consider an arbitrary word $x \in (L \sqcup\!\sqcup_T \Sigma^*) \cap (L \rightsquigarrow_{\tau(T)} \Sigma^*)$. Then there exist $y_1, y_2 \in L$ such that $x \in y_1 \sqcup\!\sqcup_T \Sigma^*$ and $x \in y_2 \rightsquigarrow_{\tau(T)} \Sigma^*$. By Lemma 5.8.1, we have that $y_2 \in x \sqcup\!\sqcup_T \Sigma^*$. Thus, $y_1 \,\omega_T\, x \,\omega_T\, y_2$. By the $T$-convexity of $L$, $x \in L$. Thus, the inclusion is established.

The reverse implication is similar. Let $(L \sqcup\!\sqcup_T \Sigma^*) \cap (L \rightsquigarrow_{\tau(T)} \Sigma^*) \subseteq L$. Let $y_1, y_2 \in L$ and $x \in \Sigma^*$ be such that $y_1 \,\omega_T\, x \,\omega_T\, y_2$. Then $x \in y_1 \sqcup\!\sqcup_T \Sigma^*$ and $y_2 \in x \sqcup\!\sqcup_T \Sigma^*$. Again, Lemma 5.8.1 implies that $x \in y_2 \rightsquigarrow_{\tau(T)} \Sigma^*$. Thus, $x \in L$, by our assumed inclusion, and $L$ is $T$-convex. ■

**Corollary 6.6.2** *Let $T \subseteq \{0,1\}^*$ be reflexive. Then $L \subseteq \Sigma^*$ is $T$-convex if and only if $(L \sqcup\!\sqcup_T \Sigma^*) \cap (L \rightsquigarrow_{\tau(T)} \Sigma^*) = L$.*

**Proof.** We show that if $T$ is reflexive, then for all $L \subseteq \Sigma^*$,

$$L \subseteq (L \sqcup\!\sqcup_T \Sigma^*) \cap (L \rightsquigarrow_{\tau(T)} \Sigma^*). \tag{6.13}$$

If $T$ is reflexive, then $0^* \subseteq T$ and $(L \sqcup\!\sqcup_T \Sigma^*) \supseteq (L \sqcup\!\sqcup_{0^*} \{\epsilon\}) = L$. Further, if $T \supseteq 0^*$ then $\tau(T) \supseteq i^*$ and $(L \rightsquigarrow_{\tau(T)} \Sigma^*) \supseteq (L \rightsquigarrow_{i^*} \{\epsilon\}) = L$. Thus, we have established (6.13). ■

We now turn to decidability:

**Corollary 6.6.3** *Let $T \subseteq \{0,1\}^*$ be a regular set of trajectories. Given a regular language $L$, it is decidable whether $L$ is $T$-convex.*

**Proof.** As $L, T$ are regular, so are $L \sqcup\!\sqcup_T \Sigma^*$, $L \rightsquigarrow_{\tau(T)} \Sigma^*$ and $(L \sqcup\!\sqcup_T \Sigma^*) \cap (L \rightsquigarrow_{\tau(T)} \Sigma^*)$. Thus, the inclusion in Lemma 6.6.1 is decidable. ■

We now turn to our main result of this section:

**Theorem 6.6.4** *Let $\Sigma$ be an alphabet and $T \subseteq \{0,1\}^*$. For all languages $L \subseteq \Sigma^+$, the following two conditions are equivalent:*

*(i) $L$ is a $\widehat{T}$-code;*

*(ii) $L$ is a $\widehat{T}$-convex $T$-code.*

**Proof.** (i) $\Rightarrow$ (ii): Let $L \subseteq \Sigma^+$ be a $\widehat{T}$-code. Then as $T \subseteq \widehat{T}$, $L$ is a $T$-code as well. Assume that $u\ \omega_{\widehat{T}}\ v\ \omega_{\widehat{T}}\ w$, with $u, w \in L$. As $\widehat{T}$ is transitive, by definition, $u\ \omega_{\widehat{T}}\ w$. Thus, $u = w$, as $u, w \in L$. Now, by the antisymmetry of $\widehat{T}$, $v\ \omega_{\widehat{T}}\ u$ and $u\ \omega_{\widehat{T}}\ v$ imply $v = u \in L$. Thus, $L$ is $\widehat{T}$-convex.
(ii) $\Rightarrow$ (i): Let $L \subseteq \Sigma^+$ be a $T$-code, as well as being $\widehat{T}$-convex.

Recall the operator $\Omega_T$ given by (6.11). Let $T_i = \Omega_T^i(T)$. Then $\widehat{T} = \cup_{i\geq 0} T_i$, by (6.12). We establish (by induction) that $L$ is a $T_i$-code for all $i \geq 0$. The result will then follow. To see this, assume $L$ is a $T_i$-code for all $i \geq 0$. Let $x, y \in L$ be such that $x\ \omega_{\widehat{T}}\ y$. Then there exists $t \in \widehat{T}$ such that $y \in x \sqcup\!\sqcup_t z$ for some $z \in \Sigma^*$. As $t \in \widehat{T}$, there exists $i \geq 0$ such that $t \in T_i$, Thus, $x\ \omega_{T_i}\ y$ and then $x = y$, as required.

We now establish by induction on $i \geq 0$ that $L$ is a $T_i$-code. For $i = 0$, $T_0 = T$. Thus, $L$ is a $T$-code by assumption.

Let $i > 0$ and assume that $L$ is a $T_{i-1}$-code. Let $x, y \in L$ be chosen so that $x\ \omega_{T_i}\ y$. Thus, there exist $t \in T_i$ and $z \in \Sigma^*$ such that $y \in x \sqcup\!\sqcup_t z$. We have that $t \in T_i = \Omega_T(T_{i-1}) = T \cup T_{i-1} \cup \psi(\sigma^{-1}(T_{i-1}) \cap \varphi^{-1}(T_{i-1}))$. If $t \in T \cup T_{i-1}$, then, as $y \in x \sqcup\!\sqcup_t z$, by induction $x = y$.

Consider then the case when $t \in \psi(\sigma^{-1}(T_{i-1}) \cap \varphi^{-1}(T_{i-1}))$. Let $t_0, t_1 \in T_{i-1}$ be such that $t \in \psi(\varphi^{-1}(t_0) \cap \sigma^{-1}(t_1))$. By definition of $\psi, \sigma, \varphi$, we know that we can write

$$t_0 = \prod_{k=1}^{n} 0^{i_k} 1^{j_k}$$

for some $n \in \mathbb{N}$ and $i_k, j_k \in \mathbb{N}$ for all $1 \leq k \leq n$, as well as $t_1 = \prod_{k=1}^{n} s_k$ where $|s_k| = i_k$ for all $1 \leq k \leq n$. Further,

$$t = \prod_{k=1}^{n} s_k 1^{j_k}.$$

As $y \in x \sqcup\!\sqcup_t z$, we can write $x = \prod_{k=1}^{n} x_k$, $z = \prod_{k=1}^{n} \alpha_k \beta_k$, where $x_k, \alpha_k, \beta_k \in \Sigma^*$ satisfy $|x_k| = |s_k|_0$, $|\alpha_k| = |s_k|_1$ and $|\beta_k| = j_k$ for all $1 \leq k \leq n$. Further, let $y = \prod_{k=1}^{n} \gamma_k \beta_k$ where $\gamma_k \in x_k \sqcup\!\sqcup_{s_k} \alpha_k$ for all $1 \leq k \leq n$.

Let $\alpha = \prod_{k=1}^{n} \alpha_k$, $\beta = \prod_{k=1}^{n} \beta_k$ and $\gamma = \prod_{k=1}^{n} \gamma_k$. Then we note that

$$\begin{aligned} y &\in \gamma \sqcup\!\sqcup_{t_0} \beta; \\ \gamma &\in x \sqcup\!\sqcup_{t_1} \alpha. \end{aligned}$$

As $t_0, t_1 \in T_{i-1} \subseteq \widehat{T}$, we conclude that $x\ \omega_{T_{i-1}}\ \gamma\ \omega_{T_{i-1}}\ y$, as well as $x\ \omega_{\widehat{T}}\ \gamma\ \omega_{\widehat{T}}\ y$, and thus $\gamma \in L$, by the $\widehat{T}$-convexity of $L$.

Finally, we note that $\gamma\ \omega_{T_{i-1}}\ y$ implies that $\gamma = y$, as $L$ is a $T_{i-1}$-code by induction. Similarly, $x\ \omega_{T_{i-1}}\ \gamma$ implies that $\gamma = x$. We conclude that $x = y$ and, since $x, y \in L$ were chosen arbitrarily, $L$ is a $T_i$-code. ■

Theorem 6.6.4 was known for the case $O = 0^*1^*0^*$, which corresponds to outfix codes, see, e.g., Shyr and Thierrin [185, Prop. 2]. In this case, $\widehat{O} = H = (0 + 1)^*$, which corresponds to hypercodes. Theorem 6.6.4 was known to Guo *et al.* [57, Prop. 2] in a slightly weaker form for $B = 0^*1^* + 1^*0^*$. In this case, $\widehat{B} = I = 1^*0^*1^*$, and the convexity is with respect to the factor (or subword) ordering. See also Long [136, Sect. 5] for the case of shuffle codes.

## 6.7 Closure Properties

We now consider the closure properties of $\mathcal{P}_T$.

### 6.7.1 Closure under Boolean Operations

We note immediately that $\mathcal{P}_T$ is closed under intersection with arbitrary languages, provided the intersection is non-empty:

**Lemma 6.7.1** *Let $\Sigma$ be an alphabet and $T \subseteq \{0, 1\}^*$. Let $L_0 \in \mathcal{P}_T(\Sigma)$ and $L_1 \subseteq \Sigma^+$. If $L_0 \cap L_1 \neq \emptyset$, then $L_0 \cap L_1 \in \mathcal{P}_T(\Sigma)$.*

Further, it is clear that $\mathcal{P}_T$ is closed under union if and only if $T \subseteq 0^* + 1^*$.

**Lemma 6.7.2** *Let $\Sigma$ be an alphabet with $|\Sigma| > 1$ and $T \subseteq \{0,1\}^*$. Then $\mathcal{P}_T(\Sigma)$ is closed under union if and only if $T \subseteq 0^* + 1^*$.*

**Proof.** If $T \subseteq 0^* + 1^*$, then $\mathcal{P}_T(\Sigma) = 2^{\Sigma^+} - \{\emptyset\}$. Thus, it is clear that $\mathcal{P}_T(\Sigma)$ is closed under union.

If $T \cap \overline{0^* + 1^*} \neq \emptyset$, then let $t \in T$ be such that $|t|_0, |t|_1 \neq 0$. Let $t_0 = 0^{|t|_0}$. As $|t|_1 \neq 0$, $t_0 \neq t$. It suffices to note that $\{t\}, \{t_0\} \in \mathcal{P}_T(\Sigma)$, but that $\{t, t_0\} \notin \mathcal{P}_T(\Sigma)$. ■

For completeness, we consider closure of $\mathcal{P}_T(\Sigma)$ under non-empty complement relative to $\Sigma^+$:

**Lemma 6.7.3** *Let $\Sigma$ be an alphabet with $|\Sigma| > 1$. Let $T \subseteq \{0,1\}^*$. Then there exists $L \in \mathcal{P}_T(\Sigma)$ such that $\overline{L} \cap \Sigma^+ \neq \emptyset$ and $\overline{L} \cap \Sigma^+ \notin \mathcal{P}_T(\Sigma)$ if and only if $T \not\subseteq 0^* + 1^*$.*

**Proof.** If $T \subseteq 0^* + 1^*$, then $\mathcal{P}_T(\Sigma) = 2^{\Sigma^+} - \{\emptyset\}$. Assume there exists $L \in \mathcal{P}_T(\Sigma)$ such that $\overline{L} \cap \Sigma^+ \neq \emptyset$ and $\overline{L} \cap \Sigma^+ \notin \mathcal{P}_T(\Sigma)$ Thus, $\overline{L} \cap \Sigma^+ \notin 2^{\Sigma^+} - \{\emptyset\}$. As $\overline{L} \cap \Sigma^+ \neq \emptyset$, we must have that $\overline{L} \cap \Sigma^+ \notin 2^{\Sigma^+}$, i.e., $\overline{L} \cap \Sigma^+ \not\subseteq \Sigma^+$, which is absurd.

If $T \cap \overline{0^* + 1^*} \neq \emptyset$, then let $t \in T$ be such that $|t|_0, |t|_1 \neq 0$. Let $0, 1 \in \Sigma$ without loss of generality.

Let $t_1 = 1^{|t|_1}$ and $t_0 = 0^{|t|_0}$. Note that the three trajectories $t, t_0, t_1$ are all distinct. We conclude by noting that $L = \{t_1\} \in \mathcal{P}_T(\Sigma)$, but $\Sigma^+ - \{t_1\} \supseteq \{t_0, t\}$. Thus, $L \in \mathcal{P}_T(\Sigma)$ but $\overline{L} \cap \Sigma^+ \notin \mathcal{P}_T(\Sigma)$.
■

### 6.7.2 Closure under Catenation

**Theorem 6.7.4** *Let $T \subseteq \{0,1\}^*$ be a set of trajectories such that $s(T) \cup p(T) \subseteq T$. Then $\mathcal{P}_T$ is closed under catenation.*

**Proof.** Let $L_i \in \mathcal{P}_T$ for $i = 1, 2$. Assume that

$$(L_1 L_2 \sqcup\!\sqcup_T x) \cap L_1 L_2 \neq \emptyset$$

for some $x \in \Sigma^*$. We will demonstrate that $x = \epsilon$. Let $\alpha_i, \beta_i \in L_i$ for $i = 1, 2$ be such that

$$\beta_1 \beta_2 \in \alpha_1 \alpha_2 \sqcup\!\sqcup_T x.$$

Let $t \in T$ be such that $\beta_1\beta_2 \in \alpha_1\alpha_2 \sqcup\!\sqcup_t x$. Let $x = x_1x_2$ and $t = t_1t_2$ be chosen so that $\beta_1\beta_2 \in (\alpha_1 \sqcup\!\sqcup_{t_1} x_1)(\alpha_2 \sqcup\!\sqcup_{t_2} x_2)$. We distinguish two cases:

(a) $|\alpha_1| + |x_1| \geq |\beta_1|$. Then there exists $\gamma \in \Sigma^*$ such that

$$\begin{aligned} \beta_1\gamma &\in \alpha_1 \sqcup\!\sqcup_{t_1} x_1; \\ \beta_2 &\in \gamma\,(\alpha_2 \sqcup\!\sqcup_{t_2} x_2). \end{aligned}$$

Let $t_2' = 1^{|\gamma|}t_2$ and $x_2' = \gamma\, x_2$. Then, as $|\gamma| \leq |t_1|$, $t_2' \in s(T) \subseteq T$ and thus $\beta_2 \in \alpha_2 \sqcup\!\sqcup_{t_2'} x_2'$ implies that $x_2' = \epsilon$. In particular, $x_2 = \gamma = \epsilon$. As $\gamma = \epsilon$, $\beta_1 \in \alpha_1 \sqcup\!\sqcup_{t_1} x_1$. Note that $t_1 \in p(T) \subseteq T$. Thus, $L_1$ a $T$-code implies that $x_1 = \epsilon$ and hence $x = x_1x_2 = \epsilon$.

(b) $|\alpha_1| + |x_1| < |\beta_1|$. Let $\gamma \in \Sigma^+$ be such that

$$\begin{aligned} \beta_1 &\in (\alpha_1 \sqcup\!\sqcup_{t_1} x_1)\gamma\,; \\ \gamma\beta_2 &\in \alpha_2 \sqcup\!\sqcup_{t_2} x_2. \end{aligned}$$

Let $t_1' = t_1 1^{|\gamma|} \in p(T) \subseteq T$, as $|\gamma| \leq |t_2|$, and let $x_1' = x_1\gamma$. Then $\beta_1 \in (\alpha_1 \sqcup\!\sqcup_{t_1'} x_1')$. As $L_1$ is a $T$-code, $x_1' = \epsilon$. This contradicts that $\gamma \in \Sigma^+$.

Thus, $x = \epsilon$ and $L_1L_2$ is a $T$-code. ■

We note that Theorem 6.7.4 can also be proven as follows: as $p(T) \cup s(T) \subseteq T$, $T$ is both cancellative and leviesque. By Jürgensen *et al.* [99, Prop. 10], this implies that $\mathcal{P}_T$ is closed under catenation.

### 6.7.3 Closure under Inverse Morphism

We now turn to inverse morphism. Let $n \geq 1$. Let $T \subseteq (0^*1^*)^n$ be a bounded regular language such that there exist $a_i, b_i, c_i, d_i$ for $1 \leq i \leq n$ such that

$$T = \prod_{i=1}^{n} 0^{a_i}(0^{b_i})^*1^{c_i}(1^{d_i})^*. \tag{6.14}$$

(We assume throughout that $T \subseteq (0^*1^*)^n$; similar proofs follow if, e.g., $T \subseteq (0^*1^*)^n 0^*$). Let

$$\begin{aligned} I_j &= \{a_j + b_j m : m \geq 0\} \quad \forall 1 \leq j \leq n; \\ K_j &= \{c_j + d_j m : m \geq 0\} \quad \forall 1 \leq j \leq n. \end{aligned}$$

Let $I'_j = I_j \setminus \{0\}$ for all $1 \leq j \leq n$.

Let $\varphi : \Delta^* \to \Sigma^*$ be a morphism. We define $[\varphi], [\varphi^{-1}] : \mathbb{N} \to 2^{\mathbb{N}}$ as follows:

$$\begin{aligned} [\varphi](m) &= \{|x| : x \in \varphi(\Sigma^m)\}; \\ [\varphi^{-1}](m) &= \{|x| : x \in \varphi^{-1}(\Sigma^m)\}. \end{aligned}$$

We extend these functions naturally to operate on $2^{\mathbb{N}}$ as, e.g., $[\varphi](S) = \bigcup_{s \in S} [\varphi](s)$.

We now prove a generalization of a result on infix and outfix codes established by Ito *et al.* [77, Prop. 6.5].

**Theorem 6.7.5** *Let $T \subseteq (0^*1^*)^n$ be a bounded regular set of trajectories as given by (6.14). Let $\varphi : \Delta^* \to \Sigma^*$ be a morphism satisfying*

*(a) $\emptyset \neq [\varphi^{-1}](I_j) \subseteq I_j$ for all $1 \leq j \leq n$.*

*(b) there exists $j$ with $1 \leq j \leq n$ such that $\emptyset \neq [\varphi^{-1}](I'_j) \subseteq I'_j$.*

*(c) $[\varphi](I_j) \subseteq I_j$ for all $1 \leq j \leq n$.*

*(d) $[\varphi](K_j) \subseteq K_j$ for all $1 \leq j \leq n$.*

*Then $\mathcal{P}_T(\Sigma)$ is closed under $\varphi^{-1}$ if and only if*

$$\{|x| : x \in \varphi^{-1}(\epsilon)\}^n \cap \left( \prod_{j=1}^{n} K_j - \{0\}^n \right) = \emptyset. \tag{6.15}$$

**Proof.** Assume that (6.15) fails. Let $x_j$ for $1 \leq j \leq n$ be such that $x_j \in \varphi^{-1}(\epsilon)$ and $|x_j| \in K_j$. By (6.15), $x = \prod_{i=1}^{n} x_i \neq \epsilon$. Let $k_j = |x_j|$ for $1 \leq j \leq n$.

By (a), let $i_j \in I_j$ be such that $[\varphi^{-1}](i_j) \neq \emptyset$ for all $1 \leq j \leq n$, and such there exist $j_0$ satisfying $1 \leq j_0 \leq n$, $i_{j_0} \neq 0$ and $[\varphi^{-1}](i_{j_0})$ contains a non-zero element, by (b). Thus, $\varphi^{-1}(\Sigma^{i_j}) \neq \emptyset$. Let

$u_j \in \Sigma^{i_j}$ be such that there exist $v_j \in \varphi^{-1}(u_j)$ for all $1 \leq j \leq n$. As $i_{j_0} \neq 0$, $u = \prod_{j=1}^{n} u_j \neq \epsilon$, and as we can choose $v_{j_0} \in \varphi^{-1}(u_{j_0})$ to be a non-empty word, $v = \prod_{j=1}^{n} v_j \neq \epsilon$. Further, by (a), $|v_j| \in I_j$. Let $\ell_j = |v_j|$ for $1 \leq j \leq n$.

Consider $t = \prod_{j=1}^{n} 0^{\ell_j} 1^{k_j}$. As $\ell_j \in I_j$ and $k_j \in K_j$, $t \in T$. We now define a $T$-code $L \subseteq \Sigma^+$ such that $\varphi^{-1}(L)$ is not a $T$-code.

Consider $L = \{u\} \subseteq \Sigma^+$. Trivially, $L$ is a $T$-code. Let $w = \prod_{j=1}^{n} v_j x_j$. Note that $\varphi(v) = \varphi(v_1) \cdots \varphi(v_n) = u_1 \cdots u_n = u$, and that $\varphi(w) = \prod_{j=1}^{n} \varphi(v_j)\varphi(x_j) = \prod_{j=1}^{n} u_j \cdot \epsilon = u$. Thus, $v, w \in \varphi^{-1}(L)$. Further, $v \neq \epsilon$ implies that $w \neq \epsilon$.

The fact that $\varphi^{-1}(L)$ is not a $T$-code now follows, since $w \in \varphi^{-1}(L) \cap (v \sqcup\!\sqcup_t x) \subseteq \varphi^{-1}(L) \cap (\varphi^{-1}(L) \sqcup\!\sqcup_T \Delta^+)$.

For the reverse implication, let $L \subseteq \Sigma^+$ be a $T$-code such that $\varphi^{-1}(L)$ is not a $T$-code. Then there exists $t \in T$, $u, v \in \varphi^{-1}(L)$ and $x \in \Delta^+$ such that $v \in u \sqcup\!\sqcup_T x$. As $\varphi(u), \varphi(v) \in L \subseteq \Sigma^+$, $u, v \in \Delta^+$.

Consider $t = \prod_{j=1}^{n} 0^{i_j} 1^{k_j}$ for some $i_j \in I_j$ and $k_j \in K_j$ for $1 \leq j \leq n$. Then $v = \prod_{j=1}^{n} u_j x_j$. for $|u_j| = i_j$, $|x_j| = k_j$, $1 \leq j \leq n$. Consider that

$$\begin{aligned} \varphi(v) &= \prod_{i=1}^{n} \varphi(u_j)\varphi(x_j), \\ \varphi(u) &= \prod_{i=1}^{n} \varphi(u_j), \\ \varphi(x) &= \prod_{i=1}^{n} \varphi(x_j). \end{aligned}$$

Let $\ell_j = |\varphi(u_j)|$ and $m_j = |\varphi(x_j)|$ for $1 \leq j \leq n$. By assumptions (c) and (d), $\ell_j \in I_j$ and $m_j \in K_j$. Thus,

$$t' = \prod_{j=1}^{n} 0^{\ell_j} 1^{m_j} \in T.$$

Then we may easily observe that

$$\varphi(v) \in \varphi(u) \sqcup\!\sqcup_{t'} \varphi(x).$$

As $\varphi(v), \varphi(u) \in L$, a $T$-code, $\varphi(x) = \epsilon$, and, in particular, $\varphi(x_j) = \epsilon$ for all $1 \leq j \leq n$. Thus,

recalling that $k_j = |x_j|$ and $x \neq \epsilon$, we note that

$$[k_1, \cdots, k_n] \in \{|x| \ : \ x \in \varphi^{-1}(\epsilon)\}^n \cap \left( \prod_{j=1}^{n} K_j - \{0\}^n \right).$$

This completes the proof. ■

### 6.7.4 Closure under Reversal

For a word $w = w_1 w_2 \cdots w_n$, where $w_i \in \Sigma$, its *reversal*, denoted $w^R$, is given by $w^R = w_n w_{n-1} \cdots w_1$. If $L \subseteq \Sigma^*$ is a language, then its reversal is $L^R = \{w^R \ : \ w \in L\}$. For a class of languages $\mathcal{C}$, let $\mathcal{C}^R = \{L^R \ : \ L \in \mathcal{C}\}$.

**Lemma 6.7.6** *For all $T \subseteq \{0,1\}^*$, the following equality holds: $\mathcal{P}_{T^R} = \mathcal{P}_T^R$.*

**Proof.** It suffices to show that $\mathcal{P}_{T^R} \subseteq \mathcal{P}_T^R$.

Let $L \in \mathcal{P}_{T^R}$. Then we have that $L \cap (L \sqcup\!\sqcup_{T^R} \Sigma^+) = \emptyset$. Assume that $L \notin \mathcal{P}_T^R$ and thus $L^R \notin \mathcal{P}_T$. Let $x, y \in L^R$, $t \in T$ and $z \in \Sigma^+$ be such that $x \in y \sqcup\!\sqcup_t z$. Then we note (see, e.g., Mateescu *et al.* [147, Rem. 4.9(ii)]) that $x^R \in y^R \sqcup\!\sqcup_{t^R} z^R$. But as $x^R, y^R \in L$, $t^R \in T^R$, and $z^R \in \Sigma^+$, this contradicts that $L$ is a $T^R$-code. Thus, $L \in \mathcal{P}_T^R$. ■

**Corollary 6.7.7** *Let $T \subseteq \{0,1\}^*$. Then $\mathcal{P}_T^R = \mathcal{P}_T$ if and only if $T = T^R$.*

## 6.8 Maximal $T$-codes

Let $T \subseteq \{0,1\}^*$. We say that $L \in \mathcal{P}_T(\Sigma)$ is a *maximal* $T$-code if, for all $L' \in \mathcal{P}_T(\Sigma)$, $L \subseteq L'$ implies $L = L'$. Denote the set of all maximal $T$-codes over an alphabet $\Sigma$ by $\mathcal{M}_T(\Sigma)$. Note that the alphabet $\Sigma$ is crucial in the definition of maximality. By Zorn's Lemma, we can easily establish that every $L \in \mathcal{P}_T(\Sigma)$ is contained in some element of $\mathcal{M}_T(\Sigma)$. Again, the proof is a specific instance of a result from dependency theory. We may also prove the following result using dependency theory; the result is also clear in our case:

**Lemma 6.8.1** *Let $T_1 \subseteq T_2$. Then for all $\Sigma$, $\mathcal{M}_{T_2}(\Sigma) \subseteq \mathcal{M}_{T_1}(\Sigma)$.*

### 6.8.1 Decidability and Maximal $T$-Codes

Unlike showing that every $T$-code can be embedded in a maximal $T$-code, to our knowledge, dependency theory has not addressed the problem of deciding whether a language is a maximal code under some dependence system. We address this problem for $T$-codes now. We first require the following technical lemma, which is interesting in its own right (specific cases were known for, e.g., prefix codes [18, Prop. 3.1, Thm. 3.3], hypercodes [185, Cor. to Prop. 11], as well as biprefix and outfix codes [134, Lemmas 3.3 and 3.5]). Let $\tau : \{0,1\}^* \to \{i,d\}^*$ be again given by $\tau(0) = i$ and $\tau(1) = d$.

**Lemma 6.8.2** *Let $T \subseteq \{0,1\}^*$. Let $\Sigma$ be an alphabet. For all $L \in \mathcal{P}_T(\Sigma)$, $L \in \mathcal{M}_T(\Sigma)$ if and only if*

$$L \cup (L \sqcup\!\sqcup_T \Sigma^+) \cup (L \rightsquigarrow_{\tau(T)} \Sigma^+) = \Sigma^+. \tag{6.16}$$

**Proof.** Let $L \in \mathcal{P}_T(\Sigma) - \mathcal{M}_T(\Sigma)$. Then there exists $x \in \Sigma^+$ such that $L \cup \{x\} \in \mathcal{P}_T(\Sigma)$, but $x \notin L$. Thus, assume, contrary to what we want to prove, that $x \in (L \sqcup\!\sqcup_T \Sigma^+) \cup (L \rightsquigarrow_{\tau(T)} \Sigma^+)$.

If $x \in L \sqcup\!\sqcup_T \Sigma^+$, then certainly $x \in (L \cup \{x\}) \sqcup\!\sqcup_T \Sigma^+$, by the monotonicity of $\sqcup\!\sqcup_T$. But this contradicts that $L \cup \{x\}$ is a $T$-code.

If $x \in L \rightsquigarrow_{\tau(T)} \Sigma^+$, then by the monotonicity of $\rightsquigarrow_{\tau(T)}$, $x \in (L \cup \{x\}) \rightsquigarrow_{\tau(T)} \Sigma^+$. But this contradicts that $L \cup \{x\}$ is a $T$-code, by (6.7). Thus, $x \notin L \cup (L \sqcup\!\sqcup_T \Sigma^+) \cup (L \rightsquigarrow_{\tau(T)} \Sigma^+)$.

For the reverse implication, assume that $L \in \mathcal{M}_T(\Sigma)$. Then for all $x \in \Sigma^+$ with $x \notin L$, there exist $y \in L, z \in \Sigma^+$ such that either $x \in y \sqcup\!\sqcup_T z$ or $y \in x \sqcup\!\sqcup_T z$. The second membership is equivalent to $x \in y \rightsquigarrow_{\tau(T)} z$. Thus, we have $x \in (L \sqcup\!\sqcup_T \Sigma^+) \cup (L \rightsquigarrow_{\tau(T)} \Sigma^+)$ for all $x \in \Sigma^+ - L$. The result then follows. ■

**Corollary 6.8.3** *Let $T \subseteq \{0,1\}^*$ be a regular set of trajectories. Given a regular language $L \subseteq \Sigma^+$, it is decidable whether $L \in \mathcal{M}_T(\Sigma)$.*

**Proof.** By Lemma 6.3.9, we can decide whether $L \in \mathcal{P}_T(\Sigma)$. If not, then certainly $L \notin \mathcal{M}_T(\Sigma)$. Otherwise, since $T, L$ are regular, then the languages $L, L \sqcup\!\sqcup_T \Sigma^+$, $L \rightsquigarrow_{\tau(T)} \Sigma^+$, as well as $L \cup$

$(L \sqcup\!\sqcup_T \Sigma^+) \cup (L \rightsquigarrow_{\tau(T)} \Sigma^+)$ are regular. Thus, the equality (6.16) is decidable. ■

Similar results were also obtained by Kari *et al.* [108, Sect. 5]. We now consider the decidability of being a maximal $T$-code for finite languages. Our goal is to give a class of sets of trajectories larger than REG such that for any $T$ in our class, it is decidable whether an arbitrary finite language is a maximal $T$-code.

We first introduce some notation. Let $T \subseteq \{0,1\}^*$. For any $n \geq 0$, let $\eta_n(T) = \{t \in T \ : \ |t|_0 = n\}$. Clearly, $\cup_{n\geq 0}\eta_n(T) = T$.

Before we begin, we require some preliminary lemmas. Recall that a *semilinear set* over $\mathbb{N}^k$ is a finite union of sets of the form $\{\mathbf{u} + \sum_{i=1}^n c_i \mathbf{v_i} \ : \ c_i \in \mathbb{N}\}$ where $\mathbf{u}, \mathbf{v_i} \in \mathbb{N}^k$. The following lemma can be found in Ginsburg [50, Cor. 5.3.2]:

**Lemma 6.8.4** *Let $T \subseteq w_1^* w_2^*$ for $w_1, w_2 \in \{0,1\}^*$. Then $T$ is a CFL if and only if $\{(m,n) \ : \ w_1^m w_2^n \in T\}$ is a semilinear set.*

**Lemma 6.8.5** *Let $T \subseteq w_1^* w_2^*$ for $w_1, w_2 \in \{0,1\}^*$. If $w_1, w_2$ are given and $T$ is an effectively given CFL, then for all $n \geq 1$, $\eta_n(T)$ is an effectively regular language.*

For example, let $T = \{0^m 1^m \ : \ m \geq 0\} \subseteq 0^*1^*$. Then $\eta_n(T) = \{0^n 1^n\}$ for all $n \geq 0$. If $T = (01)^*1^*$, then $\eta_n(T) = (01)^n 1^*$. We note that we cannot relax the conditions of Lemma 6.8.5 to $T \subseteq w_1^* w_2^* w_3^*$, since, e.g., $T = \{1^n 0^m 1^n \ : \ n, m \geq 0\} \subseteq 1^*0^*1^*$, but $\eta_m(T) = \{1^n 0^m 1^n \ : \ n \geq 0\}$, which is not regular if $m > 0$.

**Proof.** Let $T \subseteq w_1^* w_2^*$ for $w_1, w_2 \in \{0,1\}^*$. Let $S$ be the semilinear set such that $w_1^{\alpha_1} w_2^{\alpha_2} \in T$ if and only if $(\alpha_1, \alpha_2) \in S$. Since the union of regular languages is regular, we can assume without loss of generality that $S$ is linear, i.e., there exist $m, k_1, k_2 \geq 0$ and $p_1, r_i \geq 0$ for all $1 \leq i \leq m$ such that

$$S = \{(k_1, k_2) + \sum_{i=1}^{m} n_i(p_i, r_i) \ : \ (n_1, \ldots, n_m) \in \mathbb{N}^m\}.$$

We assume without loss of generality that $(p_j, r_j) \neq (0,0)$ for all $1 \leq j \leq m$, otherwise, we can simply remove this index from our set without affecting $S$. We distinguish between four cases:

(a) $w_1 w_2 \in 1^* + 0^*$. In this case, as $T$ is a unary CFL, it is known that $T$ is a regular language. Thus, so is $\eta_n(T) = T \cap (1^*0)^n 1^*$.

(b) $w_1 \in 1^*$. By case (a), we can assume that $w_2 \notin 1^*$, i.e., that $|w_2|_0 \neq 0$. As $w_1 \in 1^*$, there exists $\alpha \geq 0$ such that

$$T = \{1^{\alpha(k_1 + \sum_{i=1}^m n_i p_i)} w_2^{k_2 + \sum_{i=1}^m n_i r_i} \ : \ (n_1, \ldots, n_m) \in \mathbb{N}^m\}.$$

Let $I \subseteq \mathbb{N}^m$ be defined so that

$$I = \{(n_1, \ldots, n_m) \ : \ |w_2|_0 (k_2 + \sum_{i=1}^m n_i r_i) = n\}.$$

From this, we can see that

$$\eta_n(T) = \{1^{\alpha(k_1 + \sum_{i=1}^m n_i p_i)} w_2^{k_2 + \sum_{i=1}^m n_i r_i} \ : \ (n_1, \ldots, n_m) \in \mathbb{N}^m\}.$$

By reordering if necessary, let $0 \leq m' \leq m$ be the index such that for all $j \leq m'$, $r_j \neq 0$ and for all $m' < j \leq m$, $r_j = 0$. Let $\varphi : I \to \mathbb{N}^{m'}$ be given by $\varphi(n_1, n_2, \ldots, n_m) = (n_1, n_2, \ldots, n_{m'})$. Note that $\varphi^{-1}(\varphi(I)) = I$ as we have that if $(n_1, \ldots, n_m) \in I$, for all $m' < j \leq m$,

$$(n_1, n_2, \ldots, n_{j-1}, n'_j, n_{j+1}, \ldots, n_m) \in I$$

for all $n'_j \in \mathbb{N}$.

Further, note that $\varphi(I)$ is finite, since for all $(n_1, \ldots, n_{m'}) \in \varphi(I)$ and all $j \leq m'$, $n_j$ satisfies

$$n_j \leq \frac{1}{r_j} \left( \frac{n}{|w_2|_0} - k_2 \right).$$

Thus, we can conclude that

$$\eta_n(T) = \{1^{\alpha(k_1 + \sum_{i=1}^{m'} n_i p_i)} ( \prod_{i=m'+1}^{m} (1^{\alpha p_i})^* ) w_2^{k_2 + \sum_{i=1}^{m'} n_i r_i}$$
$$: \ (n_1, \ldots, n_{m'}) \in \varphi(I)\}.$$

and that $\eta_n(T)$ is regular.

(c) $w_2 \in 1^*$. Thus, consider that $\eta_n(T^R) = \eta_n(T)^R$. As $T^R \subseteq (w_2^R)^*(w_1^R)^*$, by (a) or (b), $\eta_n(T^R)$ is regular. As the regular languages are closed under reversal, $\eta_n(T)$ is regular.

(d) $w_1, w_2 \notin 1^*$. Let $I \subseteq \mathbb{N}^m$ be defined by

$$I = \{(n_1, \ldots, n_m) \in \mathbb{N}^m : |w_1|_0(k_1 + \sum_{i=1}^{m} n_i p_i) + |w_2|_0(k_2 + \sum_{i=1}^{m} n_i r_i) = n\}.$$

Note that $I$ is finite, as $|w_1|_0, |w_2|_0 \neq 0$ and $(p_i, r_i) \neq (0, 0)$ for all $1 \leq i \leq m$. Further, we have that

$$\eta_n(T) = \{w_1^{k_1+\sum_{i=1}^{m} n_i p_i} w_2^{k_2+\sum_{i=1}^{m} n_i r_i} : (n_1, \ldots, n_m) \in I\}.$$

From this, we note that $\eta_n(T)$ is finite.

Thus, $\eta_n(T)$ is regular. ■

We are now ready to give our positive decidability result:

**Theorem 6.8.6** *Let $T \subseteq \{0, 1\}^*$ be a CFL such that $T \subseteq w_1^* w_2^*$ for $w_1, w_2 \in \{0, 1\}^*$, where $w_1, w_2$ are given. If $F$ is a finite set, then we can decide whether $F$ is a maximal $T$-code. Furthermore, all constructions are effective.*

**Proof.** Let $T \subseteq w_1^* w_2^*$ be a CFL. Let $F$ be our finite set and let $\ell(F) = \{|x| : x \in F\}$ and $\ell_F = \max\{\ell : \ell \in \ell(F)\}$. First, we note that we can find $T^{\leq \ell_F} = T \cap \{0, 1\}^{\leq \ell_F}$, and that

$$F \rightsquigarrow_{\tau(T)} \Sigma^+ = F \rightsquigarrow_{\tau(T^{\leq \ell_F})} \Sigma^+,$$

which is thus a regular language, since $F$, $\Sigma^+$, $\tau(T^{\leq \ell_F})$ are, as well.

Second, we note that $\eta(T) = \cup_{\ell \in \ell(F)} \eta_\ell(T)$ is a regular language, since $\ell(F)$ is finite, and $\eta_\ell(T)$ is regular by Lemma 6.8.5. Further, we note that

$$F \sqcup\!\sqcup_T \Sigma^+ = F \sqcup\!\sqcup_{\eta(T)} \Sigma^+,$$

which is regular, by the regularity of $F$, $\Sigma^+$ and $\eta(T)$.

Thus, we conclude that $F \cup (F \rightsquigarrow_{\tau(T)} \Sigma^+) \cup (F \sqcup\!\sqcup_T \Sigma^+)$ is a regular language, and thus, we can determine whether this language is equal to $\Sigma^+$. Thus, by Lemma 6.8.2, we can determine whether $F$ is a maximal $T$-code. ■

### 6.8.2 Transitivity and Embedding $T$-codes

Given a class of codes $\mathcal{C}$, and a language $L \in \mathcal{C}$ of given complexity, there has been much research into whether or not $L$ can be *embedded in* (or *completed to*) a maximal element $L' \in \mathcal{C}$ of the same complexity, i.e., a maximal code $L' \in \mathcal{C}$ with $L \subseteq L'$. Finite and regular languages in these classes of codes are of particular interest. For instance, we note that every regular code can be completed to a maximal regular code, while the same is not true for finite codes or finite biprefix codes.

We now show an interesting result on embedding $T$-codes in maximal $T$-codes while preserving complexity. For example, we will show that if $T$ is transitive and regular and $L$ is a regular $T$-code, then we can embed $L$ in a maximal $T$-code which is also regular.

Our construction is a generalization of a result due to Lam [128]. In particular, we define two transformations on languages. Let $T$ be a set of trajectories and $L \subseteq \Sigma^+$ be a language. Then define $U_T(L), V_T(L) \subseteq \Sigma^+$ as

$$\begin{aligned} U_T(L) &= \Sigma^+ - (L \sqcup\!\sqcup_T \Sigma^+ \cup L \rightsquigarrow_{\tau(T)} \Sigma^+); \\ V_T(L) &= U_T(L) - (U_T(L) \sqcup\!\sqcup_T \Sigma^+). \end{aligned}$$

First, we note the following two properties of $U_T(L)$, $V_T(L)$:

**Lemma 6.8.7** *Let $T \subseteq \{0,1\}^*$ be a set of trajectories and $L \in \mathcal{P}_T(\Sigma)$. Then $L \subseteq U_T(L)$ and $L \subseteq V_T(L)$.*

**Proof.** We establish first that $L \subseteq U_T(L)$. Let $x \in L$, but assume that $x \notin U_T(L)$. Then $x \in L \sqcup\!\sqcup_T \Sigma^+$ or $x \in L \rightsquigarrow_{\tau(T)} \Sigma^+$. In the first case, we have $L \cap (L \sqcup\!\sqcup_T \Sigma^+) \neq \emptyset$, contradicting that $L$ is a $T$-code. The second case also contradicts that $L$ is a $T$-code, since then $L \cap (L \rightsquigarrow_{\tau(T)} \Sigma^+) \neq \emptyset$, contradicting (6.7).

We now establish $L \subseteq V_T(L)$. Assume not, then as $L \subseteq U_T(L)$, we must have that $L \cap (U_T(L) \sqcup\!\sqcup_T \Sigma^+) \neq \emptyset$. Assume that $y \in U_T(L)$, $z \in \Sigma^+$ and $x \in L$ are chosen so that $x \in y \sqcup\!\sqcup_T z$. Thus $y \in x \rightsquigarrow_{\tau(T)} z \subseteq L \rightsquigarrow_{\tau(T)} \Sigma^+$, contradicting that $y \in U_T(L)$. Thus, $L \subseteq V_T(L)$. ■

**Theorem 6.8.8** *Let $T \subseteq \{0,1\}^*$ be transitive. Let $\Sigma$ be an alphabet. Then for all $L \in \mathcal{P}_T(\Sigma)$, the language $V_T(L)$ contains $L$ and $V_T(L) \in \mathcal{M}_T(\Sigma)$.*

**Proof.** By Lemma 6.8.7, $L \subseteq V_T(L)$. That $V_T(L)$ is a $T$-code follows from Lemma 6.3.11 applied to $U_T(L)$. Thus, it remains to show that for all $z \in \Sigma^+ - V_T(L)$, $V_T(L) \cup \{z\}$ is not a $T$-code.

Let $z \notin V_T(L)$ be arbitrary. We distinguish two cases:

(a) if $z \notin U_T(L)$, then $z \in (L \sqcup\!\sqcup_T \Sigma^+) \cup (L \rightsquigarrow_{\tau(T)} \Sigma^+)$. If $z \in L \sqcup\!\sqcup_T \Sigma^+ \subseteq V_T(L) \sqcup\!\sqcup_T \Sigma^+$, then $V_T(L) \cup \{z\} \notin \mathcal{P}_T(\Sigma)$. If $z \in L \rightsquigarrow_{\tau(T)} \Sigma^+ \subseteq V_T(L) \rightsquigarrow_{\tau(T)} \Sigma^+$, then again (this time by (6.7)), $V_T(L) \cup \{z\} \notin \mathcal{P}_T(\Sigma)$.

(b) if $z \in U_T(L) - V_T(L)$, then $z \in U_T(L) \sqcup\!\sqcup_T \Sigma^+$. Let $y \in U_T(L)$ be a shortest word such that $z \in y \sqcup\!\sqcup_T \Sigma^+$. We claim that $y \in V_T(L)$. If this were not the case, then as $y \in U_T(L) - V_T(L)$, we have that $y \in U_T(L) \sqcup\!\sqcup_T \Sigma^+$, by definition of $V_T(L)$. Let $y' \in U_T(L)$ be such that $y \in y' \sqcup\!\sqcup_T \Sigma^+$. Thus, we have that $y'\omega_T y \omega_T z$. By transitivity of $T$, $y'\omega_T z$, i.e., $z \in y' \sqcup\!\sqcup_T \Sigma^*$. As $|y'| < |y| < |z|$, we certainly have that $z \in y' \sqcup\!\sqcup_T \Sigma^+$ in particular. But as $|y'| < |y|$, this contradicts our choice of $y$. Thus, $y \in V_T(L)$. But $y, z \in V_T(L) \cup \{z\}$ and $z \in y \sqcup\!\sqcup_T \Sigma^+$ imply that $V_T(L) \cup \{z\} \notin \mathcal{P}_T(\Sigma)$.

Thus, $V_T(L)$ is a maximal $T$-code. ■

There are several consequences of Theorem 6.8.8. We note only one important corollary:

**Corollary 6.8.9** *Let $T \subseteq \{0,1\}^*$ be transitive and regular. Then every regular (resp., recursive) $T$-code is contained in a maximal regular (resp., recursive) $T$-code.*

Corollary 6.8.9 was given for $T = 1^*0^*1^*$ and regular $T$-codes by Lam [128, Prop. 3.2]. Further research into the case when $T$ is not transitive is necessary (for example, the proofs of Zhang

and Shen [206] and Bruyère and Perrin [19] on embedding regular biprefix codes are much more involved than our construction, and do not seem to be easily generalized).

We can extend our embedding results to finite languages with one additional constraint on $T$, namely completeness. The following technical lemma is easily proven:

**Lemma 6.8.10** *Let $T \subseteq \{0,1\}^*$ be complete. Then for all $y \in \Sigma^*$ and for all $m \leq |y|$, there exists $z \in \Sigma^m$ such that $y \in z \sqcup\!\sqcup_T \Sigma^*$. Further, if $m < |y|$, $y \in z \sqcup\!\sqcup_T \Sigma^+$.*

We now show that for transitive and complete sets of trajectories $T$, finite $T$-codes can be completed to finite maximal $T$-codes.

**Corollary 6.8.11** *Let $T \subseteq \{0,1\}^*$ be transitive and complete. Let $\Sigma$ be an alphabet. Then for all finite $F \in \mathcal{P}_T(\Sigma)$, there exists a finite language $F' \in \mathcal{M}_T(\Sigma)$ such that $F \subseteq F'$. Further, if $T$ is effectively regular, and $F$ is effectively given, we can effectively construct $F'$.*

**Proof.** Let $F$ be a finite language and $n = \max\{|x| : x \in F\}$. As $F \in \mathcal{P}_T(\Sigma)$, $n \neq 0$. We first establish the following claim: for all $y \in \Sigma^+$ with $|y| > n$, there exists $u \in U_T(F)$ such that $y \in u \sqcup\!\sqcup_T \Sigma^+$.

Let $y \in \Sigma^+$ be such that $|y| > n$. Then by Lemma 6.8.10, there exists $z$ such that $|z| = n$ and $y \in z \sqcup\!\sqcup_T \Sigma^+$. Note that as $n \neq 0$, $z \in \Sigma^+$. If $z \in U_T(F)$, we have established the claim with $u = z$. Thus, assume that $z \notin U_T(F)$. By definition of $U_T(F)$, we have that $z \in (F \sqcup\!\sqcup_T \Sigma^+) \cup (F \rightsquigarrow_{\tau(T)} \Sigma^+)$. However, $|x| < n$ for all $x \in F \rightsquigarrow_{\tau(T)} \Sigma^+$. Thus, we have that $z \in F \sqcup\!\sqcup_T \Sigma^+ \subseteq U_T(F) \sqcup\!\sqcup_T \Sigma^+$, the inclusion being valid by Lemma 6.8.7. Let $u \in U_T(F)$ be such that $z \in u \sqcup\!\sqcup_T \Sigma^+$. Then $u \,\omega_T\, z$ and $z \,\omega_T\, y$. Thus, by transitivity, $u \,\omega_T\, y$. As $|u| < |y|$, this implies that $y \in u \sqcup\!\sqcup_T \Sigma^+$. Thus, our claim is proven.

We now establish that $V_T(F)$ is finite. Let $y$ be an arbitrary word such that $|y| > n$. By our claim, $y \in U_T(F) \sqcup\!\sqcup_T \Sigma^+$. But by definition of $V_T(F)$, this implies that $y \notin V_T(F)$. Thus, $V_T(F) \subseteq \Sigma^{\leq n}$. Therefore, the conditions of the corollary are met by $V_T(F)$, by Theorem 6.8.8. This completes the proof. ■

In practice, the condition that $T$ be complete is not very restrictive, since natural operations seem to typically be defined by a complete set of trajectories.

In Section 6.9.3 below, we will give alternate conditions on $T$ that ensure that every finite $T$-code can be embedded in a finite maximal $T$-code. However, this result will be a trivial consequence of the fact that for such $T$, all $T$-codes are finite.

We now show that there exist $T$ which are not transitive, and for which the above results do not hold. It is known, for example, that there exist finite biprefix codes which cannot be embedded in a maximal finite biprefix code (see, e.g., Bruyère and Perrin [19, Sect. 3]). We present the following two examples, as well; in the first case, $T$ is regular but not transitive, and for all regular $T$-codes $L$, $L$ cannot be embedded in any maximal CF $T$-code. In the second example, $T$ is not complete, and no finite $T$-code can be embedded in a maximal finite $T$-code.

**Example 6.8.12:** Let $T = (01)^*$; then $\sqcup\!\sqcup_T$ is known as perfect or balanced literal shuffle. Clearly, $T$ is not transitive. Let $\Sigma = \{a\}$. We claim that for all regular languages $L \subseteq a^*$, $L$ is not a maximal $T$-code.

Let $L \subseteq a^*$ be regular. As $L$ is a unary regular language, it is well-known that $L$ corresponds to an ultimately periodic set of natural numbers. That is, there exist $n_0$, $p \in \mathbb{N}$ with $p > 0$ such that for all $n > n_0$, $a^n \in L$ if and only if $a^{n+p} \in L$.

Let $r = \min\{kp : k \geq 1, kp > n_0\}$. Then we have two cases:

(a) if $a^r \in L$, then $a^{2r} \in L$ as well. Thus, as $a^{2r} \in a^r \sqcup\!\sqcup_T a^r$, $L$ is not a $T$-code.

(b) if $a^r \notin L$, then $a^{2r} \notin L$ as well. Thus, consider $L \cup \{a^{2r}\}$. If $L$ is a $T$-code, then as $a^{2r} \notin L \sqcup\!\sqcup_T a^+$ and $L \cap (a^{2r} \sqcup\!\sqcup_T a^+) = \emptyset$, we have that $L \cup \{a^{2r}\}$ is a $T$-code as well. Thus, $L$ is not a maximal $T$-code.

Thus, there are no regular languages in $\mathcal{M}_T(\{a\})$. Further, since the unary context-free and unary regular languages coincide, there are no context-free languages in $\mathcal{M}_T(\{a\})$, either. Thus, e.g., the $T$-code $\{a\}$ cannot be embedded in any regular (or context-free) maximal $T$-code.

We note in passing that one maximal $T$-code containing $\{a\}$ is given by $L = \{a^{c_n} : n \geq 1\}$

where $\{c_n\}_{n\geq 1} = \{1, 3, 4, 5, 7, 9, 11, \dots\}$ is the lexicographically least sequence of positive integers satisfying $m \in \{c_n\} \iff 2m \notin \{c_n\}$. This sequence has received some attention in the literature, and has connections to the Thue-Morse word. We point the reader to A003159 in Sloane [188] for details and references. Clearly, $L$ is not regular. □

**Example 6.8.13:** Let $T = \{0^j 1^{2i} 0^j \ : \ i, j \geq 0\}$. Then $\sqcup\!\sqcup_T$ is the balanced insertion operation. Note that $T$ is transitive, but not complete. Let $\Sigma$ be an alphabet and let $L_o = \{x \in \Sigma^+ \ : \ |x| \equiv 1 \pmod 2\}$. Then for all $L \in \mathcal{P}_T(\Sigma)$, $L \cup L_o \in \mathcal{P}_T(\Sigma)$. Thus, there are no finite maximal $T$-codes. □

## 6.9 Finiteness of all $T$-codes

In this section, we investigate $T \subseteq \{0, 1\}^*$ such that all $\mathcal{P}_T$ codes are finite. It is a well-known result that all hypercodes ($T = \{0, 1\}^*$) are finite, which can be concluded from a result due to Higman [64].

We define the following classes of sets of trajectories:

$$\begin{aligned}
\mathfrak{F}_R &= \{T \in \{0, 1\}^* \ : \ \mathcal{P}_T \cap \text{REG} \subseteq \text{FIN}\};\\
\mathfrak{F}_C &= \{T \in \{0, 1\}^* \ : \ \mathcal{P}_T \cap \text{CF} \subseteq \text{FIN}\};\\
\mathfrak{F}_H &= \{T \in \{0, 1\}^* \ : \ \mathcal{P}_T \subseteq \text{FIN}\}.
\end{aligned}$$

The class $\mathfrak{F}_H$ is of particular importance. If $T$ is a partial order and $T \in \mathfrak{F}_H$, then $T$ is a *well partial order*[1]. This is a subject of tremendous research, not only in the larger theory of partial orders (see the survey of Kruskal [125]), but also within formal language theory as well. Without trying to be exhaustive, we note the work of Jullien [96], Haines [58], van Leeuwen [197], Ehrenfeucht *et al.* [47], Ilie [72, 73], Ilie and Salomaa [80] and Harju and Ilie [59] on well partial orders relating to words. We also refer the reader to the survey of results presented by de Luca and Varricchio [33, Sect. 5].

[1]Recall that we say that $T$ has property $P$ if and only if $\omega_T$ has property $P$.

To begin, we give conditions on $T$ which ensure all regular (or context-free) $T$-codes are finite.

### 6.9.1 Finiteness of Regular $T$-codes

Let $T \subseteq \{0,1\}^*$. Define the *insertion behaviour* of $T$, denoted $ib(T)$, as

$$ib(T) = \{(n_1, n_2, n_3) \in \mathbb{N}^3 \ : \ 0^{n_1}1^{n_2}0^{n_3} \in T\}.$$

Say that $T$ is REG-*pumping compliant* if, for all $i, j, k \in \mathbb{N}$ $(j > 0)$, there exists $j'$ with $0 \leq j' < j$ such that

(i) if $j' = 0$, then $ib(T) \cap \{(i + jm_1, jm_2, k + jm_3) \ : \ m_1, m_3 \geq 0, m_2 > 0\} \neq \emptyset$.

(ii) if $1 \leq j' < j$, then $ib(T) \cap \{(i + j' + jm_1, jm_2, k - j' + jm_3) \ : \ m_1 \geq 0, m_2, m_3 > 0\} \neq \emptyset$.

The use of the terminology 'REG-pumping compliant' will become clear in the following lemma:

**Lemma 6.9.1** *Let $T \subseteq \{0,1\}^*$. If $T$ is* REG-*pumping compliant, then $T \in \mathfrak{F}_R$.*

**Proof.** Let $R \in$ REG be an infinite regular language over $\Sigma$. By the pumping lemma for regular languages, there exist $u, v, w \in \Sigma^*$ such that $v \neq \epsilon$ and $uv^*w \subseteq R$. Let $i = |u|$, $j = |v|$ and $k = |w|$. Note that $j \neq 0$. Let $j'$ be the natural number implied by the REG-pumping compliance condition.

If $j' = 0$, then let $m_1, m_2, m_3$ be chosen so that $m_1, m_3 \geq 0$, $m_2 > 0$ and $(i + jm_1, jm_2, k + jm_3) \in ib(T)$. Let $t = 0^{i+jm_1}1^{jm_2}0^{k+jm_3}$. By definition, $t \in T$. Consider $x = uv^{m_1+m_3}w \in R$ and $y = v^{m_2}$. As $m_2 \neq 0$ and $v \neq \epsilon$, $y \neq \epsilon$. We note that

$$x \sqcup\!\sqcup_t y \ni uv^{m_1} \cdot v^{m_2} \cdot v^{m_3}w = uv^{m_1+m_2+m_3}w.$$

Thus, $(R \sqcup\!\sqcup_T \Sigma^+) \cap R \neq \emptyset$ and $R \notin \mathcal{P}_T$.

If $1 \leq j' < j$, let $m_1 \geq 0, m_2, m_3 > 0$ be chosen so that

$$(i + j' + jm_1, jm_2, k - j' + jm_3) \in ib(T),$$

and hence $t = 0^{i+j'+jm_1}1^{jm_2}0^{k+(j-j')+j(m_3-1)} \in T$. Let $v_1 \in \Sigma^*$ be the prefix of $v$ of length $j'$ and let $v = v_1v_2$ for some $v_2 \in \Sigma^*$.

Consider $x = uv^{m_1+m_3}w \in R$ and $y = (v_2v_1)^{m_2} \neq \epsilon$. Then

$$x \sqcup\!\sqcup_t y \ni uv^{m_1}v_1 \cdot v_2(v_1v_2)^{m_2-1}v_1 \cdot v_2v^{m_3-1}w = uv^{m_1+m_2+m_3}w.$$

Again, $(R \sqcup\!\sqcup_T \Sigma^+) \cap R \neq \emptyset$ and thus $R \notin \mathcal{P}_T$. Thus, $\mathcal{P}_T$ contains no infinite regular languages. ■

The condition of being REG-pumping compliant is not very restrictive. Clearly, if $T \supseteq 0^*1^*0^*$, then $T$ is REG-pumping compliant (in this case, Lemma 6.9.1 is a corollary of a result on outfix codes due to Ito *et al.* [77]). For a broader class of examples, we can consider immune languages.

**Lemma 6.9.2** *Let $T \subseteq \{0,1\}^*$ be a set of trajectories such that $\overline{T} \cap 0^*1^*0^*$ is* REG*-immune. Then $T$ is* REG*-pumping compliant.*

**Proof.** Let $i \geq 0$, $j > 0$, $k \geq 0$ be arbitrary. Consider

$$T_0 = T_0(i,j,k) = 0^i(0^j)^*(1^j)^+(0^j)^*0^k.$$

As $T_0$ is an infinite regular language, $T_0$ is not a subset of $\overline{T} \cap 0^*1^*0^*$. Thus, $T_0 \cap \overline{(\overline{T} \cap 0^*1^*0^*)} = T_0 \cap (T \cup \overline{0^*1^*0^*}) \neq \emptyset$. As $T_0 \subseteq 0^*1^*0^*$, this implies that $T_0 \cap T \neq \emptyset$. Thus, there exist $m_1 \geq 0$, $m_2 > 0$ and $m_3 \geq 0$ such that $0^{i+jm_1}1^{jm_2}0^{k+jm_3} \in T$, i.e., $(i + jm_1, jm_2, k + jm_3) \in ib(T)$. Thus, the REG-pumping compliant conditions are met with $j' = 0$. ■

Next, we show that if $T \subseteq 0^*1^*0^*$, then REG-pumping compliance is necessary to ensure that there are no infinite regular languages in $\mathcal{P}_T$.

**Lemma 6.9.3** *Let $T \subseteq 0^*1^*0^*$ be not* REG*-pumping compliant. Then $\mathcal{P}_T(\Sigma)$ contains an infinite regular language for all $\Sigma$ with $|\Sigma| \geq 2$.*

**Proof.** Let $i, j, k \in \mathbb{N}$ be arbitrary such that $i \geq 0$, $j > 0$, $k \geq 0$,

$$ib(T) \cap \{(i + jm_1, jm_2, k + jm_3) \ : \ m_1, m_3 \geq 0, m_2 > 0\} = \emptyset.$$

and for all $1 \leq j' < j$,

$$ib(T) \cap \{(i + j' + jm_1, jm_2, k - j' + jm_3) \ : \ m_1 \geq 0, m_2, m_3 > 0\} = \emptyset.$$

Let $a, b \in \Sigma$ $(a \neq b)$ and $R = a^i(b^j)^*a^k$. We claim that $R \in \mathcal{P}_T(\Sigma)$. Assume not. Then there exist $\ell_1 > \ell_2 \geq 0$ such that

$$a^i b^{j\ell_1} a^k \in a^i b^{j\ell_2} a^k \sqcup\!\sqcup_T z$$

for some $z \in \{a, b\}^+$. By observation, $z = b^{j(\ell_1 - \ell_2)}$. Thus, let $t \in T$ be chosen so that

$$a^i b^{j\ell_1} a^k \in a^i b^{j\ell_2} a^k \sqcup\!\sqcup_t b^{j(\ell_1 - \ell_2)}.$$

Then as $T \subseteq 0^*1^*0^*$, $t = 0^{i+\alpha j+j'}1^{j(\ell_1-\ell_2)}0^{(j-j')+(\ell_2-\alpha-1)j+k}$ for some $\alpha$ and $j'$ with either $0 \leq \alpha \leq \ell_2$ and $j' = 0$ or $0 \leq \alpha < \ell_2 - 1$ and $1 \leq j' < j$. If $j' = 0$, then $(i+\alpha j, j(\ell_1-\ell_2), k+(\ell_2-\alpha)j) \in ib(T)$ while if $j' \neq 0$, then $(i + j' + \alpha j, j(\ell_1 - \ell_2), k - j' + (\ell_2 - \alpha)j) \in ib(T)$, which are both contradictions. ■

### 6.9.2 Finiteness of Context-free $T$-codes

Let $T \subseteq \{0, 1\}^*$. Define the *2–insertion behaviour* of $T$, denoted $2ib(T)$, as follows:

$$2ib(T) = \{(n_1, n_2, \ldots, n_5) \in \mathbb{N}^5 \ : \ 0^{n_1}1^{n_2}0^{n_3}1^{n_4}0^{n_5} \in T\}.$$

We use $2ib(T)$ to define the notion of CF-*pumping compliance*. The idea is the same as REG-pumping compliance, but with more cases. In particular, say that $T$ is CF-pumping compliant if, for all $i, j_1, j_2, k, \ell \in \mathbb{N}$, with $j_1 + j_2 > 0$, there exist $j_1', j_2' \in \mathbb{N}$ such that $0 \leq j_i' < j_i$ for $i = 1, 2$ and $2ib(T) \cap P \neq \emptyset$, where $P$ is defined as follows:

(a) if $j_1' = j_2' = 0$, then

$$\begin{aligned} P \ = \ & \{(i + j_1\alpha_1, j_1\beta, k + j_1\alpha_2 + j_2\alpha_3, j_2\beta, \ell + j_2\alpha_4) \\ & : \ \alpha_m, \beta \in \mathbb{N}, (1 \leq m \leq 4), \beta > 0, \alpha_1 + \alpha_2 = \alpha_3 + \alpha_4\}. \end{aligned}$$

(b) if $1 \leq j_1' < j_1$ and $j_2' = 0$, then

$$\begin{aligned} P &= \{(i + j_1' + j_1\alpha_1, j_1\beta, k - j_1' + j_1\alpha_2 + j_2(\alpha_3 + \gamma_1), j_2\beta, \ell + j_2(\alpha_4 + \gamma_2)) \\ &\quad : \alpha_m, \beta, \gamma_p \in \mathbb{N}, (1 \leq m \leq 4, 1 \leq p \leq 2), \\ &\quad \beta, \alpha_2 > 0, \alpha_1 + \alpha_2 = \alpha_3 + \alpha_4 + 1, \gamma_1 + \gamma_2 = 1\}. \end{aligned}$$

(c) if $j_1' = 0$ and $1 \leq j_2' < j_2$, then

$$\begin{aligned} P &= \{(i + j_1(\alpha_1 + \gamma_1), j_1\beta, k + j_2' + j_1(\alpha_2 + \gamma_2) + j_2\alpha_3, j_2\beta, \ell - j_2' + j_2\alpha_4) \\ &\quad : \alpha_m, \beta, \gamma_p \in \mathbb{N}, (1 \leq m \leq 4, 1 \leq p \leq 2), \\ &\quad \beta, \alpha_4 > 0, \alpha_1 + \alpha_2 + 1 = \alpha_3 + \alpha_4, \gamma_1 + \gamma_2 = 1\}. \end{aligned}$$

(d) if $1 \leq j_1' < j_1$ and $1 \leq j_2' < j_2$, then

$$\begin{aligned} P &= \{(i + j_1' + j_1\alpha_1, j_1\beta, k - j_1' + j_2' + j_1\alpha_2 + j_2\alpha_3, j_2\beta, \ell - j_2' + j_2\alpha_4) \\ &\quad : \alpha_m, \beta \in \mathbb{N}, (1 \leq m \leq 4),\ \beta, \alpha_2, \alpha_4 > 0, \alpha_1 + \alpha_2 = \alpha_3 + \alpha_4\}. \end{aligned}$$

**Lemma 6.9.4** *Let $T \subseteq \{0,1\}^*$. If $T$ is* CF*-pumping compliant, then $T \in \mathfrak{F}_C$.*

**Proof.** Let $L \in$ CF be an infinite language which is a subset of $\Sigma^+$. Then by the pumping lemma for CFLs, there exist $u, v, w, x, y \in \Sigma^*$ such that $vx \neq \epsilon$ and $\{uv^m wx^m y \ : \ m \geq 0\} \subseteq L$. Let $i = |u|$, $j_1 = |v|$, $k = |w|$, $j_2 = |x|$ and $\ell = |y|$. Let $j_1', j_2'$ be the natural numbers implied by the CF-pumping compliance of $T$. We consider the case $j_1' = 0$ and $1 \leq j_2' < j_2$. The other cases are similar (the differences are similar to the differences between the cases in the proof of Lemma 6.9.1).

Let $\alpha_m, \beta, \gamma_p \in \mathbb{N}$ for $1 \leq m \leq 4$ and $1 \leq p \leq 2$ be such that

$$(i + j_1(\alpha_1 + \gamma_1), j_1\beta, k + j_2' + j_1(\alpha_2 + \gamma_2) + j_2\alpha_3, j_2\beta, \ell - j_2' + j_2\alpha_4) \in 2ib(T). \tag{6.17}$$

Further, we have that $\beta, \alpha_4 > 0$, $\alpha_1 + \alpha_2 + 1 = \alpha_3 + \alpha_4$ and $\gamma_1 + \gamma_2 = 1$, i.e., one of $\gamma_p = 0$ and other is equal to one. Consider that

$$uv^{\alpha_1+\alpha_2+1} wx^{\alpha_3+\alpha_4} y, uv^{\alpha_1+\alpha_2+1+\beta} wx^{\alpha_3+\alpha_4+\beta} y \in L.$$

Further, if $x = x_1x_2$ where $x_1, x_2 \in \Sigma^*$ and $|x_1| = j_2'$, then

$$uv^{\alpha_1+\alpha_2+1+\beta}wx^{\alpha_3+\alpha_4+\beta}y \in z_1 \cdot z_2 \cdot z_3 \sqcup\!\sqcup_t v^\beta(x_2x_1)^\beta$$

where

$$\begin{aligned} z_1 &= uv^{\alpha_1+\gamma_1}, \\ z_2 &= v^{\alpha_2+\gamma_2}wx^{\alpha_3}x_1, \\ z_3 &= x_2x^{\alpha_4-1}y, \\ t &= 0^{i+j_1(\alpha_1+\gamma_1)}1^{j_1\beta}0^{k+j_2'+j_1(\alpha_2+\gamma_2)+j_2\alpha_3}1^{j_2\beta}0^{\ell-j_2'+j_2\alpha_4}. \end{aligned}$$

By (6.17), $t \in T$. Note also that

$$z_1z_2z_3 = uv^{\alpha_1+\alpha_2+1}wx^{\alpha_3+\alpha_4}y \in L.$$

As $vx \neq \epsilon$ and $\beta > 0$, $v^\beta(x_2x_1)^\beta \neq \epsilon$. Thus, $L \notin \mathcal{P}_T$. ■

Note that if $T \supseteq 0^*1^*0^*1^*0^*$ then $T$ satisfies the conditions of Lemma 6.9.4. This instance of our result is also a corollary of a result due to Thierrin and Yu [193, Prop. 3.3(2)].

### 6.9.3 Finiteness of $T$-codes

We now turn to the question of the existence of arbitrary infinite languages in a class of $T$-codes. We first show that if $T$ is bounded, then there is an infinite $T$-code.

**Theorem 6.9.5** *Let $T \subseteq \{0,1\}^*$ be a bounded set of trajectories. Then for all $\Sigma$ with $|\Sigma| > 1$, $\mathcal{P}_T(\Sigma)$ contains an infinite language.*

**Proof.** Let $T \subseteq \{0,1\}^*$ be a bounded language. Then there exist $k \geq 0$ and $w_1, w_2, \ldots, w_k \in \{0,1\}^*$ such that $T \subseteq w_1^*w_2^* \cdots w_k^*$. By Lemma 6.3.2, if we can establish that there is an infinite $T'$-code, where $T' = w_1^* \cdots w_k^*$, the result will follow. Thus, without loss of generality, we let $T = w_1^*w_2^* \cdots w_k^*$.

If $w_1 = w_2 = \cdots = w_k = \epsilon$, then $T = \{\epsilon\}$, and thus $\mathcal{P}_T(\Sigma) = 2^{\Sigma^+} - \{\emptyset\}$, which clearly contains an infinite language.

Otherwise, there exists $i_0$ with $1 \leq i_0 \leq k$ such that $w_{i_0} \neq \epsilon$. For all $1 \leq i \leq k$, let $\alpha_i = |w_i|$. Let $a, b \in \Sigma$ be distinct letters, and define $L_T \subseteq \{a, b\}^+$ by

$$L_T = \{(\prod_{i=1}^{k} a^m b^{\alpha_i}) a^m \ : \ m \geq 0\}.$$

We have that $L_T \subseteq \{a, b\}^+$ as $\alpha_{i_0} \neq 0$. We claim $L_T \in \mathcal{P}_T(\Sigma)$. Assume not. Then there exist $m_1, m_2 \in \mathbb{N}$ with $m_1 > m_2$, $t \in T$ and $z \in \Sigma^+$ such that

$$(\prod_{i=1}^{k} a^{m_1} b^{\alpha_i}) a^{m_1} \in (\prod_{i=1}^{k} a^{m_2} b^{\alpha_i}) a^{m_2} \sqcup\!\sqcup_t z.$$

Thus, we have that $z = a^{(k+1)(m_1 - m_2)}$. Further, let $t_i \in \{0, 1\}^*$ for $1 \leq i \leq k+1$ be defined so that

$$t = (\prod_{i=1}^{k} t_i 0^{\alpha_i}) t_{k+1},$$

where $|t_i|_0 = m_2$ and $|t_i|_1 = m_1 - m_2$ for all $1 \leq i \leq k+1$. As $t \in T$, there exist $j_i \in \mathbb{N}$, $1 \leq i \leq k$, such that $t = \prod_{i=1}^{k} w_i^{j_i}$. Thus, we have that

$$\sum_{i=1}^{k} \alpha_i j_i = (\sum_{i=1}^{k} |t_i| + \alpha_i) + |t_{k+1}|,$$

and so

$$\sum_{i=1}^{k} \alpha_i j_i \geq \sum_{i=1}^{k} |t_i| + \alpha_i.$$

Let $\ell$ with $1 \leq \ell \leq k$ be the minimal index such that

$$\sum_{i=1}^{\ell} \alpha_i j_i \geq \sum_{i=1}^{\ell} |t_i| + \alpha_i. \tag{6.18}$$

Note that $j_\ell > 0$, since if $j_\ell = 0$, then $\ell - 1$ satisfies (6.18) as well, contrary to our choice of $\ell$ (if $\ell = 1$ and $j_1 = 0$, then $|t_1| = 0$, which is a contradiction to $|t_1| = m_1 > 0$).

Let $u_1 = \prod_{i=1}^{\ell-1} t_i 0^{\alpha_i}$, $u_2 = (\prod_{i=\ell+1}^{k} t_i 0^{\alpha_i}) t_{k+1}$, $s_1 = \prod_{i=1}^{\ell-1} w_i^{j_i}$ and $s_2 = \prod_{i=\ell+1}^{k} w_i^{j_i}$. Thus, we have that

$$u_1 t_\ell 0^{\alpha_\ell} u_2 = s_1 w_\ell^{j_\ell} s_2$$

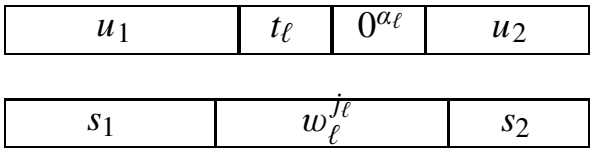


Figure 6.1: Two factorizations of $t$.

with $|u_1| \geq |s_1|$ and $|u_1| + |t_\ell| + \alpha_\ell \leq |s_1| + \alpha_\ell \cdot j_\ell$. The situation is summarized in Fig. 6.1. Thus, we have that $w_\ell^{j_\ell}$ contains a block of zeroes of length $\alpha_\ell$. As $|w_\ell| = \alpha_\ell$ and $j_\ell \neq 0$, this implies that $w_\ell = 0^{\alpha_\ell}$. But then as $t_\ell$ is a factor of $w_\ell^{j_\ell}$, we also have that $t_\ell \in 0^*$. Thus, $|t_\ell|_1 = 0$, and $m_1 = m_2$, a contradiction. ■

Further, there exist uncountably many unbounded trajectories $T$ such that $\mathcal{P}_T$ contains infinite–even infinite *regular*–languages. Infinitely many of these are unbounded regular sets of trajectories.

**Theorem 6.9.6** *Let $T \subseteq \{0,1\}^*$ be a set of trajectories such that there exists $n \geq 0$ such that $T \subseteq 0^{\leq n}1(0+1)^*$. Then for all $\Sigma$ with $|\Sigma| > 1$, $\mathcal{P}_T(\Sigma)$ contains an infinite regular language.*

**Proof.** Let $n \geq 0$ and $T^{(n)} = 0^{\leq n}1(0+1)^*$. By Lemma 6.3.2, it suffices to prove that $\mathcal{P}_{T^{(n)}}(\Sigma)$ contains an infinite regular language. Let $a, b \in \Sigma$ be distinct letters. Consider the regular language $R_n = a^{n+1}b^*$. Assume that $R_n \notin \mathcal{P}_{T^{(n)}}(\Sigma)$. Thus, there exist $i \geq 0$, $t_0 \in T^{(n)}$ and $z \in \{a,b\}^+$ such that $(a^{n+1}b^i \sqcup\!\sqcup_{t_0} z) \cap R_n \neq \emptyset$. Let $t_0 = 0^m 1 t_2$ for some $n \geq m \geq 0$ and $t_2 \in \{0,1\}^*$. Consider that

$$a^{n+1}b^i \sqcup\!\sqcup_{t_0} z = a^m z_1 (a^{n+1-m}b^i \sqcup\!\sqcup_{t_2} z_2)$$

where $z = z_1 z_2$ and $z_1 \in \{a,b\}$.

By assumption, $a^m z_1 (a^{n+1-m}b^i \sqcup\!\sqcup_{t_2} z_2) \cap R_n \neq \emptyset$, so that $z_1 = a$. But now,

$$(a^{n+1-m}b^i \sqcup\!\sqcup_{t_2} z_2) \cap a^{n-m}b^* \neq \emptyset,$$

which is clearly impossible, since $|x|_a \geq n+1-m$ for all $x \in a^{n+1-m}b^i \sqcup\!\sqcup_{t_2} z_2$. ■

The following corollary holds by Lemma 6.7.6.

**Corollary 6.9.7** *Let $T \subseteq \{0,1\}^*$ be a set of trajectories such that there exists $n \geq 0$ such that $T \subseteq (0+1)^*10^{\leq n}$. Then for all $\Sigma$ with $|\Sigma| > 1$, $\mathcal{P}_T(\Sigma)$ contains an infinite regular language.*

We now turn to defining sets $T$ of trajectories such that all $T$-codes are finite. The following proof is generalized from the case $H = (0+1)^*$ found in, e.g., Lothaire [140] or Conway [28, pp. 63–64].

**Lemma 6.9.8** *Let $n, m \geq 1$ be such that $m \mid n$. Let $T_{n,m} = (0^n + 1^m)^*0^{\leq n-1}$. Then $T_{n,m} \in \mathfrak{F}_H$.*

**Proof.** In what follows, let $\omega = \omega_{T_{n,m}}$. Assume that there exists an infinite $T_{n,m}$-code. Then there exists an infinite sequence $\{x_i\}_{i\geq 1}$ which is $\omega$-free, i.e., $i < j$ implies $x_i \ \omega \ x_j$ does not hold. As $T_{n,m} \supseteq 0^*$, $\omega$ is reflexive and we have that $x_i \neq x_j$ for all $i > j \geq 1$.

We now choose (using the axiom of choice) a minimal infinite $\omega$-free sequence as follows: let $y_1$ be the shortest word which begins an infinite $\omega$-free sequence. Let $y_2$ be the shortest word such that $y_1, y_2$ begins an infinite $\omega$-free sequence. We continue in this way. Let $\{y_i\}_{i\geq 1}$ be the resulting sequence. Clearly, $\{y_i\}_{i\geq 1}$ is an infinite $\omega$-free sequence.

As $\omega$ is reflexive, $y_i \neq y_j$ for all $i > j \geq 1$. Therefore, $|y_i| \leq n$ for only finitely many $i \in \mathbb{N}$. Furthermore, since there are only finitely many words of length $n$, there exist $y \in \Sigma^n$ and $\{i_j\}_{j\geq 1} \subseteq \mathbb{N}$ such that $y$ is a prefix of $y_{i_j}$ for all $j \geq 1$. In particular, for all $j \geq 1$, let $u_j \in \Sigma^*$ be the word such that $y_{i_j} = yu_j$. Consider the sequence

$$Y = \{y_1, y_2, y_3, \cdots, y_{i_1-1}, u_1, u_2, \cdots\}.$$

Clearly, as $n \geq 1$, $|u_1| < |y_{i_1}|$. Thus, $Y$ is an infinite sequence which comes before $\{y_i\}_{i\geq 1}$ in our ordering of infinite $\omega$-free sequences, and so two words in $Y$ must be comparable under $\omega$. By assumption, $y_{j_1} \not\omega \ y_{j_2}$ for all $1 \leq j_1 < j_2 \leq i_1 - 1$. Thus, there are two remaining cases:

(i) there exist $1 \leq j \leq i_1 - 1$ and $k \geq 1$ such that $y_j \omega u_k$. Thus, let $t \in T_{n,m}$ and $\alpha \in \Sigma^*$ be chosen so that $u_k \in y_j \sqcup\!\sqcup_t \alpha$. Consider $t' = 1^n t \in T_{n,m}$. Then $y_{i_k} = yu_k \in y(y_j \sqcup\!\sqcup_t \alpha) = y_j \sqcup\!\sqcup_{t'} y\alpha$. Therefore, $y_j \ \omega \ y_{i_k}$. As $j \leq i_1 - 1 < i_k$, this is a contradiction.

(ii) there exist $k > \ell \geq 1$ such that $u_\ell \; \omega \; u_k$. Let $\alpha \in \Sigma^*$ and $t \in T_{n,m}$ be such that $u_k \in u_\ell \sqcup\!\sqcup_t \alpha$. Consider $t' = 0^n t \in T_{n,m}$. Then $y_{i_k} = y u_k \in y(u_\ell \sqcup\!\sqcup_t \alpha) = y u_\ell \sqcup\!\sqcup_{t'} \alpha = y_{i_\ell} \sqcup\!\sqcup_{t'} \alpha$. Thus $y_{i_\ell} \; \omega \; y_{i_k}$. As $\ell < k$, this is a contradiction.

We have arrived at a contradiction. ■

As another class of examples, Ehrenfeucht *et al.* [47, p. 317] note that $\{1^n, 0\}^* \in \mathfrak{F}_H$ for all $n \geq 1$ (their other results, though elegant and interesting, do not otherwise seem to be applicable to our situation).

Note that $T_{1,1} = \{0, 1\}^*$. Let $T_n = T_{n,n}$. For all $1 \leq i < j$, $\mathcal{P}_{T_i} \neq \mathcal{P}_{T_j}$, as $0^i 1^i \in T_i - T_j$. Thus, by Lemma 6.3.6, the classes of $T_i$- and $T_j$-codes are distinct.

**Corollary 6.9.9** *There are infinitely many $T \subseteq \{0, 1\}^*$ which define distinct classes $\mathcal{P}_T$ satisfying $\mathcal{P}_T \subseteq$ FIN.*

Further, the following is immediate:

**Corollary 6.9.10** *Let $T \subseteq \{0, 1\}^*$ be such that $T_n \subseteq T$ for some $n \geq 1$. Then $\mathcal{P}_T \subseteq$ FIN.*

Ilie [73, Sect. 7.7] also gives a class of partial orders which we may phrase in terms of sets of trajectories. In particular, define the set of functions

$$\mathcal{G} = \{g : \mathbb{N} \to \mathbb{N} \; : \; g(0) = 0 \text{ and } 1 \leq g(n) \leq n \text{ for all } n \geq 1\}.$$

Then for all $g \in \mathcal{G}$, we define

$$T_g = \{1^* \prod_{k=1}^{m} (0^{i_k} 1^*) \; : \; i_k \geq 0 \; \forall 1 \leq k \leq m; \; m = g(\sum_{k=1}^{m} i_k)\}.$$

We denote the *upper limit* of a sequence $\{s_n\}_{n \geq 1}$ by $\overline{\lim}_{n \to \infty} s_n$. We have the following result [73, Thm. 7.7.8]:

**Theorem 6.9.11** *Let $g \in \mathcal{G}$. Then $T_g \in \mathfrak{F}_H \iff \overline{lim}_{n \to \infty} \frac{n}{g(n)} < \infty$.*

### 6.9.4 Decidability and Finiteness Conditions

We now consider decidability of membership in $\mathcal{P}_T$ if $T$ satisfies the conditions of the previous sections. We have the following positive decidability results:

**Theorem 6.9.12** *Let $T$ be recursive. If $T \in \mathfrak{F}_R$ (resp., $T \in \mathfrak{F}_C$, $T \in \mathfrak{F}_H$) then given a regular (resp., context-free, context-free) language $L$, it is decidable whether $L \in \mathcal{P}_T$.*

**Proof.** We establish the result for $T \in \mathfrak{F}_C$. The case $T \in \mathfrak{F}_H$ is an instance of this case and the case $T \in \mathfrak{F}_R$ is very similar. Let $T \in$ REC and $T \in \mathfrak{F}_C$. Let $L \in$ CF. We first check if $L$ is infinite. If it is, then certainly $L \notin \mathcal{P}_T$, so we answer no.

If $L$ is finite, then we can effectively find a list of all words in $L$ (consider putting $L$ in Chomsky Normal Form (CNF); see Hopcroft and Ullman [68] for an introduction to CNF). Let $F = L$, where $F$ is some effectively given finite set. Then by Lemma 6.3.10, we can decide whether $L = F \in \mathcal{P}_T$.

■

One might hope for an undecidability result of the following type, which would complement Theorem 6.3.9: for a fixed $T \in$ REG (perhaps with some reasonable assumption, e.g., completeness), then it is undecidable, given a CFL $L$, whether $L \in \mathcal{P}_T$. Theorem 6.9.12 shows us that we cannot hope for a simple such result, since we need to restrict ourselves to those $T$ which do not lie in $\mathfrak{F}_C$ in this case.

### 6.9.5 Up and Down Sets

Let $L \subseteq \Sigma^*$ and $T \subseteq \{0,1\}^*$ Define $\text{DOWN}_T(L)$, $\text{UP}_T(L)$ as

$$\text{DOWN}_T(L) = L \rightsquigarrow_{\tau(T)} \Sigma^*;$$

$$\text{UP}_T(L) = L \sqcup\!\sqcup_T \Sigma^*.$$

Our notation roughly follows Harju and Ilie [59], where $\text{DOWN}_T(L)$ is denoted $\text{DOWN}_{\omega_T}(L)$ and $\text{UP}_T(L)$ is denoted $\text{DOWN}_{\omega_T^{-1}}(L)$.

Our aim in this section is, given $T$, to characterize the complexity $\mathrm{UP}_T(L)$ and $\mathrm{DOWN}_T(L)$ for arbitrary $L$. We will have a particular interest in those $T \in \mathfrak{F}_H$ which are partial orders. Let $\mathfrak{F}_H^{(po)}$ denote the class of all trajectories $T \in \mathfrak{F}_H$ which are partial orders.

Haines [58] observed that for $T = (0+1)^*$, $\mathrm{UP}_T(L)$ and $\mathrm{DOWN}_T(L)$ are regular languages for all $L$. There is an elegant generalization of Haines' result due to Harju and Ilie [59]: If we restrict our attention to those $T \in \mathfrak{F}_H$ which are compatible, then $\mathrm{UP}_T(L)$ and $\mathrm{DOWN}_T(L)$ are still regular languages for all languages $L$. We recall this in the following result, which is a specific case of a result due to Harju and Ilie [59, Thm. 6.3]:[2]

**Theorem 6.9.13** *Let $T \in \mathfrak{F}_H$ be compatible. Let $L \subseteq \Sigma^*$ be a language. Then $\mathrm{UP}_T(L)$, $\mathrm{DOWN}_T(L)$ are regular languages.*

The following corollary is an interesting consequence:

**Corollary 6.9.14** *Let $T \in \mathfrak{F}_H$ satisfy $0^* \subseteq T^*$. Let $L \subseteq \Sigma^*$ be a language. Then the languages $\mathrm{UP}_{T^*}(L)$, $\mathrm{DOWN}_{T^*}(L)$ are regular.*

**Proof.** If $0^* \subseteq T^*$ then $T^*$ is clearly compatible by Corollary 6.4.14. Further, as $T \subseteq T^*$, we have $T^* \in \mathfrak{F}_H$. The result now follows by Theorem 6.9.13. ■

We now consider arbitrary $T \in \mathfrak{F}_H^{(po)}$ and seek to characterize the complexity of $\mathrm{UP}_T(L)$ and $\mathrm{DOWN}_T(L)$. By the same proofs as given for $H = (0+1)^*$ (see, e.g., Harrison [62, Sect. 6.6]), we have the following results:

**Lemma 6.9.15** *Let $T \subseteq \mathfrak{F}_H^{(po)}$. Let $L \subseteq \Sigma^*$. Then*

*(a) there exists a finite language $F \subseteq \Sigma^*$ such that $\mathrm{UP}_T(L) = \mathrm{UP}_T(F)$.*

*(b) there exists a finite language $G \subseteq \Sigma^*$ such that $\mathrm{DOWN}_T(L) = \overline{\mathrm{UP}_T(G)}$.*

We now characterize the complexity of $\mathrm{UP}_T(L)$ and $\mathrm{DOWN}_T(L)$ for all $L$, based on the complexity of $T$:

[2]Note that what Harju and Ilie call monotone, we call compatible.

**Theorem 6.9.16** *Let $\mathcal{C}$ be a cone. Let $T \in \mathfrak{F}_H^{(po)}$ be an element of $\mathcal{C}$. Then for all $L \subseteq \Sigma^*$, $\mathrm{UP}_T(L) \in \mathcal{C}$ and $\mathrm{DOWN}_T(L) \in$ co-$\mathcal{C}$.*

**Proof.** Let $L \subseteq \Sigma^*$. Then there exists $F \subseteq \Sigma^*$ such that $\mathrm{UP}_T(L) = \mathrm{UP}_T(F) = F \sqcup\!\sqcup_T \Sigma^*$. By the closure properties of cones under $\sqcup\!\sqcup_T$, $\mathrm{UP}_T(L) \in \mathcal{C}$. A similar proof shows that $\mathrm{DOWN}_T(L) \in$ co-$\mathcal{C}$. ■

### 6.9.6 $T$-Convexity Revisited

We now turn to the complexity of $T$-convex languages:

**Theorem 6.9.17** *Let $\mathcal{C}$ be a cone. Let $T \in \mathfrak{F}_H^{(po)}$ be an element of $\mathcal{C}$. Then every $T$-convex language is an element of $\mathcal{C} \wedge$ co-$\mathcal{C}$.*

**Proof.** Let $T \in \mathfrak{F}_H^{(po)}$. As $T$ is a partial order, it is reflexive. Thus, if $L$ is a $T$-convex language, we have that $L = \mathrm{UP}_T(L) \cap \mathrm{DOWN}_T(L)$ by Corollary 6.6.2. Thus, by Theorem 6.9.16, the result follows. ■

The following corollary is immediate, based on the closure properties of the recursive and regular languages:

**Corollary 6.9.18** *Let $T \in$ REG (resp., REC) be such that $T \in \mathfrak{F}_H^{(po)}$. If $L$ is a $T$-convex language, then $L \in$ REG (resp., REC).*

Corollary 6.9.18 was known for the case of $H = (0+1)^*$ and $L \in$ REG, see Thierrin [192, Cor. to Prop. 3]. Further, we can also establish the following result:

**Theorem 6.9.19** *Let $T \in \mathfrak{F}_H$ be compatible. Then every $T$-convex language is regular.*

Consider the sets $E_n = \{0, 1^n\}^*$. As noted by Ehrenfeucht *et al.* [47], $E_n \in \mathfrak{F}_H$. As $E_n = E_n^*$ and $0^* \subseteq E_n$, $E_n$ is compatible. Thus, we have that every $E_n$-convex language is regular.

## 6.10 Conclusions

We have introduced the notion of a $T$-code, and examined its properties. Many results which are known in the literature are specific instances of general results on $T$-codes. However, the notion of a $T$-code is not so general as to prevent interesting results from being obtained. We feel that the framework of $T$-codes is very suitable for further analysis of the general structure of the many classes of codes which it generalizes. Further research into this area should prove very useful.

# Chapter 7

# Language Equations

## 7.1 Introduction

The study of language equations, that is, the investigation of solutions to systems of equations in which there are unknowns and fixed language constants, has been the subject of much research in many varied areas. In this chapter, we seek to unify previous results in the theory of language equations initially investigated by Kari [106]. The language operations we consider are shuffle and deletion along trajectories.

We first investigate language equations of the form $L_1 = X \sqcup\!\sqcup_T L_2$, or $L_1 = X \rightsquigarrow_T L_2$, where $T$ is a fixed set of trajectories and $L_1, L_2$ are fixed languages. Equations of this form have previously been studied by Kari [106]. However, by the closure properties for shuffle and deletion on trajectories, we are able to claim positive decidability results when $L_1$, $L_2$ and $T$ are regular.

We also investigate decomposition results for a certain class of trajectories. The problem of decompositions using shuffle on trajectories was initially suggested by Mateescu *et al.* [147] as a possible means of representing complex languages as a combination of simpler languages. For instance, given a language $L$, if $L = L_1 \sqcup\!\sqcup_T L_2$, and if the combined complexity of $L_1, L_2$ and $T$ are less than the complexity of $L$ (for some appropriate measure of complexity) then the triple $[L_1, L_2, T]$ can serve as a more compact representation of the language $L$. Shuffle decompositions

for arbitrary shuffle $T = (0+1)^*$ was studied by Câmpeanu *et al.* [21]. When $T = 0^*1^*$, the decomposition of languages into *prime* parts, that is, into languages which cannot be further decomposed, was studied by Salomaa and Yu [176].

We conclude by investigating systems of equations using shuffle on trajectories. We obtain preliminary results showing that the invertibility of shuffle on trajectories allows some analysis of systems of equations rather than simply individual language equations.

Before continuing, we note that the equations we consider in this chapter are known as *implicit language equations* by, e.g., Leiss [131, Sect. 2.6.2]. Implicit language equations are of the form $R = \varphi$, where $R$ is a fixed language and $\varphi$ is a formula involving constant languages and unknowns connected by language operations. In contrast, *explicit language equations* are of the form $X = \varphi$, where $X$ is an unknown. We consider some explicit language equations in Section 8.11.

## 7.2 Solving One-Variable Equations

We begin by examining equations with one unknown. We find positive decidability results in these cases, provided that the languages involved are regular.

### 7.2.1 Solving $X \sqcup\!\sqcup_T L = R$ and $X \rightsquigarrow_T L = R$

The following is a result of Kari [106, Thm. 4.6]:

**Theorem 7.2.1** *Let $L$, $R$ be languages over $\Sigma$ and $\diamond, \star$ be two binary word operations, which are left-inverses to each other. If the equation $X \diamond L = R$ has a solution $X \subseteq \Sigma^*$, then the language*

$$R' = \overline{\overline{R} \star L}$$

*is also a solution of the equation. Moreover, $R'$ is a superset of all other solutions of the equation.*

By Theorem 7.2.1, Theorem 5.8.1 and Lemma 5.3.1, we note the following corollary:

**Corollary 7.2.2** *Let $T \subseteq \{0,1\}^*$. Let $T, L, R$ be regular languages. Then it is decidable whether the equation $X \sqcup_T L = R$ has a solution $X$.*

The idea is the same as discussed by Kari [106, Thm. 2.3]: we compute $R'$ given in Theorem 7.2.1, and check whether $R'$ is a solution to the desired equation. Since all languages involved are regular and the constructions are effective, we can test for equality of regular languages. Also, we note the following corollary, which is established in the same manner as Corollary 7.2.2:

**Corollary 7.2.3** *Let $T \subseteq \{i,d\}^*$. Let $T, L, R$ be regular languages. Then it is decidable whether the equation $X \rightsquigarrow_T L = R$ has a solution $X$.*

### 7.2.2 Solving $L \sqcup_T X = R$ and $L \rightsquigarrow_T X = R$

The following is also a result of Kari [106, Thm. 4.2]:

**Theorem 7.2.4** *Let $L, R$ be languages over $\Sigma$ and $\diamond, \star$ be two binary word operations, which are right-inverses to each other. If the equation $L \diamond X = R$ has a solution $X \subseteq \Sigma^*$, then the language*

$$R' = \overline{L \star \overline{R}}$$

*is also a solution of the equation. Moreover, $R'$ is a superset of all other solutions of the equation.*

Thus, the following result is easily shown, by appealing to Theorem 5.8.2:

**Corollary 7.2.5** *Let $T \subseteq \{0,1\}^*$. Let $T, L, R$ be regular languages. Then it is decidable whether the equation $L \sqcup_T X = R$ has a solution $X$.*

We now consider the decidability of solutions to the equation $L \rightsquigarrow_T X = R$ where $T$ is a fixed set of trajectories, $L, R$ are regular languages and $X$ is unknown. We have the following result, which is an immediate corollary of Theorem 5.8.3:

**Corollary 7.2.6** *Let $T \subseteq \{i,d\}^*$. Let $T, L, R$ be regular languages. Then it is decidable whether the equation $L \rightsquigarrow_T X = R$ has a solution $X$.*

### 7.2.3 Solving $\{x\} \sqcup\!\sqcup_T L = R$

In this section, we briefly address the problem of finding solutions to equations of the form

$$\{x\} \sqcup\!\sqcup_T L = R$$

where $T$ is a fixed regular set of trajectories, $L$, $R$ are regular languages, and $x$ is an unknown word. This is a generalization of the results of Kari [106].

**Theorem 7.2.7** *Let $\Sigma$ be an alphabet. Let $T \subseteq \{0,1\}^*$ be a fixed regular set of trajectories. Then for all regular languages $R, L \subseteq \Sigma^*$, it is decidable whether there exists a word $x \in \Sigma^*$ such that $\{x\} \sqcup\!\sqcup_T L = R$.*

**Proof.** Let $r = \min\{|y| \ : \ y \in R\}$. Given a DFA for $R$, it is clear that we can compute $r$ by breadth-first search. Then note that $|z| = |x| + |y|$ for all $z \in x \sqcup\!\sqcup_T y$ (regardless of $T$). Thus, it is clear that if $x$ exists satisfying $\{x\} \sqcup\!\sqcup_T L = R$, then $|x| \leq r$. Our algorithm then simply considers all words $x$ of length at most $r$, and checks whether $\{x\} \sqcup\!\sqcup_T L = R$ holds. ■

### 7.2.4 Solving $\{x\} \rightsquigarrow_T L = R$

In this section, we are concerned with decidability of the existence of solutions to the equation

$$\{x\} \rightsquigarrow_T L = R$$

where $x$ is a word in $\Sigma^*$, and $L$, $R$, $T$ are regular languages. Equations of this form have previously been considered by Kari [106]. Our constructions generalize those of Kari directly.

We begin with the following technical lemma:

**Lemma 7.2.8** *Let $\Sigma$ be an alphabet. Then for all sets of trajectories $T \subseteq \{i,d\}^*$, and for all $R, L \subseteq \Sigma^*$, the following equality holds:*

$$\overline{(\overline{R} \sqcup\!\sqcup_{\tau^{-1}(T)} L)} = \{x \in \Sigma^* \ : \ \{x\} \rightsquigarrow_T L \subseteq R\}.$$

**Proof.** Let $x$ be a word such that $\{x\} \rightsquigarrow_T L \subseteq R$, and assume, contrary to what we want to prove, that $x \in \overline{R} \sqcup\!\sqcup_{\tau^{-1}(T)} L$. Then there exist $y \in \overline{R}, z \in L$ and $t \in \tau^{-1}(T)$ such that $x \in y \sqcup\!\sqcup_t z$. By Theorem 5.8.1,

$$y \in x \rightsquigarrow_{\tau(t)} z.$$

As $\tau(t) \in T$, we conclude that $y \in (\{x\} \rightsquigarrow_T L) \cap \overline{R}$. Thus $\{x\} \rightsquigarrow_T L \subseteq R$ does not hold, contrary to our choice of $x$. Thus $x \in \overline{(\overline{R} \sqcup\!\sqcup_{\tau^{-1}(T)} L)}$.

For the reverse inclusion, let $x \in \overline{(\overline{R} \sqcup\!\sqcup_{\tau^{-1}(T)} L)}$. Further, assume that $(\{x\} \rightsquigarrow_T L) \cap \overline{R} \neq \emptyset$. In particular, there exist words $z \in L$ and $t \in T$ such that

$$x \rightsquigarrow_t z \cap \overline{R} \neq \emptyset.$$

Let $y$ be some word in this intersection. As $y \in x \rightsquigarrow_t z$, by Theorem 5.8.1, we have that $x \in y \sqcup\!\sqcup_{\tau^{-1}(t)} z$. Thus, $x \in \overline{R} \sqcup\!\sqcup_{\tau^{-1}(T)} L$, contrary to our choice of $x$. This proves the result. ■

Thus, we can state the main result of this section:

**Theorem 7.2.9** *Let $\Sigma$ be an alphabet. Let $T \subseteq \{i, d\}^*$ be an arbitrary regular set of trajectories. Then the problem "Does there exist a word $x$ such that $\{x\} \rightsquigarrow_T L = R$" is decidable for regular languages $L, R$.*

**Proof.** Let $L, R$ be regular languages. We note that if $R$ is infinite, then the answer to our problem is no; there can only be finitely many deletions along the set of trajectories $T$ from a finite word $x$. Thus, assume that $R$ is finite. Then we can construct the following regular language:

$$P = \overline{(\overline{R} \sqcup\!\sqcup_{\tau^{-1}(T)} L)} - \bigcup_{S \subsetneq R} \overline{(\overline{S} \sqcup\!\sqcup_{\tau^{-1}(T)} L)}.$$

Note that $\subsetneq$ denotes proper inclusion. We claim that $P = \{x \ : \ \{x\} \rightsquigarrow_T L = R\}$.

Assume $x \in P$. Then by Lemma 7.2.8, we have that

$$x \in \{x \ : \ \{x\} \rightsquigarrow_T L \subseteq R\}, \tag{7.1}$$

$$x \notin \{x \ : \ \{x\} \rightsquigarrow_T L \subseteq S \subsetneq R\}. \tag{7.2}$$

Thus, we must have that $\{x\} \rightsquigarrow_T L = R$, since $\{x\} \rightsquigarrow_T L$ is a subset of $R$, but is not contained in any proper subset of $R$.

Similarly, if $\{x\} \rightsquigarrow_T L = R$, then by Lemma 7.2.8, we have that $x \in \overline{(\overline{R} \sqcup\!\sqcup_{\tau^{-1}(T)} L)}$. But as $\{x\} \rightsquigarrow_T L$ is not contained in any $S$ with $S \subsetneq R$, we have that $x \notin \bigcup_{S \subsetneq R} \overline{(\overline{S} \sqcup\!\sqcup_{\tau^{-1}(T)} L)}$. Thus, $x \in P$.

Thus, if $R$ is finite, to decide if a word $x$ exists satisfying $\{x\} \rightsquigarrow_T L = R$, we construct $P$ and test if $P \neq \emptyset$. Since $P$ will be regular, this can be done effectively (as we have noted, if $R$ is infinite, we answer no). ■

## 7.3 Decidability of Shuffle Decompositions

Say that a language $L$ has a *non-trivial shuffle decomposition* with respect to a set of trajectories $T \subseteq \{0,1\}^*$ if there exist $X_1, X_2 \neq \{\epsilon\}$ such that $L = X_1 \sqcup\!\sqcup_T X_2$.

In this section, we are concerned with giving a class of sets of trajectories $T \subseteq \{0,1\}^*$ such that it is decidable, given a regular language $R$, whether $R$ has a non-trivial shuffle decomposition with respect to $T$. For $T = (0+1)^*$, this is an open problem [21, 75]. While we do not settle this open problem, we establish a unified generalization of the results of Kari and Kari and Thierrin [105, 106, 114, 117], which leads to a large class of examples of sets of trajectories where the shuffle decomposition problem is decidable.

Our focus will be on letter-bounded regular sets of trajectories, which we studied in Section 5.5. We will require the following result of Ginsburg and Spanier [54] on bounded regular languages:

**Theorem 7.3.1** *Let $L \subseteq w_1^* w_2^* \cdots w_n^*$ be a regular language. Then there exist $N \geq 1$, and $b_{j,k}, c_{j,k} \in \mathbb{N}$ for all $1 \leq j \leq N$ and $1 \leq j \leq n$ such that*

$$L = \bigcup_{j=1}^{N} w_1^{b_{j,1}} (w_1^{c_{j,1}})^* \cdots w_n^{b_{j,n}} (w_n^{c_{j,n}})^*. \tag{7.3}$$

From results due to Ginsburg and Spanier (see Ginsburg [50, Thm. 5.5.2]) and Szilard *et al.* [190, Thm. 2], we have the following result:

**Corollary 7.3.2** *Let $L \subseteq \Sigma^*$ be a bounded regular language. Then we can effectively compute $w_1, \ldots, w_n \in \Sigma^*$, $N \geq 1$ and $b_{j,k}, c_{j,k} \in \mathbb{N}$ for all $1 \leq j \leq N$ and $1 \leq k \leq n$ such that (7.3) holds.*

We will also require the following observation:

**Lemma 7.3.3** *Let $T \subseteq \{i, d\}^*$ be a letter-bounded regular set of trajectories. Then $T$ is a finite union of $i$-regular sets of trajectories.*

**Proof.** Let $m \geq 0$ and $T \subseteq (i^*d^*)^m i^*$. Then by Theorem 7.3.1, there exist $N \geq 1$, $b_{j,k}, c_{j,k} \in \mathbb{N}$ with $1 \leq j \leq N$ and $1 \leq k \leq 2m+1$ such that

$$T = \bigcup_{j=1}^{N} \left( \prod_{k=1}^{m} i^{b_{j,2k-1}} (i^{c_{j,2k-1}})^* d^{b_{j,2k}} (d^{c_{j,2k}})^* \right) i^{b_{j,2m+1}} (i^{c_{j,2m+1}})^*.$$

Let $T_j = \left( \prod_{k=1}^{m} \#_k (d^{b_{j,2k}} (d^{c_{j,2k}})^*) \right) \#_{m+1}$ for all $1 \leq j \leq N$. Let $\varphi_j$ be defined by $\varphi_j(d) = \{d\}$ and $\varphi_j(\#_k) = i^{b_{j,2k-1}} (i^{c_{j,2k-1}})^*$ for all $1 \leq j \leq m+1$. Then note that $T = \bigcup_{j=1}^{N} \varphi_j(T_j)$. The result thus holds, as $\varphi_j(T_j)$ is $i$-regular for all $1 \leq j \leq N$. ■

We first require a small detour to demonstrate that given a letter-bounded regular set of trajectories, we can compute $m$ such that $T \subseteq (i^*d^*)^m i^*$ (such an $m$ necessarily exists, as is easily observed).

**Lemma 7.3.4** *Let $T \subseteq \{i, d\}^*$ be a letter-bounded regular set of trajectories. Suppose that $T \subseteq w_1^* \cdots w_n^*$ where $w_j \in \{i, d\}^*$, with natural numbers $N, b_{j,k}, c_{j,k}$ with $1 \leq j \leq N$ and $1 \leq k \leq n$ such that*

$$T = \bigcup_{j=1}^{N} w_1^{b_{j,1}} (w_1^{c_{j,1}})^* \cdots w_n^{b_{j,n}} (w_n^{c_{j,n}})^*. \tag{7.4}$$

*If $w_\ell \notin i^* + d^*$ for some $1 \leq \ell \leq n$, then $c_{j,\ell} = 0$ for all $1 \leq j \leq N$.*

**Proof.** Suppose that $w_\ell \notin i^* + d^*$ and that there exists $j$ with $1 \leq j \leq N$ and $c_{j,\ell} \neq 0$. Then there exist $u, v \in \{i, d\}^*$ such that $u(w_\ell^{c_{j,\ell}})^* v \subseteq T$. Therefore, for any natural number $m$, we can choose a word $x$ in $T$ such that more than $m$ blocks of occurrences of $i$ (resp., $d$) are separated by blocks of occurrences of $d$ (resp., $i$). Thus, we cannot have that $T \subseteq (i^*d^*)^m i^*$ for any $m$ and, from this, we can easily see that $T$ is not letter-bounded. ■

The following observation is also useful:

**Fact 7.3.5** *Let $w = w_1 \cdots w_n \in \Sigma^*$ be a word with $w_i \in \Sigma$. Then for all $i \geq 0$, the following inclusion holds:*

$$w^{\leq i} \subseteq (\prod_{i=1}^{n} w_i^*)^i.$$

*In particular, any finite subset of $w^*$ is letter-bounded.*

**Theorem 7.3.6** *Let $T \subseteq \{i, d\}^*$ be a letter-bounded regular set of trajectories. Then we can effectively calculate $m \geq 1$ such that $T \subseteq (i^*d^*)^m i^*$.*

**Proof.** As $T \subseteq \{i, d\}^*$ is bounded and regular, by Corollary 7.3.2, we can effectively determine $w_1, \ldots, w_n \in \{i, d\}^*$ such that $T \subseteq w_1^* w_2^* \cdots w_n^*$, $N \geq 1$, and $b_{j,k}, c_{j,k}$ for all $1 \leq j \leq N$ and $1 \leq k \leq n$ such that

$$T = \bigcup_{j=1}^{N} w_1^{b_{j,1}} (w_1^{c_{j,1}})^* \cdots w_n^{b_{j,n}} (w_n^{c_{j,n}})^*. \tag{7.5}$$

If $w_j \in i^* + d^*$ for all $1 \leq j \leq n$, then we can easily find an $m$ to satisfy our conditions. Suppose $w_j \notin i^* + d^*$ for some $1 \leq j \leq n$. Let $S = \{j \ : \ 1 \leq j \leq n, w_j \in i^* + d^*\}$. Then by Lemma 7.3.4, if $k \notin S$ then $c_{j,k} = 0$ for all $1 \leq j \leq N$.

Thus, we can effectively determine, for all $k \notin S$, $\alpha_k = \max\{b_{j,k} \ : \ 1 \leq j \leq N\}$. Now we note that, using Fact 7.3.5,

$$m \leq \left( \sum_{j \notin S} \alpha_j \cdot |w_j| \right) + |S| + 2$$

(the last term reflects the possibility of needing to change an expression of the form $T \subseteq (d^*i^*)^k d^*$ to $T \subseteq (i^*d^*)^{k+1} i^*$). Thus, we can test, for all $m$ in this range, the resulting (decidable) inclusion $T \subseteq (i^*d^*)^m i^*$. ■

We now return to our investigations of shuffle decompositions. We have the following corollary of Theorem 5.5.1.

**Corollary 7.3.7** *Let $T \subseteq \{i, d\}^*$ be a letter-bounded regular set of trajectories. Then for all regular languages $R$, there are only finitely many regular languages $L'$ such that $L' = R \rightsquigarrow_T L$ for some*

*language $L$. Furthermore, given effective constructions for $T$ and $R$, we can effectively construct a finite set $\mathcal{S}$ of regular languages such that if $L' = R \rightsquigarrow_T L$ for some language $L$, then $L' \in \mathcal{S}$.*

**Proof.** Let $R$ be a regular language accepted by a DFA $M = (Q, \Sigma, \delta, q_0, F)$. Let $T \subseteq (i^*d^*)^m i^*$ for some $m \geq 0$ be a regular set of trajectories. By Theorem 7.3.6, such an $m$ is computable. Then by Lemma 7.3.3 and Corollary 7.3.2, there exist $n \geq 0$ and $T_i \in \mathfrak{I}$ for $1 \leq i \leq n$ such that $T = \cup_{i=1}^{n} T_i$. By (5.5), we know that if $Q_R(T_i, L) = Q_R(T_i, L')$, then $R \rightsquigarrow_{T_i} L = R \rightsquigarrow_{T_i} L'$ for all $1 \leq i \leq n$.

Note that, for all $L \subseteq \Sigma^*$ and all $1 \leq i \leq n$, $Q_R(T_i, L) \subseteq Q^{2m}$. As $Q^{2m}$ is a finite set, there are only finitely many languages of the form $R \rightsquigarrow_{T_i} L$. This set can be obtained by considering all possible choices of sets $Q' \subseteq Q^{2m}$, and constructing the regular language from (5.5) with $Q' = Q_R(T_i, L)$ (duplicates may also then be removed, as we can compare the resulting regular languages).

Let $\mathcal{S}_i$ be the finite set of regular languages of the form $R \rightsquigarrow_{T_i} L$. As

$$R \rightsquigarrow_T L = \bigcup_{i=1}^{n} R \rightsquigarrow_{T_i} L,$$

we have that if $L'$ is of the form $L' = R \rightsquigarrow_T L$, then $L' = \cup_{i=1}^{n} L_i$ where $L_i \in \mathcal{S}_i$ for all $1 \leq i \leq n$. There are again only finitely many languages in $\{\cup_{i=1}^{n} L_i : L_i \in \mathcal{S}_i\}$. This establishes the result. ■

**Theorem 7.3.8** *Let $T \subseteq \{0, 1\}^*$ be a letter-bounded regular set of trajectories. Let $R$ be a regular language over an alphabet $\Sigma$. Then there exists a natural number $n \geq 1$ such that there are $n$ distinct regular languages $Y_i$ with $1 \leq i \leq n$ such that for any $L \subseteq \Sigma^*$ the following are equivalent:*

*(a) there exists a solution $Y \subseteq \Sigma^*$ to the equation $L \sqcup\!\sqcup_T Y = R$;*

*(b) there exists an index $i$ with $1 \leq i \leq n$ such that $L \sqcup\!\sqcup_T Y_i = R$.*

*The languages $Y_i$ can be effectively constructed, given effective constructions for $T$ and $R$. Further, if $Y$ is a solution to $L \sqcup\!\sqcup_T Y = R$, then there is some $1 \leq i \leq n$ such that $Y \subseteq Y_i$.*

**Proof.** Let $T, R$ be given. Let $\mathcal{S}_1(T, R)$ be the finite set of languages of the form $\overline{R} \rightsquigarrow_{\pi(T)} L$ for some $L \subseteq \Sigma^*$. This set is finite and effectively constructible by Corollary 7.3.7. Let $\mathcal{S}(T, R) =$ co-$\mathcal{S}_1(T, R)$.

Let $L$ be arbitrary. Thus, if $L \sqcup\!\sqcup_T Y = R$, then $Y \subseteq X$ for some $X \in \mathcal{S}(T, R)$ by Theorems 5.8.2 and 7.2.4, and $L \sqcup\!\sqcup_T X = R$. Further, each language in $\mathcal{S}(T, R)$ is regular, by Corollary 7.3.7. Thus, (a) implies (b). The implication (b) implies (a) is trivial. ■

The symmetric result also holds:

**Theorem 7.3.9** *Let $T \subseteq \{0, 1\}^*$ be a letter-bounded regular set of trajectories. Let $R$ be a regular language over an alphabet $\Sigma$. Then there exists a natural number $n \geq 1$ such that there are $n$ distinct regular languages $Z_i$ with $1 \leq i \leq n$ such that for any $L \subseteq \Sigma^*$ the following are equivalent:*

*(a) there exists a solution $Z \subseteq \Sigma^*$ to the equation $Z \sqcup\!\sqcup_T L = R$;*

*(b) there exists an index $i$ with $1 \leq i \leq n$ such that $Z_i \sqcup\!\sqcup_T L = R$.*

*The languages $Z_i$ can be effectively constructed, given effective constructions for $T$ and $R$. Further, if $Z$ is a solution to $Z \sqcup\!\sqcup_T L = R$, then there is some $1 \leq i \leq n$ such that $Z \subseteq Z_i$.*

We can now give the main result of this section, which states that the shuffle decomposition problem is decidable for letter-bounded regular sets of trajectories:

**Theorem 7.3.10** *Let $T \subseteq \{0, 1\}^*$ be a letter-bounded regular set of trajectories. Then given a regular language $R$, it is decidable whether there exist $X_1, X_2$ such that $X_1 \sqcup\!\sqcup_T X_2 = R$.*

**Proof.** Let $\mathcal{S}(T, R)$ be the set of languages described by Theorem 7.3.8 and, analogously, let $\mathcal{T}(T, R)$ be the set of languages described by Theorem 7.3.9.

We now note the result follows since if $X_1 \sqcup\!\sqcup_T X_2 = R$ has a solution $[X_1, X_2]$, it also has a solution in $\mathcal{S}(T, R) \times \mathcal{T}(T, R)$, since $\sqcup\!\sqcup_T$ is monotone. Thus, we simply test all the finite (non-trivial) pairs in $\mathcal{S}(T, R) \times \mathcal{T}(T, R)$ for the desired equality. ■

This result was known for catenation, $T = 0^*1^*$ (see, e.g., Kari and Thierrin [117]). However, it also holds for, e.g., the following operations: insertion ($0^*1^*0^*$), $k$-insertion ($0^*1^*0^{\leq k}$ for fixed $k \geq 0$), and bi-catenation ($1^*0^* + 0^*1^*$).

We also note that if the equation $X_1 \sqcup\!\sqcup_T X_2 = R$ has a solution, where $R$ is a regular language and $T$ is a letter-bounded regular set of trajectories, then the equation also has solution $Y_1 \sqcup\!\sqcup_T Y_2 = R$ where $Y_1, Y_2$ are regular languages. This result is well-known for $T = 0^*1^*$ (see, e.g., Choffrut and Karhumäki [25]). For $T = (0 + 1)^*$, this problem is open [21, Sect. 7].

### 7.3.1 1-thin sets of trajectories

Recall that a language $L$ is *1-thin* if $|L \cap \Sigma^n| \leq 1$ for all $n \geq 0$. We now prove that if $T \subseteq \{0, 1\}^*$ is a fixed 1-thin set of trajectories, given a regular language $R$, it is decidable whether $R$ has a shuffle decomposition with respect to $T$.

Define the right-useful solutions to $L \sqcup\!\sqcup_T X = R$ as

$$use_T^{(r)}(X; L) = \{x \in X \ : \ L \sqcup\!\sqcup_T x \neq \emptyset\}. \tag{7.6}$$

The left-useful solutions, denoted $use_T^{(\ell)}(X; L)$, are defined similarly for the equation $X \sqcup\!\sqcup_T L = R$.

**Theorem 7.3.11** *Let $T \subseteq \{0, 1\}^*$ be a 1-thin regular set of trajectories. Given a regular language $R$, it is decidable whether $R$ has a shuffle decomposition with respect to $T$.*

**Proof.** Let $L_1 = R \rightsquigarrow_{\tau(T)} \Sigma^*$ and $L_2 = R \rightsquigarrow_{\pi(T)} \Sigma^*$. Then we claim that

$$\exists X_1, X_2 \text{ such that } R = X_1 \sqcup\!\sqcup_T X_2 \iff L_1 \sqcup\!\sqcup_T L_2 = R. \tag{7.7}$$

The right-to-left implication is trivial. To prove the reverse implication, we first show that if $X_1 \sqcup\!\sqcup_T X_2 = R$, then $use_T^{(\ell)}(X_1; X_2) \subseteq L_1$ and $use_T^{(r)}(X_2; X_1) \subseteq L_2$.

We show only that $use_T^{(\ell)}(X_1; X_2) \subseteq L_1$. The other inclusion is proven similarly. Let $x \in use_T^{(\ell)}(X_1; X_2)$. Then there is some $y \in X_2$ such that $x \sqcup\!\sqcup_T y \neq \emptyset$. As $X_1 \sqcup\!\sqcup_T X_2 = R$, we must

have that $z \in R$ for all $z \in x \sqcup\!\sqcup_T y$. Thus, by Theorem 5.8.1, $x \in z \rightsquigarrow_{\tau(T)} y \subseteq L_1$. The inclusion is proven. Thus,

$$R = X_1 \sqcup\!\sqcup_T X_2 = use_T^{(\ell)}(X_1; X_2) \sqcup\!\sqcup_T use_T^{(r)}(X_2; X_1) \subseteq L_1 \sqcup\!\sqcup_T L_2.$$

To conclude the proof, we need only establish the inclusion $L_1 \sqcup\!\sqcup_T L_2 \subseteq R$.

Let $x \in L_1$. Thus, there exist $\alpha \in R$, $\beta \in \Sigma^*$ and $t \in T$ such that $x \in \alpha \rightsquigarrow_t \beta$. Thus, $\{\alpha\} = x \sqcup\!\sqcup_t \beta$. Now, as $\alpha \in R = X_1 \sqcup\!\sqcup_T X_2$, there is some $x_1 \in X_1$, $x_2 \in X_2$ and $t' \in T$ such that $\{\alpha\} = x_1 \sqcup\!\sqcup_{t'} x_2$.

Consider now that $|t| = |\alpha| = |t'|$. As $T$ is 1-thin, this implies that $t = t'$. Thus,

$$x \sqcup\!\sqcup_t \beta = x_1 \sqcup\!\sqcup_t x_2,$$

from which it is clear that $x = x_1$ and $x_2 = \beta$. Thus, $x \in X_1$. A similar argument establishes that $L_2 \subseteq X_2$. Therefore $L_1 \sqcup\!\sqcup_T L_2 \subseteq X_1 \sqcup\!\sqcup_T X_2 = R$. Thus, we have established that $R = L_1 \sqcup\!\sqcup_T L_2$ and (7.7) holds. The useful solutions are nontrivial iff $L_1, L_2 \neq \{\epsilon\}$. ■

We note that Theorem 7.3.10 and Theorem 7.3.11 do not apply to all sets of trajectories. Thus, to our knowledge, the question of the decidability of the existence of solutions to $R = X_1 \sqcup\!\sqcup_T X_2$ for a given regular language $R$ is still open in the following cases (for details on literal and initial literal shuffle, see Berard [16]):

(a) arbitrary shuffle: $T = (0 + 1)^*$;

(b) literal shuffle: $T = (0^* + 1^*)(01)^*(0^* + 1^*)$;

(c) initial literal shuffle: $T = (01)^*(0^* + 1^*)$.

## 7.4 Solving Quadratic Equations

Let $T \subseteq \{0, 1\}^*$ be a letter-bounded regular set of trajectories. We can also consider solutions $X$ to the equation $X \sqcup\!\sqcup_T X = R$, for regular languages $R$. This is a generalization of a result due to Kari and Thierrin [114].

**Theorem 7.4.1** *Fix a letter-bounded regular set of trajectories $T \subseteq \{0,1\}^*$. Then it is decidable whether there exists a solution $X$ to the equation $X \sqcup_T X = R$ for a given regular language $R$.*

**Proof.** Let $\mathcal{S}(T, R)$ be the set of languages described by Theorem 7.3.8, and, analogously, let $\mathcal{T}(T, R)$ be the set of languages described by Theorem 7.3.9.

Assume the equation $X \sqcup_T X = R$ has a solution. Then we claim that it also has a regular solution. Let $X$ be a language such that $X \sqcup_T X = R$. Then, in particular, $X$ is a solution to the equation $X \sqcup_T Y = R$, where $X$ is fixed and $Y$ is a variable. Thus, by Theorem 7.3.8, there is some regular language $Y_i \in \mathcal{S}(T, R)$ such that $X \sqcup_T Y_i = R$. Further, $X \subseteq Y_i$. Analogously, considering the equation $X \sqcup_T Y_i = R$, $X \subseteq Z_j$ for some regular language $Z_j \in \mathcal{T}(T, R)$. Thus, $X \subseteq Y_i \cap Z_j$, and $Z_j \sqcup_T Y_i = R$.

Let $X_0 = Y_i \cap Z_j$. Then note that $R = X \sqcup_T X \subseteq X_0 \sqcup_T X_0 \subseteq Z_j \sqcup_T Y_i = R$. The inclusions follow by the monotonicity of $\sqcup_T$. Thus, $X_0 \sqcup_T X_0 = R$. By construction, $X_0$ is regular.

Thus, to decide whether there exists $X$ such that $X \sqcup_T X = R$, we construct the set

$$\mathcal{U}(T, R) = \{Y_i \cap Z_j \ : \ Y_i \in \mathcal{S}(T, R), Z_j \in \mathcal{T}(T, R)\},$$

and test each language for equality. If a solution exists, we answer yes. Otherwise, we answer no. ■

## 7.5 Existence of Trajectories

In this section, we consider the following problem: given languages $L_1$, $L_2$ and $R$, does there exist a set of trajectories $T$ such that $L_1 \sqcup_T L_2 = R$? We prove this to be decidable when $L_1$, $L_2$, $R$ are regular languages.

**Theorem 7.5.1** *Let $L_1, L_2, R \subseteq \Sigma^*$ be regular languages. Then it is decidable whether there exists a set $T \subseteq \{0,1\}^*$ of trajectories such that $L_1 \sqcup_T L_2 = R$.*

**Proof.** Let

$$T_0 = \{t \in \{0,1\}^* \ : \ \forall x \in L_1, y \in L_2, x \sqcup_t y \subseteq R\}. \tag{7.8}$$

Note that the following are equivalent definitions of $T_0$:

$$T_0 = \{t \in \{0,1\}^* \ : \ \forall x \in L_1, y \in L_2, (x \sqcup_t y \neq \emptyset \Rightarrow x \sqcup_t y \subseteq R)\}; \tag{7.9}$$

$$T_0 = \{t \in \{0,1\}^* \ : \ \forall x \in L_1 \cap \Sigma^{|t|_0}, y \in L_2 \cap \Sigma^{|t|_1}, (x \sqcup_t y \subseteq R)\}. \tag{7.10}$$

Then we claim that

$$\exists T \subseteq \{0,1\}^* \text{ such that } (L_1 \sqcup_T L_2 = R) \iff L_1 \sqcup_{T_0} L_2 = R.$$

The right-to-left implication is trivial. Assume that there is some $T \subseteq \{0,1\}^*$ such that $L_1 \sqcup_T L_2 = R$. Let $t \in T$. Then for all $x \in L_1$ and $y \in L_2$, $x \sqcup_t y \subseteq L_1 \sqcup_T L_2 = R$. Thus, $t \in T_0$ by definition, and $T_0 \supseteq T$.

Thus, note that $R = L_1 \sqcup_T L_2 \subseteq L_1 \sqcup_{T_0} L_2$. It remains to establish that $L_1 \sqcup_{T_0} L_2 \subseteq R$. But this is clear from the definition of $T_0$. Thus $L_1 \sqcup_{T_0} L_2 = R$ and the claim is established.

We now establish that $T_0$ is regular and effectively constructible; to do this, we establish instead that $\overline{T_0} = \{0,1\}^* - T_0$ is regular.

Let $M_j = (Q_j, \Sigma, \delta_j, q_j, F_j)$ be a complete DFA accepting $L_j$ for $j = 1, 2$. Let $M_r = (Q_r, \Sigma, \delta_r, q_r, F_r)$ be a complete DFA accepting $R$. Define an NFA $M = (Q, \{0,1\}, \delta, q_0, F)$ where $Q = Q_1 \times Q_2 \times Q_r$, $q_0 = [q_1, q_2, q_r]$, $F = F_1 \times F_2 \times (Q_r - F_r)$, and $\delta$ is defined as follows:

$$\delta([q_j, q_k, q_\ell], 0) = \{[\delta_1(q_j, a), q_k, \delta_r(q_\ell, a)] \ : \ a \in \Sigma\} \quad \forall [q_j, q_k, q_\ell] \in Q_1 \times Q_2 \times Q_r,$$

$$\delta([q_j, q_k, q_\ell], 1) = \{[q_j, \delta_2(q_k, a), \delta_r(q_\ell, a)] \ : \ a \in \Sigma\} \quad \forall [q_j, q_k, q_\ell] \in Q_1 \times Q_2 \times Q_r.$$

Then we note that $\delta$ has the following property: for all $t \in \{0,1\}^*$,

$$\delta([q_1, q_2, q_r], t) = \{[\delta(q_1, x), \delta(q_2, y), \delta(q_r, x \sqcup_t y)] \ : \ x, y \in \Sigma^*, |x| = |t|_0, |y| = |t|_1\}.$$

By (7.10), if $t \in \overline{T_0}$ there is some $x, y \in \Sigma^*$ such that $x \in L_1$, $y \in L_2$, $|x| = |t|_0$, $|y| = |t|_1$ but $x \sqcup_t y \cap \overline{R} \neq \emptyset$. This is exactly what is reflected by the choice of $F$. Thus, $L(M) = \overline{T_0}$.

Thus, as $T_0$ is effectively regular, to determine whether there exists $T$ such that $L_1 \sqcup\!\sqcup_T L_2 = R$, we construct $T_0$ and test $L_1 \sqcup\!\sqcup_{T_0} L_2 = R$. ■

Note that the proof of Theorem 7.5.1 is similar in theme to the proofs of, e.g., Kari [106, Thm. 4.2, Thm. 4.6]: they each construct a maximal solution to an equation, and that solution is regular. The maximal solution is then tested as a possible solution to the equation to determine if any solutions exist. However, unlike the results of Kari, Theorem 7.5.1 does not use the concept of an inverse operation.

We can also repeat Theorem 7.5.1 for the case of deletion along trajectories. The results are identical, with the proof following by the substitution of $T_0 = \{t \in \{i,d\}^* : \forall x \in L_1, y \in L_2, x \rightsquigarrow_t y \subseteq R\}$. The proof that $T_0$ is regular differs slightly from that above; we leave the construction to the reader. Thus, we have the following result:

**Theorem 7.5.2** *Let $L_1, L_2, R \subseteq \Sigma^*$ be regular languages. Then it is decidable whether there exists a set $T \subseteq \{i,d\}^*$ of trajectories such that $L_1 \rightsquigarrow_T L_2 = R$.*

## 7.6 Undecidability of One-Variable Equations

We now turn to establishing undecidability results for equations with one unknown. We focus on the case where one of the remaining languages is an LCFL, and the other is a regular language.

Let $\Pi_0, \Pi_1 : \{0,1\}^* \to \{0,1\}^*$ be the projections given by $\Pi_0(0) = 0$, $\Pi_0(1) = \epsilon$ and $\Pi_1(1) = 1$, $\Pi_1(0) = \epsilon$. We say that $T \subseteq \{0,1\}^*$ is *left-enabling* (resp., *right-enabling*) if $\Pi_0(T) = 0^*$ (resp., $\Pi_1(T) = 1^*$).

We first show that if a set of trajectories is regular and left- or right-enabling, then it is undecidable whether the corresponding equation has a solution:

**Theorem 7.6.1** *Fix $T \subseteq \{0,1\}^*$ to be a regular set of left-enabling (resp., right-enabling) trajectories. For a given LCFL $L$ and regular language $R$, it is undecidable whether or not $L \sqcup\!\sqcup_T X = R$ (resp., $X \sqcup\!\sqcup_T L = R$) has a solution $X$.*

**Proof.** Let $T$ be left-enabling. Let $\Sigma$ be an alphabet of size at least two and let $\#, \$ \notin \Sigma$. Let $R = (\Sigma^+ + \#^+) \sqcup_T \$^*$. By the closure properties of $\sqcup_T$, and the fact that $T$ is regular, $R$ is a regular language. Let $L \subseteq \Sigma^+$ be an arbitrary LCFL and $L_\# = L + \#^+$. Note that $L_\#$ is an LCFL. We claim that

$$L_\# \sqcup_T X = R \text{ has a solution } \iff L = \Sigma^+. \tag{7.11}$$

This will establish the result, since it is undecidable whether an arbitrary LCFL $L \subseteq \Sigma^+$ satisfies $L = \Sigma^+$.

First, if $L = \Sigma^+$, then note that $X = \$^*$ is a solution for $L_\# \sqcup_T X = R$. Second, assume that $X$ is a solution for $L_\# \sqcup_T X = R$. It is clear that for all $X$,

$$L_\# \sqcup_T X = R \iff L_\# \sqcup_T use_T^{(r)}(X; L_\#) = R, \tag{7.12}$$

where $use_T^{(r)}(X, L)$ is defined by (7.6). Thus, we will focus on useful solutions to the equation $L_\# \sqcup_T X = R$.

Now, we note that, assuming that $use_T^{(r)}(X, L_\#)$ is a solution to $L_\# \sqcup_T use_T^{(r)}(X; L_\#) = R$, $use_T^{(r)}(X, L_\#)$ cannot contain words with letters from $\Sigma$, because words in $R$ do not contain words with both # and letters from $\Sigma$.

In particular, let $x \in use_T^{(r)}(X, L_\#) \subseteq X$. Then there exists $y \in L_\#$ (in particular, $y \neq \epsilon$) such that $y \sqcup_T x \neq \emptyset$. Consider the word $\#^{|y|}$. As $y$ and $\#^{|y|}$ have the same length, we must have that $\#^{|y|} \sqcup_T x \neq \emptyset$.

Consider any $z \in \#^{|y|} \sqcup_T x \subseteq L_\# \sqcup_T X$. As $|y| \neq 0$, $|z|_\# > 0$. As $L_\# \sqcup_T X = R$, we must have that $z \in (\Sigma^+ + \#^+) \sqcup_T \$^*$. Thus, $z \in (\# + \$)^+$, and consequently, $x \in (\# + \$)^*$. Thus, $use_T^{(r)}(X, L_\#) \subseteq (\# + \$)^*$.

Let $\Pi_\Sigma : (\Sigma \cup \{\#, \$\})^* \to \Sigma^*$ be the projection onto $\Sigma$. Now as $T$ is left-enabling, note that

$\Pi_\Sigma(R) = \Sigma^+$, by definition of $R = (\Sigma^+ + \#^+) \sqcup\!\sqcup_T \$^*$. Thus,

$$\begin{aligned}
\Sigma^+ &= \Pi_\Sigma(R) = \Pi_\Sigma(L_\# \sqcup\!\sqcup_T X) \\
&= \Pi_\Sigma(L_\# \sqcup\!\sqcup_T use_T^{(r)}(X, L_\#)) \subseteq \Pi_\Sigma(L_\# \sqcup\!\sqcup_T (\# + \$)^*) \\
&= \Pi_\Sigma((L + \#^+) \sqcup\!\sqcup_T (\# + \$)^*) = \Pi_\Sigma((L \sqcup\!\sqcup_T (\# + \$)^*) + (\#^+ \sqcup\!\sqcup_T (\# + \$)^*)) \\
&= \Pi_\Sigma(L \sqcup\!\sqcup_T (\# + \$)^*) \\
&= L \subseteq \Sigma^+.
\end{aligned}$$

The last equality is valid since $T$ is left-enabling, and therefore, for all $x \in L$, there is some $j \geq 0$ such that $x \sqcup\!\sqcup_T (\$ + \#)^j \neq \emptyset$. We conclude that $L = \Sigma^+$, and thus, by (7.11), the result follows. The proof in the case that $T$ is right-enabling is similar. ■

Say that a set $T \subseteq \{0, 1\}^*$ of trajectories if *left-preserving* (resp., *right-preserving*) if $T \supseteq 0^*$ (resp., $T \supseteq 1^*$). Note that if $T$ is complete, then it is both left- and right-preserving.

We can give an incomparable result which removes the condition that $T$ must be regular, but must strengthen the conditions on words in $T$. Namely, $T$ must be left-preserving rather than left-enabling:

**Theorem 7.6.2** *Fix $T \subseteq \{0, 1\}^*$ to be a set of left-preserving (resp., right-preserving) trajectories. Given an LCFL $L$ and a regular language $R$, it is undecidable whether there exists a language $X$ such that $L \sqcup\!\sqcup_T X = R$ (resp., $X \sqcup\!\sqcup_T L = R$).*

**Proof.** Let $T$ be left-preserving (the proof when $T$ is right-preserving is similar). It is clear that for all $X$,

$$L \sqcup\!\sqcup_T X = R \iff L \sqcup\!\sqcup_T use_T^{(r)}(X; L) = R.$$

Thus, we will focus on useful solutions to our equation.

Let $\Sigma$ be our alphabet and $\# \notin \Sigma$. Let $L \subseteq \Sigma^+$ be an arbitrary LCFL. Let $L_\# = L + \#^+$. Note that $\epsilon \notin L_\#$ and that $L_\#$ is an LCFL. We claim that $L_\# \sqcup\!\sqcup_T use_T^{(r)}(X; L_\#) = \Sigma^+ + \#^+$ if and only if $L = \Sigma^+$ and $use_T^{(r)}(X; L_\#) = \{\epsilon\}$.

First, assume that $L = \Sigma^+$ and $use_T^{(r)}(X; L_\#) = \{\epsilon\}$. Then $L_\# = \Sigma^+ + \#^+$ and

$$\begin{aligned} L_\# \sqcup\!\sqcup_T X &= L_\# \sqcup\!\sqcup_T use_T^{(r)}(X; L_\#) \\ &= (\Sigma^+ + \#^+) \sqcup\!\sqcup_T \{\epsilon\} \\ &= (\Sigma^+ + \#^+), \end{aligned}$$

since $T \supseteq 0^*$.

Now, assume that $L_\# \sqcup\!\sqcup_T use_T^{(r)}(X; L_\#) = \Sigma^+ + \#^+$. Let $x \in use_T^{(r)}(X; L_\#)$. Then there exists $y \in L_\#$ ($y \neq \epsilon$) such that $y \sqcup\!\sqcup_T x \neq \emptyset$. Consider $\#^{|y|}$. As $|y| = |\#^{|y|}|$, we must have that $\#^{|y|} \sqcup\!\sqcup_T x \neq \emptyset$.

For all $z \in \#^{|y|} \sqcup\!\sqcup_T x \subseteq L_\# \sqcup\!\sqcup_T use_T^{(r)}(X; L_\#)$, as $|y| \neq 0$, $|z|_\# > 0$. Further,

$$z \in L_\# \sqcup\!\sqcup_T use_T^{(r)}(X; L_\#) = \Sigma^+ + \#^+.$$

Thus, we must have that $z \in \#^+$ and that $x \in \#^*$. Thus, $use_T^{(r)}(X; L_\#) \subseteq \#^*$.

We now show that $\epsilon \in use_T^{(r)}(X; L_\#)$. As $L_\# \sqcup\!\sqcup_T use_T^{(r)}(X; L_\#) = \Sigma^+ + \#^+$, for all $y \in \Sigma^+$, there exist $\alpha \in L_\#$ and $\beta \in use_T^{(r)}(X; L_\#) \subseteq \#^*$ such that $y \in \alpha \sqcup\!\sqcup_T \beta$. If $\beta \neq \epsilon$, then $|y|_\# > 0$. Thus $\alpha = y$, and $\beta = \epsilon \in use_T^{(r)}(X; L_\#)$. This also demonstrates that $\Sigma^+ \subseteq L_\#$, which implies that $L = \Sigma^+$.

It remains to show that $use_T^{(r)}(X; L_\#) = \{\epsilon\}$. Let $\#^i \in use_T^{(r)}(X; L_\#)$ for some $i > 0$. Then, there is some $y \in L_\# = \Sigma^+ + \#^+$ such that $y \sqcup\!\sqcup_T \#^i \neq \emptyset$.

If $y \in \Sigma^+$, then for all $z \in y \sqcup\!\sqcup_T \#^i$, $|z|_\Sigma, |z|_\# > 0$, which contradicts that $z \in \Sigma^+ + \#^+$, since $L_\# \sqcup\!\sqcup_T use_T^{(r)}(X; L_\#) = \Sigma^+ + \#^+$. Thus, $y \in \#^+$. But then let $y' \in \Sigma^+$ be chosen so that $|y| = |y'|$. We have that $y' \in L_\#$ as well. We are thus reduced to the first case with $y'$ and $\#^i$, and our assumption that $\#^i \in use_T^{(r)}(X; L_\#)$ is therefore false.

We have established that a (useful) solution to the equation

$$(L + \#^+) \sqcup\!\sqcup_T X = (\Sigma^+ + \#^+)$$

exists if and only if $L = \Sigma^+$. Therefore, the existence of such solutions must be undecidable. ■

Note that as $use_T^{(r)}(X; L_\#) = \{\epsilon\}$, Theorem 7.6.2 remains undecidable even if the required (useful) language is required to be a singleton.

We also note that if $R$ and $L$ are interchanged in the equations of the statements of Theorem 7.6.2 or Theorem 7.6.1, the corresponding problems are still undecidable. The proofs are trivial, and are left to the reader.

## 7.7 Undecidability of Shuffle Decompositions

It has been shown [21] that it is undecidable whether a context-free language has a nontrivial shuffle decomposition with respect to the set of trajectories $\{0, 1\}^*$. Here we extend this result for arbitrary complete regular sets trajectories.

If $T$ is a complete set of trajectories, then any language $L$ has decompositions $L \sqcup\!\sqcup_T \{\epsilon\}$ and $\{\epsilon\} \sqcup\!\sqcup_T L$. Below we exclude these trivial decompositions; all other decompositions of $L$ are said to be nontrivial.

**Theorem 7.7.1** *Let $T$ be any fixed complete regular set of trajectories. For a given context-free language $L$ it is undecidable whether or not there exist languages $X_1, X_2 \neq \{\epsilon\}$ such that $L = X_1 \sqcup\!\sqcup_T X_2$.*

**Proof.** Let $P = (u_1, \ldots, u_k; v_1, \ldots, v_k)$, $k \geq 1$, $u_i, v_i \in \Sigma^*$, $i = 1, \ldots, k$, be an arbitrary PCP instance. We construct a context-free language $L(P)$ such that $L(P)$ has a nontrivial decomposition along the set of trajectories $T$ if and only if the instance $P$ does not have a solution.

Choose $\Omega = \Sigma \cup \{a, b, \#, \flat_1, \flat_2, \natural_1, \natural_2, \$_1, \$_2\}$, where $\{a, b, \#, \flat_1, \flat_2, \natural_1, \natural_2, \$_1, \$_2\} \cap \Sigma = \emptyset$. Let

$$L_0 = \left(\flat_1^+(\Sigma \cup \{a, b, \#\})^*\natural_1^+ \cup \flat_2^+(\Sigma \cup \{a, b, \#\})^*\natural_2^+\right) \sqcup\!\sqcup_T (\$_1^+ \cup \$_2^+). \tag{7.13}$$

Define

$$\begin{aligned} L_1' \;=\; & \{ab^{i_1} \cdots ab^{i_m} \# u_{i_m} \cdots u_{i_1} \# \mathrm{rev}(v_{j_1}) \cdots \mathrm{rev}(v_{j_n}) \# b^{j_n} a \cdots b^{j_1} a \\ & : i_1, \ldots, i_m, j_1, \ldots, j_n \in \{1, \ldots, k\},\ m, n \geq 1\} \end{aligned}$$

and let

$$L_1 = L_0 - [\natural_1^+ L_1' \natural_1^+ \sqcup\!\sqcup_T \$_2^+].$$

Using the fact that $T$ is regular, it is easy to see that a nondeterministic pushdown automaton $M$ can verify that a given word is not in $\natural_1^+ L_1' \natural_1^+ \sqcup\!\sqcup_T \$_2^+$. On input $w$, using the finite state control $M$ keeps track of the unique trajectory $t$ (if it exists) such that $w \rightsquigarrow_{\tau(t)} \$_2^* \in \natural_1^+(\Sigma \cup \{a, b, \#\})^* \natural_1^+$ and $w \rightsquigarrow_{\pi(t)} \natural_1^+(\Sigma \cup \{a, b, \#\})^* \natural_1^+ \in \$_2^*$. If $t \notin T$, $M$ accepts. Also if $t$ does not exist, $M$ accepts. Using the stack $M$ can verify that $w \rightsquigarrow_{\tau(t)} \$_2^* \notin \natural_1^+ L_1' \natural_1^+$ by guessing where the word violates the definition of $L_1'$. Note that this verification can be interleaved with the computation checking whether $t$ is in $T$. Since $L_0$ is regular, it follows that $L_1$ is context-free.

Define

$$\begin{aligned} L_2' \;=\; & \{ab^{i_1} \cdots ab^{i_m} \# w \# \mathrm{rev}(w) \# b^{i_m} a \cdots b^{i_1} a \\ & : \; w \in \Sigma^*, i_1, \ldots, i_m \in \{1, \ldots, k\}, \; m \geq 1\} \end{aligned}$$

and let

$$L_2 = L_0 - [\natural_1^+ L_2' \natural_1^+ \sqcup\!\sqcup_T \$_2^+].$$

As above it is seen that $L_2$ is context-free. It follows that also the language

$$L(P) = L_1 \cup L_2 = L_0 - [\natural_1^+ (L_1' \cap L_2') \natural_1^+ \sqcup\!\sqcup_T \$_2^+] \tag{7.14}$$

is context-free.

First consider the case where the PCP instance $P$ does not have a solution. Now $L_1' \cap L_2' = \emptyset$ and (7.13) gives a nontrivial decomposition for $L(P) = L_0$ along the set of trajectories $T$.

Secondly, consider the case where the PCP instance $P$ has a solution. This means that there exists a word

$$w_0 \in L_1' \cap L_2'. \tag{7.15}$$

For the sake of contradiction we assume that we can write

$$L(P) = X_1 \sqcup\!\sqcup_T X_2, \tag{7.16}$$

where $X_1, X_2 \neq \{\epsilon\}$.

We establish a number of properties that the languages $X_1$ and $X_2$ must necessarily satisfy. We first claim that it is not possible that

$$\text{alph}(X_1) \cap \{\flat_i, \natural_i\} \neq \emptyset \text{ and } \text{alph}(X_2) \cap \{\flat_j, \natural_j\} \neq \emptyset \tag{7.17}$$

where $\{i, j\} = \{1, 2\}$. If the above relations would hold, then the completeness of $T$ would imply that $X_1 \sqcup\!\sqcup_T X_2$ has some word containing a letter from $\{\flat_1, \natural_1\}$ and a letter from $\{\flat_2, \natural_2\}$. This is impossible since $X_1 \sqcup\!\sqcup_T X_2 \subseteq L_0$.

Let $\Phi = \{\flat_1, \flat_2, \natural_1, \natural_2\}$. Since $L(P)$ has both words that contain letters $\flat_1$, $\natural_1$ and words that contain letters $\flat_2$, $\natural_2$, by (7.17) the only possibility is that all the letters of $\Phi$ "come from" one of the components $X_1$ and $X_2$. We assume in the following that

$$\text{alph}(X_2) \cap \Phi = \emptyset. \tag{7.18}$$

This can be done without loss of generality since the other case is completely symmetric (we can just interchange the letters 0 and 1 in $T$.)

Next we show that

$$\text{alph}(X_2) \cap (\Sigma \cup \{a, b, \#\}) = \emptyset. \tag{7.19}$$

Let $\Pi_\Phi : \Omega^* \to \Phi^*$ be the projection onto $\Phi$. Since $\Pi_\Phi(L(P)) = \flat_1^+\natural_1^+ \cup \flat_2^+\natural_2^+$ and $X_2$ does not contain any letters of $\Phi$, it follows that $\Pi_\Phi(X_1) = \flat_1^+\natural_1^+ \cup \flat_2^+\natural_2^+$. Thus if (7.19) does not hold, the completeness of $T$ implies that $X_1 \sqcup\!\sqcup_T X_2$ contains words where a letter from $\Sigma \cup \{a, b, \#\}$ occurs before a letter from $\{\flat_1, \flat_2\}$ or after a letter from $\{\natural_1, \natural_2\}$. Hence (7.19) holds.

Since $X_2 \neq \{\epsilon\}$, the equations (7.18) and (7.19) imply that

$$\text{alph}(X_2) \cap \{\$_1, \$_2\} \neq \emptyset.$$

Since $L(P)$ has words with letters $\$_1$, other words with letters $\$_2$, and no words containing both letters $\$_1$, $\$_2$, using again the completeness of $T$ it follows that

$$\text{alph}(X_1) \cap \{\$_1, \$_2\} = \emptyset. \tag{7.20}$$

Now consider the word $w_0 \in L_1' \cap L_2'$ given by (7.15). We have $\flat_i w_0 \natural_i \sqcup\!\sqcup_T \$_i \subseteq L_i$, $i = 1, 2$, and let $u_i \in \flat_i w_0 \natural_i \sqcup\!\sqcup_T \$_i$, $i = 1, 2$, be arbitrary. We can write

$$u_i = x_{i,1} \sqcup\!\sqcup_{t_i} x_{i,2}, \text{ such that } x_{i,j} \in X_j, t_i \in T, i = 1, 2, j = 1, 2.$$

By (7.18), (7.19) and (7.20) we have

$$X_1 \subseteq (\Phi \cup \Sigma \cup \{a, b, \#\})^* \text{ and } X_2 \subseteq \{\$_1, \$_2\}^*$$

and hence

$$x_{i,1} = \flat_i w_0 \natural_i, \quad x_{i,2} = \$_i, \quad i = 1, 2.$$

Now $x_{1,1} \sqcup\!\sqcup_{t_1} x_{2,2} \subseteq X_1 \sqcup\!\sqcup_T X_2$. Let $\eta = \flat_1 w_0 \natural_1 \sqcup\!\sqcup_{t_1} \$_2 \in x_{1,1} \sqcup\!\sqcup_T x_{2,2}$, Then $\eta \notin L(P)$ by the choice of $w_0$ and (7.14). This contradicts (7.16). ■

In the proof of Theorem 7.7.1, whenever the CFL has a nontrivial decomposition along the set of trajectories $T$, it has a decomposition where the component languages are, in fact, regular. This gives the following result:

**Corollary 7.7.2** *Let $T$ be any fixed complete regular set of trajectories. For a given context-free language $L$ it is undecidable whether or not*

*(a) there exist regular languages $X_1, X_2 \neq \{\epsilon\}$ such that $L = X_1 \sqcup\!\sqcup_T X_2$.*

*(b) there exist context-free languages $X_1, X_2 \neq \{\epsilon\}$ such that $L = X_1 \sqcup\!\sqcup_T X_2$.*

## 7.8 Undecidability of Existence of Trajectories

We now turn to undecidability results for problems involving the existence of a set of trajectories satisfying a certain equation.

**Lemma 7.8.1** *Given an LCFL $L$, it is undecidable whether there exists $T \subseteq \{0, 1\}^*$ such that $L \sqcup\!\sqcup_T \{\epsilon\} = \Sigma^*$ (resp., whether $\{\epsilon\} \sqcup\!\sqcup_T L = \Sigma^*$).*

**Proof.** We claim that

$$\exists\, T \subseteq \{0,1\}^* \text{ such that } L \sqcup\!\sqcup_T \{\epsilon\} = \Sigma^* \iff L = \Sigma^*.$$

If $L = \Sigma^*$, then $T = 0^*$ satisfies the equation. Assume that there exists $T$ such that $L \sqcup\!\sqcup_T \{\epsilon\} = \Sigma^*$. Then for all $x \in \Sigma^*$, there exist $y \in L$ and $t \in T$ such that $x \in y \sqcup\!\sqcup_t \epsilon$. But this only happens if $x = y$ and $t = 0^{|x|}$. Thus, $x \in L$. Therefore, $L = \Sigma^*$. This establishes the first part of the lemma. The second follows on noting that $T \subseteq \{0,1\}^*$ satisfies $L \sqcup\!\sqcup_T \{\epsilon\} = \Sigma^*$ iff $sym_s(T)$ satisfies $\{\epsilon\} \sqcup\!\sqcup_{sym_s(T)} L = \Sigma^*$. ■

We will require some additional constructs before proving the remaining case. For a set $I \subseteq \mathbb{N}$, let $\Sigma^I = \{x \in \Sigma^* : |x| \in I\}$.

We show the following undecidability result:

**Lemma 7.8.2** *Given an LCFL $L$, it is undecidable whether there exists $I \subseteq \mathbb{N}$ such that $L = \Sigma^I$.*

**Proof.** We appeal to Corollary 2.5.4. Let $P(L)$ be true if $L = \Sigma^I$ for some $I \subseteq \mathbb{N}$. Note that $P$ is non-trivial, as, e.g., $P(\{a^n b^n : n \geq 0\})$ does not hold. Further, $P(\Sigma^*)$ is true, since $\Sigma^* = \Sigma^{\mathbb{N}}$ in our notation. Note that $P$ is preserved under quotient, since if $L = \Sigma^I$ and $a \in \Sigma$ is arbitrary, then $L/a = \Sigma^{I'}$ where $I' = \{x : x + 1 \in I\}$. Thus, we can apply Corollary 2.5.4 and it is undecidable whether $P(L)$ holds for an arbitrary LCFL $L$. ■

We note that a similar undecidability result was proven by Kari and Sosík [112, Lemma 8.1] for proving undecidability results of the following form: given $R_1, R_2 \in$ REG and $L \in$ CF, does $L \sqcup\!\sqcup_T R_1 = R_2$ hold? Necessary and sufficient conditions on regular sets of trajectories $T$ such that the above problem is decidable are given by Kari and Sosík. The related undecidability proof given by Kari and Sosík uses a reduction from PCP.

**Lemma 7.8.3** *Given an LCFL $L$, it is undecidable whether there exists $T \subseteq \{0,1\}^*$ such that $\Sigma^* \sqcup\!\sqcup_T \{\epsilon\} = L$.*

**Proof.** Let $L$ be an LCFL. Then we claim that

$$\exists T \subseteq \{0,1\}^* \text{ such that } L = \Sigma^* \sqcup\!\sqcup_T \{\epsilon\} \iff \exists I \subseteq \mathbb{N} \text{ such that } L = \Sigma^I.$$

($\Leftarrow$): If $I \subseteq \mathbb{N}$ is such that $L = \Sigma^I$, then let $T = \{0\}^I$. Then we can easily see that $L = \Sigma^* \sqcup\!\sqcup_T \{\epsilon\}$. ($\Rightarrow$): If $T \subseteq \{0,1\}^*$ exists such that $L = \Sigma^* \sqcup\!\sqcup_T \{\epsilon\}$, then by definition of shuffle on trajectories, we can assume without loss of generality that $T \subseteq 0^*$. Let $I = \{i \ : \ 0^i \in T\}$. Then we can see that $L = \Sigma^I$. Therefore, since it is undecidable whether $L = \Sigma^I$, we have established the result. ■

To summarize, we have established the following result:

**Corollary 7.8.4** *Given an LCFL $L$ and regular languages $R_1$, $R_2$, it is undecidable whether there exists $T \subseteq \{0,1\}^*$ such that (a) $R_1 \sqcup\!\sqcup_T R_2 = L$, (b) $R_1 \sqcup\!\sqcup_T L = R_2$ or (c) $L \sqcup\!\sqcup_T R_1 = R_2$.*

We now turn to deletion along trajectories.

**Lemma 7.8.5** *Given an LCFL $L$, it is undecidable whether there exists $T \subseteq \{i,d\}^*$ such that $L \rightsquigarrow_T \{\epsilon\} = \Sigma^*$.*

**Proof.** Let $\Sigma$ be an alphabet of size at least two, and let $L \subseteq \Sigma^*$ be an LCFL. Then we can verify that

$$\exists T \subseteq \{i,d\}^* \text{ such that } L \rightsquigarrow_T \{\epsilon\} = \Sigma^* \iff L = \Sigma^*, T \supseteq i^*.$$

The right-to-left implication is easily verified. For the reverse implication, let $T \subseteq \{i,d\}^*$ be such that $L \rightsquigarrow_T \{\epsilon\} = \Sigma^*$. Let $x \in \Sigma^*$ be arbitrary. Then there exist $y \in L$ and $t \in T$ such that $x \in y \rightsquigarrow_t \epsilon$. By definition, $y = x$ and $t = i^{|x|}$. From this we can see that $L = \Sigma^*$ and $T \supseteq i^*$. ■

**Lemma 7.8.6** *Given an LCFL $L$, it is undecidable whether there exists $T \subseteq \{i,d\}^*$ such that $\Sigma^* \rightsquigarrow_T \{\epsilon\} = L$.*

**Proof.** It is easy to verify that

$$\exists T \subseteq \{i,d\}^* \text{ such that } L = \Sigma^* \rightsquigarrow_T \{\epsilon\} \iff \exists I \subseteq \mathbb{N} \text{ such that } L = \Sigma^I.$$

($\Leftarrow$): If $I \subseteq \mathbb{N}$ is such that $L = \Sigma^I$, then let $T = \{i\}^I$. Then we can easily see that $L = \Sigma^* \rightsquigarrow_T \{\epsilon\}$. ($\Rightarrow$): If $T \subseteq \{i, d\}^*$ exists such that $L = \Sigma^* \rightsquigarrow_T \{\epsilon\}$, then by definition of deletion along trajectories, we can assume without loss of generality that $T \subseteq i^*$. Let $I = \{j \ : \ i^j \in T\}$. Then we can see that $L = \Sigma^I$. ■

**Lemma 7.8.7** *Given a LCFL $L \subseteq \{a, b\}^*$, it is undecidable whether there exists $T \subseteq \{i, d\}^*$ such that $(\overline{a}a + \overline{b}b)^* \rightsquigarrow_T L = (\overline{a} + \overline{b})^*$, where $\{\overline{a}, \overline{b}\}$ is a marked copy of $\{a, b\}$.*

**Proof.** Let $R_1 = (\overline{a}a + \overline{b}b)^*$ and $R_2 = (\overline{a} + \overline{b})^*$. We show that there exists $T \subseteq \{i, d\}^*$ such that $R_1 \rightsquigarrow_T L = R_2$ holds if and only if $L = \{a, b\}^*$.

Assume that there exists $T \subseteq \{i, d\}^*$ such that $R_1 \rightsquigarrow_T L = R_2$. Let $x \in R_2$ be arbitrary. Then there exist $y \in R_1$, $z \in L$ and $t \in T$ such that $x \in y \rightsquigarrow_t z$. Let $y = \prod_{i=1}^{m} \overline{y_i} y_i$ where $y_i \in \{a, b\}$. As $R_2 \subseteq \{\overline{a}, \overline{b}\}^*$, and $L \subseteq \{a, b\}^*$, we must have that $t = (id)^m$ and $z = \overline{x}$. Thus, $L = \{a, b\}^*$, since $x$ was chosen arbitrarily in $R_2$.

The converse equality $R_1 \rightsquigarrow_T L = R_2$ with $L = \{a, b\}^*$ and $T = (id)^*$ is easily verified. Thus, as it is undecidable whether $L = \{a, b\}^*$, the result is established. ■

Thus, we have demonstrated the following result:

**Corollary 7.8.8** *Given an LCFL $L$ and regular languages $R_1$, $R_2$, it is undecidable whether there exists $T \subseteq \{i, d\}^*$ such that (a) $R_1 \rightsquigarrow_T R_2 = L$, (b) $L \rightsquigarrow_T R_1 = R_2$, or (c) $R_1 \rightsquigarrow_T L = R_2$.*

## 7.9 Systems of Language Equations

The study of language equations is part of the larger study of systems of language equations and their solutions. In this section, we move from our previous work on the study single language equations to the study of systems of language equations.

Leiss makes the following strong criticism in his monograph on language equations:

> Somewhat related results, for a single equation in a single variable, were reported in [Kari] [106]; however, this paper restricts the class $\mathrm{EX}_A(\mathrm{CONST}; \mathrm{OP}, X_1, \ldots, X_n)$ in

> our terminology [the set of all systems of language equations over the alphabet $A$ in the variables $X_1, \ldots, X_n$, with operations from OP and taking constant languages, and having a solution in, the class of languages CONST] to one where only a single operator occurs, which, moreover, is assumed invertible with respect to words (not languages). Effectively, this excludes the standard language equations; the equations considered in [106] are essentially word equations. Even more damaging for the generality of these results, only a single equation can be treated at a time, since in order to be able to talk about (nontrivial) systems of equations, it is necessary to have at least two variables present in at least one equation. Therefore, this paper does not contribute significantly toward our goal of establishing a theory of language equations. [131, p. 127]

Leiss does not address the results of Kari and Thierrin [117], which extends the criticized work of Kari to deal with decomposition of languages via catenation. This is perhaps because these results only deal with catenation, and not a general set of operations. However the results of Section 7.3 deal with a large class of operations defined by shuffle on trajectories and therefore introduce a situation where "at least two variables [are] present in at least one equation". Thus, the results of Section 7.3 suggest that the framework introduced by Kari [106], and extended by Kari and Thierrin [117], does represent a valid contribution to the theory of language equations.

In this section, we seek to extend this study of language equations even further and directly address the criticisms of Leiss by considering systems of equations involving shuffle on trajectories. We again focus on decidability of the existence of solutions to a system of equations. We feel again that this shows that the equations considered have merit in the theory of language equations.

We consider systems of equations of the following form. Let $n \geq 1$. Let $\Sigma$ be an alphabet and $R_1, \ldots, R_n$ be regular languages over $\Sigma$. Let $X_1, \ldots, X_m$ be variables. Further, let $Y_{i,1}, Y_{i,2} \in \{X_1, \ldots, X_m\} \cup \text{REG}$ for all $1 \leq i \leq n$ (i.e., $Y_{i,j}$ is either a variable or a regular language over $\Sigma$). Let $T_i \subseteq \{0, 1\}^*$ for all $1 \leq i \leq n$ be regular sets of trajectories, subject to the condition that if $Y_{i,1}, Y_{i,2}$ are both variables, then $T_i$ is letter-bounded. We define our system of equations as follows: for all $1 \leq i \leq n$, let $E_i$ be the equation

$$E_i : \; R_i = Y_{i,1} \sqcup\!\sqcup_{T_i} Y_{i,2}. \tag{7.21}$$

Our problem then is the following: Given the system of equations $E_i$ for $1 \leq i \leq n$, does there exist a solution $[X_1, \ldots, X_m]$?

**Theorem 7.9.1** *Let $E_i$ with $1 \leq i \leq n$ be a system of equations as given by (7.21) and the description above. It is decidable whether there exists a solution $[X_1, \ldots, X_m]$ to the system.*

**Proof.** Let $1 \leq i \leq n$. Let $\mathcal{S}(T_i, R_i)$ and $\mathcal{T}(T_i, R_i)$ be the sets of languages described by Theorems 7.3.8 and 7.3.9, respectively. For all $1 \leq i \leq n$ and $1 \leq j \leq m$, define sets $\mathcal{V}_j^{(i)}$ of languages as follows:

(a) If $Y_{i,1} = X_j$ and $Y_{i,2} \subseteq \Sigma^*$, then $\mathcal{V}_j^{(i)} = \{\overline{\overline{R_i} \rightsquigarrow_{\tau(T_i)} Y_{i,2}}\}$.

(b) If $Y_{i,1} \subseteq \Sigma^*$ and $Y_{i,2} = X_j$, then $\mathcal{V}_j^{(i)} = \{\overline{\overline{R_i} \rightsquigarrow_{\pi(T_i)} Y_{i,2}}\}$.

(c) If $Y_{i,1} = X_j$ and $Y_{i,2} \in \{X_1, \ldots, X_m\} - \{X_j\}$, then $\mathcal{V}_j^{(i)} = \mathcal{S}(T_i, R_i)$.

(d) If $Y_{i,1} \in \{X_1, \ldots, X_m\} - \{X_j\}$ and $Y_{i,2} = X_j$, then then $\mathcal{V}_j^{(i)} = \mathcal{T}(T_i, R_i)$.

(e) If $Y_{i,1} = Y_{i,2} = X_j$, then $\mathcal{V}_j^{(i)} = \{L_1 \cap L_2 \ : L_1 \in \mathcal{S}(T_i, R_i), L_2 \in \mathcal{T}(T_i, R_i)\}$.

(f) If $Y_{i,1}, Y_{i,2} \in (\{X_1, \ldots, X_m\} - \{X_j\}) \cup \text{REG}$, then $\mathcal{V}_j^{(i)} = \{\Sigma^*\}$.

For all $1 \leq j \leq m$, let

$$\mathcal{V}_j = \{\bigcap_{i=1}^{n} Z^{(i)} \ : \ Z^{(i)} \in \mathcal{V}_j^{(i)}\}.$$

**Claim 7.9.2** *The system $E_i$ ($1 \leq i \leq n$) has a solution $[X_1, \ldots, X_m]$ iff it has a solution in $\prod_{j=1}^{m} \mathcal{V}_j$ $(= \mathcal{V}_1 \times \mathcal{V}_2 \times \cdots \times \mathcal{V}_m)$.*

**Proof.** ($\Leftarrow$): Trivial.

($\Rightarrow$): Assume there exists a solution $[X_1, \ldots, X_m]$. Let $1 \leq j \leq m$ be arbitrary. We show that as $X_j$ is a solution to each of the equations $E_i$ in which $X_j$ appears, there is also a language $Z_j \in \mathcal{V}_j$ which is a solution, and $X_j \subseteq Z_j$. Let $1 \leq i \leq n$ be chosen so that $X_j \in \{Y_{i,1}, Y_{i,2}\}$. There are five cases:

(a) $X_j = Y_{i,1}$ and $Y_{i,2} \subseteq \Sigma^*$. Then $E_i$ is given by $R_i = X_j \sqcup\!\sqcup_{T_i} Y_{i,2}$. By Theorems 7.2.1 and 5.8.1, we have that $X_j \subseteq \overline{\overline{R_i} \rightsquigarrow_{\tau(T_i)} Y_{i,2}}$. Thus, let $Z_j^{(i)} = \overline{\overline{R_i} \rightsquigarrow_{\tau(T_i)} Y_{i,2}}$. We also have that $R_i = Z_j^{(i)} \sqcup\!\sqcup_{T_i} Y_{i,2}$ and that $Z_j^{(i)} \in \mathcal{V}_j^{(i)}$.

(b) $X_j = Y_{i,2}$ and $Y_{i,1} \subseteq \Sigma^*$: similar to case (a).

(c) $X_j = Y_{i,1}$ and $Y_{i,2} \in \{X_1, \ldots, X_m\} - \{X_j\}$. Then we have that $X_j \subseteq Z_j^{(i)}$ for some $Z_j^{(i)} \in \mathcal{V}_j^{(i)}$ by Theorem 7.3.8. Further, $R_i = Z_j^{(i)} \sqcup\!\sqcup_{T_i} Y_{i,2}$.

(d) $X_j = Y_{i,2}$ and $Y_{i,1} \in \{X_1, \ldots, X_m\} - \{X_j\}$. This case is similar to case (c).

(e) $X_j = Y_{i,1} = Y_{i,2}$. Then $E_i$ is given by $R_i = X_j \sqcup\!\sqcup_{T_i} X_j$. By the proof of Theorem 7.4.1, we have that $X_j \subseteq Z_j^{(i)}$ for some $Z_j^{(i)} \in \mathcal{V}_j^{(i)}$. Further, $Z_j^{(i)} \sqcup\!\sqcup_{T_i} Z_j^{(i)} = R_i$.

Thus, we have that for all $1 \leq i \leq n$ and $1 \leq j \leq m$, if $X_j$ appears in $E_i$, then $X_j \subseteq Z_j^{(i)}$ for some $Z_j^{(i)} \in \mathcal{V}_j^{(i)}$. Further, replacing $X_j$ by $Z_j^{(i)}$ in $E_i$ also yields a solution. If $X_j$ does not appear in $E_i$, then $Z_j^{(i)} = \Sigma^*$, and $X_j \subseteq Z_j^{(i)}$. Let

$$Z_j = \bigcap_{i=1}^{n} Z_j^{(i)}.$$

Note that $X_j \subseteq Z_j$ and that $Z_j \in \mathcal{V}_j$.

We now show that replacing $X_j$ with $Z_j$ still results in a solution to the system of equations $E_i$ with $1 \leq i \leq n$, i.e., that $[X_1, \ldots, X_{j-1}, Z_j, X_{j+1}, \ldots, X_m]$ is a solution to the system. Consider an arbitrary $i$ with $1 \leq i \leq n$ where $X_j$ appears in $E_i$. There are again five cases:

(a) $X_j = Y_{i,1}$ and $Y_{i,2} \subseteq \Sigma^*$. Then $R_i = X_j \sqcup\!\sqcup_{T_i} Y_{i,2}$. By Theorems 7.2.1 and 5.8.1, we have that

$$\begin{aligned} R_i &= X_j \sqcup\!\sqcup_{T_i} Y_{i,2} \subseteq Z_j \sqcup\!\sqcup_{T_i} Y_{i,2} \\ &\subseteq Z_j^{(i)} \sqcup\!\sqcup_{T_i} Y_{i,2} = R_i. \end{aligned}$$

The inclusions are due to the monotonicity of $\sqcup\!\sqcup_{T_i}$. Thus, $Z_j$ satisfies $E_i$.

(b) $X_j = Y_{i,2}$ and $Y_{i,1} \subseteq \Sigma^*$: similar to case (a).

(c) $X_j = Y_{i,1}$ and $Y_{i,2} \in \{X_1, \ldots, X_m\} - \{X_j\}$. Let $1 \leq \ell \leq m$ be chosen so that $X_\ell = Y_{j,2}$. Then note that

$$\begin{aligned} R_i &= X_j \sqcup\!\sqcup_{T_i} X_\ell \subseteq Z_j \sqcup\!\sqcup_{T_i} X_\ell \\ &\subseteq Z_j^{(i)} \sqcup\!\sqcup_{T_i} X_\ell = R_i. \end{aligned}$$

The inclusions are again by the monotonicity of $\sqcup\!\sqcup_{T_i}$ and the final equality is due to Theorem 7.3.8. Thus, $Z_j$ is a solution to equation $E_i$.

(d) $X_j = Y_{i,2}$ and $Y_{i,1} \in \{X_1, \ldots, X_m\} - \{X_j\}$. This case is similar to case (c).

(e) $X_j = Y_{i,1} = Y_{i,2}$. Then $E_i$ is given by $R_i = X_j \sqcup\!\sqcup_{T_i} X_j$. Again, we have that

$$\begin{aligned} R_i &= X_j \sqcup\!\sqcup_{T_j} X_j \subseteq Z_j \sqcup\!\sqcup_{T_i} Z_j \\ &\subseteq Z_j^{(i)} \sqcup\!\sqcup_{T_i} Z_j^{(i)} = R_i. \end{aligned}$$

Thus, $Z_j$ is a solution to $E_i$.

Thus, we have established that if a solution $[X_1, \ldots, X_m]$ exists, each $X_j$ may be replaced by some $Z_j \in \mathcal{V}_j$. This establishes the claim. ■

We now return to our main proof. We know that each set $\mathcal{V}_j^{(i)}$ is finite and contains only regular languages, each of which may be effectively constructed. Thus, there are finitely many effectively regular languages in $\mathcal{V}_j$ for all $1 \leq j \leq m$, and the set $\prod_{j=1}^{m} \mathcal{V}_j$ consists of finitely many $m$-tuples of effectively regular languages. We can test each of these $m$-tuples for equality. This gives an effective procedure for determining whether solutions to this systems of equations exist. ■

We note that the systems we consider cannot be reduced to a single language equation in the manner of Baader and Narendran [13] (see also Baader and Küsters [11]) since our equations do not involve an explicit union operation.

We also note that for systems of equations as given by (7.21), if the system has a solution $[X_1, \ldots, X_m]$, it also has a solution $[Y_1, \ldots, Y_m]$ which consists of regular languages. We refer to the reader to Choffrut and Karhumäki [25] and Polak [167] for a discussion of systems of language equations involving catenation ($T = 0^*1^*$), Kleene closure and union.

## 7.10 Conclusions

In this chapter, we have considered language equations involving shuffle and deletion on trajectories. Positive decidability results have been obtained when the fixed languages involved in our equations are regular. When context-free languages are involved, undecidability results have been obtained. We have also considered systems of equations involving shuffle on trajectories.

In particular, we have made progress in the problem posed by shuffle decompositions. The question of whether a regular language $R$ has a non-trivial shuffle decomposition $R = X_1 \sqcup_T X_2$ when $T = (0+1)^*$ remains open. However, for a substantial and practically significant class of sets of trajectories–namely, the regular letter-bounded sets of trajectories–we have positively answered the shuffle decomposition problem.

We have also investigated decidability problems for equations where the unknown is the set of trajectories. While we have solved the decidability problems for regular languages and LCFLs, the constructions used are distinct from those in the remainder of this chapter, as they do not explicitly involve the use of an inverse operation.

# Chapter 8

# Iteration of Trajectory Operations

## 8.1 Introduction

Iterated concatenation, known as Kleene closure, is one of the defining operations of regular languages, and its properties are well known. As is commonly noted, "regular expressions without the star operator define only finite languages " [201, p. 77] (others, e.g., Salomaa [175] also express the same idea). There are many fundamental and deep results in formal language theory related to Kleene closure: we mention only the study of primitive words and star-height as examples. In this chapter, we examine iteration of trajectory-based operations, such as iterated (arbitrary) shuffle, which has been a topic for active research for the past 25 years [55, 85, 86, 87, 88, 93, 170, 182, 198].

We generalize the study of quotients and residuals with respect to an operation, which have been studied for particular operations by Câmpeanu *et al.* [21], Ito *et al.* [78, 79] and Kari and Thierrin [115]. We show that the smallest language containing $L$ and closed under shuffle along $T$ (or deletion along $T$) is the (positive) iteration closure of $L$ under $T$.

We also examine the concepts of *shuffle bases* and *extended shuffle bases*. These have been previously studied by Ito *et al.* [82], Ito *et al.* [78, 79], Ito and Silva [80] and Hsiao *et al.* [69]. These notions are related to the concept of $T$-codes introduced in Chapter 6.

Some of the work in this chapter has previously appeared in the more general setting of *word*

*operations*, as studied by Hsiao *et al.* [69]. However, we present the results below for several important reasons. First, the framework on shuffle on trajectories and deletion along trajectories yields closure properties which do not necessarily hold in the more general setting of word operations. Further, we have presented our results with slightly modified definitions which we feel are more natural. These modified definitions allow us to drop certain assumptions which were necessary in the setting of word operations, and also allow us to make interesting conclusions to the classes of $T$-codes which were not done in the more general setting.

## 8.2 Definitions

We first define the iterated shuffle operations relative to a given set $T$ of trajectories. Let $T \subseteq \{0,1\}^*$ be a set of trajectories. Then, for all languages $L \subseteq \Sigma^*$ and all $i \geq 0$, we define $(\sqcup\!\sqcup_T)^i(L)$ as follows:

$$\begin{aligned}
(\sqcup\!\sqcup_T)^0(L) &= \{\epsilon\} \\
(\sqcup\!\sqcup_T)^1(L) &= L \\
(\sqcup\!\sqcup_T)^{i+1}(L) &= \left((\sqcup\!\sqcup_T)^i(L) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^i(L)\right) \cup (\sqcup\!\sqcup_T)^i(L) \quad \forall i \geq 1. \qquad (8.1)
\end{aligned}$$

Note that we do not require that $T$ defines an associative operation. Further, we define $(\sqcup\!\sqcup_T)^*(L)$ and $(\sqcup\!\sqcup_T)^+(L)$ as

$$\begin{aligned}
(\sqcup\!\sqcup_T)^*(L) &= \bigcup_{i\geq 0} (\sqcup\!\sqcup_T)^i(L); \\
(\sqcup\!\sqcup_T)^+(L) &= \bigcup_{i\geq 1} (\sqcup\!\sqcup_T)^i(L).
\end{aligned}$$

Similarly, we define iterated deletion along a set $T$ of trajectories. Let $T \subseteq \{i,d\}^*$ be a set of trajectories. Then, for all $L \subseteq \Sigma^*$ and all $i \geq 0$, we define $(\rightsquigarrow_T)^i(L)$ as follows:

$$\begin{aligned}
(\rightsquigarrow_T)^0(L) &= \{\epsilon\}; \\
(\rightsquigarrow_T)^1(L) &= L; \\
(\rightsquigarrow_T)^{i+1}(L) &= \left((\rightsquigarrow_T)^i(L) \rightsquigarrow_T (\rightsquigarrow_T)^i(L)\right) \cup (\rightsquigarrow_T)^i(L) \quad \forall i \geq 1. \qquad (8.2)
\end{aligned}$$

We again do not require that $T$ defines an associative operation. Further, we define $(\rightsquigarrow_T)^*(L)$ and $(\rightsquigarrow_T)^+(L)$ as

$$\begin{aligned}(\rightsquigarrow_T)^*(L) &= \bigcup_{i\geq 0}(\rightsquigarrow_T)^i(L);\\ (\rightsquigarrow_T)^+(L) &= \bigcup_{i\geq 1}(\rightsquigarrow_T)^i(L).\end{aligned}$$

We also require an auxiliary operation $L_1[\rightsquigarrow_T]^i L_2$ which is defined recursively for all $i \geq 0$ as follows:

$$\begin{aligned}L_1[\rightsquigarrow_T]^0 L_2 &= L_1\\ L_1[\rightsquigarrow_T]^{i+1} L_2 &= (L_1[\rightsquigarrow_T]^i L_2) \rightsquigarrow_T L_2 \quad \forall i \geq 1.\end{aligned}$$

We then set

$$L_1[\rightsquigarrow_T]^* L_2 = \bigcup_{i\geq 0} L_1[\rightsquigarrow_T]^i L_2.$$

## 8.3 Iterated Shuffle on Trajectories

We begin our investigation with iterated shuffle on trajectories. We require some preliminary discussion regarding our definition and an alternate definition. Then we discuss some examples of iterated shuffle on trajectories before beginning our examination of the operation.

### 8.3.1 Left-Associativity and a Simplified Definition

Let $T \subseteq \{0, 1\}^*$. We say that $T$ is *left-associative* if, for all $\alpha, \beta, \gamma \in \Sigma^*$,

$$\alpha \sqcup\!\sqcup_T (\beta \sqcup\!\sqcup_T \gamma) \subseteq (\alpha \sqcup\!\sqcup_T \beta) \sqcup\!\sqcup_T \gamma.$$

Note that associativity implies left-associativity. Further, we can verify that $T = 0^*1^*0^*$ (insertion) is left-associative but not associative. Initial literal shuffle, given by $T = (01)^*(0^* + 1^*)$, is not left-associative. Left-associativity is also called left-inclusiveness [69]. Several of the results obtained

by Hsiao *et al.* [69] are similar to our results, but must include the condition of left-associativity, due to a slightly less general definition of iterated operations, given by the recurrence

$$\begin{aligned} (\sqcup\!\sqcup_T)^0_X(L) &= \{\epsilon\} \\ (\sqcup\!\sqcup_T)^1_X(L) &= L \\ (\sqcup\!\sqcup_T)^{i+1}_X(L) &= (\sqcup\!\sqcup_T)^i_X(L) \sqcup\!\sqcup_T L \quad \forall i \geq 1 \end{aligned} \tag{8.3}$$

instead of (8.1). Definitions (8.1 and (8.3) agree when the operation is left-associative. For example, our Theorem 8.6.5 in Section 8.6.2 below is given by Hsiao *et al.* [69] for left-associative word operations.

We now give a characterization of left-associativity. It is a weakening of the corresponding result of Mateescu *et al.* [147] on associativity. Let $D = \{x, y, z\}$. Then let $\tau, \sigma, \varphi, \psi : D^* \to \{0,1\}^*$ be the morphisms given by

$$\begin{array}{cccccccccccc} \sigma(x) & = & 0, & \tau(x) & = & 0, & \varphi(x) & = & 0, & \psi(x) & = & \epsilon, \\ \sigma(y) & = & 0, & \tau(y) & = & 1, & \varphi(y) & = & 1, & \psi(y) & = & 0, \\ \sigma(z) & = & 1, & \tau(z) & = & 1, & \varphi(z) & = & \epsilon, & \psi(z) & = & 1. \end{array}$$

Then the following result follows by the same proof as in Mateescu *et al.* [147, Prop. 4.7]:

**Theorem 8.3.1** *Let $T \subseteq \{0,1\}^*$. Then $T$ is left-associative if and only if*

$$\tau^{-1}(T) \cap \psi^{-1}(T) \subseteq \sigma^{-1}(T) \cap \varphi^{-1}(T).$$

*Thus, if $T$ is regular, it is decidable if $T$ is left-associative.*

Let $(\sqcup\!\sqcup_T)^*_X$, $(\sqcup\!\sqcup_T)^+_X$ be the iterated versions of $\sqcup\!\sqcup_T$ defined by (8.3) instead of (8.1), i.e.,

$$(\sqcup\!\sqcup_T)^*_X(L) = \bigcup_{i \geq 0} (\sqcup\!\sqcup_T)^i_X(L); \tag{8.4}$$

$$(\sqcup\!\sqcup_T)^+_X(L) = \bigcup_{i \geq 1} (\sqcup\!\sqcup_T)^i_X(L). \tag{8.5}$$

$$\tag{8.6}$$

Again, we note that it is not hard to establish that $(\sqcup\!\sqcup_T)^*_X(L) = (\sqcup\!\sqcup_T)^*(L)$ for all $L$ if $T$ is left-associative.

### 8.3.2 Some examples

We begin by noting the most well-studied iteration operation, that of Kleene closure. If $T = 0^*1^*$, then $\sqcup\!\sqcup_T$ defines the concatenation operation. Note that $T = 0^*1^*$ is associative. Thus

$$(\cdot)^*(L) = L^* = \{w_1 w_2 w_3 \cdots w_n \ : \ n \geq 0, w_i \in L\}.$$

We note that if $L$ is regular, then $L^*$ is regular.

If $T = (0 + 1)^*$, we get the operation of shuffle-closure, which has been well-studied in the literature (see Gischer [55], Jędrzejowicz [87, 88, 89, 91, 92], Kari and Thierrin [118], Ito *et al.* [79] as well as much work in software specification [6, 83, 182, 170]). Let us denote this case by $(\sqcup\!\sqcup)^*(L)$. We note that $(\sqcup\!\sqcup)^*(L)$ does not preserve regularity, even if $L$ is a singleton set, as

$$(\sqcup\!\sqcup)^*(\{ab\}) \cap a^*b^* = \{a^n b^n \ : \ n \geq 0\}.$$

We also note that the CFLs are not closed under $(\sqcup\!\sqcup)^*$, in fact, there exist a finite set $F$ such that $(\sqcup\!\sqcup)^*(F)$ is not a CFL. Let $L = \{abc, acb, bac, bca, cab, cba\}$. Then we can see that

$$(\sqcup\!\sqcup)^*(L) = \{w \in \{a, b, c\}^* \ : \ |w|_a = |w|_b = |w|_c\}.$$

Using a grammar-based argument, Gischer [55] proved that the closure of the regular languages under $(\sqcup\!\sqcup)^*$ is a proper subset of the CSLs. For an LBA-based proof of inclusion, see Jędrzejowicz [87]. We note that with a simple extension of the result of Jędrzejowicz, we can show that $(\sqcup\!\sqcup_T)^*(L) \in$ CS for all $T, L \in$ CS with $T$ left-associative.

If $T = 0^*1^*0^*$, we get the insertion operation, $\leftarrow$. Iterated insertion has been studied by Ito *et al.* [78], Kari and Thierrin [118] and Holzer and Lange [67]. Again, we note that $(\leftarrow)^*(\{ab\}) \cap a^*b^* = \{a^n b^n \ : \ n \geq 0\}$. Thus, iterated insertion of a singleton can result in non-regular sets.

If $T = 0^*1^* + 1^*0^*$, $\sqcup\!\sqcup_T$ is the bi-catenation operation (see Shyr and Yu [187] or Hsiao *et al.* [69]). Note that $L_1 \sqcup\!\sqcup_T L_2 = L_1L_2 + L_2L_1$. Thus,

$$(\sqcup\!\sqcup_T)^2(L) = L^2 + L^2 = L^2$$

and $(\sqcup\!\sqcup_T)^*(L) = L^*$. Thus, iterated bi-catenation preserves regularity.

### 8.3.3 Iteration and Density

We now turn to examining the relation of the density of a set of trajectories to the closure properties of its iteration operation. Recall that the density of a language was defined in Section 4.3. We begin by noting that preserving regularity is incomparable with density of $T$. We note that $T = 0^*1^*$ has density $O(n)$, and that the associated operation (Kleene closure) preserves regularity. However, there exist constant density sets of trajectories whose iteration closure does not preserve regularity, even when applied to finite sets:

**Lemma 8.3.2** *Let $T = 10^*1$. Then $p_T(n) = 1$ but $(\sqcup\!\sqcup_T)^*(\{ab\}) = \{a^n b^n \ : \ n \geq 0\}$.*

We now show that the set of trajectories in Lemma 8.3.2 can be considered the simplest constant-density regular set of trajectories which does not preserve regularity, when considering $(\sqcup\!\sqcup_T)^*_X$. In particular, we show that if $T$ has constant density, then $(\sqcup\!\sqcup_T)^*_X$ preserves finiteness unless $T \supseteq u(0^c)^*v$ for some $u, v \in \{0, 1\}^*$ and $c \geq 1$.

**Lemma 8.3.3** *Let $T = \cup_{i=1}^{k} u_i v_i^* w_i$ for $u_i, v_i, w_i \in \{0, 1\}^*$ and $|v_i|_1 > 0$ for all $1 \leq i \leq k$. Then for all finite languages $L$, $(\sqcup\!\sqcup_T)^*_X(L)$ is finite.*

**Proof.** For all $1 \leq i \leq k$, let $\beta_i = |u_i w_i|_1$ and $\eta_i = |v_i|_1 > 0$. Let $\beta = \max_{1 \leq i \leq k}\{\beta_i\}$ and $\eta = \max_{1 \leq i \leq k}\{\eta_i\}$.

Let $1 \leq i \leq k$. If, for all $x \in L$, there is no $\ell \geq 0$ such that $|x| = \beta_i + \ell\eta_i$, then for all $X \subseteq \Sigma^*$, $X \sqcup\!\sqcup_{u_i v_i^* w_i} L = \emptyset$. Thus, without loss of generality, we can assume that for each $1 \leq i \leq k$, there is some $x \in L$ and $\ell \geq 0$ such that $|x| = \beta_i + \ell\eta_i$, otherwise, we can replace $T$ with

$$T' = \bigcup_{\substack{1 \leq j \leq k \\ j \neq i}} u_j v_j^* w_j.$$

For all $1 \leq i \leq k$, let $\ell_i = \max\{\ell \geq 0 \ : \ \exists u \in L \text{ such that } |u| = \beta_i + \ell\eta_i\}$. Since $L$ is finite, $\ell_i$ exists. Let $\lambda = \max_{1 \leq i \leq k}\{\ell_i\}$.

Let $x \in (\sqcup\!\sqcup_T)^*_X(L)$. We show that the length of $x$ is bounded above by $\beta + \lambda\eta$. As $x \in (\sqcup\!\sqcup_T)^*_X(L)$, either $x = \epsilon$, or there is some $y \in (\sqcup\!\sqcup_T)^*_X(L)$ and $z \in L$ such that $x \in y \sqcup\!\sqcup_T z$. Let $1 \leq i \leq k$ and $s \geq 0$ be such that $x \in y \sqcup\!\sqcup_t z$ where $t = u_i v_i^s w_i$.

As $\eta_i \neq 0$, $|t|_1 = \beta_i + s\eta_i > \beta_i + \ell_i\eta_i$. As $|z| = |t|_1$, we have that $s \leq \ell_i$ by choice of $\ell_i$. Now $|x| = |t| = \beta_i + s\eta_i \leq \beta_i + \ell_i\eta_i \leq \beta + \lambda\eta$. ■

Consider the following particular case of a result due to Szilard *et al.* [190]:

**Lemma 8.3.4** *Let $L \subseteq \Sigma^*$ be a regular language such that $p_L(n) \in O(1)$. Then $L$ is a finite union of terms of the form $uv^*w$ for words $u, v, w \in \Sigma^*$.*

Then, the following corollary is immediate.

**Corollary 8.3.5** *Let $T \subseteq \{0,1\}^*$ be a regular set of trajectories such that $p_T(n) \in O(1)$, and the closure of the finite languages under $(\sqcup\!\sqcup_T)^*_X$ contains an infinite language. Then $T \supseteq u(0^c)^*v$ for some $u, v \in \{0,1\}^*$ and $c \geq 1$.*

We note, however, that if we drop the condition that we use $(\sqcup\!\sqcup_T)^*_X$ instead of $(\sqcup\!\sqcup_T)^*$, the result no longer holds. Consider $T = (01)^*$. Then we have that $(\sqcup\!\sqcup_T)^*(\{ab\}) \supseteq \{a^n b^n \ : \ n \geq 0\}$.

We now turn to regular sets of trajectories whose iteration closure contains non-CF languages. Note that if $T = 10^*110^*$, $(\sqcup\!\sqcup_T)^*_X(\{abc\}) \cap a^*b^*c^* = \{a^n b^n c^n \ : \ n \geq 0\}$. Thus, there is a linear density regular set of trajectories whose iteration closure of singletons contains non-CF languages. However, we also have the following example: for $T = (01)^*$, $(\sqcup\!\sqcup_T)^*(\{abc\}) = \{a^{2^n} b^{2^n} c^{2^n} \ : \ n \geq 0\}$. We summarize the minimal known density of regular languages and regular sets of trajectories witnessing non-closure properties for iterated shuffle on trajectories in Table 8.1.

## 8.4 Iterated Deletion

We now consider iterated deletion operations. We first note that the finite languages are closed under iterated deletion:

| | $(\sqcup\!\sqcup_T)^*$ | | $(\sqcup\!\sqcup_T)^*_X$ | |
|---|---|---|---|---|
| | $p_T(n)$ | $p_L(n)$ | $p_T(n)$ | $p_L(n)$ |
| non-regular | $O(1)$ | $O(1)$ | $O(1)$ | $O(1)$ |
| non-CF | $O(1)$ | $O(1)$ | $O(n)$ | $O(1)$ |

Figure 8.1: Summary of minimum-density regular languages and regular sets of trajectories demonstrating non-closure properties for iterated shuffle on trajectories.

**Lemma 8.4.1** *Let $L \subseteq \Sigma^{\leq m}$ for some $m \geq 0$. Then for all $T \subseteq \{i, d\}^*$, $(\rightsquigarrow_T)^*(L) \subseteq \Sigma^{\leq m}$.*

Second, we show that an alternate definition will suffice for some operations we will consider here. This alternate definition will somewhat simplify the results in this section. Call a set of trajectories $T \subseteq \{i, d\}^*$ *del-left-preserving* if $T \supseteq i^*$. Consider the following definitions:

$$\begin{aligned}
(\rightsquigarrow_T)^0_X(L) &= \{\epsilon\} \\
(\rightsquigarrow_T)^1_X(L) &= L \\
(\rightsquigarrow_T)^{i+1}_X(L) &= (\rightsquigarrow_T)^i_X(L) \rightsquigarrow_T ((\rightsquigarrow_T)^i_X(L) \cup \{\epsilon\}) \forall i \geq 1
\end{aligned} \tag{8.7}$$

We also define $(\rightsquigarrow_T)^*_X$ and $(\rightsquigarrow_T)^+_X$:

$$(\rightsquigarrow_T)^*_X(L) = \bigcup_{i \geq 0} (\rightsquigarrow_T)^i_X(L); \tag{8.8}$$

$$(\rightsquigarrow_T)^+_X(L) = \bigcup_{i \geq 1} (\rightsquigarrow_T)^i_X(L). \tag{8.9}$$

The following result motivates the above definitions:

**Theorem 8.4.2** *Let $T \subseteq \{i, d\}^*$ be a del-left-preserving set of trajectories. Then for all $L \subseteq \Sigma^*$, $(\rightsquigarrow_T)^*(L) = (\rightsquigarrow_T)^*_X(L)$.*

**Proof.** The result is immediate on noting the following two identities, which are obvious from the definition of $\rightsquigarrow_T$:

$$X \rightsquigarrow_T (Y + Z) = X \rightsquigarrow_T Y + X \rightsquigarrow_T Z; \tag{8.10}$$

$$X \rightsquigarrow_T \{\epsilon\} = X \rightsquigarrow_{T \cap i^*} \{\epsilon\}. \tag{8.11}$$

Further, it is clear that $X \rightsquigarrow_{i^*} \{\epsilon\} = X$. ■

### 8.4.1 Iterated Scattered Deletion

In this section, we consider a problem of Ito *et al.* [79] on iterated scattered deletion[1]. Recall that if $T = (0+1)^*$, we denote $\rightsquigarrow_T$ by $\rightsquigarrow$. Ito *et al.* [79] asked whether the regular languages are closed under $(\rightsquigarrow)^+$. We show that they are not.

Let $k \geq 2$ be arbitrary, and let $\Sigma_k = \{\alpha_i, \beta_i, \gamma_i, \eta_i\}_{i=1}^k$. Then we define $L_k \subseteq \Sigma_k^*$ as

$$L_k = \prod_{i=1}^{k} (\alpha_i \beta_i)^* \prod_{i=1}^{k} (\gamma_i \eta_i)^* + \bigcup_{i=1}^{k} \beta_i \eta_i.$$

We claim that

$$(\rightsquigarrow)^+(L_k) \cap \prod_{i=1}^{k} \alpha_i^+ \prod_{i=1}^{k} \gamma_i^+ = \{\alpha_1^{i_1} \alpha_2^{i_2} \cdots \alpha_k^{i_k} \gamma_1^{i_1} \gamma_2^{i_2} \cdots \gamma_k^{i_k} \ : \ i_j \geq 1\}. \tag{8.12}$$

and that $(\rightsquigarrow)^+(L_k)$ cannot be expressed as the intersection of $k-1$ context-free languages.

We first establish (8.12). Let $(i_1, i_2, \cdots, i_k) \in \mathbb{N}^k$. Then note that

$$\prod_{j=1}^{k} \alpha_j^{i_j} \prod_{j=1}^{k} \gamma_j^{i_j} \in (\cdots (\prod_{j=1}^{k} (\alpha_j \beta_j)^{i_j} \prod_{j=1}^{k} (\gamma_j \eta_j)^{i_j}) [\rightsquigarrow]^{i_1} \beta_1 \eta_1) \cdots) [\rightsquigarrow]^{i_k} \beta_k \eta_k.$$

Intuitively, we delete matching pairs of $\beta_j$ and $\eta_j$ from the word $\theta = \prod_{j=1}^k (\alpha_j \beta_j)^{i_j} \prod_{j=1}^k (\gamma_j \eta_j)^{i_j}$, and leave only occurrences of $\alpha_j$ and $\gamma_j$, of which we must necessarily have equal numbers, by choice of our word $\theta$. This establishes the right-to-left inclusion of (8.12). We now show the reverse inclusion. First, note that if $\theta \in (\rightsquigarrow)^+(L_k)$, then we can write $\theta = x_1 x_2 \cdots x_k y_1 y_2 \cdots y_k$ where $x_i \in \{\alpha_i, \beta_i\}^*$ and $y_i \in \{\gamma_i, \eta_i\}^*$. To prove the left-to-right inclusion of (8.12), we will establish the following stronger claim:

**Claim 8.4.3** *Let $x_1 x_2 \cdots x_k y_1 y_2 \cdots y_k \in (\rightsquigarrow)^+(L_k)$ where $x_i \in \{\alpha_i, \beta_i\}^*$ and $y_i \in \{\gamma_i, \eta_i\}^*$ for all $1 \leq i \leq k$. Then for all $1 \leq i \leq k$, the following equalities hold:*

$$|x_i|_{\alpha_i} - |x_i|_{\beta_i} = |y_i|_{\gamma_i} - |y_i|_{\eta_i}. \tag{8.13}$$

[1]Note: Since this research appeared in Bull. EATCS [36], I have been informed that this problem has been previously solved; see Ito and Silva [80], where the authors show that there exists a regular language $R$ such that $(\rightsquigarrow)^+(R)$ is not a CFL. We note that the results here were found independently, and extend those of Ito and Silva.

**Proof.** Let $z = x_1x_2\cdots x_ky_1y_2\cdots y_k \in (\rightsquigarrow)^+(L_k)$. Then there exists some $i \geq 1$ such that $z \in (\rightsquigarrow)^i(L_k)$. The proof is by induction on $i$. For $i = 1$, $z \in L_k$. Thus, we see that either

(a) for all $1 \leq \ell \leq k$, $x_\ell = (\alpha_\ell\beta_\ell)^{j_\ell}$ for some $j_\ell \geq 0$ and $y_\ell = (\gamma_\ell\eta_\ell)^{j'_\ell}$ for some $j'_\ell \geq 0$, in which case $|x_\ell|_{\alpha_\ell} - |x_\ell|_{\beta_\ell} = 0 = |y_\ell|_{\gamma_\ell} - |y_\ell|_{\eta_\ell}$; or

(b) $z = \beta_\ell\eta_\ell$ for some $1 \leq \ell \leq k$. Thus, $|x_\ell|_{\alpha_\ell} - |x_\ell|_{\beta_\ell} = -1 = |y_\ell|_{\gamma_\ell} - |y_\ell|_{\eta_\ell}$ and $x_j = y_j = \epsilon$ for all $1 \leq j \leq k$ with $j \neq \ell$.

Thus, the result holds for $i = 1$.

Assume the claim holds for all natural numbers less than $i$. Let $z \in (\rightsquigarrow)^i(L_k)$. Then there exists some $\theta \in (\rightsquigarrow)^{i-1}(L_k)$ and $\zeta \in (\rightsquigarrow)^{i-1}(L_k) \cup \{\epsilon\}$ such that $z \in \theta \rightsquigarrow \zeta$. If $\zeta = \epsilon$, then $z = \theta$ and the result holds by induction. Thus, let

$$\begin{aligned} \theta &= u_1u_2\cdots u_kv_1v_2\cdots v_k \\ \zeta &= s_1s_2\cdots s_kt_1t_2\cdots t_k \end{aligned}$$

with $u_\ell, s_\ell \in \{\alpha_\ell, \beta_\ell\}^*$ and $v_\ell, t_\ell \in \{\gamma_\ell, \eta_\ell\}^*$ for all $1 \leq \ell \leq k$. Then note that for all $1 \leq \ell \leq k$,

$$\begin{aligned} |x_\ell|_{\alpha_\ell} &= |u_\ell|_{\alpha_\ell} - |s_\ell|_{\alpha_\ell}; \\ |x_\ell|_{\beta_\ell} &= |u_\ell|_{\beta_\ell} - |s_\ell|_{\beta_\ell}; \\ |y_\ell|_{\gamma_\ell} &= |v_\ell|_{\gamma_\ell} - |t_\ell|_{\gamma_\ell}; \\ |y_\ell|_{\eta_\ell} &= |v_\ell|_{\eta_\ell} - |t_\ell|_{\eta_\ell}. \end{aligned}$$

Thus, by induction, we can easily establish that the desired equalities hold. ■

We now show that $(\rightsquigarrow)^+(L_k)$ cannot be expressed as the intersection of $k-1$ context-free languages. Let $\mathrm{CF}_k$ be the class of languages which are expressible as the intersection of $k$ CFLs. The following lemma is obvious, since the CFLs are closed under intersection with regular languages, (for further closure properties of $\mathrm{CF}_k$, see, e.g., Latta and Wall [129]).

**Lemma 8.4.4** *$\mathrm{CF}_k$ is closed under intersection with regular languages.*

We will also require the following lemma:

**Lemma 8.4.5** *Let $L_1, L_2 \in \mathrm{CF}_k$ be such that there exist disjoint regular languages $R_1, R_2$ such that $L_i \subseteq R_i$ for $i = 1, 2$. Then $L_1 \cup L_2 \in \mathrm{CF}_k$.*

**Proof.** Let $X_i, Y_i \in \mathrm{CF}$ for $1 \leq i \leq k$ be chosen so that $L_1 = \cap_{i=1}^{k} X_i$ and $L_2 = \cap_{i=1}^{k} Y_i$. Then without loss of generality, we may assume that $X_j \subseteq R_1$ and $Y_j \subseteq R_2$ for $1 \leq j \leq k$; if not, we may replace $X_i$ with $X_i \cap R_1$ and $Y_i$ with $Y_i \cap R_2$ as necessary. Both intersections are still CFLs.

Thus, note that $L_1 \cup L_2 = (X_1 \cup Y_1) \cap \cdots \cap (X_k \cup Y_k)$. As $X_i \cup Y_i \in \mathrm{CF}$, the result immediately follows. ■

The following result is due to Liu and Weiner [133, Thm. 8]:

**Theorem 8.4.6** *Let $k \geq 2$. Let $L_k'' = \{\alpha_1^{i_1}\alpha_2^{i_2}\cdots\alpha_k^{i_k}\alpha_1^{i_1}\alpha_2^{i_2}\cdots\alpha_k^{i_k} \ : \ i_j \geq 0\}$. Then $L_k'' \in \mathrm{CF}_k - \mathrm{CF}_{k-1}$.*

However, we prove the following corollary, which will be more useful to us:

**Corollary 8.4.7** *Let $L_k' = \{\alpha_1^{i_1}\alpha_2^{i_2}\cdots\alpha_k^{i_k}\alpha_1^{i_1}\alpha_2^{i_2}\cdots\alpha_k^{i_k} \ : \ i_j \geq 1\}$. Then $L_k' \in \mathrm{CF}_k - \mathrm{CF}_{k-1}$.*

**Proof.** The sufficiency of $k$ intersections is obvious, by Lemma 8.4.4. We prove only the necessity of $k$ intersections. The proof is by induction. For $k = 2$, the result can be established by the pumping lemma. Let $k > 2$ and $S \subset [k]$. Denote by $L_k^{(S)}$ the language

$$L_k^{(S)} = \{\prod_{j\in S}\alpha_j^{i_j}\prod_{j\in S}\alpha_j^{i_j} \ : \ i_j \geq 1\}.$$

Further, note that

$$L_k^{(S)} \subseteq (\prod_{j\in S}\alpha_j^{+})^2.$$

Let $R_S = (\prod_{j\in S}\alpha_j^{+})^2$. Then note that $R_S \cap R_{S'} = \emptyset$ for all $S, S' \subseteq [k]$ (including the possibility that $S = [k]$) with $S \neq S'$. If $S \subset [k]$, where the inclusion is proper, then $L_k^{(S)} \in \mathrm{CF}_{k-1}$.

Assume that $L_k'$ can be expressed as the intersection of $k-1$ CFLs. We then note that

$$L_k'' = L_k' \cup \bigcup_{S\subsetneq[k]} L_k^{(S)}.$$

By Lemma 8.4.5, $L_k'' \in \mathrm{CF}_{k-1}$, a contradiction. This completes the proof. ■

Thus, we may state our main result:

**Theorem 8.4.8** *For all $k \geq 2$, there exists an $O(n^{2k-1})$-density bounded regular language $L_k$ such that $(\rightsquigarrow)^+(L_k)$ cannot be expressed as the intersection of $k-1$ context-free languages.*

**Proof.** Let $\Delta_k = \{\gamma_i, \alpha_i\}_{i=1}^k$. Let $h_k : \Delta_k^* \to \Delta_k^*$ be given by $h_k(\alpha_i) = h_k(\gamma_i) = \alpha_i$ for all $1 \leq i \leq k$. Let $D_k = (\rightsquigarrow)^+(L_k)$ and $R_k = \prod_{i=1}^k \alpha_i^+ \prod_{i=1}^k \gamma_i^+$. If $D_k \in \text{CF}_{k-1}$, then $D_k \cap R_k \in \text{CF}_{k-1}$ as well, by Lemma 8.4.4. We claim that this implies that $h_k(D_k \cap R_k)$ is in $\text{CF}_{k-1}$.

Let $X_1, X_2, \ldots, X_{k-1} \in \text{CF}$ be chosen so that

$$D_k \cap R_k = \cap_{i=1}^{k-1} X_i.$$

The inclusion $h_k(D_k \cap R_k) \subseteq \cap_{i=1}^{k-1} h_k(X_i)$ is easily verified. We now show the reverse inclusion. First, we may assume without loss of generality that $X_j \subseteq R_k$ for all $1 \leq j \leq k-1$. If not, let $X_j' = X_j \cap R_k$. By the closure properties of the CFLs, $X_j' \in \text{CF}$, and we still have $D_k \cap R_k = \cap_{i=1}^{k-1} X_i'$.

Let $x \in \cap_{i=1}^{k-1} h_k(X_i)$. Let $y_i \in X_i$ be such that $h_k(y_i) = x$ for $1 \leq i \leq k-1$. By assumption, we can write

$$y_j = \prod_{i=1}^{k} \alpha_i^{\ell_i^{(j)}} \prod_{i=1}^{k} \gamma_i^{m_i^{(j)}}$$

for some $\ell_i^{(j)}, m_i^{(j)} \geq 1$ for $1 \leq i \leq k$ and $1 \leq j \leq k-1$. Thus, by definition of $h_k$,

$$h_k(y_j) = \prod_{i=1}^{k} \alpha_i^{\ell_i^{(j)}} \prod_{i=1}^{k} \alpha_i^{m_i^{(j)}},$$

for all $1 \leq j \leq k-1$. As $h_k(y_j) = x$, for all $1 \leq j \leq k-1$, we must have that $\ell_i^{(j)} = \ell_i^{(j')}$ and $m_i^{(j)} = m_i^{(j')}$ for $1 \leq i \leq k$ and all $1 \leq j, j' \leq k-1$. Thus $y_1 = \cdots = y_{k-1} \in \cap_{i=1}^{k-1} X_i$ and $x \in h_k(\cap_{i=1}^{k-1} X_i) = h_k(D_k \cap R_k)$. Therefore, $h_k(D_k \cap R_k) = \cap_{i=1}^{k-1} h_k(X_i)$. As $h_k(X_i)$ is a CFL for all $1 \leq i \leq k-1$, $h_k(D_k \cap R_k) \in \text{CF}_{k-1}$. But now note that

$$h_k(D_k \cap R_k) = L_k',$$

by (8.12). This contradicts Corollary 8.4.7. Thus, $D_k$ cannot be expressed as the intersection of $k-1$ CFLs. ■

We now consider another example of representing non-regular languages by the iterated scattered deletion of a regular language. Let $\tilde{\Sigma}$ be a copy of $\Sigma$. Recall that $com(u)$ is the set of all words which can be obtained by permuting the letters of $u$, i.e.,

$$com(u) = \{v \in \Sigma^* \ : \ \forall a \in \Sigma, |u|_a = |v|_a\}.$$

**Theorem 8.4.9** *Let $L = \{u\tilde{v} \ : \ u \in \Sigma^*, v \in com(u)\}$. Then there exist regular languages $R_1, R_2$ such that $(\rightsquigarrow)^+(R_1) \cap R_2 = L$.*

**Proof.** Let $\hat{\Sigma}$, $\check{\Sigma}$ be two additional copies of $\Sigma$. Let $R_1 = (\bigcup_{a\in\Sigma}(a\hat{a}))^*(\bigcup_{a\in\Sigma}\tilde{a}\check{a})^* + \bigcup_{a\in\Sigma}\hat{a}\check{a}$. Let $R_2 = \Sigma^*\tilde{\Sigma}^*$.

We can establish, in the same manner as Theorem 8.4.8, that $(\rightsquigarrow)^+(R_1) \cap R_2 = L$. Again, the key step is to show that for all $x_1x_2 \in (\rightsquigarrow)^+(R_1)$, where $x_1 \in (\Sigma + \hat{\Sigma})^*$ and $x_2 \in (\tilde{\Sigma} + \check{\Sigma})^*$, the following equality holds for all $a \in \Sigma$:

$$|x_1|_a - |x_1|_{\hat{a}} = |x_2|_{\tilde{a}} - |x_2|_{\check{a}}.$$

This is easily established by induction. ■

Note that $L \in \text{CF}_{|\Sigma|}$. To see this, consider the language

$$L_a = \{w\tilde{u} \ : \ w, u \in \Sigma^*, |w|_a = |u|_a\}$$

for all $a \in \Sigma$. Then $L_a \in \text{CF}$, and $L = \cap_{a\in\Sigma} L_a$. The fact that $L \in \text{CF}_{|\Sigma|}$ is in contrast to the fact that the language $\{w\tilde{w} \ : \ w \in \Sigma^*\}$ is not in $\text{CF}_k$ for any $k \geq 1$, which was established by Wotschke [200]. Thus, there is a significant difference, in terms of descriptional complexity, between the language of (marked) squares and the language $L$ of (marked) "Abelian squares". We refer the reader to Jędrezejowicz and Szepietowski [93] for a discussion of $L$ versus $\{ww \ : \ w \in \Sigma^*\}$ as it relates to mildly context-sensitive families of languages and iterated shuffle.

We have the following open problem:

**Open Problem 8.4.10** *For all regular languages $R$, does there exist a $k \geq 1$ such that $(\rightsquigarrow)^+(R) \in \text{CF}_k$?*

### 8.4.2 Density and Iterated Deletion

From Theorem 8.4.8, we note that the $O(n^2)$-density set $T = i^*di^*di^*$ of trajectories yields an operation $(\rightsquigarrow_T)^*$ which does not preserve regularity. Indeed, if $L = (ba)^*(ce)^* + ac$, then

$$(\rightsquigarrow_T)^*(L) \cap b^*e^* = \{b^n e^n \ : \ n \geq 0\}.$$

Is is open whether there is a set $T$ of trajectories with $p_T(n) \in o(n^2)$ which does not preserve regularity.

We note by Theorem 8.4.8, there is an $O(n)$-density regular language $L$ such that $(\rightsquigarrow)^*(L)$ is not regular. Thus, we can ask the following open question:

**Open Problem 8.4.11** *Given a regular language $L$ such that $p_L(n) \in O(1)$, is $(\rightsquigarrow)^*(L)$ regular?*

We note that if $T = i^*di^*di^*di^*$ then using $L = (a_1a_2)^*(b_1b_2)^*(c_1c_2)^* + a_1b_1c_1$, we see that $(\rightsquigarrow_T)^*(L)$ is not a CFL. Again, we do not know if there exists a regular set $T \subseteq \{i, d\}^*$ with $p_T(n) \in o(n^3)$ such that the closure of the regular languages under $(\rightsquigarrow_T)^*$ contains non-CF languages. We summarize the best-known minimal densities in Table 8.4.2.

| | $p_T(n)$ | $p_L(n)$ |
|---|---|---|
| non-regular | $O(n^2)$ | $O(n)$ |
| non-CF | $O(n^3)$ | $O(n^2)$ |

Figure 8.2: Summary of minimum-density regular languages and regular sets of trajectories demonstrating non-closure properties for iterated deletion along trajectories.

## 8.5 Additional Closure Properties

We now consider some additional closure properties. We are motivated by an open problem of Ito and Silva [80] on the closure properties of the CSLs under iterated scattered deletion. We will require the following theorem, which can be found, in a slightly less general version, in, e.g., Salomaa [174, Thm. 9.9]:

**Theorem 8.5.1** *Let $\Sigma$ be an alphabet, and $a \notin \Sigma$. Let $s \in \Omega(\log(n))$ be any space constructible function. Then for all $L \in$ RE over $\Sigma$, there exists $L' \in$ NSPACE$(s)$ such that $L' \subseteq a^*L$ and for all $x \in L$, there exists $i \geq 0$ such that $a^i x \in L'$.*

For all $T \subseteq \{i, d\}^*$, let $\text{suff}(T) = \{t \in \{i, d\}^* \ : \ \exists t' \in \{i, d\}^* \text{ such that } t't \in T\}$. Our stated closure property is given below:

**Theorem 8.5.2** *Let $s \in \Omega(\log(n))$ be a space-constructible function. Let $d^*i^* \subseteq T$. If there exists $L$ such that $(\rightsquigarrow_{suff(T)})^+(L) \in$ RE $-$ NSPACE$(s)$, then NSPACE$(s)$ is not closed under $(\rightsquigarrow_T)^+$.*

**Proof.** Let $L \subseteq \Sigma^*$ be a language such that $(\rightsquigarrow_{\text{suff}(T)})^+(L) \in$ RE $-$ NSPACE$(s)$. Let $a \notin \Sigma$ and $L_0 \subseteq a^* \left((\rightsquigarrow_{\text{suff}(T)})^+(L)\right)$ be the language in NSPACE$(s)$ described by Theorem 8.5.1. Let $L_1 = L_0 + a^*$. Clearly, $L_1 \in$ NSPACE$(s)$. We claim that $(\rightsquigarrow_T)^+(L_1) \cap \Sigma^+ = (\rightsquigarrow_{\text{suff}(T)})^+(L)$.

($\supseteq$): Let $x \in (\rightsquigarrow_{\text{suff}(T)})^+(L)$. Then there exists $j \geq 0$ such that $a^j x \in L_1$. As $a^j \in L_1$ and $T \supseteq d^*i^* \ni d^j i^{|x|}$, $x \in a^j x \rightsquigarrow_T a^j \subseteq (\rightsquigarrow_T)^+(L_1)$.

($\subseteq$): To prove this inclusion, we prove the stronger claim that

$$(\rightsquigarrow_T)^+(L_1) \subseteq a^*(\rightsquigarrow_{\text{suff}(T)})^*(L).$$

Let $x \in (\rightsquigarrow_T)^+(L_1)$. Then there exists $j \geq 1$ such that $x \in (\rightsquigarrow_T)^j(L_1)$. The proof of our claim is by induction on $j$.

For $j = 1$, $x \in L_1 \subseteq a^*(\rightsquigarrow_{\text{suff}(T)})^+(L) + a^*$. Thus, the result clearly holds. Let $j > 1$ and assume the result holds for all natural numbers less than $j$. As $x \in (\rightsquigarrow_T)^j(L_1)$, then either $x \in (\rightsquigarrow_T)^{j-1}(L_1)$, whereby the result clearly holds by induction, or there exist $x_1, x_2 \in (\rightsquigarrow_T)^{j-1}(L_1)$ and $t \in T$ such that $x \in x_1 \rightsquigarrow_t x_2$. By induction, $x_i = a^{k_i} u_i$ for $k_i \geq 0$, and $u_i \in (\rightsquigarrow_{\text{suff}(T)})^*(L)$, for $i = 1, 2$.

There are three cases:

(a) $u_1, u_2 \in (\rightsquigarrow_{\text{suff}(T)})^+(L)$. Then $x \in x_1 \rightsquigarrow_t x_2$ implies that $t = t_1 t_2$ where $t_1$ satisfies $|t_1| = k_1$ and $|t_1|_d = k_2$. Thus, $x \in a^{k_1 - k_2}(u_1 \rightsquigarrow_{t_2} u_2)$. Let $u \in u_1 \rightsquigarrow_{t_2} u_2$ be such that $x = a^{k_1 - k_2} u$. Then note that $t_2 \in \text{suff}(T)$ and thus $u \in u_1 \rightsquigarrow_{t_2} u_2 \subseteq (\rightsquigarrow_{\text{suff}(T)})^+(L)$. Thus, $x \in a^*(\rightsquigarrow_{\text{suff}(T)})^*(L)$.

(b) $u_2 = \epsilon$. Then $x_2 = a^{k_2}$ and $x_1 = a^{k_1}u_1$ where $u_1 \in (\rightsquigarrow_{\text{suff}(T)})^*(L_1)$. Then note that necessarily, $x = a^{k_2-k_1}u_1$. Thus, $x \in a^*(\rightsquigarrow_{\text{suff}(T)})^*(L_1)$.

(c) $u_1 = \epsilon$ and $u_2 \in (\rightsquigarrow_T)^+(L)$ such that $u_2 \neq \epsilon$. Then $x_1 \rightsquigarrow_t x_2 = \emptyset$.

Thus, we have established that

$$(\rightsquigarrow_T)^+(L_1) \cap \Sigma^+ = (\rightsquigarrow_{\text{suff}(T)})^+(L). \tag{8.14}$$

Assume, contrary to what we want to prove, that NSPACE$(s)$ is closed under $(\rightsquigarrow_T)^+$. As $L_1 \in$ NSPACE$(s)$, $(\rightsquigarrow_{\text{suff}(T)})^+(L) \in$ NSPACE$(s)$ by (8.14). This contradicts our choice of $L$. Thus, we have established the result. ■

For scattered deletion, $T = (i+d)^*$, and thus $T = \text{suff}(T)$. Thus, if CS $=$ NSPACE$(n)$ is closed under $(\rightsquigarrow)^+$, then for all $L \in$ RE $-$ CS, $(\rightsquigarrow)^+(L) \in$ CS. The closure of CS under $(\rightsquigarrow_T)^+$ is an open problem posed by Ito and Silva [80].

## 8.6 $T$-Closure of a Language

We will now investigate the natural problem of, given $L$, finding the smallest language which is closed under $\sqcup\!\sqcup_T$ and contains $L$. Classically, it is known that the smallest language containing $L$ which is closed under concatenation is $L^+$. This question has also been examined by Ito *et al.* [78, 79] and Kari and Thierrin [115] for other operations modeled by shuffle on trajectories. We will require some notions about quotients and residuals, which we discuss first.

### 8.6.1 Shuffle-$T$ Quotient

Let $T \subseteq \{0,1\}^*$. In this section, we describe the shuffle-$T$ quotient of a language $L$ with respect to a language $L_1$, and show that if $L$, $L_1$ and $T$ are regular, the shuffle-$T$ quotient of $L_1$ with respect to $L$ is again regular.

Let $\Sigma$ be an alphabet and $L \subseteq \Sigma^*$. Then the *shuffle-$T$ quotient of $L$ with respect to $L_1$*, denoted

$sq_T(L; L_1)$, is given by

$$sq_T(L; L_1) = \{x \in \Sigma^* \ : \ \forall y \in L_1, y \sqcup\!\sqcup_T x \subseteq L\}.$$

For arbitrary shuffle $T = (0+1)^*$, the shuffle quotient has been examined by Câmpeanu *et al.* [21]. Ito *et al.* have examined (arbitrary) shuffle residual [79], which is given by $sq_T(L; L)$ for $T = (0+1)^*$ and insertion residual [78], which is given by $sq_T(L; L)$ for $T = 0^*1^*0^*$. Kari and Thierrin [115] have studied $k$-insertion residuals, given by $sq_T(L; L)$ for $T = 0^*1^*0^{\leq k}$ for arbitrary $k \geq 0$. Hsiao *et al.* [69] consider right residuals for more general word operations. Our main result of this section is the following:

**Theorem 8.6.1** *For all $L \subseteq \Sigma^*$, $sq_T(L; L_1) = \overline{(\overline{L} \rightsquigarrow_{\pi(T)} L_1)}$. Thus, if $L, L_1, T$ are regular, so is $sq_T(L; L_1)$, and it can be effectively constructed.*

**Proof.** Let $x \in sq_T(L; L_1)$. Assume, contrary to what we want to prove, that $x \in \overline{L} \rightsquigarrow_{\pi(T)} L_1$. Thus, there exist $t \in T$, $y \in \overline{L}$ and $z \in L_1$ such that $x \in y \rightsquigarrow_{\pi(t)} z$. By Theorem 5.8.2, $y \in z \sqcup\!\sqcup_t x$. As $x \in sq_T(L; L_1)$ and $z \in L_1$, $z \sqcup\!\sqcup_t x \subseteq L$. However, $y \in \overline{L}$, a contradiction.

Let $x \notin sq_T(L; L_1)$. Thus, there exists some $u \in L_1$ such that $u \sqcup\!\sqcup_T x \cap \overline{L} \neq \emptyset$. Let $y$ be some word in this intersection, and let $t \in T$ be such that $y \in u \sqcup\!\sqcup_t x$. Thus by Theorem 5.8.2, $x \in y \rightsquigarrow_{\pi(t)} u \subseteq \overline{L} \rightsquigarrow_{\pi(T)} L_1$. Thus, we conclude that $\overline{sq_T(L; L_1)} \subseteq \overline{L} \rightsquigarrow_{\pi(T)} L_1$.

The fact that the regular languages are effectively closed under deletion along regular trajectories implies that $sq_T(L; L_1)$ is a regular language. This completes the proof. ■

Note that Theorem 8.6.1 gives an alternate proof that if $L_1, L_2$ are regular, then the (arbitrary) shuffle quotient of $L_1$ and $L_2$ is regular (this was originally proven by Câmpeanu *et al.* [21, Lemma 4]). Further, Theorem 8.6.1 was proven for $L = L_1$ and $T = (0+1)^*$ by Ito *et al.* [79, Prop. 2.4], for $L = L_1$ and $T = 0^*1^*0^*$ by Ito *et al.* [78, Prop. 2.3], and for $L = L_1$ and $T = 0^*1^*0^{\leq k}$ for fixed $k \geq 0$ by Kari and Thierrin [115, Prop. 2.3]. An equivalent formulation of Theorem 8.6.1 was given by Hsiao *et al.* [69, Prop. 30], but without explicitly using the notion of inverse operations. Further, in their framework of word operations, we cannot conclude any closure properties.

### 8.6.2 $T$-closure

Let $r_T(L) = sq_T(L; L)$, which we call the *shuffle-$T$ residual* of $L$. A language $L \subseteq \Sigma^*$ such that $L \subseteq r_T(L)$ is said to be *shuffle-$T$ closed*. Define

$$C_T(L) = \{L' \subseteq \Sigma^* \ : \ L \subseteq L' \subseteq r_T(L')\}.$$

Then $C_T(L)$ is the set of all shuffle-$T$ closed languages containing $L$ ($C_T(L) \neq \emptyset$ as $\Sigma^* \in C_T(L)$). Further, define

$$cl_T(L) = \bigcap_{L' \in C_T(L)} L'.$$

Then $cl_T(L)$ is the smallest $T$-closed language containing $L$; we call $cl_T(L)$ the *shuffle-$T$ closure of $L$*.

**Proposition 8.6.2** *Let $T \subseteq \{0, 1\}^*$. Then $L$ is shuffle-$T$ closed if and only if $L \sqcup_T L \subseteq L$.*

**Proof.** Let $L$ be shuffle-$T$ closed. Then for all $x \in L$, we have $x \in r_T(L)$. Thus, for all $u \in L$, $u \sqcup_T x \subseteq L$. Clearly then, $L \sqcup_T L \subseteq L$.

For the reverse implication, let $L \sqcup_T L \subseteq L$. Then let $x \in L$; we show $x \in r_T(L)$. As $L \sqcup_T L \subseteq L$, for all $y \in L$, $y \sqcup_T x \subseteq L$. Thus, by definition, $x \in r_T(L)$. ■

Proposition 8.6.2 was noted for scattered deletion by Ito *et al.* [79], for sequential deletion by Ito *et al.* [78] and for $k$-deletion by Kari and Thierrin [115]. For catenation, a weakened version of the only-if portion of the result is sometimes given as an easy undergraduate exercise (see, e.g., Martin [141, Ex. 2.22]). In the framework of general word operations, a variant of Proposition 8.6.2 is given by Hsiao *et al.* [69, Prop. 24].

**Corollary 8.6.3** *Let $T \subseteq \{0, 1\}^*$ be a regular set of trajectories. Given a regular language $L$, it is decidable whether $L$ is shuffle-$T$ closed.*

We now seek to give a characterization of $cl_T(L)$ for all $T \subseteq \{0, 1\}^*$. The following fact is obvious from the definition of $(\sqcup_T)^i$:

**Fact 8.6.4** *For all* $L \subseteq \Sigma^*$ *and all* $i \geq 1$, $(\sqcup\!\sqcup_T)^i(L) \subseteq (\sqcup\!\sqcup_T)^{i+1}(L)$.

**Theorem 8.6.5** *Let* $T \subseteq \{0,1\}^*$. *Then* $cl_T(L) = (\sqcup\!\sqcup_T)^+(L)$.

**Proof.** Note that $L = (\sqcup\!\sqcup_T)^1(L) \subseteq (\sqcup\!\sqcup_T)^+(L)$. To show that $cl_T(L) \subseteq (\sqcup\!\sqcup_T)^+(L)$, it suffices to show that $(\sqcup\!\sqcup_T)^+(L)$ is shuffle-$T$ closed. We appeal to Proposition 8.6.2. In particular, we show that $(\sqcup\!\sqcup_T)^+(L) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^+(L) \subseteq (\sqcup\!\sqcup_T)^+(L)$. Let $x, y \in (\sqcup\!\sqcup_T)^+(L)$. Then there exist $j, k \geq 1$ such that $x \in (\sqcup\!\sqcup_T)^j(L)$ and $y \in (\sqcup\!\sqcup_T)^k(L)$. Let $m = \max(j,k)$. Then clearly $x, y \in (\sqcup\!\sqcup_T)^m(L)$. Thus, by definition of $(\sqcup\!\sqcup_T)^m(L)$,

$$x \sqcup\!\sqcup_T y \subseteq (\sqcup\!\sqcup_T)^{m+1}(L) \subseteq (\sqcup\!\sqcup_T)^+(L).$$

The inclusion is proven.

We now show that $(\sqcup\!\sqcup_T)^+(L) \subseteq cl_T(L)$. Again, the proof is by induction on $i$: we show $(\sqcup\!\sqcup_T)^i(L) \subseteq cl_T(L)$ for all $i \geq 1$.

For $i = 1$, $L \subseteq cl_T(L)$ by definition of $cl_T(L)$. Now let $i > 1$ and assume the result holds for all integers less than $i$. Consider

$$\begin{aligned}
(\sqcup\!\sqcup_T)^i(L) &= \left((\sqcup\!\sqcup_T)^{i-1}(L) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^{i-1}(L)\right) + (\sqcup\!\sqcup_T)^{i-1}(L) \\
&\subseteq (cl_T(L) \sqcup\!\sqcup_T cl_T(L)) + cl_T(L) \\
&\subseteq cl_T(L) + cl_T(L) = cl_T(L)
\end{aligned}$$

where the first inclusion is by induction on $i$, and the second inclusion is by the fact that $cl_T(L)$ is $T$-closed (by definition of $cl_T(L)$), and Proposition 8.6.2. ■

Theorem 8.6.5 was also proven for sequential insertion by Ito *et al.* [78, Prop. 2.4], and for arbitrary shuffle by Ito *et al.* [79, Prop. 2.6].

Recall that $(\sqcup\!\sqcup_T)^*_X$ is the iterated version of $\sqcup\!\sqcup_T$ defined by (8.4). As we have stated, it is not hard to establish that $(\sqcup\!\sqcup_T)^*_X(L) = (\sqcup\!\sqcup_T)^*(L)$ for all $L$ if $T$ is left-associative. Thus, we can conclude that if $T$ is left-associative, $cl_T(L) = (\sqcup\!\sqcup_T)^+_X(L)$ for all $L$. We now show that the requirement that $T$ be left-associative is necessary for $cl_T(L) = (\sqcup\!\sqcup_T)^+_X(L)$.

**Lemma 8.6.6** *There exist a singleton language $L$ and set $T$ of non-left-associative trajectories such that $cl_T(L) \neq (\sqcup\!\sqcup_T)^+_X(L)$.*

**Proof.** We show, in fact, that infinitely many pairs $(L, T)$ exist satisfying the lemma. Let $k \geq 1$, and $T_k = 0^*1^*0^{\leq k}$. Then $\sqcup\!\sqcup_{T_k} = \overset{k}{\leftarrow}$, the $k$-insertion operation, studied by Kari and Thierrin [115]. For each $k \geq 1$, we define $L_k = \{ba^k\}$. Then we claim that

$$cl_{T_k}(L_k) \neq (\overset{k}{\leftarrow})^+_X(L_k). \tag{8.15}$$

We establish first that

$$\{b^i a^{ki} : i \geq 1\} \subseteq cl_{T_k}(L_k).$$

As $L_k \subseteq cl_{T_k}(L_k)$, $ba^k \in cl_{T_k}(L_k)$. For each $i > 1$, $ba^k, b^i a^{ki} \in cl_{T_k}(L_k)$ imply that $b^{i+1}a^{k(i+1)} \in ba^k \overset{k}{\leftarrow} b^i a^{ki} \subseteq cl_{T_k}(L_k)$.

Now, we note that $b^3(a+b)^* \cap (\overset{k}{\leftarrow})^+_X(L_k) = \emptyset$. To see this, note that if $x \in (\overset{k}{\leftarrow})^i_X(L_k)$, then $|x| = ki$ and $|x|_b = i$. We can then prove, by induction, that at most 2 occurrences of $b$ can occur at the start of any word in $(\overset{k}{\leftarrow})^i_X(L_k)$, since $k$-insertions always happen "close to the right end" of the word. Thus, we can establish (8.15). ■

Note that Lemma 8.6.6 corrects an error in Kari and Thierrin [115, Prop. 2.4], where it is claimed that $cl_{T_k}(L) = (\sqcup\!\sqcup_{T_k})^+_X(L)$ for each $T_k = 0^*1^*0^{\leq k}$.

### 8.6.3 Codes and Shuffle-Closed Languages

We now show that $T$-codes are shuffle-$T$ closed if and only if they are trivially shuffle-$T$ closed.

**Lemma 8.6.7** *Let $T \subseteq \{0,1\}^*$. Let $L \in \mathcal{P}_T(\Sigma)$. Then $L$ is shuffle-$T$ closed if and only if $L \sqcup\!\sqcup_T L = \emptyset$.*

**Proof.** If $L \sqcup\!\sqcup_T L = \emptyset$, then clearly $L \sqcup\!\sqcup_T L \subseteq L$, and $L$ is shuffle-$T$ closed.

Let $L \in \mathcal{P}_T(\Sigma)$. Then note that necessarily $L \subseteq \Sigma^+$. Let $L$ be shuffle-$T$ closed. Assume there exist $x, y \in L$ such that $x \sqcup\!\sqcup_T y \neq \emptyset$. Let $z \in x \sqcup\!\sqcup_T y$. Thus, $z \in L$ as $L$ is shuffle-$T$ closed. Therefore, $z \in L \cap (L \sqcup\!\sqcup_T \Sigma^+)$, contradicting that $L$ is a $T$-code. ■

**Corollary 8.6.8** *Let $T \subseteq \{0,1\}^*$ be complete. Let $L \in \mathcal{P}_T(\Sigma)$. Then $L$ is shuffle-$T$ closed if and only if $L = \emptyset$.*

## 8.7 Deletion Closure of a Language

We now consider the problem of, given a language $L$, finding the smallest language which contains $L$ and is closed under $\rightsquigarrow_T$.

### 8.7.1 Del-$T$ Quotient

Let $T \subseteq \{i,d\}^*$. In this section, we describe the del-$T$ quotient of a language with respect to a language $L$, and show that if $L, L_1, T$ are regular, the del-$T$ quotient of $L_1$ with respect to $L$ is again regular.

We define the set of *$T$-scattered subwords* as follows:

$$scs_T(L) = L \rightsquigarrow_{sym_d(T)} \Sigma^*.$$

Note that

$$scs_T(L) = \{u \in \Sigma^* \ : \ \exists v \in \Sigma^* \text{ such that } u \sqcup\!\sqcup_{\pi^{-1}(T)} v \cap L \neq \emptyset\}.$$

Further, note that if $L, T$ are regular, then $scs_T(L)$ is regular. As examples, note that

(a) when $T = i^*d^*i^*$, we have $scs_T(L) = sub(L)$, the subwords of $L$ (e.g., Ito *et al.* [78]), given by

$$sub(L) = \{u \in \Sigma^* \ : \ \exists x, y \in \Sigma^* \text{ such that } xuy \in L\}.$$

(b) if $T = (i+d)^*$, we have that $scs_T(L) = sps(L)$, the scattered (or sparse) subwords of $L$ (e.g., Ito *et al.* [79]), given by

$$sps(L) = \{u \in \Sigma^* \ : \ \exists v \in \Sigma^* \text{ such that } u \sqcup\!\sqcup v \cap L \neq \emptyset\}.$$

Let $\Sigma$ be an alphabet and $L \subseteq \Sigma^*$. Then the *deletion-$T$ quotient of $L$ with respect to $L_1$*, denoted $dq_T(L; L_1)$, is given by

$$dq_T(L; L_1) = \{x \in scs_T(L) \ : \ \forall y \in L_1, y \rightsquigarrow_T x \subseteq L\}.$$

Ito *et al.* have examined scattered deletion residual [79], which is given by $dq_T(L; L)$ for $T = (i + d)^*$ and deletion residual [78], which is given by $dq_T(L; L)$ for $T = i^*d^*i^*$. For $k \geq 1$, Kari and Thierrin [115] have studied $k$-deletion residual, which is given by $dq_T(L; L)$ for $T = i^*d^*i^{\leq k}$. Note that we could also define

$$dq'_T(L; L_1) = \{x \in \Sigma^* \ : \ \forall y \in L_1, y \rightsquigarrow_T x \subseteq L\}.$$

In this case, we get

$$dq'_T(L; L_1) = dq_T(L; L_1) \cup \overline{scs_T(L)}.$$

Our main result of the section is the following:

**Theorem 8.7.1** *For all $L \subseteq \Sigma^*$, $dq_T(L; L_1) = \overline{(L_1 \rightsquigarrow_{sym_d(T)} \overline{L})} \cap scs_T(L)$. Thus, if $L, L_1, T$ are regular, so is $dq_T(L; L_1)$, and it can be effectively constructed.*

**Proof.** Let $x \in dq_T(L; L_1)$. Immediately, we have that $x \in scs_T(L)$. Assume, contrary to what we want to prove, that $x \in L_1 \rightsquigarrow_{sym_d(T)} \overline{L}$. Thus, there exist $t \in T$, $y \in \overline{L}$ and $z \in L_1$ such that $x \in z \rightsquigarrow_{sym_d(t)} y$. By Theorem 5.8.3, $y \in z \rightsquigarrow_t x$. As $x \in dq_T(L; L_1)$ and $z \in L_1$, $z \rightsquigarrow_t x \subseteq L$. However, $y \in \overline{L}$, a contradiction.

Let $x \in \overline{L_1 \rightsquigarrow_{sym_d(T)} \overline{L}} \cap scs_T(L)$. Assume, contrary to what we want to prove, that $x \notin dq_T(L; L_1)$. As $x \in scs_T(L)$, this implies that there exists a word $y \in L_1$ such that $y \rightsquigarrow_T x \cap \overline{L} \neq \emptyset$. Let $u$ be some word in this intersection. Thus, there is some $t \in T$ such that $u \in y \rightsquigarrow_t x$. By Theorem 5.8.3, $x \in y \rightsquigarrow_{sym_d(t)} u \subseteq L_1 \rightsquigarrow_{sym_d(T)} \overline{L}$. This contradicts our choice of $x$. ■

Theorem 8.7.1 was proven by Ito *et al.* for the cases where $L = L_1$ and $T = (i + d)^*$ [79, Prop. 4.2] as well as $L = L_1$ and $T = i^*d^*i^*$ [78, Prop. 3.2]. Theorem 8.7.1 was proven by Kari and Thierrin [115] for the case of $L = L_1$ and $T = i^*d^*i^{\leq k}$ for fixed $k \geq 1$.

### 8.7.2 $T$-del-closure

Let $T \subseteq \{i, d\}^*$ Let $dr_T(L) = dq_T(L; L)$, which we call the *del-$T$ residual* of $L$. A language $L$ such that $L \cap scs_T(L) \subseteq dr_T(L)$ is said to be *del-$T$ closed*.

We first note a class of trajectories for which the annoyance of dealing with $scs_T(L)$ is removed. We call a set $T \subseteq \{i, d\}^*$ *del-left-preserving* with respect to $L$ if $i^{|x|} \in T$ for all $x \in L$. Note that if $T$ is del-left-preserving with respect to every language $L$ then $T$ is del-left-preserving. If $sym_d(T)$ is del-left-preserving (with respect to $L$), we say that $T$ is *sym-del-left-preserving* (with respect to $L$), or *sdl-preserving*.

**Lemma 8.7.2** *Let $L \subseteq \Sigma^*$. If $T$ is sdl-preserving with respect to $L$, then $L \subseteq scs_T(L)$.*

**Proof.** Note in this case that $scs_T(L) = L \rightsquigarrow_{sym_d(T)} \Sigma^* \supseteq L \rightsquigarrow_{sym_d(T)} \{\epsilon\} = L$, as $sym_d(T)$ is del-left-preserving. ■

Note that if $T$ is sdl-preserving, $T \supseteq d^*$. This is satisfied by, for example, right- and left-quotient, and sequential, bi-polar, $k$- and scattered deletion.

Consider that if $L \subseteq scs_T(L)$, then clearly $L = scs_T(L) \cap L$. This leads to the following observation:

**Proposition 8.7.3** *Let $T$ be sdl-preserving with respect to $L$. Then $L$ is del-$T$ closed if and only if $L \subseteq dr_T(L)$.*

For defining the $T$-del-closure of a language, we need the following notation. Define

$$dC_T(L) = \{L' \subseteq \Sigma^* \ : \ L \subseteq L' \text{ and } L' \cap scs_T(L') \subseteq dr_T(L')\}.$$

Then $dC_T(L)$ is the set of all del-$T$ closed languages containing $L$ ($dC_T(L) \neq \emptyset$ as $\Sigma^* \in dC_T(L)$). Further, define $dcl_T(L) = \cap_{L' \in dC_T(L)} L'$. It is not hard to see that

$$scs_T(dcl_T(L)) \subseteq \bigcap_{L' \in dC_T(L)} scs_T(L'), \tag{8.16}$$

and that

$$dcl_T(L) \cap scs_T(dcl_T(L)) \subseteq \bigcap_{L' \in dC_T(L)} dr_T(L). \tag{8.17}$$

With this, we can see that $dcl_T(L)$ is the smallest $T$-del-closed language containing $L$; we call $dcl_T(L)$ the *del-$T$ closure of $L$*.

**Proposition 8.7.4** *Let $T \subseteq \{i, d\}^*$. Then $L$ is del-$T$ closed if and only if $L \rightsquigarrow_T (L \cap scs_T(L)) \subseteq L$.*

**Proof.** The proof is similar to that of Lemma 8.6.2. Let $L$ be del-$T$ closed. Then for all $x \in L \cap scs_T(L)$, we have $x \in dr_T(L)$. Thus, for all $u \in L$, $u \rightsquigarrow_T x \subseteq L$. Clearly then, $L \rightsquigarrow_T (L \cap scs_T(L)) \subseteq L$.

For the reverse implication, let $L \rightsquigarrow_T (L \cap scs_T(L)) \subseteq L$. Consider $x \in L \cap scs_T(L)$; we show $x \in dr_T(L)$. As $L \rightsquigarrow_T (L \cap scs_T(L)) \subseteq L$, for all $y \in L$, $y \rightsquigarrow_T x \subseteq L$. Thus, by definition, as $x \in scs_T(L)$, we also have $x \in dr_T(L)$. ■

**Corollary 8.7.5** *Let $L \subseteq \Sigma^*$. Let $T \subseteq \{i, d\}^*$ be sdl-preserving with respect to $L$. Then $L$ is del-$T$-closed if and only if $L \rightsquigarrow L \subseteq L$.*

Corollary 8.7.5 was noted by Ito *et al.* for $T = (i + d)^*$ [79] and $T = i^*d^*i^*$ [78]. We can also generalize an interesting result of Ito *et al.* [78, 79] and Kari and Thierrin [115, Prop. 3.3]. Call a set of trajectories $T$ *square-enabling* if $\Psi(T) \supseteq \{(n, n) : n \geq 0\}$.

**Lemma 8.7.6** *Let $T \subseteq \{0, 1\}^*$ be square-enabling, and such that $\tau(T)$ is sdl-preserving. Let $L$ be a shuffle-$T$ closed language. Then $L$ is del-$\tau(T)$ closed if and only if $L = L \rightsquigarrow_{\tau(T)} L$.*

**Proof.** If $L = L \rightsquigarrow_{\tau(T)} L$, then by Corollary 8.7.5, $L$ is $\tau(T)$-del-closed.

Now, assume that $L$ is $\tau(T)$-del-closed. Again by Corollary 8.7.5, $L \supseteq L \rightsquigarrow_{\tau(T)} L$. Thus, let $x \in L$. we must show $x \in L \rightsquigarrow_{\tau(T)} L$.

As $L$ is shuffle-$T$ closed, $x \sqcup\!\sqcup_T x \subseteq L$. As $T$ is square-enabling, $x \sqcup\!\sqcup_T x \neq \emptyset$. Thus, let $t \in T$ and $y \in L$ be chosen so that $y \in x \sqcup\!\sqcup_t x$. By Theorem 5.8.2, $x \in y \rightsquigarrow_{\tau(t)} x$. Thus, $x \in L \rightsquigarrow_{\tau(T)} L$, as required. ■

Let $T \subseteq \{i, d\}^*$. Let $L$ be a language. We now give a characterization of the $T$-del-closure of a language $L$, when $T$ is sdl-preserving.

**Fact 8.7.7** *For all $T \subseteq \{i, d\}^*$ and $L \subseteq \Sigma^*$, $(\rightsquigarrow_T)^i(L) \subseteq (\rightsquigarrow_T)^{i+1}(L)$.*

**Theorem 8.7.8** *Let $L \subseteq \Sigma^*$ be a language and $T \subseteq \{i, d\}^*$ be sdl-preserving. Then $dcl_T(L) = (\rightsquigarrow_T)^+(L)$.*

**Proof.** Note that $L \subseteq (\rightsquigarrow_T)^+(L)$. Then to show that $dcl_T(L) \subseteq (\rightsquigarrow_T)^+(L)$, it suffices to show that $(\rightsquigarrow_T)^+(L)$ is del-$T$ closed. We appeal to Corollary 8.7.5. In particular, we show that

$$(\rightsquigarrow_T)^+(L) \rightsquigarrow_T (\rightsquigarrow_T)^+(L) \subseteq (\rightsquigarrow_T)^+(L).$$

Let $u, v \in (\rightsquigarrow_T)^+(L)$. Then there exist $i, j \geq 1$ such that $u \in (\rightsquigarrow_T)^i(L)$ and $v \in (\rightsquigarrow_T)^j(L)$. Let $k = \max(i, j)$. This implies that $u, v \in (\rightsquigarrow_T)^k(L)$. Thus, $u \rightsquigarrow v \subseteq (\rightsquigarrow_T)^{k+1}(L)$ by definition of $(\rightsquigarrow_T)^k$. We conclude that $(\rightsquigarrow_T)^+$ is del-$T$ closed and thus $dcl_T(L) \subseteq (\rightsquigarrow_T)^+(L)$.

To show the reverse inclusion, we show by induction on $i$ that $(\rightsquigarrow_T)^i(L) \subseteq dcl_T(L)$ for all $i \geq 1$. For $i = 1$, $L \subseteq dcl_T(L)$ by definition of $dcl_T(L)$. Now assume the result holds for all natural numbers less than $i$. Consider

$$\begin{aligned}
(\rightsquigarrow_T)^i(L) &= \left((\rightsquigarrow_T)^{i-1}(L) \rightsquigarrow_T (\rightsquigarrow_T)^{i-1}(L)\right) + (\rightsquigarrow_T)^{i-1}(L); \\
&\subseteq (dcl_T(L) \rightsquigarrow_T dcl_T(L)) + dcl_T(L); \\
&\subseteq (dcl_T(L)) + dcl_T(L) = dcl_T(L),
\end{aligned}$$

where the first inclusion is by induction on $i$, and the second inclusion is by Corollary 8.7.5, and the fact that $dcl_T(L)$ is $T$-del-closed. ■

Theorem 8.7.8 was proven by Ito *et al.* in the case of scattered [79, Prop. 4.4] and sequential [78, Prop. 3.4] deletion. Theorem 8.7.8 was proven by Kari and Thierrin [115, Prop. 3.4] for the case of $k$-deletion.

We now show that sdl-preservation is necessary in the alternate definition of $(\rightsquigarrow_T)^+_X(L)$ given by (8.9):

**Lemma 8.7.9** *There exist a set of trajectories $T$ and infinitely many languages $L$ such that $T$ is not sdl-preserving with respect to $L$ and $dcl_T(L) \neq (\rightsquigarrow_T)^+_X(L)$.*

**Proof.** We introduce a rather extreme example of a set $T$ satisfying the conditions. Let $T = d^*$. Then note that unless $L \subseteq \{\epsilon\}$, $T$ is not sdl-preserving with respect to $L$. Further, note that

$$L_1 \rightsquigarrow_T L_2 = \begin{cases} \{\epsilon\} & \text{if } L_1 \cap L_2 \neq \emptyset. \\ \emptyset & \text{otherwise.} \end{cases}$$

Then let $L$ be any language such that $\{\epsilon\}$ is properly contained in $L$. Then $L \rightsquigarrow_T L = \{\epsilon\}$ and by induction, we can show that $(\rightsquigarrow_T)^i_X(L) \rightsquigarrow_T ((\rightsquigarrow_T)^i_X(L) + \epsilon) = \{\epsilon\}$, for all $i \geq 2$. Thus, $(\rightsquigarrow_T)^+_X(L) = \{\epsilon\}$. But clearly in this case, $dcl_T(L) \neq \{\epsilon\}$, since $L \subseteq dcl_T(L)$ by definition. ■

## 8.8 $T$-Shuffle Base

We now extend the notion of shuffle base, examined by Ito *et al.* [78, 79] and Kari and Thierrin [115]. Let $L \subseteq \Sigma^+$ be a shuffle-$T$-closed language (the following definitions can be given for $L \subseteq \Sigma^*$, as was done by Ito *et al.* [78, 79] and Kari and Thierrin [115], but we restrict this possibility to allow for simpler definitions; the results below can also be given for the more complete definitions of Ito *et al.* and Kari and Thierrin without much difficulty). The *shuffle-T base* of $L$, denoted by $J_T(L)$, is the set of words in $L$ which cannot be expressed as the shuffle along $T$ of words in $L$. Thus,

$$J_T(L) = \{u \in L \ : \ u \notin L \sqcup_T L\}.$$

**Proposition 8.8.1** *Let $T$ be left-associative and $L \subseteq \Sigma^+$ be shuffle-$T$ closed. Then*

$$L = (\sqcup_T)^+(J_T(L)). \tag{8.18}$$

**Proof.** As $L$ is shuffle-$T$ closed, $L = (\sqcup_T)^+(L)$. Thus, it suffices to show that $(\sqcup_T)^+(L) = (\sqcup_T)^+(J_T(L))$. As $T$ is left-associative, we reduce this to showing following equality:

$$(\sqcup_T)^+_X(L) = (\sqcup_T)^+_X(J_T(L)).$$

To see that $(\sqcup\!\sqcup_T)_X^+(J_T(L)) \subseteq (\sqcup\!\sqcup_T)_X^+(L)$, we note that $J_T(L) \subseteq L$ and $(\sqcup\!\sqcup_T)_X^+$ is a monotone operator, since $\sqcup\!\sqcup_T$ is. For the reverse inclusion, let $x \in (\sqcup\!\sqcup_T)_X^+(L)$. Then we note that as $\epsilon \notin L$, if $x \in (\sqcup\!\sqcup_T)_X^i(L)$, then $i \leq |x|$. Thus, let $h_x$ be the maximum $i$ such that $x \in (\sqcup\!\sqcup_T)_X^i(L)$. We prove that $x \in (\sqcup\!\sqcup_T)_X^+(J_T(L))$ by induction on $h_x$.

For $h_x = 1$, $x \in L$ and $x \notin (\sqcup\!\sqcup_T)_X^2(L) = L \sqcup\!\sqcup_T L$. Thus, by definition, $x \in J_T(L) \subseteq (\sqcup\!\sqcup_T)_X^+(J_T(L))$. Assume the result holds for all $x$ with $h_x < h$ for some $h > 1$. Let $x$ be a word such that $h_x = h$. Then $x \in (\sqcup\!\sqcup_T)_X^h(L)$, and there exist $y \in (\sqcup\!\sqcup_T)_X^{h-1}(L)$ and $z \in L$ such that $x \in y \sqcup\!\sqcup_T z$. By induction, $y \in (\sqcup\!\sqcup_T)_X^+(J_T(L))$. If $z \in J_T(L)$, we are done. Therefore, assume that $z \in L - J_T(L)$. There exist $u, v \in L$ such that $z = u \sqcup\!\sqcup_T v$. Thus, $x \in y \sqcup\!\sqcup_T (u \sqcup\!\sqcup_T v) \subseteq (y \sqcup\!\sqcup_T u) \sqcup\!\sqcup_t v$, as $T$ is left-associative. But now $y \in (\sqcup\!\sqcup_T)_X^{h-1}(L)$, $u, v \in L$ imply that $x \in (\sqcup\!\sqcup_T)_X^{h+1}(L)$, a contradiction to our choice of $x$. Thus, $z \in J_T(L)$ and $x \in (\sqcup\!\sqcup_T)_X^+(J_T(L))$. ■

We now demonstrate that if $L$ and $T$ are regular and $L$ is shuffle-$T$ closed, then $J_T(L)$ is also regular.

**Lemma 8.8.2** *Let $T \subseteq \{0,1\}^*$ be a regular set of trajectories. If $L \subseteq \Sigma^+$ is regular and shuffle-$T$-closed, then $J_T(L)$ is regular.*

**Proof.** The proof will rely on establishing that

$$L - J_T(L) = L \sqcup\!\sqcup_T L. \tag{8.19}$$

Let $x \in L - J_T(L)$. Then as $x \notin J_T(L)$, there exist $u, v \in L$ such that $x \in u \sqcup\!\sqcup_T v$. Thus $x \in L \sqcup\!\sqcup_T L$. Now, let $x \in L \sqcup\!\sqcup_T L$. As $L$ is shuffle-$T$-closed, $L \sqcup\!\sqcup_T L \subseteq L$. Thus $x \in L$. As $x \in L \sqcup\!\sqcup_T L$, $x \notin J_T(L)$ by definition. This establishes (8.19). Now, as $L \sqcup\!\sqcup_T L$, $J_T(L) \subseteq L$, we have that $J_T(L) = L - L \sqcup\!\sqcup_T L$. Since $L$ and $L \sqcup\!\sqcup_T L$ are regular, $J_T(L)$ is regular. ■

Lemma 8.8.2 offers alternate proofs of the corresponding results by Ito *et al.* for scattered deletion [79, Prop. 5.1] and sequential deletion [78, Prop. 5.1], and by Kari and Thierrin for $k$-insertion [115, Prop. 2.5].

## 8.9 Shuffle-Free Languages

In this section, we consider the notion of shuffle-free languages. This was first examined by Ito *et al.* [82]. Hsiao *et al.* [69, Sect. 5] also considered a similar notion for arbitrary word operations. We show that for shuffle on trajectories, we can obtain results relating shuffle-free languages and $T$-codes.

We adopt the following shorthand:

$$(\sqcup\!\sqcup_T)^{\geq 2}(L) = \bigcup_{i \geq 2} (\sqcup\!\sqcup_T)^i(L).$$

Let $T \subseteq \{0, 1\}^*$. Then we say that $\emptyset \neq L \subseteq \Sigma^+$ is *sh-$T$-free* if

$$(\sqcup\!\sqcup_T)^{\geq 2}(L) \cap L = \emptyset.$$

Thus, $L$ is sh-$T$-free if, for all $u_1, u_2, \ldots, u_k \in L$ ($k \geq 2$), there is no way to shuffle the $u_j$s together with $\sqcup\!\sqcup_T$ to get a word from $L$. The concept of sh-$T$-free is an extension of the corresponding notion, sh-free, for arbitrary shuffle; this was introduced by Ito *et al.* [82].

The following lemma will be useful:

**Lemma 8.9.1** *Let $T \subseteq \{0, 1\}^*$. Let $L_1, L_2$ be two sh-$T$-free languages such that $(\sqcup\!\sqcup_T)^+(L_1) = (\sqcup\!\sqcup_T)^+(L_2)$. Then $L_1 = L_2$.*

**Proof.** Assume not. Let $x \in L_1 - L_2$, without loss of generality. Since $x \in L_1 \subseteq (\sqcup\!\sqcup_T)^+(L_1) = (\sqcup\!\sqcup_T)^+(L_2)$, either $x \in L_2$ or there exist $u_1, u_2 \in (\sqcup\!\sqcup_T)^+(L_2)$ such that $x \in u_1 \sqcup\!\sqcup_T u_2$. As $x \notin L_2$ by assumption, let $x \in u_1 \sqcup\!\sqcup_T u_2$. We now note that $u_1, u_2 \in (\sqcup\!\sqcup_T)^+(L_2) = (\sqcup\!\sqcup_T)^+(L_1)$. Thus, $x \in (\sqcup\!\sqcup_T)^+(L_1) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^+(L_1) \subseteq (\sqcup\!\sqcup_T)^{\geq 2}(L_1)$. This contradicts that $L_1$ is sh-$T$-free. ■

Let $L \subseteq \Sigma^*$. We say that $B_T(L) \subseteq L \cap \Sigma^+$ is an *extended sh-$T$-base* of $L$ if $B_T(L)$ is sh-$T$-free and $L \cap \Sigma^+ \subseteq (\sqcup\!\sqcup_T)^+(B_T(L))$.

**Lemma 8.9.2** *Let $T \subseteq \{0, 1\}^*$ and let $L \subseteq \Sigma^*$. Then the extended sh-$T$-base of $L$ is unique.*

**Proof.** Let $B_1$, $B_2$ be two extended sh-$T$-bases of $L$.

Let $x \in B_1$. Then $x \neq \epsilon$ by definition. Further, $x \in \Sigma^+ \cap L \subseteq (\sqcup\!\sqcup_T)^+(B_2)$, by the fact that $B_2$ is an extended-$T$-base of $L$. Thus, $B_1 \subseteq (\sqcup\!\sqcup_T)^+(B_2)$, and

$$(\sqcup\!\sqcup_T)^+(B_1) \subseteq (\sqcup\!\sqcup_T)^+((\sqcup\!\sqcup_T)^+(B_2)) \subseteq (\sqcup\!\sqcup_T)^+(B_2),$$

where the last inclusion is valid by Theorem 8.6.5. By symmetry, we also have that $(\sqcup\!\sqcup_T)^+(B_2) \subseteq (\sqcup\!\sqcup_T)^+(B_1)$.

As $B_1$, $B_2$ are sh-$T$-free, we have that $B_1 = B_2$, by Lemma 8.9.1. ■

We now show that every language has a sh-$T$-base. Consider the following languages [69, 82]:

$$\begin{aligned} B_0 &= \emptyset; \\ K_i &= L - (\sqcup\!\sqcup_T)^+(\bigcup_{j=0}^{i-1} H_j); \quad \forall i \geq 1 \\ B_i &= \{x \,:\, x \in K_i \text{ and } |x| \leq |y| \quad \forall y \in K_i\}; \quad \forall i \geq 1 \\ B &= \bigcup_{i \geq 1} B_i. \end{aligned}$$

Then it is straight-forward to establish that $B$ is a sh-$T$-base for $L$ (see, e.g., Hsiao *et al.* [69, Prop. 12]).

There is an interesting relation between extended sh-$T$-bases and $T$-codes. This emphasizes the naturalness of the definition of $T$-codes.

**Lemma 8.9.3** *Let $T \subseteq \{0,1\}^*$ be a set of trajectories such that $\pi(T)$ is sdl-preserving. If $B_T(L)$ is the extended sh-$T$-base of any del-$\pi(T)$ closed language $L$, then $B_T(L)$ is a $T$-code.*

**Proof.** Let $B_T(L)$ be the extended sh-$T$-base of $L$. Suppose $B_T(L)$ is not a $T$-code. Then there exist $u, v \in B_T(L) \subseteq L$ such that $v \in u \sqcup\!\sqcup_T w$ for some $w \in \Sigma^+$. By Theorem 5.8.2, $w \in v \rightsquigarrow_{\pi(T)} u$.

As $u, v \in L$, $w \in L$ by Corollary 8.7.5. Thus, $w \in L \cap \Sigma^+ \subseteq (\sqcup\!\sqcup_T)^+(B_T(L))$. But then $v \in u \sqcup\!\sqcup_T w \subseteq (\sqcup\!\sqcup_T)^{\geq 2}(B_T(L))$. Thus, $v \in B_T(L) \cap (\sqcup\!\sqcup_T)^{\geq 2}(B_T(L))$, a contradiction. ■

Lemma 8.9.3 was established in a slightly weaker form by Ito and Silva [80, Prop. 3.2] for the case $T = (0+1)^*$. Hsiao *et al.* [69, Prop. 22] note the result for $T = 0^*1^* + 1^*0^*$.

The converse is given in a more general form as follows:

**Lemma 8.9.4** *Let $T \subseteq \{0,1\}^*$ be a set of trajectories such that $\pi(T)$ is sdl-preserving. Let $L \in \mathcal{P}_T(L)$. Then $L \cup \{\epsilon\}$ is del-$\pi(T)$ closed.*

**Proof.** As $L \in \mathcal{P}_T(L)$, $L \rightsquigarrow_{\pi(T)} L \subseteq \{\epsilon\}$, by (6.8). Consider that

$$\begin{aligned} & L \cup \{\epsilon\} \rightsquigarrow_{\pi(T)} L \cup \{\epsilon\} \\ = \; & (L \rightsquigarrow_{\pi(T)} L) \cup (\{\epsilon\} \rightsquigarrow_{\pi(T)} L \cup \{\epsilon\}) \cup (L \rightsquigarrow_{\pi(T)} \{\epsilon\}) \\ \subseteq \; & \{\epsilon\} \cup \{\epsilon\} \cup L = L \cup \{\epsilon\}. \end{aligned}$$

Therefore, $L \cup \{\epsilon\}$ is del-$\pi(T)$ closed, by Corollary 8.7.5. ■

## 8.10 $T$-Primitive Words

A word $w \in \Sigma^*$ is said to be *primitive* if $w = u^k$ implies $u = w$ and $k = 1$. For instance, the words $aab$ and $ababbb$ are primitive, while $aabaab = (aab)^2$ is not. The concept of primitive words is one of the most studied in formal language theory, and poses one of the most well-known of all open problems in formal language theory concerning the complexity of the set of all primitive words. The concept of a primitive word has been extended from concatenation to insertion and shuffle by Kari and Thierrin [118] and to arbitrary word operations by Hsiao *et al.* [69]. In this section, we consider primitivity with respect to a given shuffle on trajectories operation. We show that under our general definition of $(\sqcup\!\sqcup_T)^*$, every word has a $T$-primitive root. We also consider analogues of the Lyndon-Schützenberger Theorems for shuffle on trajectories.

### 8.10.1 $T$-Primitivity and $T$-roots

In this section, we consider primitive words with respect to a set of trajectories. Our definition will be with respect to our definition of iteration, which will alleviate certain restrictions which were

imposed by Hsiao *et al.* [69].

Let $T \subseteq \{0, 1\}^*$. Given a word $w \in \Sigma^*$, we say that $w$ is *$T$-primitive* if $w \in (\sqcup\!\sqcup_T)^n(u)$ implies $u = w$ and $n = 1$. Let $Q_T(\Sigma)$ be the set of all $T$-primitive words over an alphabet $\Sigma$. For all $T \subseteq \{0, 1\}^*$ and $w \in \Sigma^*$, let

$$\sqrt[T]{w} = \{u \in Q_T(\Sigma) \ : \ w \in (\sqcup\!\sqcup_T)^+(u)\}.$$

We call $\sqrt[T]{w}$ the set of *$T$-primitive roots* of $w$.

It is well known (see, e.g, Lothaire [140, Prop. 1.3.1]) that every non-empty word has a unique primitive (i.e., $T$-primitive for $T = 0^*1^*$) root, that is, for $T = 0^*1^*$, $|\sqrt[T]{w}| = 1$ for all $w \in \Sigma^+$. Kari and Thierrin [118] note that for $T = 0^*1^*0^*$, this does not hold, as, e.g., $babb, bbab \in \sqrt[T]{(b^2a)^{n+1}b^{n+1}}$ for all $n \geq 1$. However, we now note that for all $T$, every non-empty word has at least one $T$-primitive root. We will require the following lemma:

**Lemma 8.10.1** *Let $u \in \Sigma^*$. Then $(\sqcup\!\sqcup_T)^+((\sqcup\!\sqcup_T)^+(u)) \subseteq (\sqcup\!\sqcup_T)^+(u)$.*

**Proof.** The proof follows by the fact that $(\sqcup\!\sqcup_T)^+(u)$ is shuffle-$T$ closed, by Theorem 8.6.5. ■

**Theorem 8.10.2** *Let $T \subseteq \{0, 1\}^*$. Let $w \in \Sigma^+$. Then $|\sqrt[T]{w}| \geq 1$.*

**Proof.** Let $w \in \Sigma^+$ be arbitrary. If $w \in Q_T(\Sigma)$, then we are done, as $w \in \sqrt[T]{w}$. Thus, assume that $w$ is not $T$-primitive.

Let $w \in (\sqcup\!\sqcup_T)^i(u)$ for some $u \in \Sigma^*$ and $i \geq 2$. If $u$ is $T$-primitive, then we are done, as $u \in \sqrt[T]{w}$.

Note that $|u| < |w|$ as $i \geq 2$. Now, if $u$ is not $T$-primitive, then $u \in (\sqcup\!\sqcup_T)^j(v)$ for some $v \in \Sigma^*$ and $j \geq 2$. Note that $w \in (\sqcup\!\sqcup_T)^+((\sqcup\!\sqcup_T)^+(v)) \subseteq (\sqcup\!\sqcup_T)^+(v)$. Thus, if $v$ is $T$-primitive, we are done.

Otherwise, as $|v| < |u| < |w|$, we note that as we continue this process, we eventually reach a word $x$ such that $w \in (\sqcup\!\sqcup_T)^+(x)$ and $x$ is $T$-primitive. This completes the proof. ■

### 8.10.2 Freeness and Uniqueness of $T$-Primitive Roots

We now turn to uniqueness of $T$-primitive roots. We will require the notion of a free semigroup, see, e.g., Shyr [184]. Recall that a semigroup is a set $S$ equipped with an associative binary operation; it does not necessarily have an identity element. Let $M$ be a semigroup. A non-empty subset $B \subseteq M$ is said to be a *base* for $M$ if and only if for all $u_1, u_2, \ldots, u_n, v_1, v_2, \ldots, v_m \in B$, the equality

$$u_1 u_2 \cdots u_n = v_1 v_2 \cdots v_m$$

implies that $n = m$ and $u_i = v_i$ for all $1 \leq i \leq n$. Note that $\Sigma$ is a base for $\Sigma^+$ as a semigroup under concatenation. A semigroup $M$ is said to be *free* if there exists some base $B$ such that $B^* = M$ (it can be easily shown that such a base must be unique). Levi [132] gives conditions on a semigroup being free. Let $T$ be an associative, deterministic set of trajectories. We say that $T$ is *free* if the semigroup $M = (\Sigma^+, \sqcup\!\sqcup_T)$ is free[2].

As we will be dealing exclusively with associative sets of trajectories, we will use the operation $(\sqcup\!\sqcup_T)^+_X$, which we gave in (8.5).

We first note that, besides concatenation, there exist non-trivial operations which are free. For instance, the operation of balanced insertion, given by $T = \{0^k 1^{2j} 0^k \ : \ k, j \geq 0\}$ is both deterministic and associative (see Mateescu *et al.* [147]) and is free, as is easily verified using Levi's conditions.

We now give an open problem concerning freeness:

**Open Problem 8.10.3** *Given $T$ regular (or context-free), is it decidable whether $T$ is free?*

It does not seem that Levi's conditions are helpful in this regard. Further, consider the following test: let $L = \Sigma^+ - (\Sigma^+ \sqcup\!\sqcup_T \Sigma^+)$. Test if $L$ is a $*$-T-code (i.e., every word in $(\sqcup\!\sqcup_T)^+(L)$ is uniquely generated). Then $T$ is free if and only if $L$ is a $*$-T-code.

[2]We could also easily frame the current discussion in terms of free monoids (i.e., semigroups with identities), and say that $T$ is free if the monoid $M = (\Sigma^*, \sqcup\!\sqcup_T, \epsilon)$ is free. However, as noted by Mateescu *et al.* [147, Rem. 4.7], we only need to require that $0^* + 1^* \subseteq T$ to ensure that $\epsilon$ is the identity element for $M$.

Since $T$ is regular implies that $L$ is regular, we expect that this would a plausible test for the freeness of $T$, since it is decidable whether a regular language is a $*$-code. However, the well-known algorithm to determine if a regular language is a $*$-code (e.g., Berstel and Perrin [18, Thm. I.3.1]) relies on the fact that $\Sigma^*$ is a free monoid under concatenation. Thus, it does not seem immediately possible to generalize this proof to other $T \subseteq \{0, 1\}^*$.

Levi's conditions are occasionally referred to as Levi's Lemma (see, e.g., Allouche and Shallit [3, Sect. 1.4]), which can be stated as follows for our purposes:

**Lemma 8.10.4** *Let $T$ be deterministic, associative and free. Then for all $u, v, x, y \in \Sigma^*$, $u \sqcup\!\sqcup_T v = x \sqcup\!\sqcup_T y$ implies that there exists $z \in \Sigma^*$ such that either $u = x \sqcup\!\sqcup_T z$ or $x = u \sqcup\!\sqcup_T z$.*

Levi's lemma (with $T = 0^*1^*$) is necessary for two of the most fundamental and elegant theorems in formal language theory and combinatorics on words, the Lyndon-Schützenberger Theorems (see, e.g., Allouche and Shallit [3, Sect. 1.4]):

**Theorem 8.10.5** *Let $x, z \in \Sigma^+$, $y \in \Sigma^*$. Then the following are equivalent:*

*(a) $xy = yz$;*

*(b) there exist $u, v \in \Sigma^*$, $e \geq 0$ such that $x = uv$, $z = vu$ and $y = (uv)^e u$.*

**Theorem 8.10.6** *Let $x, y \in \Sigma^+$. Then the following three conditions are equivalent:*

*(a) $xy = yx$;*

*(b) there exist integers $i, j > 0$ such that $x^i = y^j$;*

*(c) there exist $z \in \Sigma^+$ and integers $k, \ell > 0$ such that $x = z^k$ and $y = z^\ell$.*

We now show that freeness is the essential property in proving a generalization of the Lyndon-Schützenberger Theorems for $\sqcup\!\sqcup_T$.

The additional condition necessary to generalize the first Lyndon-Schützenberger Theorem is the possession of a unit element (see Mateescu *et al.* [147, Sect. 4.4]), which is the condition that $0^* + 1^* \subseteq T$. Thus, if $T$ has a unit element, $x \in (x \sqcup\!\sqcup_T \epsilon) \cap (\epsilon \sqcup\!\sqcup_T x)$ for all $x \in \Sigma^*$.

**Theorem 8.10.7** *Let $T$ be an associative, deterministic, free set of trajectories with a unit element. Let $x, z \in \Sigma^+$ and $y \in \Sigma^*$ be such that $x \sqcup_T y, y \sqcup_T z \neq \emptyset$. Then the following are equivalent:*

*(a)* $x \sqcup_T y = y \sqcup_T z$;

*(b) there exist* $u, v \in \Sigma^*$, $e \geq 0$ *such that*

$$\begin{aligned} x &= u \sqcup_T v, \\ z &= v \sqcup_T u, \\ y &= (\sqcup_T)^e_X(u \sqcup_T v) \sqcup_T u. \end{aligned}$$

**Proof.** ((a) $\Rightarrow$ (b)): Let $x \sqcup_T y = y \sqcup_T z$. The proof is by induction on $|y|$. The base case for $y \in \Sigma^*$ is $|y| = 0$. In this case, $x = x \sqcup_T \epsilon$ and $z = \epsilon \sqcup_T z$, as $x \sqcup_T \epsilon, \epsilon \sqcup_T z \neq \emptyset$, by assumption. We conclude that $x = z$ and (b) holds with $u = \epsilon$, $v = x = z$ and $e = 0$.

Assume the result holds for all words $y'$ with $0 \leq |y'| < |y|$. Let $x \sqcup_T y = y \sqcup_T z$. By Lemma 8.10.4, there exists $w \in \Sigma^*$ such that either $x = y \sqcup_T w$ and $z = w \sqcup_T y$ or $y = x \sqcup_T w$ and $y = w \sqcup_T z$.

In the first case, let $u = y$, $v = w$ and $e = 0$. Then $y = u = (\sqcup_T)^e_X(u \sqcup_T v) \sqcup_T u$, as $T$ is ST-strict. Further, $x = u \sqcup_T v$ and $z = v \sqcup_T u$.

In the second case, we have that $x \sqcup_T w = w \sqcup_T z$. Note that $|w| = |y| - |z| < |y|$ as $z \in \Sigma^+$. Therefore, by induction, there exist $u, v \in \Sigma^*$ and $e \geq 0$ such that $x = u \sqcup_T v$, $z = v \sqcup_T u$ and $w = (\sqcup_T)^e_X(u \sqcup_T v) \sqcup_T u$. Note that

$$y = x \sqcup_T w = (u \sqcup_T v) \sqcup_T (\sqcup_T)^e_X(u \sqcup_T v) \sqcup_T u = (\sqcup_T)^{e+1}_X(u \sqcup_T v) \sqcup_T u,$$

by the associativity of $T$. Thus, (b) is satisfied.

((b) $\Rightarrow$ (a)): Note that

$$\begin{aligned} x \sqcup\!\sqcup_T y &= (u \sqcup\!\sqcup_T v) \sqcup\!\sqcup_T (\ \sqcup\!\sqcup_T)^e_X (u \sqcup\!\sqcup_T v) \sqcup\!\sqcup_T u \\ &= u \sqcup\!\sqcup_T v \sqcup\!\sqcup_T \underbrace{(u \sqcup\!\sqcup_T v) \sqcup\!\sqcup_T \cdots \sqcup\!\sqcup_T (u \sqcup\!\sqcup_T v)}_{e \text{ times.}} \sqcup\!\sqcup_T u \\ &= \underbrace{(u \sqcup\!\sqcup_T v) \sqcup\!\sqcup_T \cdots \sqcup\!\sqcup_T (u \sqcup\!\sqcup_T v)}_{e \text{ times.}} \sqcup\!\sqcup_T u \sqcup\!\sqcup_T v \sqcup\!\sqcup_T u \\ &= y \sqcup\!\sqcup_T z. \end{aligned}$$

This proves the desired equality. ■

Call a set $T$ of trajectories *concatenation-like* if $T$ is deterministic, associative, free, and satisfies the following property, which we call *power-enabling*: for all $n, k \geq 0$, $(kn, n) \in \Psi(T) \Rightarrow ((k+1)n, n) \in \Psi(T)$. Note that balanced insertion is power-enabling and hence concatenation-like.

We will require the following proposition, which is easily proven by induction:

**Proposition 8.10.8** *Let $T$ be associative. Then for all $k, \ell > 0$, and all $z \in \Sigma^+$,*

$$(\ \sqcup\!\sqcup_T)^k_X((\ \sqcup\!\sqcup_T)^\ell_X(z)) = (\ \sqcup\!\sqcup_T)^{k\ell}_X(z).$$

We may now state and prove our second generalized Lyndon-Schützenberger Theorem:

**Theorem 8.10.9** *Let $T$ be concatenation-like. Let $x, y \in \Sigma^+$ be words such that $x \sqcup\!\sqcup_T y$ and $y \sqcup\!\sqcup_T x$ are non-empty. Then the following three conditions are equivalent:*

*(a) $x \sqcup\!\sqcup_T y = y \sqcup\!\sqcup_T x$;*

*(b) there exist integers $i, j > 0$ such that $\emptyset \neq (\ \sqcup\!\sqcup_T)^i_X(x) = (\ \sqcup\!\sqcup_T)^j_X(y)$;*

*(c) there exist $z \in \Sigma^+$ and $k, \ell > 0$ such that $x = (\ \sqcup\!\sqcup_T)^k_X(z)$ and $y = (\ \sqcup\!\sqcup_T)^\ell_X(z)$.*

**Proof.** ((a) $\Rightarrow$ (c)): Assume that $x \neq y$ and $\emptyset \neq x \sqcup\!\sqcup_T y = y \sqcup\!\sqcup_T x$. We show the result by induction on $|xy|$. As $x, y \in \Sigma^+$, the base case is $|xy| = 2$. Thus, $x, y \in \Sigma$. Thus, $x \sqcup\!\sqcup_T y = y \sqcup\!\sqcup_T x$, and $T$ deterministic implies that $x = y$, contrary to our assumption.

Assume that the result holds for all $x, y \in \Sigma^+$ with $|xy| < n$. Let $|xy| = n$. Let $|x| \geq |y|$. As $x \sqcup\!\sqcup_T y = y \sqcup\!\sqcup_T x$, by Levi's property, there exists $w \in \Sigma^*$ such that $x = y \sqcup\!\sqcup_T w$. Note that

$y \sqcup\!\sqcup_T x = x \sqcup\!\sqcup_T y = (y \sqcup\!\sqcup_T w) \sqcup\!\sqcup_T y = y \sqcup\!\sqcup_T (w \sqcup\!\sqcup_T y)$ by the associativity of $T$. Thus, by the determinism of $T$, $x = w \sqcup\!\sqcup_T y$ and $w \sqcup\!\sqcup_T y = y \sqcup\!\sqcup_T w$.

If $w = y$, then $x = y \sqcup\!\sqcup_T y$, and (c) follows with $z = y$ and $k = 2$, $\ell = 1$. Thus, assume that $w \neq y$.

If $|w| = 0$, then $x = y \sqcup\!\sqcup_T w$ implies that $x = y$. Again, this contradicts our assumptions on $x, y$. Thus, $|w| > 0$. Note that $|wy| = |x| < |xy|$. Thus, as $w \sqcup\!\sqcup_T y = y \sqcup\!\sqcup_T w$, by induction there exist $z \in \Sigma^+$, $k, \ell > 0$ such that $w = (\sqcup\!\sqcup_T)^k_X(z)$ and $y = (\sqcup\!\sqcup_T)^\ell_X(z)$. Thus,

$$x = (\sqcup\!\sqcup_T)^\ell_X(z) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^k_X(z).$$

Thus, $x \in (\sqcup\!\sqcup_T)^+_X(z)$ by the closure of $(\sqcup\!\sqcup_T)^+_X(z)$ under $\sqcup\!\sqcup_T$. As $T$ is deterministic, we have that there is some $m > 0$ such that $x = (\sqcup\!\sqcup_T)^m_X(z)$. Thus, (c) is satisfied.

((c) $\Rightarrow$ (b)): Let $z \in \Sigma^+$, $k, \ell > 0$ be such that $x = (\sqcup\!\sqcup_T)^k_X(z)$ and $y = (\sqcup\!\sqcup_T)^\ell_X(z)$. Using Proposition 8.10.8, we note that

$$(\sqcup\!\sqcup_T)^\ell_X(x) = (\sqcup\!\sqcup_T)^\ell_X((\sqcup\!\sqcup_T)^k_X(z)) = (\sqcup\!\sqcup_T)^{\ell k}_X(z) = (\sqcup\!\sqcup_T)^k_X((\sqcup\!\sqcup_T)^\ell_X(z)) = (\sqcup\!\sqcup_T)^k_X(y).$$

By the fact that $(\sqcup\!\sqcup_T)^k_X(z) = x$, $((k-1)|z|, |z|) \in \Psi(T)$. Thus, $((\ell \cdot k - 1)|z|, |z|) \in \Psi(T)$ and both $(\sqcup\!\sqcup_T)^\ell_X(x)$ and $(\sqcup\!\sqcup_T)^k_X(y)$ are non-empty. Thus, (b) is satisfied.

((b) $\Rightarrow$ (a)): Let $i, j > 0$ be such that $\emptyset \neq (\sqcup\!\sqcup_T)^i_X(x) = (\sqcup\!\sqcup_T)^j_X(y)$. Note that if $|x| = |y|$, then $|(\sqcup\!\sqcup_T)^i_X(x)| = |(\sqcup\!\sqcup_T)^j_X(y)|$ implies that $i = j$. Thus, by the determinism of $T$, $x = y$ and (a) holds.

Thus, without loss of generality, assume that $|x| > |y|$. Thus, by the associativity of $T$,

$$x \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^{i-1}_X(x) = (\sqcup\!\sqcup_T)^i_X(x) = (\sqcup\!\sqcup_T)^j_X(y) = y \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^{j-1}_X(y).$$

(We let $(\sqcup\!\sqcup_T)^{j-1}_X(y) = \epsilon$ if $j = 1$, and similarly for $x$ and $i$.) Thus, by Levi's property, there exists some $w \in \Sigma^+$ such that $x = y \sqcup\!\sqcup_T w$. Thus,

$$(\sqcup\!\sqcup_T)^j_X(y) = (\sqcup\!\sqcup_T)^i_X(x) = (\sqcup\!\sqcup_T)^i_X(y \sqcup\!\sqcup_T w).$$

Note that

$$(\sqcup\!\sqcup_T)^i_X(y \sqcup\!\sqcup_T w) = \underbrace{(y \sqcup\!\sqcup_T w) \sqcup\!\sqcup_T (y \sqcup\!\sqcup_T w) \sqcup\!\sqcup_T \cdots \sqcup\!\sqcup_T (y \sqcup\!\sqcup_T w)}_{i \text{ times}}.$$

By the determinism of $T$, $(\sqcup\!\sqcup_T)^j_X(y) = (\sqcup\!\sqcup_T)^i_X(y \sqcup\!\sqcup_T w)$ implies that

$$\begin{aligned}
(\sqcup\!\sqcup_T)^{j-1}_X(y) &= w \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^{i-1}_X(y \sqcup\!\sqcup_T w) \\
\Rightarrow \quad (\sqcup\!\sqcup_T)^{j-1}_X(y) \sqcup\!\sqcup_T y &= w \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^{i-1}_X(y \sqcup\!\sqcup_T w) \sqcup\!\sqcup_T y \\
\Rightarrow \quad (\sqcup\!\sqcup_T)^j_X(y) &= (\sqcup\!\sqcup_T)^i_X(w \sqcup\!\sqcup_T y) \\
\Rightarrow \quad (\sqcup\!\sqcup_T)^i_X(y \sqcup\!\sqcup_T w) &= (\sqcup\!\sqcup_T)^i_X(w \sqcup\!\sqcup_T y).
\end{aligned}$$

By the determinism of $T$, $y \sqcup\!\sqcup_T w = w \sqcup\!\sqcup_T y$. Thus,

$$\begin{aligned}
x \sqcup\!\sqcup_T y &= (y \sqcup\!\sqcup_T w) \sqcup\!\sqcup_T y \\
&= y \sqcup\!\sqcup_T (w \sqcup\!\sqcup_T y) \\
&= y \sqcup\!\sqcup_T x.
\end{aligned}$$

Thus, (a) is satisfied, as $y \neq x$ and $y \sqcup\!\sqcup_T x = x \sqcup\!\sqcup_T y \neq \emptyset$. ■

Our main corollary of the generalized Lyndon-Schützenberger Theorems concerns the uniqueness of $T$-primitive roots:

**Corollary 8.10.10** *Let $T$ be concatenation-like. Then for all $u \in \Sigma^+$, $u$ has a unique $T$-primitive root.*

**Proof.** Let $u \in \Sigma^+$, and assume that $v_1, v_2 \in \sqrt[T]{u}$. Then as $T$ is concatenation-like, we have $u = (\sqcup\!\sqcup_T)^{i_1}(v_1)$ and $u = (\sqcup\!\sqcup_T)^{i_2}(v_2)$ for some $i_1, i_2 \geq 1$. Thus, there exist $j_1, j_2$ such that $u = (\sqcup\!\sqcup_T)^{j_1}_X(v_1)$ and $u = (\sqcup\!\sqcup_T)^{j_2}_X(v_2)$.

By Theorem 8.10.9, there exist $z \in \Sigma^+$ and $k_1, k_2 \geq 1$ such that $v_1 = (\sqcup\!\sqcup_T)^{k_1}_X(z)$ and $v_2 = (\sqcup\!\sqcup_T)^{k_2}_X(z)$. As $v_1, v_2 \in Q_T(\Sigma)$, we must have that $k_1 = k_2 = 1$ and $z = v_1 = v_2$. Thus, $|\sqrt[T]{u}| = 1$, as required. ■

Thus, we see that if $T$ is catenation-like, then each word has a unique $T$-primitive root. This applies, e.g., to balanced insertion.

## 8.11 Language Equations Revisited

In Chapter 7, we have studied language equations involving shuffle and deletion along trajectories. In this section, we revisit this topic to consider some equation forms which we did not consider in Chapter 7.

In particular, we note that all of language equations we have considered have the form

$$L = X \diamond Y$$

where $X, Y$ (and occasionally $\diamond$) may be unknown, but the language $L$ is always fixed. We recall that, in the terminology of, e.g., Leiss [131, Sect. 2.6], such equations are called *implicit language equations*. In this section, we consider *explicit language equations*, which we recall are of the form $X = \alpha$ where $X$ is unknown, and $\alpha$ is an equation involving constants, language operations (including $\sqcup\!\sqcup_T$) and unknowns.

Clearly, explicit language equations are of considerable interest; indeed, the fundamental notion of CFGs is equivalent to explicit systems of language equations of the form $X = \alpha$ where $\alpha$ is an expression involving catenation and union [10, Sect. 2]. Extensions of the context-free grammar formalism to capture larger classes of languages via grammars have also been studied, for example, conjunctive [157] and Boolean [159] grammars, both recently introduced by Okhotin. Explicit language equations are crucial to both these studies.

Consideration of language equations with unknowns on both sides of the equality sign must involve some caution, however, especially with regard to our interest thus far in answering questions such as "is it decidable whether this equation has a solution?" Consider the equation $X = L_1 \sqcup\!\sqcup_T L_2$, where $L_1, L_2$ are languages and $T$ is a set of trajectories. Clearly, this equation has a solution $X = L_1 \sqcup\!\sqcup_T L_2$. Note that this assumes nothing about the complexity of $L_1, L_2$ or $T$. Other equations possess trivial solutions; $X = X \sqcup\!\sqcup_T L$ has a solution $X = \emptyset$ regardless of $L$ and $T$.

Thus, in this section, our focus will shift somewhat from decidability, which is often trivial, to characterizations of solutions. Our results will be similar in spirit to Arden's Lemma (Arden [9],

cf., Salomaa [173, Ch. 3]) which states that the equation $X = XR_1 + R_2$ has a unique solution $R_1^* R_2$ if $\epsilon \notin R_1$. Further, $R_1^* R_2$ is the least solution (by inclusion) of $X = XR_1 + R_2$, independent of $R_1$ (see, e.g., Conway [28, Thm. 3, p. 27]). We will see that such results can be extended to $\sqcup\!\sqcup_T$, under certain assumptions on $T$, and that related equations have similar characterizations.

### 8.11.1 Arden-like Equations

Our first consideration will be the following equation:

$$X = X \sqcup\!\sqcup_T L_1 + L_2,$$

where $X$ is unknown, and $L_1, L_2$ are arbitrary languages. In the case where $T = 0^*1^*$, the characterization of the unique solution of this equation (under the assumption that $\epsilon \notin L_1$) is known as Arden's Lemma (however, the identity $L_2 L_1^* = L_2 L_1^* L_1 + L_2$ was previously known, see, e.g., Kleene [119, Eq. (9), p. 24]). We will require some conditions on $T$ in order for a similar result to hold.

**Theorem 8.11.1** *Let $T \subseteq \{0,1\}^*$ be left-preserving and associative. Let $L_1, L_2$ be arbitrary languages. The least solution to the equation*

$$X = X \sqcup\!\sqcup_T L_1 + L_2, \tag{8.20}$$

*is the language $L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^*(L_1)$.*

**Proof.** As $T$ is associative, it will suffice to establish that the language $L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^*_X(L_1)$ is the least solution to (8.20). Recall that $(\sqcup\!\sqcup_T)^*_X$ is defined by (8.4).

We first show that $L = L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^*_X(L_1)$ is a solution to (8.20). We establish the inclusion $L \sqcup\!\sqcup_T L_1 + L_2 \subseteq L$, by proving that

$$L_2 \subseteq L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^*_X(L_1) \tag{8.21}$$

and that

$$\left(L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^i_X(L_1)\right) \sqcup\!\sqcup_T L_1 \subseteq L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^*_X(L_1) \tag{8.22}$$

for all $i \geq 0$. We note that (8.21) holds since $T$ is left-preserving and $\epsilon \in (\sqcup\!\sqcup_T)^*_X(L_1)$. We now establish (8.22) for all $i \geq 0$. Note that for arbitrary $i \geq 0$,

$$\begin{aligned}(L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^i_X(L_1)) \sqcup\!\sqcup_T L_1 &= L_2 \sqcup\!\sqcup_T ((\sqcup\!\sqcup_T)^i_X(L_1) \sqcup\!\sqcup_T L_1) \\ &= L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^{i+1}_X(L_1) \\ &\subseteq L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^*_X(L_1).\end{aligned}$$

We now establish the inclusion $L \sqcup\!\sqcup_T L_1 + L_2 \supseteq L$. Let $x \in L = L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^*_X(L_1)$. Then there exists $i \geq 0$ such that $x \in L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^i_X(L_1)$. For $i = 0$, $x \in L_2 \sqcup\!\sqcup_T \{\epsilon\} \subseteq L_2$. Thus, $x \in L_2 \subseteq L \sqcup\!\sqcup_T L_1 + L_2$. Let $i > 0$. Then

$$\begin{aligned}x &\in L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^i_X(L_1) \\ &= L_2 \sqcup\!\sqcup_T ((\sqcup\!\sqcup_T)^{i-1}_X(L_1) \sqcup\!\sqcup_T L_1) \\ &= (L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^{i-1}_X(L_1)) \sqcup\!\sqcup_T L_1 \\ &\subseteq (L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^*_X(L_1)) \sqcup\!\sqcup_T L_1 + L_2.\end{aligned}$$

Thus, $L$ is a solution to (8.20). We now show that it is the least solution to this equation. Let $X_0$ be the least solution to (8.20). We show that $X_0 \supseteq L$. First, we note that

$$X_0 = X_0 \sqcup\!\sqcup_T L_1 + L_2 \supseteq L_2 = L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^0_X(L_1).$$

Let $i \geq 0$. Assume that $X_0 \supseteq L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^i_X(L_1)$. Then we have that

$$\begin{aligned}X_0 \supseteq X_0 \sqcup\!\sqcup_T L_1 &\supseteq (L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^i_X(L_1)) \sqcup\!\sqcup_T L_1 \\ &\supseteq L_2 \sqcup\!\sqcup_T ((\sqcup\!\sqcup_T)^i_X(L_1) \sqcup\!\sqcup_T L_1) \\ &\supseteq L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^{i+1}_X(L_1).\end{aligned}$$

Thus, $X_0 \supseteq L_2 \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^*_X(L_1) = L$. This completes the proof. ■

We can also give the following symmetric result.

**Theorem 8.11.2** *Let $T \subseteq \{0,1\}^*$ be right-preserving and associative. Let $L_1, L_2$ be arbitrary languages. Then the least solution to the equation*

$$X = L_1 \sqcup_T X + L_2, \tag{8.23}$$

*is the language $(\sqcup_T)^*(L_1) \sqcup_T L_2$.*

Further results in this area are likely. For instance, Salomaa considers systems of equations of the form

$$X_i = \sum_{j=1}^{n} X_j L_{j,i} + R_i$$

for $1 \leq i \leq n$, and investigates their solutions [173, Ch. 3, Sect. 2]. Systems of this form with $X_j \sqcup_T L_{j,i}$ are clear generalizations (for a single fixed $T \subseteq \{0,1\}^*$), and results can likely be obtained in the same manner as Theorems 8.11.1 and 8.11.2.

### 8.11.2 A Language Equation for $(\sqcup_T)^+(L)$

Due to the conditions placed on $T$ in the previous section, the following question seems natural: given arbitrary $T$ and $L$, can we find a language equation for which $(\sqcup_T)^+(L)$ is the minimal solution? We give this equation in the following theorem:

**Theorem 8.11.3** *Let $T \subseteq \{0,1\}^*$. For any language $L \subseteq \Sigma^*$, the least solution to the equation*

$$X = X \sqcup_T X + L \tag{8.24}$$

*is $X = (\sqcup_T)^+(L)$.*

**Proof.** We first show that $(\sqcup_T)^+(L)$ is a solution of (8.24). Consider that

$$\begin{aligned} (\sqcup_T)^+(L) \sqcup_T (\sqcup_T)^+(L) + L &\subseteq (\sqcup_T)^+(L) + L \\ &\subseteq (\sqcup_T)^+(L). \end{aligned}$$

The last inclusion is by definition of $(\sqcup_T)^+(L)$ and the first is by Proposition 8.6.2 and Theorem 8.6.5. To prove the reverse inclusion, let $x \in (\sqcup_T)^+(L)$. Then there exists $i \geq 1$ such that $x \in$

$(\sqcup\!\sqcup_T)^i(L)$. If $i = 1$, then $x \in L \subseteq (\sqcup\!\sqcup_T)^+(L) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^+(L) + L$, and the inclusion holds. Assume then that $i > 1$ and for all $y \in (\sqcup\!\sqcup_T)^j(L)$ with $j < i$, $y \in (\sqcup\!\sqcup_T)^+(L) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^+(L) + L$. Let $x \in (\sqcup\!\sqcup_T)^i(L)$. Then by definition, $x \in (\sqcup\!\sqcup_T)^{i-1}(L)$ or $x \in (\sqcup\!\sqcup_T)^{i-1}(L) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^{i-1}(L)$. In the first case, the result holds by induction. Otherwise,

$$\begin{aligned} x &\in (\sqcup\!\sqcup_T)^{i-1}(L) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^{i-1}(L) \\ &\subseteq ((\sqcup\!\sqcup_T)^+(L) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^+(L) + L) \sqcup\!\sqcup_T ((\sqcup\!\sqcup_T)^+(L) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^+(L) + L) \\ &\subseteq ((\sqcup\!\sqcup_T)^+(L) + L) \sqcup\!\sqcup_T ((\sqcup\!\sqcup_T)^+(L) + L) \\ &\subseteq (\sqcup\!\sqcup_T)^+(L) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^+(L) + L. \end{aligned}$$

Here, the first inclusion is by induction, the second is by Proposition 8.6.2 and Theorem 8.6.5, and the third is by the distributivity of $(\sqcup\!\sqcup_T)^+$ over union. Thus,

$$(\sqcup\!\sqcup_T)^+(L) = (\sqcup\!\sqcup_T)^+(L) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^+(L) + L.$$

Let $X_0 \subseteq \Sigma^*$ be the least solution of (8.24). As $(\sqcup\!\sqcup_T)^+(L)$ is a solution of (8.24), $X_0 \subseteq (\sqcup\!\sqcup_T)^+(L)$. It remains to show the reverse inclusion, which is again accomplished by induction, by showing that $(\sqcup\!\sqcup_T)^i(L) \subseteq X_0$ for all $i \geq 1$.

For $i = 1$, $L \subseteq L + X_0 \sqcup\!\sqcup_T X_0 = X_0$. Let $i > 1$ and assume that $(\sqcup\!\sqcup_T)^{i-1}(L) \subseteq X_0$. Then note that

$$\begin{aligned} X_0 &= X_0 \sqcup\!\sqcup_T X_0 + L \\ &\supseteq X_0 \sqcup\!\sqcup_T X_0 \\ &\supseteq (\sqcup\!\sqcup_T)^{i-1}(L) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^{i-1}(L). \end{aligned}$$

Thus $X_0 \supseteq (\sqcup\!\sqcup_T)^{i-1}(L)$ and $X_0 \supseteq (\sqcup\!\sqcup_T)^{i-1}(L) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^{i-1}(L)$, i.e., $X_0 \supseteq (\sqcup\!\sqcup_T)^{i-1}(L) + (\sqcup\!\sqcup_T)^{i-1}(L) \sqcup\!\sqcup_T (\sqcup\!\sqcup_T)^{i-1}(L) = (\sqcup\!\sqcup_T)^i(L)$. This establishes the inclusion. Thus, $X_0 = (\sqcup\!\sqcup_T)^+(L)$, establishing the result. ■

## 8.12 Conclusions

In this chapter, we have considered the iteration of shuffle on trajectory and deletion on trajectory operations. Our work generalizes previous work by Ito *et al.* [78, 79] and Kari and Thierrin [115] on particular shuffle and deletion along trajectories operations. Our work is also similar to that of Hsiao *et al.* [69] on iteration of arbitrary word operations. However, by considering a more general definition of iterated shuffle on trajectories, we are able to overcome some of the conditions imposed by Hsiao *et al.* in their study of iterated binary word operations.

After considering iterated shuffle and deletion along trajectories and the closure of languages under shuffle and deletion along trajectories, we have investigated the notions of shuffle base, extended shuffle base, primitivity and the Lyndon-Schützenberger Theorems, all considered in trajectory-based contexts. These notions generalize well to shuffle and deletion along trajectories, and many key concepts hold. We note that in the context of investigating the Lyndon-Schützenberger Theorems, we raise the problem of when $\sqcup\!\sqcup_T$ defines a free operation. This fundamental problem remains open.

Finally, we have returned to the topic of language equations. For explicit equations, the problem of decidability of the existence of solutions becomes trivial, and we have instead focused on characterizing least solutions to language equations. We have found explicit least solutions for two key explicit language equations involving $\sqcup\!\sqcup_T$. One generalizes a well-known language equation for Kleene closure which has been studied for fifty years. The second is a general language equation formulated so that its least solution is $(\sqcup\!\sqcup_T)^+(L)$.

# Chapter 9

# Conclusions

## 9.1 Results and Focus

In this thesis, we have examined word operations defined by trajectories, and related problems. Our focus is on four areas: descriptional complexity, codes, language equations and iterated operations.

Chapter 4 considered the problem of state complexity for shuffle on trajectories. As shuffle on trajectories defines a very general family of operations on languages, examining the state complexity of shuffle on trajectories is a challenging problem. The work in Chapter 4 is the first research in state complexity which does not examine a particular operation, but a class of language operations. We have seen that the density of the set of trajectories proves useful for characterizing the state complexity of the resulting shuffle on trajectories operations. We have given upper and lower bounds on the state complexity of shuffle on trajectories for sets of trajectories which are slender, i.e., sets of trajectories which only contain a constant number of trajectories for any fixed length. In this case, we see that there is a substantial advantage over the case where the number of trajectories of any given length is not restricted.

In Chapter 5, we have introduced the operation of deletion along trajectories. Several natural deletion-like operations are particular cases of deletion along trajectories, including quotient,

scattered deletion and sequential deletion. The most crucial theoretical aspect of deletion along trajectories is that it serves as an inverse to shuffle on trajectories. The closure properties of deletion along trajectories are different from those of shuffle on trajectories, and we have investigated these properties. The most interesting property is the similarity between the closure properties of deletion along trajectories and proportional removals. This similarity yields many non-regular sets of trajectories for which the associated deletion on trajectories operation preserves regularity.

Chapter 6 investigates classes of languages defined by shuffle on trajectories, and their relation to traditional classes of codes. This investigation has given much insight into the nature of code classes by shifting the focus from classes of languages to the language-theoretic properties of the associated sets of trajectories. We have addressed many natural lines of research in the theory of codes, in particular, questions of maximality, embedding of codes, finiteness properties and the relationship between convexity of languages and code-theoretic properties. While other general mechanisms for defining classes of code-like languages have been presented in the literature, we have found that shuffle on trajectories attains a desirable balance between expressive power on the one hand and the ability to obtain interesting results on the other. In particular, decidability results are obtained through the known closure properties of shuffle and deletion along trajectories.

Our consideration of these classes of codes in Chapter 6 has also led naturally to the definition of a binary relation on the set of all finite words for each set of trajectories. We have investigated algebraic properties of these binary relations; of particular interest is whether a set of trajectories defines a transitive binary relation. Decidability of these properties of the binary relation defined by a set of trajectories are also considered.

As deletion along trajectories constitutes an inverse to shuffle on trajectories, we can therefore investigate language equations involving both shuffle and deletion along trajectories. This is the focus of Chapter 7. We have investigated several different equation forms, and have obtained both positive and negative decidability results for the problem of determining whether an equation possesses a solution. We have also considered systems of equations with shuffle on trajectories, an important step in the investigation of language equations involving shuffle on trajectories.

Finally, Chapter 8 has investigated the questions raised by considering iterated versions of shuffle and deletion along trajectories operations. With a very natural definition of iterated shuffle on trajectories, denoted $(\sqcup\!\sqcup_T)^+(L)$, we can give, for all languages $L$ and all sets of trajectories $T \subseteq \{0,1\}^*$ (resp., all $T \subseteq \{i,d\}^*$ with $T \supseteq i^*$), a characterization of the smallest language containing $L$ and closed under $\sqcup\!\sqcup_T$ (resp., $\rightsquigarrow_T$). We have also studied explicit language equations, i.e., language equations of the form $X = \alpha$ where $X$ is a variable and $\alpha$ is an expression involving $\sqcup\!\sqcup_T$. This study is a fundamental first step towards developing grammar systems involving shuffle on trajectories. In our study, we focus on characterization results for elementary explicit language equations involving shuffle on trajectories. For all $T \subseteq \{0,1\}^*$, we find that the equation $X = X \sqcup\!\sqcup_T X + L$ has least solution $(\sqcup\!\sqcup_T)^+(L)$.

## 9.2 Open Problems

In this section, we survey some of the more interesting open problems we have considered in this thesis. This is not a complete description of all open problems in this thesis.

Chapter 4 investigated the descriptional complexity of shuffle on trajectories. We concluded with the following conjecture:

**Conjecture 9.2.1** *For all $T \subseteq \{0,1\}^*$ with $p_T(n) \in \Omega(2^n)$, there exists $L_1, L_2$ such that $sc(L_1 \sqcup\!\sqcup_T L_2) = 2^{\Omega(sc(l_1)sc(L_2))}$.*

In chapter 6, we investigated $T$-codes, which are a natural generalization of many classes of codes. From a theoretical standpoint, the most interesting open problem is characterizing those $T$ for which all $T$-codes are finite:

**Open Problem 9.2.2** *What are necessary and sufficient conditions on a set $T \subseteq \{0,1\}^*$ of trajectories such that $\mathcal{P}_T(\Sigma) \subseteq$ FIN?*

In particular, a characterization for Open Problem 9.2.2 which solves the following problem is of the most interest:

**Open Problem 9.2.3** *Given a (regular, context-free) set $T \subseteq \{0,1\}^*$ of trajectories, is it decidable whether $\mathcal{P}_T(\Sigma) \subseteq \text{FIN}$?*

One fundamental property of sets of trajectories which we have not resolved is freeness, which we discussed in Section 8.10.2.

**Open Problem 9.2.4** *Let $T \subseteq \{0,1\}^*$ be a regular (or context-free) set of trajectories. Is it decidable whether the semigroup $M = (\Sigma^*, \sqcup\!\sqcup_T)$ is a free semigroup?*

There are several open problems concerning iteration of shuffle and deletion on trajectories. The following problem is easily stated, but a solution does not seem obvious:

**Open Problem 9.2.5** *For all $R \in \text{REG}$, does there exist $k \geq 1$ such that $(\rightsquigarrow)^+(R) \in \text{CF}_k$?*

## 9.3 Further Research Directions

In this section, we examine some research directions which we hope to explore in the future.

### 9.3.1 Confluence of $\omega_T$

Given a transitive, reflexive binary relation $\rho$ on $\Sigma^*$, we say that a language $L \subseteq \Sigma^*$ is *confluent with respect to $\rho$* if, for all $x, y \in L$, there exists $z \in L$ such that $x \,\rho\, z$ and $y \,\rho\, z$.

Ilie [73] has investigated the confluence property for arbitrary binary relations, and examined decidability problems for specific relations, including the prefix, suffix and factor orders. Investigating the decidability of the confluence problem for arbitrary $\omega_T$ remains an interesting research question.

### 9.3.2 Codes Defined by Multiple Sets of Trajectories

Let $n \geq 1$ and $T_i \subseteq \{0,1\}^*$ for $1 \leq i \leq n$. Define $\mathbf{T} = \{T_1, \ldots, T_n\}$. Consider the relation $\omega_{\mathbf{T}}$ on $\Sigma^*$ defined by

$$x \; \omega_{\mathbf{T}} \; y \iff \bigwedge_{i=1}^{n} x \; \omega_{T_i} \; y$$

for all $x, y \in \Sigma^*$. Define $\mathcal{P}_{\mathbf{T}}(\Sigma)$ as the set of all anti-chains under $\omega_{\mathbf{T}}$. There has been interest in $\mathcal{P}_{\mathbf{T}_{\text{ps}}}(\Sigma)$ for $\mathbf{T}_{\text{ps}} = \{0^*1^*, 1^*0^*\}$, see Jürgensen and Konstantinidis [97, pp. 550–551] for references and a discussion of this class.

Further, we can define a related class $\mathcal{Q}^m_{\mathbf{T}}(\Sigma)$ as follows: $L \in \mathcal{Q}^m_{\mathbf{T}}(\Sigma)$ if and only if for all $L' \subseteq L$ with $|L'| \leq m$, $L' \in \cup_{i=1}^{n} \mathcal{P}_{T_i}(\Sigma)$. For $\mathcal{Q}^m_{\mathbf{T}}(\Sigma)$, other than $\mathbf{T}_{\text{ps}}$, the classes $\mathbf{T}_{\text{io}} = \{0^*1^*0^*, 1^*0^*1^*\}$ [139], as well as $\mathbf{T}_{k-\text{io}} = \{(1^*0^*)^k1^*, (0^*1^*)^k0^*\}$ and $\mathbf{T}_{k-\text{ps}} = \{(0^*1^*)^k, (1^*0^*)^k\}$ for $k \geq 1$ (see Long *et al.* [138, Defns. 5 and 6] or Long [137]) have received attention.

It is easily established that $\mathcal{Q}^2_{\mathbf{T}}(\Sigma) = \mathcal{P}_{\mathbf{T}}(\Sigma)$ and that $\mathcal{Q}^{n+i}_{\mathbf{T}}(\Sigma) = \mathcal{Q}^n_{\mathbf{T}}(\Sigma)$ for all $\mathbf{T}$ with $|\mathbf{T}| = n$ and for all $i \geq 0$. However, other problems related to these classes of languages do not seem to be so easy. For instance, the decidability of membership in $\mathcal{P}_{\mathbf{T}_{\text{ps}}}(\Sigma)$ (see Ito *et al.* [76] or Jürgensen *et al.* [98]) relies intrinsically on the nature of the members of $\mathbf{T}_{\text{ps}}$. The corresponding problem for $\mathbf{T}_{\text{io}}$ also relies on the nature of the sets of trajectories involved [38]. It appears to be a very challenging problem to determine the decidability of membership in $\mathcal{P}_{\mathbf{T}}(\Sigma)$ for arbitrary $\mathbf{T} = \{T_1, \ldots, T_n\}$, where each $T_i$ is regular. Kari *et al.* [108, Thm. 4.7] have solved a similar decision problem for two sets of trajectories in their framework of *bond-free property*. However, their approach does not seem to be adaptable to our situation.

### 9.3.3 Semantic Trajectory-Based Operations

By slightly expanding the notion of a trajectory, we find that trajectory-based operations have several interesting applications. Emerging uses of word operations, especially in modeling operations on strands of DNA, show promise for using shuffle and deletion on trajectories or related trajectory-based variants. Work has already been conducted in this area by Mateescu [144] and more recently

by Kari *et al.* [108]. Both of these works focus on using the shuffle on trajectories model to operations on DNA.

However, there is another approach to adapting the trajectory-based framework to model situations such as operations on DNA strands. This approach considers what are known as *semantic* operations. In the paper which introduced shuffle on trajectories, Mateescu *et al.* make the following distinction between *syntactic* and *semantic* operations on words:

> We introduce and investigate new methods [shuffle on trajectories] to define parallel composition of words and languages. These methods are based on syntactic constraints on the shuffle operations. The constraints are referred to as syntactic constraints since they do not concern properties of the words that are shuffled, or properties of the letters that occur in these words.
> Instead, the constraints involve the general strategy to switch from one word to another word. Once such a strategy is defined, the structure of the words that are shuffled does not play any role.
> However, constraints that take into consideration the inner structure of the words that are shuffled together are referred to as semantic constraints. [147, p. 2]

With the current focus on operations on DNA, we see semantic operations not as a clumsy sibling to syntactic operations, but a challenging area of investigation. We feel that a suitable extension of the shuffle and deletion along trajectories model would provide much insight into the nature of semantic operations. A reasonable model exists which shares many similarities to the current shuffle on trajectories model, but adds sufficient semantic power to encompass many interesting semantic operations, including those investigated by several authors, including Daley *et al.* [29, 30, 31], Kari and Thierrin [116], Kari [107], Mateescu and Salomaa [149], Kudlek and Mateescu [127, 126], de Simone [34], Chen *et al.* [23] and many others; Mateescu *et al.* [146] summarize some of these semantic operations. We also note the possibility of extending the semantic framework proposed by Abdelwahed [1, 2], which is a model of combining processes in control of discrete event systems. Abdelwahed notes that, among others, the concept of shuffle on trajectories is "influential to the modelling paradigm [1, p. 8]" (called Interacting Discrete Event Systems–IDES) developed in the thesis.

It remains to be seen which results can be adapted from the syntactic case to the semantic case.

It is clear, however, that generalizations of some results–in particular, the equations considered by Daley *et al.* [29, 30]–will yield interesting results relating to bio-informatics.

# Bibliography


[1] ABDELWAHED, S. *Interacting Discrete Event Systems: Modelling, Verification and Supervisory Control*. PhD thesis, University of Toronto, 2002.

[2] ABDELWAHED, S., AND WONHAM, W. Interacting DES: Modelling and analysis. In *41st IEEE Conference on Decision and Control* (2002), pp. 1175–1180.

[3] ALLOUCHE, J.-P., AND SHALLIT, J. *Automatic Sequences: Theory, Applications, Generalizations*. Cambridge University Press, 2003.

[4] AMAR, V., AND PUTZOLU, G. On a family of linear grammars. *Inf. and Cont. 7* (1964), 283–291.

[5] AMAR, V., AND PUTZOLU, G. Generalizations of regular events. *Inf. and Cont. 8* (1965), 56–63.

[6] ARAKI, T., KAGIMASA, T., AND TOKURA, N. Relations of flow languages to Petri net languages. *Theor. Comp. Sci. 15* (1981), 51–75.

[7] ARAKI, T., AND TOKURA, N. Decision problems for regular expressions with shuffle and shuffle closure operators. *Systems, Computers, Controls 12*, 6 (1981), 46–50.

[8] ARAKI, T., TOKURA, N., AND KOSAI, S. Shuffle grammars. *Systems, Computers, Controls 14*, 4 (1983), 37–45.

[9] ARDEN, D. *Delayed logic and finite state machines*. University of Michigan Press, 1960, pp. 1–35.

[10] AUTEBERT, J.-M., BERSTEL, J., AND BOASSON, L. Context-free languages and pushdown automata. pp. 111–174. In [171].

[11] BAADER, F., AND KÜSTERS, R. Solving linear equations over regular languages. In *UNIF 2001: 15th International Workshop on Unification* (2001), F. Baader, V. Diekert, C. Tinelli, and R. Treinen, Eds., pp. 27–31.

[12] BEL-ENGUIX, G., MARTÍN-VIDE, C., AND MATEESCU, A. Dialog and splicing on routes. *Romanian Journal of Information Science and Technology 6*, 1-2 (2003), 45–59.

[13] BAADER, F., AND NARENDRAN, P. Unification of concept terms in descriptional logics. *J. Symb. Comput. 31* (2001), 277–305.

[14] BALCÁZAR, J., DIAZ, J., AND GABARRÓ, J. *Structural Complexity II.* Springer, 1990.

[15] BARIL, J.-L., AND VAJNOVSZKI, V. Gray code for derangements. *Disc. Appl. Math. 140* (2004), 207–221.

[16] BERARD, B. Literal shuffle. *Theor. Comp. Sci. 51* (1987), 281–299.

[17] BERSTEL, J., BOASSON, L., CARTON, O., PETAZZONI, B., AND PIN, J.-E. Operations preserving recognizable languages. In *Fundamentals of Computation Theory (FCT 2003)* (2003), A. Lingas and B. Nilsson, Eds., vol. 2751 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 343–354.

[18] BERSTEL, J., AND PERRIN, D. *Theory of Codes*. Available at http://www-igm.univ-mlv.fr/%7Eberstel/LivreCodes/Codes.html, 1996.

[19] BRUYÈRE, V., AND PERRIN, D. Maximal bifix codes. *Theor. Comp. Sci. 218* (1999), 107–121.

[20] BRZOZOWSKI, J. Roots of star events. *J. ACM 14*, 3 (1967), 466–477.

[21] CÂMPEANU, C., SALOMAA, K., AND VÁGVÖLGYI, S. Shuffle decompositions of regular languages. *Int. J. Found. Comp. Sci. 13*, 6 (2002), 799–816.

[22] CÂMPEANU, C., SALOMAA, K., AND YU, S. Tight lower bound for the state complexity of shuffle of regular languages. *J. Automata, Languages and Combinatorics 7*, 3 (2002), 303–310.

[23] CHEN, K., FOX, R., AND LYNDON, R. Free differential calculus IV: The quotient groups of lower central series. *Ann. Math., 2nd Ser. 68*, 1 (1958), 81–95.

[24] CHOFFRUT, C., AND KARHUMÄKI, J. Combinatorics on words. pp. 329–438. In [171].

[25] CHOFFRUT, C., AND KARHUMÄKI, J. On Fatou properties of rational languages. In *Where Mathematics, Computer Science, Linguistics and Biology Meet* (2000), C. Martin-Vide and V. Mitrana, Eds., Kluwer, pp. 227–235.

[26] CLERBOUT, M., ROOS, Y., AND RYL, I. Synchronization languages. *Theor. Comp. Sci. 215* (1999), 99–121.

[27] CLERBOUT, M., ROOS, Y., AND RYL, I. Synchronization languages and rewriting systems. *Inf. and Comp. 167*, 1 (2001), 46–69.

[28] CONWAY, J. *Regular Algebra and Finite Machines*. Chapman and Hall, 1971.

[29] DALEY, M. *Computational Modeling of Genetic Processes in Stichotrichous Ciliates*. PhD thesis, University of Western Ontario, 2003.

[30] DALEY, M., IBARRA, O., AND KARI, L. Closure properties and decision questions of some language classes under ciliate bio-operations. *Theor. Comp. Sci. 306*, 1 (2003), 19–38.

[31] DALEY, M., KARI, L., AND MCQUILLAN, I. Families of languages defined by ciliate bio-operations. *Theor. Comp. Sci. 320*, 1 (2004), 51–69.

[32] DASSOW, J., MITRANA, V., AND SALOMAA, A. Operations and language generating devices suggested by the genome evolution. *Theor. Comp. Sci. 270* (2002), 701–738.

[33] DE LUCA, A., AND VARRICCHIO, S. Regularity and finiteness conditions. pp. 747–810. In [171].

[34] DE SIMONE, R. Langages infinitaires et produits de mixage. *Theor. Comp. Sci. 31* (1984), 83–100.

[35] DOMARATZKI, M. State complexity of proportional removals. *J. Automata, Languages and Combinatorics 7*, 4 (2002), 455–468.

[36] DOMARATZKI, M. On iterated scattered deletion. *Bull. Eur. Assoc. Theor. Comp. Sci. 80* (2003), 159–161.

[37] DOMARATZKI, M. Deletion along trajectories. *Theor. Comp. Sci. 320*, 2–3 (2004), 293–313.

[38] DOMARATZKI, M. On the decidability of 2-infix-outfix codes. Tech. Rep. 2004-479, School of Computing, Queen's University, 2004.

[39] DOMARATZKI, M. Trajectory-based codes. *Acta Inf. 40*, 6–7 (2004), 491–527.

[40] DOMARATZKI, M. Trajectory-based embedding relations. *Fund. Inf. 59*, 4 (2004), 349–363.

[41] DOMARATZKI, M., MATEESCU, A., SALOMAA, K., AND YU, S. Deletion along trajectories and commutative closure. In *Proceedings of WORDS'03: 4th International Conference on Combinatorics on Words* (2003), T. Harju and J. Karhumäki, Eds., pp. 309–319.

[42] DOMARATZKI, M., AND OKHOTIN, A. Representing recursively enumerable languages by iterated deletion. *Theor. Comp. Sci. 314*, 3 (2004), 451–457.

[43] DOMARATZKI, M., AND SALOMAA, K. State complexity of shuffle on trajectories. In *Descriptional Complexity of Formal Systems (DCFS)* (2002), J. Dassow, M. Hoeberechts, H. Jürgensen, and D. Wotschke, Eds., pp. 95–109.

[44] DOMARATZKI, M., AND SALOMAA, K. Decidability of trajectory-based equations. To appear in *Mathematical Foundations of Computer Science 2004* (2004).

[45] DOMARATZKI, M., AND SALOMAA, K. Restricted sets of trajectories and decidability of shuffle decompositions. In *Descriptional Complexity of Formal Systems (DCFS 2004)* (2004), L. Ilie and D. Wotschke, Eds., pp. 37–51.

[46] DOMARATZKI, M., AND SALOMAA, K. State complexity of shuffle on trajectories. *Accepted, J. Automata, Languages and Combinatorics 9*, 2 (2004).

[47] EHRENFEUCHT, A., HAUSSLER, D., AND ROZENBERG, G. On regularity of context-free languages. *Theor. Comp. Sci. 23* (1983), 311–332.

[48] ELLUL, K. Descriptional complexity measures of regular languages. M.Math thesis, University of Waterloo, 2002.

[49] FLAJOLET, P., AND STEYAERT, J.-M. On sets having only hard subsets. In *Automata Languages and Programming* (1974), J. Loeckx, Ed., vol. 14 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 446–457.

[50] GINSBURG, S. *The Mathematical Theory of Context-Free Languages*. McGraw-Hill, 1966.

[51] GINSBURG, S. *Algebraic and Automata-Theoretic Properties of Formal Languages*. North-Holland, 1975.

[52] GINSBURG, S., AND SPANIER, E. Quotients of context-free languages. *J. ACM 10*, 4 (1963), 487–492.

[53] GINSBURG, S., AND SPANIER, E. Mappings of languages by two-tape devices. *J. ACM 12*, 3 (1965), 424–434.

[54] GINSBURG, S., AND SPANIER, E. H. Bounded regular sets. *Proc. Amer. Math. Soc. 17* (1966), 1043–1049.

[55] GISCHER, J. Shuffle language, Petri nets and context-sensitive grammars. *Comm. ACM 24*, 9 (1981), 597–605.

[56] GUO, L., SALOMAA, K., AND YU, S. Synchronization expressions and languages. In *Proc. 6th IEEE Symposium on Parallel and Distributed Processing* (1994), IEEE Computer Society Press, pp. 257–264.

[57] GUO, Y., SHYR, H., AND THIERRIN, G. E-Convex infix codes. *Order 3* (1986), 55–59.

[58] HAINES, L. On free monoids partially ordered by embedding. *J. Comb. Th. 6* (1969), 94–98.

[59] HARJU, T., AND ILIE, L. On quasi orders of words and the confluence property. *Theor. Comp. Sci. 200* (1998), 205–224.

[60] HARJU, T., AND KARHUMÄKI, J. Morphisms. In *Handbook of Formal Languages, Vol. I* (1997), pp. 439–510.

[61] HARJU, T., MATEESCU, A., AND SALOMAA, A. Shuffle on trajectories: The Schützenberger product and related operations. In *Mathematical Foundations of Computer Science 1998* (1998), L. Brim, J. Gruska, and J. Zlatuska, Eds., no. 1450 in Lecture Notes in Computer Science, Springer-Verlag, pp. 503–511.

[62] HARRISON, M. *Introduction to Formal Language Theory*. Addison-Wesley, 1978.

[63] HAUSLER, D., AND ZEIGER, H. Very special languages and representations of recursively enumerable languages via computation histories. *Inf. and Cont. 47* (1980), 201–212.

[64] HIGMAN, G. Ordering by divisibility in abstract algebras. *Proc. London Math. Soc. 2*, 3 (1952), 326–336.

[65] HOLZER, M., AND KUTRIB, M. State complexity of basic operations on nondeterministic finite automata. In *Implementation and Application of Automata: 7th International Conference, CIAA 2002* (2003), J.-M. Champarnaud and D. Maurel, Eds., vol. 2608 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 148–157.

[66] HOLZER, M., AND KUTRIB, M. Unary language operations and their nondeterministic state complexity. In *Developments in Language Theory: Sixth International Conference, DLT 2002* (2003), M. Ito and M. Toyama, Eds., vol. 2450 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 162–172.

[67] HOLZER, M., AND LANGE, K.-J. On the complexity of iterated insertions. In *New Trends in Formal Languages* (1997), G. Paun and A. Salomaa, Eds., vol. 1218 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 440–453.

[68] HOPCROFT, J. E., AND ULLMAN, J. D. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.

[69] HSIAO, H., HUANG, C., AND YU, S.-S. Word operation closure and primitivity of languages. *J. Universal Comp. Sci. 8*, 2 (2002), 243–256.

[70] HUNT, H., AND ROSENKRANTZ, D. Computational parallels between the regular and context-free languages. *SIAM J. Comput. 7* (1978), 99–114.

[71] IGARASHI, A., AND KOBAYASHI, N. Resource usage analysis. *ACM SIGPLAN Notices 37*, 1 (2002), 331–342.

[72] ILIE, L. Remarks on well quasi orders of words. In *Proceedings of the 3rd DLT* (1997), S. Bozapalidis, Ed., pp. 399–411.

[73] ILIE, L. *Decision Problems on Orders of Words*. PhD thesis, University of Turku, 1998.

[74] IMREH, B., ITO, M., AND KATSURA, M. On shuffle closures of commutative regular languages. In *Combinatorics, Complexity, & Logic (Auckland, 1996)* (1997), D. Bridges, C. Calude, I. Gibbons, S. Reeves, and I. Witten, Eds., Springer, pp. 276–288.

[75] ITO, M. Shuffle decomposition of regular languages. *J. Universal Comp. Sci. 8*, 2 (2002), 257–259.

[76] ITO, M., JÜRGENSEN, H., SHYR, H., AND THIERRIN, G. $n$-prefix-suffix languages. *Intl. J. Comp. Math. 30* (1989), 37–56.

[77] ITO, M., JÜRGENSEN, H., SHYR, H., AND THIERRIN, G. Outfix and infix codes and related classes of languages. *J. Comp. Sys. Sci. 43* (1991), 484–508.

[78] ITO, M., KARI, L., AND THIERRIN, G. Insertion and deletion closure of languages. *Theor. Comp. Sci. 183* (1997), 3–19.

[79] ITO, M., KARI, L., AND THIERRIN, G. Shuffle and scattered deletion closure of languages. *Theor. Comp. Sci. 245* (2000), 115–133.

[80] ITO, M., AND SILVA, P. Remark on deletions, scattered deletions and related operations on languages. In *Semigroups and Applications* (1998), J. Howie and N. Ruskuc, Eds., World Scientific, pp. 97–105.

[81] ITO, M., AND TANAKA, G. Dense property of initial literal shuffles. *Intl. J. Comp. Math. 34* (1990), 161–170.

[82] ITO, M., THIERRIN, G., AND YU, S.-S. Shuffle-closed languages. *Publ. Math. Debrecen 48*, 3–4 (1996), 317–338.

[83] ITO, T., AND NISHITANI, Y. On universality of concurrent expressions with synchronization primitives. *Theor. Comp. Sci. 19* (1982), 105–115.

[84] IWAMA, K. Unique decomposability of shuffled strings. In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing* (1983), D. Johnson *et al.*, Ed., pp. 374–381.

[85] JANTZEN, M. The power of synchronizing operations on strings. *Theor. Comp. Sci. 14* (1981), 127–154.

[86] JANTZEN, M. Extending regular expressions with iterated shuffle. *Theor. Comp. Sci. 38* (1985), 223–247.

[87] JĘDRZEJOWICZ, J. On the enlargement of the class of regular languages by the shuffle closure. *Inf. Proc. Letters 16* (1983), 51–54.

[88] JĘDRZEJOWICZ, J. Nesting of shuffle closure is important. *Inf. Proc. Letters 25* (1987), 363–367.

[89] JĘDRZEJOWICZ, J. Infinite hierarchy of expressions containing shuffle closure operator. *Inf. Proc. Letters 28*, 1 (1988), 33–37.

[90] JĘDRZEJOWICZ, J. Infinite hierarchy of shuffle expressions over a finite alphabet. *Inf. Proc. Letters 36*, 1 (1990), 13–17.

[91] JĘDRZEJOWICZ, J. Undecidability results for shuffle languages. *J. Automata, Languages and Combinatorics 1*, 2 (1997), 147–159.

[92] JĘDRZEJOWICZ, J. Structural properties of shuffle automata. *Grammars 2* (1999), 35–51.

[93] JĘDRZEJOWICZ, J., AND SZEPIETOWSKI, A. Shuffle languages are in P. *Theor. Comp. Sci. 250* (2001), 31–53.

[94] JIRÁSEK, J., JIRÁSKOVÁ, G., AND SZABARI, A. State Complexity of Concatenation and Complementation of Regular Languages. In *Pre-proceedings of CIAA 2004: Ninth International Conference on Implementations and Applications of Automata* (2004), M. Domaratzki, A. Okhotin, K. Salomaa and S. Yu, Eds., pp. 132–142.

[95] JIRÁSKOVÁ, G. State complexity of some operations on regular languages. In *Descriptional Complexity of Formal Systems: Fifth International Workshop* (2003), E. Csuhaj-Varjú, C. Kintala, D. Wotschke, and G. Vaszil, Eds., pp. 114–125.

[96] JULLIEN, P. Sur un théorème d'extension dans la théorie des mots. *CR Acad. Sc. Paris (Série A) 266* (1968), 851–854.

[97] JÜRGENSEN, H., AND KONSTANTINIDIS, S. Codes. pp. 511–600. In [171].

[98] JÜRGENSEN, H., SALOMAA, K., AND YU, S. Transducers and the decidability of independence in free monoids. *Theor. Comp. Sci. 134* (1994), 107–117.

[99] JÜRGENSEN, H., SHYR, H., AND THIERRIN, G. Codes and compatible partial orders on free monoids. In *Algebra and Order: Proceedings of the First International Symposium on Ordered Algebraic Structures, Luminy–Marseilles 1984* (1986), S. Wolfenstein, Ed., Heldermann Verlag, pp. 323–334.

[100] JÜRGENSEN, H., AND YU, S. Relations on free monoids, their independent sets, and codes. *Int. J. Comput. Math. 40* (1991), 17–46.

[101] KADRIE, A., DARE, V., THOMAS, D., AND SUBRAMANIAN, K. Algebraic properties of the shuffle over $\omega$-trajectories. *Inf. Proc. Letters 80*, 3 (2001), 139–144.

[102] KARI, L. Insertion and deletion of words: Determinism and reversibility. In *Mathematical Foundations of Computer Science 1992* (1992), I. Havel and V. Koubek, Eds., vol. 629 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 315–326.

[103] KARI, L. Generalized derivatives. *Fund. Inf. 18* (1993), 27–39.

[104] KARI, L. Insertion operations: Closure properties. *Bull. Eur. Assoc. Theor. Comp. Sci. 51* (1993), 181–191.

[105] KARI, L. Deletion operations: Closure properties. *Intl. J. Comp. Math. 52* (1994), 23–42.

[106] KARI, L. On language equations with invertible operations. *Theor. Comp. Sci. 132* (1994), 129–150.

[107] KARI, L. Power of controlled insertion and deletion. In *Results and Trends in Theoretical Computer Science* (1994), J. Karhumäki, H. Maurer, and G. Rozenberg, Eds., vol. 812 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 197–212.

[108] KARI, L., KONSTANTINIDIS, S., AND SOSÍK, P. On properties of bond-free DNA languages. Tech. Rep. 609, Computer Science Department, University of Western Ontario, 2003. Submitted for publication.

[109] KARI, L., KONSTANTINIDIS, S., AND SOSÍK, P. Bond-free languages: Formalisms, maximality and construction methods. Tech. Rep. 2004–001, Saint Mary's University Department of Mathematics and Computer Science, 2004. To appear, DNA 10.

[110] KARI, L., KONSTANTINIDIS, S., AND SOSÍK, P. Substitutions, trajectories and noisy channels. In *Pre-proceedings of CIAA 2004: Ninth International Conference on Implementations and Applications of Automata* (2004), M. Domaratzki, A. Okhotin, K. Salomaa and S. Yu, Eds., pp. 154–162.

[111] KARI, L., MATEESCU, A., SALOMAA, A., AND PĂUN, G. Deletion sets. *Fund. Inf. 19* (1993), 355–370.

[112] KARI, L., AND SOSÍK, P. Language deletions on trajectories. Tech. Rep. 606, Computer Science Department, University of Western Ontario, 2003. Submitted for publication.

[113] KARI, L., AND SOSÍK, P. On language equations with deletion. *Bull. Eur. Assoc. Theor. Comp. Sci. 83* (2004), 173–180.

[114] KARI, L., AND THIERRIN, G. $k$-catenation and applications: $k$-prefix codes. *J. Inf. Opt. Sci. 16*, 2 (1995), 263–276.

[115] KARI, L., AND THIERRIN, G. $k$-insertion and $k$-deletion closure of languages. *Soochow J. Math 21*, 4 (1995), 479–495.

[116] KARI, L., AND THIERRIN, G. Contextual insertions/deletions and computability. *Inf. and Comp. 131* (1996), 47–61.

[117] KARI, L., AND THIERRIN, G. Maximal and minimal solutions to language equations. *J. Comp. Sys. Sci. 53* (1996), 487–496.

[118] KARI, L., AND THIERRIN, G. Word insertions and primitivity. *Util. Math. 53* (1998), 49–61.

[119] KLEENE, S. Representation of events in nerve nets and finite automata. In *Automata Studies* (1956), C. Shannon and J. McCarthy, Eds., vol. 34 of *Annals of Mathematics Studies*, Princeton University Press, pp. 3–41.

[120] KOSARAJU, S. Correction to "Regularity Preserving Functions". *ACM SIGACT News 6*, 3 (1974), 22.

[121] KOSARAJU, S. Regularity preserving functions. *ACM SIGACT News 6*, 2 (1974), 16–17.

[122] KOSARAJU, S. Context-free preserving functions. *Math. Sys. Theor. 9*, 3 (1975).

[123] KOZEN, D. On regularity-preserving functions. Tech. Rep. TR95-1559, Department of Computer Science, Cornell University, 1995.

[124] KRISHNAN, P. Automatic synthesis of a subclass of schedulers in timed systems. *Theor. Comp. Sci. 298*, 3 (2003), 347–363.

[125] KRUSKAL, J. The theory of well-quasi-ordering: A frequently discovered concept. *J. Comb. Th. (A) 13* (1972), 297–305.

[126] KUDLEK, M., AND MATEESCU, A. On distributed catenation. *Theor. Comp. Sci. 180* (1997), 341–352.

[127] KUDLEK, M., AND MATEESCU, A. On mix operation. In *New Trends in Formal Languages* (1997), G. Paun and A. Salomaa, Eds., vol. 1218 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 430–439.

[128] LAM, N. Finite maximal infix codes. *Semigroup Forum 61* (2000), 346–356.

[129] LATTA, M., AND WALL, R. Intersective context-free languages. In *Lenguajes Naturales y Lenguajes Formalex IX* (1993), C. Martin-Vide, Ed., pp. 15–43.

[130] LATTEUX, M., LEGUY, B., AND RATOANDROMANANA, B. The family of one-counter languages is closed under quotient. *Acta. Inf. 22* (1985), 579–588.

[131] LEISS, E. *Language Equations*. Monographs in Computer Science. Springer, 1999.

[132] LEVI, F. On semigroups. *Bull. Calcutta Math. Soc. 36* (1944), 141–146.

[133] LIU, L., AND WEINER, P. An infinite hierarchy of intersections of context-free languages. *Math. Sys. Th. 7*, 2 (1973), 185–192.

[134] LONG, D. On nilpotency of the syntactic monoid of a language. In *Words, Languages and Combinatorics II* (1992), M. Ito and H. Jürgensen, Eds., World Scientific, pp. 279–293.

[135] LONG, D. On two infinite hierarchies of prefix codes. In *Proceedings of the Conference on Ordered Structures and Algebra of Computer Languages* (1993), K. Shum and P. Yuen, Eds., World Scientific, pp. 81–90.

[136] LONG, D. $k$-bifix codes. *Rivista di Matematica Pura ed Applicata 15* (1994), 33–55.

[137] LONG, D. *Study of Coding Theory and its Application to Cryptography*. PhD thesis, City University of Hong Kong, 2002.

[138] LONG, D., JIA, W., MA, J., AND ZHOU, D. $k$-p-infix codes and semaphore codes. *Disc. Appl. Math. 109* (2001), 237–252.

[139] LONG, D., MA, J., AND ZHOU, D. Structure of 3-infix-outfix maximal codes. *Theor. Comp. Sci. 188* (1997), 231–240.

[140] LOTHAIRE, M. *Combinatorics on Words*. Addison-Wesley, 1983.

[141] MARTIN, J. *Introduction to Languages and the Theory of Computation (3rd ed.)*. McGraw-Hill, 2003.

[142] MARTIN-VIDE, C., MATEESCU, A., ROZENBERG, G., AND SALOMAA, A. Contexts on trajectories. *Intl. J. Comp. Math. 73*, 1 (1999), 15–36.

[143] MATEESCU, A. CD grammar systems and trajectories. *Acta. Cyb. 13*, 2 (1997), 141–157.

[144] MATEESCU, A. Splicing on routes: a framework of DNA computation. In *Unconventional Models of Computation* (1998), C. Calude, J. Casti, and M. Dinneen, Eds., Springer, pp. 273–285.

[145] MATEESCU, A., AND MATEESCU, G. Associative and fair shuffle of $\omega$-words. Tech. Rep. TUCS-TR-162, University of Turku, 1998.

[146] MATEESCU, A., ROZENBERG, G., AND SALOMAA, A. Syntactic and semantic aspects of parallelism. In *Foundations of Computer Science: Potential–Theory–Cognition* (1997), C. Freksa, M. Jantzen, and R. Valk, Eds., Lecture Notes in Computer Science, Springer-Verlag, pp. 79–106.

[147] MATEESCU, A., ROZENBERG, G., AND SALOMAA, A. Shuffle on trajectories: Syntactic constraints. *Theor. Comp. Sci. 197* (1998), 1–56.

[148] MATEESCU, A., AND SALOMAA, A. Aspects of classical language theory. pp. 175–246. In [171].

[149] MATEESCU, A., AND SALOMAA, A. Parallel composition of words with re-entrant symbols. *An. Univ. Bucureşti Mat. Inform. 45*, 1 (1996), 71–80.

[150] MATEESCU, A., SALOMAA, A., SALOMAA, K., AND YU, S. On an extension of the Parikh mapping. Tech. Rep. 364, TUCS, 2000.

[151] MATEESCU, A., SALOMAA, A., AND YU, S. Factorizations of languages and commutativity conditions. *Acta Cyb. 15*, 3 (2002), 339–351.

[152] MATEESCU, A., SALOMAA, K., AND YU, S. On fairness of many-dimensional trajectories. *J. Automata, Languages and Combinatorics 5* (2000), 145–157.

[153] MEDUNA, A. Middle quotients of linear languages. *Intl. J. Comp. Math. 71* (1999), 319–335.

[154] MORIYA, T., AND YAMASAKI, H. Literal shuffle on $\omega$-languages. *Inf. Proc. Letters 59* (1996), 165–168.

[155] NICAUD, C. Average state complexity of operations on unary automata. In *Proceedings of the 24th International Symposium on Mathematical Foundations of Computer Science (MFCS 1999)* (1999), M. Kutylowski, L. Pacholski, and T. Wierzbicki, Eds., vol. 1672 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 230–240.

[156] OGDEN, W., RIDDLE, W., AND ROUNDS, W. Complexity of expressions allowing concurrency. In *Conference Record of the Fifth Annual ACM Symposium on Principles of Programming Languages* (1978), A. Aho and S. Zilles, Eds., pp. 185–194.

[157] OKHOTIN, A. Conjunctive grammars. *J. Automata, Languages and Combinatorics 6*, 4 (2001), 519–535.

[158] OKHOTIN, A. Personal communication, September 2003.

[159] OKHOTIN, A. Boolean grammars. *To appear, Inf. and Comp.* (2004).

[160] OKHOTIN, A. On the equivalence of linear conjunctive grammars to trellis automata. *RAIRO Theor. Inf. and Appl. 38*, 1 (2004), 69–88.

[161] PARKES, D. *Formal Languages and the Word Problem in Groups*. PhD thesis, University of Leicester, 2000.

[162] PARKES, D., AND THOMAS, R. Syntactic monoids and word problems. *Arabian Journal for Science and Engineering (C) 25*, 2 (2000), 81–94.

[163] PIGHIZZINI, G., AND SHALLIT, J. Unary language operations, state complexity and Jacobsthal's function. *International Journal of Foundations of Computer Science 13* (2002), 145–159.

[164] PIN, J.-E. *Varieties of Formal Languages*. Plenum, 1986.

[165] PIN, J.-E., AND SAKAROVITCH, J. Une application de la representation matricielle des transductions. *Theor. Comp. Sci. 35* (1981), 271–293.

[166] PIN, J.-E., AND SAKAROVITCH, J. Some operations and transductions that preserve rationality. In *6th GI Conference* (1983), A. Cremers and H.-P. Kriegel, Eds., vol. 145 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 277–288.

[167] POLÁK, L. Syntactic semirings and language equations. In *Implementation and Application of Automata: 7th International Conference* (2003), J.-M. Champarnaud and D. Maurel, Eds., vol. 2608 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 182–193.

[168] PĂUN, G., AND SALOMAA, A. Thin and slender languages. *Disc. Appl. Math. 61* (1995), 257–270.

[169] RAMESH KUMAR, P., AND RAJAN, A. Expletive languages. *Southeast Asian Bulletin of Mathematics 23* (1998), 187–197.

[170] RIDDLE, W. An approach to software system behaviour description. *Comp. Lang. 4* (1979), 29–47.

[171] ROZENBERG, G., AND SALOMAA, A., Eds. *Handbook of Formal Languages, Vol. 1*. Springer-Verlag, 1997.

[172] RYL, I., ROOS, Y., AND CLERBOUT, M. Generalized synchronization languages. In *Fundamentals of Computation Theory, 12th International Symposium, (FCT'99)* (1999), G. Ciobanu and G. Paun, Eds., pp. 451–462.

[173] SALOMAA, A. *Theory of Automata*. Pergamon Press, 1969.

[174] SALOMAA, A. *Formal Languages*. Academic Press, 1973.

[175] SALOMAA, A. *Jewels of Formal Language Theory*. Computer Science Press, 1981.

[176] SALOMAA, A., AND YU, S. On the decomposition of finite languages. In *Developments in Language Theory* (1999), G. Rozenberg and W. Thomas, Eds., pp. 22–31.

[177] SALOMAA, K., AND YU, S. Synchronization expressions with extended join operation. *Theor. Comp. Sci. 207* (1998), 73–88.

[178] SALOMAA, K., AND YU, S. Synchronization expressions and languages. *J. Universal Comp. Sci. 5* (1999), 610–621.

[179] SÂNTEAN, L. Six arithmetic-like operation on languages. *Cahiers de linguistique theoretique et applique 25* (1988), 65–73.

[180] SEIFERAS, J., AND MCNAUGHTON, R. Regularity-preserving relations. *Theor. Comp. Sci. 2* (1976), 147–154.

[181] SHALLIT, J. Numeration systems, linear recurrences, and regular sets. *Inf. and Comp. 113*, 2 (1994), 331–347.

[182] SHAW, A. Software descriptions with flow expressions. *IEEE Trans. Soft. Eng. SE-4*, 3 (1978), 242–254.

[183] SHOUDAI, T. A P-complete language describable with iterated shuffle. *Inf. Proc. Letters 41*, 5 (1992), 233–238.

[184] SHYR, H. *Free Monoids and Languages*. Hon Min Book Company, Taichung, Taiwan, 2001.

[185] SHYR, H., AND THIERRIN, G. Hypercodes. *Inf. and Cont. 24*, 1 (1974), 45–54.

[186] SHYR, H., AND THIERRIN, G. Codes and binary relations. In *Séminaire d'Algébre Paul Dubreil, Paris 1975–1976* (1977), A. Dold and B. Eckmann, Eds., vol. 586 of *Lecture Notes in Mathematics*, Springer-Verlag, pp. 180–188.

[187] SHYR, H., AND YU, S.-S. Bi-catenation and shuffle product of languages. *Acta Inf. 35* (1998), 689–707.

[188] SLOANE, N. *The On-Line Encyclopedia of Integer Sequences*. Published electronically at http://www.research.att.com/∼njas/sequences, 2004.

[189] STEARNS, R., AND HARTMANIS, J. Regularity preserving modifications of regular expressions. *Inf. and Cont. 6*, 1 (1963), 55–69.

[190] SZILARD, A., YU, S., ZHANG, K., AND SHALLIT, J. Characterizing regular languages with polynomial densities. In *Mathematical Foundations of Computer Science 1992* (1992), I. Havel and V. Koubek, Eds., vol. 629 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 494–503.

[191] TANAKA, G. Alternating products of prefix codes. In *Second Conference on Automata, Languages and Programming Systems: Proceedings of the conference held in Salgótarján, May 23–26, 1988* (1988), F. Géseg and I. Peák, Eds., pp. 209–213.

[192] THIERRIN, G. Convex languages. In *Automata, Languages and Programming, Colloquium, Paris, France* (1972), M. Nivat, Ed., pp. 481–492.

[193] THIERRIN, G., AND YU, S.-S. Shuffle relations and codes. *J. Inf. Opt. Sci. 12*, 3 (1991), 441–449.

[194] TULLY, E. Expletives in languages and middle units in semigroups. *Semigroup Forum 38* (1989), 77–84.

[195] VAJNOVSZKI, V. Gray visiting Motzkins. *Acta Inf. 38*, 11–12 (2002), 793–811.

[196] VAJNOVSZKI, V. A loopless algorithm for generating the permutations of a multiset. *Theor. Comp. Sci. 307*, 2 (2003), 415–431.

[197] VAN LEEUWEN, J. Effective constructions in well-partially ordered free monoids. *Disc. Math. 21* (1978), 237–252.

[198] WARMUTH, M., AND HAUSSLER, D. On the complexity of iterated shuffle. *J. Comp. Sys. Sci. 28* (1984), 345–358.

[199] WOOD, D. A factor theorem for subsets of a free monoid. *Inf. and Cont. 21* (1972), 21–26.

[200] WOTSCHKE, D. The boolean closures of deterministic and nondeterministic context-free languages. In *GI Jahrestagung* (1973), W. Brauer, Ed., vol. 1 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 113–121.

[201] YU, S. Regular languages. pp. 41–110. In [171].

[202] YU, S. State complexity of the regular languages. *J. Automata, Languages and Combinatorics 6* (2001), 221–234.

[203] YU, S. State complexity of finite and infinite regular languages. *Bull. Eur. Assoc. Theor. Comp. Sci. 76* (2002), 142–152.

[204] YU, S., ZHUANG, Q., AND SALOMAA, K. The state complexities of some basic operations on regular languages. *Theor. Comp. Sci. 125* (1994), 315–328.

[205] ZHANG, G.-Q. Automata, boolean matrices, and ultimate periodicity. *Inf. and Comp. 152* (1999), 138–154.

[206] ZHANG, L., AND SHEN, Z. Completion of recognizable bifix codes. *Theor. Comp. Sci. 145* (1995), 345–355.

# Index