Deciding Whether a Regular Language is Generated by a Splicing System

Lila Kari Steffen Kopecki

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

Abstract

(Linear) splicing as a binary word/language operation is inspired by the DNA recombination under the action of restriction enzymes and ligases, and was first introduced by Tom Head in 1987. Shortly thereafter, it was proven that the languages generated by (finite) splicing systems form a proper subclass of the class of regular languages. However, the question of whether or not one can decide if a given regular language is generated by a splicing system remained open. In this paper we give a positive answer to this question. Namely, we prove that, if a language is generated by a splicing system, then it is also generated by a splicing system whose size is a function of the size of the syntactic monoid of the input language, and which can be effectively constructed.

1 Introduction

In [10] Head described a language-theoretic operation, called *splicing*, which models DNA recombination, a cut-and-paste operation on DNA double-stranded molecules. Recall that a *DNA single-strand* is a polymer consisting of a series of the nucleotides Adenine (A), Cytosine (C), Guanine (G), and Thymine (T) attached to a linear, directed backbone. Due to the chemical structure of the backbone a DNA single-strand has a directionality; its ends are called $3'$-end and $5'$-end. Abstractly, a DNA single-strand can be viewed as a string over the four letter alphabet $\{\mathsf{A}, \mathsf{C}, \mathsf{G}, \mathsf{T}\}$. The bases A and T, respectively C and G, are *Watson-Crick-complementary*, or simply *complementary*, which means they can attach to each other via hydrogen bonds. The *complement* of a DNA single-strand $\alpha = 5'\text{-}a_1 \cdots a_n\text{-}3'$ is the strand $\overline{\alpha} = 3'\text{-}\overline{a_1} \cdots \overline{a_n}\text{-}5'$ where $a_1, \ldots, a_n$ are bases and $\overline{a_1}, \ldots, \overline{a_n}$ denote their complementary bases, respectively; note that α and $\overline{\alpha}$ have opposite orientation. A strand α and its complement $\overline{\alpha}$ can bond to each other to form a *DNA (double-)strand*.

Splicing is meant to abstract the action of two "compatible" restriction enzymes and the ligase enzyme on two DNA double-stranded molecules. The first restriction enzyme recognizes a base-sequence u_1v_1, called its *restriction site*, in any DNA string, and cuts the string containing this factor between u_1 and v_1. The second restriction enzyme, with restriction site u_2v_2, acts similarly. Assuming that the *sticky ends* obtained after these cuts are complementary, the enzyme ligase aids then the recombination (catenation) of the first segment of one cut string with the second segment of the other cut string. For example, the enzyme *Taq*I has restriction site TCGA, and the enzyme *Sci*NI has restriction site GCGC. The enzymes cut double-strands

$$\begin{array}{l} 5'\text{—}\alpha\text{—}\mathsf{T}\vert\mathsf{C}\ \mathsf{G}\ \mathsf{A}\text{—}\beta\text{—}3' \\ 3'\text{—}\overline{\alpha}\text{—}\mathsf{A}\ \overline{\mathsf{G}}\ \overline{\mathsf{C}}\vert\mathsf{T}\text{—}\overline{\beta}\text{—}5' \end{array} \quad \text{and} \quad \begin{array}{l} 5'\text{—}\gamma\text{—}\mathsf{G}\vert\mathsf{C}\ \mathsf{G}\ \mathsf{C}\text{—}\delta\text{—}3' \\ 3'\text{—}\overline{\gamma}\text{—}\mathsf{C}\ \overline{\mathsf{G}}\ \overline{\mathsf{C}}\vert\mathsf{G}\text{—}\overline{\delta}\text{—}5' \end{array}$$

along the dashed lines, respectively, leaving the first segment of the left strand with a sticky end GC which is compatible to the sticky end CG of the second segment of the right strand. The

segments can be recombined to form either the original strands or the new strand

$$\begin{array}{l} 5'\text{—}\alpha\text{—}\mathsf{T\,C\,G\,C}\text{—}\delta\text{—}3' \\ 3'\text{—}\bar{\alpha}\text{—}\mathsf{A\,G\,C\,G}\text{—}\bar{\delta}\text{—}5' \end{array} .$$

A *splicing system* is a formal language model which consists of a set of *initial words* or *axioms* I and a set of *splicing rules* R. Every word in this system represents one DNA double-strand. The most commonly used definition for a splicing rule is a quadruplet of words $r = (u_1, v_1; u_2, v_2)$. This rule splices two words $x_1u_1v_1y_1$ and $x_2u_2v_2y_2$: the words are cut between the factors u_1, v_1, respectively u_2, v_2, and the prefix (the left segment) of the first word is recombined by catenation with the suffix (the right segment) of the second word; see Figure 1 and also [18]. The words u_1v_1 and u_2v_2 are the restriction sites in the rule r. The biological example of the enzyme interaction of *Taq*I and *Sci*NI, as discussed at the beginning of this section is modeled by the rule $(\mathsf{T}, \mathsf{CGA}; \mathsf{G}, \mathsf{CGC})$; the rules $(\mathsf{TC}, \mathsf{GA}; \mathsf{GC}, \mathsf{GC})$ or $(\mathsf{TCG}, \mathsf{A}; \mathsf{GCG}, \mathsf{C})$ could be used alternatively. A splicing system generates a language which contains every word that can be obtained by successively applying rules to axioms and the intermediately produced words.

Figure 1: Splicing of the words $x_1u_1v_1y_1$ and $x_2u_2v_2y_2$ by the rule $r = (u_1, v_1; u_2, v_2)$.

Example 1. Consider the splicing system (I, R) with axiom $I = \{ab\}$ and rules $R = \{r, s\}$ where $r = (a, b; \varepsilon, ab)$ and $s = (ab, \varepsilon; a, b)$; in this paper, ε denotes the empty word. Applying the rule r to two copies of the axiom ab creates the word aab and applying the rule s to two copies of the axiom ab creates the word abb. More generally, the rule r or s can be applied to words a^ib^j and a^kb^ℓ with $i, j, k, \ell \geq 1$ in order to create the word $a^{i+1}b^\ell$ or $a^ib^{\ell+1}$, respectively. The language generated by the splicing system (I, R) is $L(I, R) = a^+b^+$.

The most natural variant of splicing systems, often referred to as *finite splicing systems*, is to consider a finite set of axioms and a finite set of rules. In this paper, by a splicing system we always mean a finite splicing system. Shortly after the introduction of splicing in formal language theory, Culik II and Harju [6] proved that splicing systems can only generate regular languages; see also [12, 17]. Gatterdam [7] gave $(aa)^*$ as an example of a regular language which cannot be generated by a splicing system; thus, the class of languages generated by splicing systems is strictly included in the class of regular languages. However, for any regular language L over an alphabet Σ, adding a marker $b \notin \Sigma$ to the left side of every word in L results in the language bL which can be generated by a splicing system [11]; for example, the language $b(aa)^*$ is generated by the axioms $\{b, baa\}$ and the rule $(baa, \varepsilon; b, \varepsilon)$.

This led to the question of whether or not one of the known subclasses of the regular languages corresponds to the class $\mathcal{S}$ of languages which can be generated by a splicing system. All investigations to date indicate that the class $\mathcal{S}$ does not coincide with another naturally defined language class. A characterization of *reflexive* splicing systems using *Schützenberger constants* was given by Bonizzoni, de Felice, and Zizza [1–3]. A splicing system is reflexive if for all rules $(u_1, v_1; u_2, v_2)$ in the system we have that $(u_1, v_1; u_1, v_1)$ and $(u_2, v_2; u_2, v_2)$ are rules in the system as well. A word v is a (Schützenberger) constant of a language L if $x_1vy_1 \in L$ and $x_2vy_2 \in L$ imply $x_1vy_2 \in L$ [19]. Recently, it was proven by Bonizzoni and Jonoska that every splicing language has a constant [5]. However, not all languages which have a constant are generated by splicing systems; for example, in the language $L = (aa)^* + b^*$ every word b^i is a constant, but L is not generated by a splicing system.

Another approach was to find an algorithm which decides whether or not a given regular language is generated by a splicing system. This problem has been investigated by Goode, Head, and Pixton [8, 9, 13], but it has only been partially solved: it is decidable whether or not a regular

language is generated by a reflexive splicing system. It is worth mentioning that a splicing system by the original definition in [10] is always reflexive. A related problem has been investigated by Kim [16]: given a regular language L and a finite set of *enzymes*, represented by set of reflexive rules R, Kim showed that it is decidable whether or not L can be generated from a finite set of axioms by using only rules from R.

In this paper we settle the decidability problem by proving that for a given regular language, it is indeed decidable whether or not the language is generated by a splicing system, not necessarily reflexive (Corollary 5.2). More precisely, for every regular language L there exists a splicing system (I_L, R_L) and if L is a splicing language, then L is generated by the splicing system (I_L, R_L). The size of this splicing system depends on the size of the syntactic monoid of L. If m is the size of the syntactic monoid of L, then all axioms in I_L and the four components of every rule in R_L have lengths in $\mathcal{O}(m^2)$ (Theorem 4.1). By results from [12, 13], we can construct a finite automaton which accepts the language generated by (I_L, R_L), compare it with a finite automaton which accepts L, and thus, decide whether L is generated by a splicing system or not. Furthermore, we prove a similar result for a more general variant of splicing that has been introduced by Pixton [17] (Theorem 3.1).[1]

The paper is organized as follows. In Section 2 we lay down the notation, recall some well-known results about syntactic monoids, and prove a pumping argument that is of importance for the proofs in the succeeding sections. Section 3 (resp. Section 4) contains the proof that a regular language L is generated by a *Pixton splicing system* (resp. *classical splicing system*) if and only if it is generated by one particular Pixton splicing system (resp. classical splicing system) whose size is bounded by the size of the syntactic monoid of L. Sections 3 and 4 can be read independently and overlap in some of their main ideas. The inclusion of both sections and the presentation order are chiefly for expository purposes: due to the features of the Pixton splicing, Section 3 introduces the main ideas in a significantly more readable way. Finally, in Section 5 we deduce the decidability results for both splicing variants.

2 Notation and Preliminaries

We assume the reader to be familiar with the fundamental concepts of language theory; see [14].

Let Σ be a finite set of *letters*, the *alphabet*; Σ^* be the set of all words over Σ; and ε denote the *empty word*. A subset L of Σ^* is a *language* over Σ. Throughout this paper, we only consider languages over the fixed alphabet Σ. Let $w \in \Sigma^*$ be a word. The length of w is denoted by $|w|$. (We use the same notation for the cardinality $|S|$ of a set S as usual.) We consider the letters of Σ to be ordered and for words $u, v \in \Sigma^*$ we denote the *length-lexicographical order* by $u \leq_{\ell\ell} v$; i. e., $u \leq_{\ell\ell} v$ if either $|u| < |v|$, or $|u| = |v|$ and u is at most v in lexicographic order. The *strict length-lexicographic order* is denoted by $<_{\ell\ell}$; we have $u <_{\ell\ell} v$ if $u \leq_{\ell\ell} v$ and $u \neq v$.

For a number $m \in \mathbb{N}$ we let $\Sigma^{\leq m}$ denote the set of words whose length is at most m, i. e., $\Sigma^{\leq m} = \bigcup_{i \leq m} \Sigma^i$. Analogously, we define $\Sigma^{<m} = \bigcup_{i<m} \Sigma^i$.

If $w = xyz$ for some $x, y, z \in \Sigma^*$, then x, y, and z are called *prefix*, *factor*, and *suffix* of w, respectively. If a prefix or suffix of w is distinct from w, it is said to be *proper*; a factor y of w is a *proper factor*, if it is neither a prefix nor a suffix of w.

Let $w = a_1 \dots a_n$ where $a_1, \dots, a_n$ are letters from Σ. By $w_{[i]}$ for $0 \leq i \leq n$ we denote a *position* in the word w: if $i = 0$, it is the position before the first letter a_1, if $i = n$ it is the position after the last letter a_n, and otherwise, it is the position between the letters a_i and a_{i+1}. We want to stress that $w_{[i]}$ is not a letter in the word w. By $w_{[i;j]}$ for $0 \leq i \leq j \leq n$ we denote the factor $a_{i+1} \cdots a_j$ which is enclosed by the positions $w_{[i]}$ and $w_{[j]}$. If $x = w_{[i;j]}$ we say the factor x starts at position $w_{[i]}$ and ends at position $w_{[j]}$. Whenever we talk about a factor x of a word w we mean a factor starting (and ending) at a certain position, even if the word x occurs as a factor at several positions in w. Let $x = w_{[i;j]}$ and $y = w_{[i';j']}$ be factors of w. We say the factors x and

[1] An extended abstract of this paper, including a shortened proof of Theorem 4.1 and Corollary 5.2 i.) was published in the conference proceedings of DNA 18 in 2012 [15]. Theorem 3.1 and Corollary 5.2 ii.) have not been published elsewhere.

y *match* (in w) if $i = i'$ and $j = j'$. The factor x is *covered* by the factor y (in w) if $i' \leq i \leq j \leq j'$. The factors x and y *overlap* (in w) if $x \neq \varepsilon$, $y \neq \varepsilon$, and $i \leq i' < j$ or $i' \leq i < j'$; in other words, if two factors x and y overlap in w, then they share at least one letter of w. Note that if a non-empty factor is covered by another factor, these two factors will also overlap. Let $x = w_{[i;j]}$ be a factor of w and let $p = w_{[k]}$ be a position in w. We say the position p *lies at the left of* x if $k \leq i$; the position p *lies at the right of* x if $k \geq j$; and the position p *lies in* x if $i < k < j$.

Every language L induces a *syntactic congruence* $\sim_L$ over words such that $u \sim_L v$ if for all words x, y

$$xuy \in L \iff xvy \in L.$$

The *syntactic class* (with respect to L) of a word u is $[u]_L = \{v \mid u \sim_L v\}$. The *syntactic monoid* of L is the quotient monoid

$$M_L = \Sigma^*/\!\sim_L = \{[u]_L \mid u \in \Sigma^*\}.$$

Example 2. The syntactic monoid M_L of the regular language $L = a^+b^+$ is given by the syntactic classes

$$M_L = \{1, [a]_L, [b]_L, [ab]_L, 0\}$$

where $1 = [\varepsilon]_L = \varepsilon$, $[a]_L = a^+$, $[b]_L = b^+$, $[ab]_L = a^+b^+$, and 0 contains all words which are not factors of a word in L. Note that 1 is the neutral element (or identity element) and 0 acts as zero in the monoid.

It is well known that a language L is regular if and only if its syntactic monoid M_L is finite. We will use two basic facts about syntactic monoids of regular languages.

Lemma 2.1 (Pumping Lemma)**.** *Let L be a regular language and let w be a word with $|w| \geq |M_L|^2$. We can factorize $w = \alpha\beta\gamma$ with $\beta \neq \varepsilon$ such that $\alpha \sim_L \alpha\beta$ and $\gamma \sim_L \beta\gamma$. In particular, $\alpha\beta\gamma \sim_L \alpha\beta^j\gamma$ for all $j \in \mathbb{N}$.*

Proof. Consider a word w with $n = |w| \geq |M_L|^2$. For $i = 0, \ldots, n$, let $X_i = [w_{[0;i]}]_L$ be the syntactic class of the prefix of w of length i and let $Y_i = [w_{[i;n]}]_L$ be the syntactic class of the suffix of w of length $n - i$. Note that $X_i \cdot Y_i = [w]_L$. By the pigeonhole principle, there are j, k with $0 \leq j < k \leq n$ such that $X_j = X_k$ and $Y_j = Y_k$. Let $\alpha = w_{[0;j]}$, $\beta = w_{[j;k]}$, and $\gamma = w_{[k;n]}$. As $\alpha \in X_j$ and $\alpha\beta \in X_k = X_j$, we see that $\alpha \sim_L \alpha\beta$ and, symmetrically, $\gamma \sim_L \beta\gamma$. □

Lemma 2.2. *Let L be a regular language. Every element $X \in M_L$ contains a word $x \in X$ with $|x| < |M_L|$.*

Proof. We define a series of sets $S_i \subseteq M_L$. We start with $S_0 = \{1\}$ (here, $1 = [\varepsilon]_L$) and let $S_{i+1} = S_i \cup \{X \cdot [a]_L \mid X \in S_i \wedge a \in \Sigma\}$ for $i \geq 0$. It is not difficult to see that $X \in S_i$ if and only if X contains a word $x \in X$ with $|x| \leq i$. As $S_i \subseteq S_{i+1}$ and M_L is finite, the series has a fixed point S_n such that $S_i = S_n$ for all $i \geq n$. Let n be the least value with this property, i. e., $S_{n-1} \subsetneq S_n$ or $n = 0$. Observe that $n < |M_L|$ as $S_0 \subsetneq S_1 \subsetneq \cdots \subsetneq S_n$. Every element $X \in M_L$ contains some word $w \in X$, thus, $X \in S_{|w|} \subseteq S_n$. We conclude that X contains a word with a length of at most $n < |M_L|$. □

2.1 A Pumping Argument

In this section we present a pumping technique that will become useful in the main proofs of Sections 3 and 4. We consider a regular language L, words α, β, γ, and a *large* even integer j such that $\alpha\beta\gamma \sim_L \alpha\beta^j\gamma$, due to the Pumping Lemma 2.1. In the proofs of Lemma 4.8 and Theorem 3.1, we need a pumping argument to replace all factors $\alpha\beta\gamma$ by $\alpha\beta^j\gamma$ in a word z in order to obtain a word $\tilde{z} \sim_L z$. As $\alpha\beta\gamma$ may be a factor of $\alpha\beta^j\gamma$, we cannot ensure that $\alpha\beta\gamma$ is not a factor of $\tilde{z}$ anymore. However, we can ensure that if $\alpha\beta\gamma = \tilde{z}_{[k;k']}$ is a factor of $\tilde{z}$, then (a) $\alpha\beta^{j/2}$ is a factor of $\tilde{z}$ starting at position $\tilde{z}_{[k]}$ or (b) $\beta^{j/2}\gamma$ is a factor of $\tilde{z}$ ending at position $\tilde{z}_{[k']}$; i. e., α is succeeded by a large number of β's or γ is preceded by a large number of β's. The next lemma is a technical

result whose purpose is to ensure that for any word z there exists a word $\tilde{z}$ such that for any factor $\alpha\beta\gamma$ in $\tilde{z}$ (a) or (b) holds, and $\tilde{z}$ is generated by applying several successive *factor replacements* $\alpha\beta\gamma \mapsto \alpha\beta^j\gamma$ to z. By a factor replacement we mean that, if z can be factorized $z = x\alpha\beta\gamma y$, then we can apply a factor replacement $\alpha\beta\gamma \mapsto \alpha\beta^j\gamma$ to z in order obtain the word new $z' = x\alpha\beta^j\gamma y$.

Lemma 2.3. *Let L be a language, let $j \geq 4$ be an even integer, and let α, β, γ be words with $\beta \neq \varepsilon$ such that $\alpha\beta\gamma \sim_L \alpha\beta^j\gamma$. For a word z we can effectively obtain a word $\tilde{z}$ by successively applying factor replacements $\alpha\beta\gamma \mapsto \alpha\beta^j\gamma$ to z such that $\tilde{z} \sim_L z$ and for all integers k, k' which satisfy $\tilde{z}_{[k;k']} = \alpha\beta\gamma$ one of the following conditions holds:*

(a) $\alpha\beta^{j/2}$ is a factor of $\tilde{z}$ starting at position $\tilde{z}_{[k]}$ or

(b) $\beta^{j/2}\gamma$ is a factor of $\tilde{z}$ ending at position $\tilde{z}_{[k']}$.

Before we prove Lemma 2.3, let us recall a basic fact about primitive words. A word p is called *primitive* if there does not exist a word $x \in \Sigma^+$ and an integer i with $i \geq 2$ such that $p = x^i$. The *primitive root* of a word $w \neq \varepsilon$ is the unique primitive word p such that $w = p^i$ for some $i \geq 1$. For a primitive word p, it is well known that if $pp = xpy$, then either $x = p$ and $y = \varepsilon$, or $x = \varepsilon$ and $y = p$. In other words, whenever p is a factor of p^n starting at position $p^n{}_{[i]}$, then $i \in |p| \cdot \mathbb{N}$ (that is, $|p|$ divides i).

For a word $w = xy$ we employ the notations $x^{-1}w = y$ and $wy^{-1} = x$. If x is not a prefix of w (resp. y is not a suffix of w), then the $x^{-1}w$ (resp. wy^{-1}) is undefined. For words $x, w \in \Sigma^*$ and $k \in \mathbb{N}$ we let $x^{-k}w = (x^k)^{-1}w$ and $wx^{-k} = w(x^k)^{-1}$.

Proof of Lemma 2.3. Let L be a language, let $j \geq 4$ be an even integer, let α, β, γ be words with $\beta \neq \varepsilon$ such that $\alpha\beta\gamma \sim_L \alpha\beta^j\gamma$, and let $\ell = |\alpha\beta\gamma|$. We use the following pumping algorithm in order to obtain $\tilde{z}$ from z:

1. let $z_0 := z$; let $n := 0$;
2. let k be the minimal position such that $z_{n[k;k+\ell]} = \alpha\beta\gamma$ and neither of the following conditions is true
   (a) $\alpha\beta^{j/2}$ is a factor of z_n starting at position $z_{n[k]}$,
   (b) $\beta^{j/2}\gamma$ is a factor of z_n ending at position $z_{n[k+\ell]}$;
   if such k does not exist, then return $\tilde{z} := z_n$ and halt;
3. let $z_{n+1} := z_{n[0;k]} \cdot \alpha\beta^j\gamma \cdot z_{n[k+\ell;|z_n|]}$; (replace the factor $z_{n[k;k+\ell]} = \alpha\beta\gamma$ by $\alpha\beta^j\gamma$)
   let $n := n + 1$;
4. repeat steps 2-4.

The pumping algorithm defines a series of words $z_0, z_1, z_2, \ldots$ where $z_0 = z$ and the word z_{n+1} is obtained from z_n by replacing one factor $\alpha\beta\gamma$ in z_n by the factor $\alpha\beta^j\gamma$. Because $\alpha\beta\gamma \sim_L \alpha\beta^j\gamma$ and by induction, we obtain that $z_n \sim_L z$ for all $n \in \mathbb{N}$. Under the assumption that the algorithm stops after N cycles, we obtain that $\tilde{z} = z_N \sim_L z$. Furthermore, the algorithm terminates only if for all factors $\tilde{z}_{[k;k+\ell]} = \alpha\beta\gamma$ in $\tilde{z}$ (a) or (b) is satisfied. Thus, in order to prove the lemma, we have to prove that the algorithm always terminates.

Let p be the primitive root of β, let m such that $\beta = p^m$, and let $y = p^{m \cdot j-2} = \beta^j p^{-2}$. (Note that y is a well-defined, non-empty word even if β is primitive, because $j \geq 4$.) For each $n < N$ we will define a unique factorization

$$z_n = x_{n,0} y x_{n,1} y x_{n,2} \cdots y x_{n,n}$$

where p is a suffix of $x_{n,i}$ for $i = 0, \ldots, n-1$ and p is a prefix of $x_{n,i}$ for $i = 1, \ldots, n$. This factorization is defined inductively: naturally, we start with $x_{0,0} = z_0 = z$. Suppose z_n is factorized in the above manner. Let k be the position in z_n such that $\alpha\beta\gamma = z_{n[k;k+\ell]}$ and

$$z_{n+1} = z_{n[0;k]} \cdot \alpha\beta^j\gamma \cdot z_{n[k+\ell;|z_n|]},$$

as described in the pumping algorithm; thus, neither (a) nor (b) holds for k in z_n. If there is no such factor, the algorithm terminates and we do not have to define z_{n+1}. We define the factorization of z_{n+1} under the assumption that $\beta = z_{n[k+|\alpha|;k+|\alpha\beta|]}$ is covered by some $x_{n,i}$ in the factorization of z_n such that $x_{n_i} = u\beta v$ for some words u, v. We split $x_{n,i}$ up in order to obtain $x_{n+1,i} = up$ and $x_{n+1,i+1} = pv$; Figure 2 illustrates a possible factorization of the words z_n and z_{n+1}. Moreover, we let $x_{n+1,i'} = x_{n,i'}$ for $i' = 0, \ldots, i-1$ and $x_{n+1,i'+1} = x_{n,i'}$ for $i' = i+1, \ldots, n$. Observe that

$$z_{n+1} = x_{n+1,0} y x_{n+1,1} y x_{n+1,2} \cdots y x_{n+1,n+1}$$

defines the desired factorization.

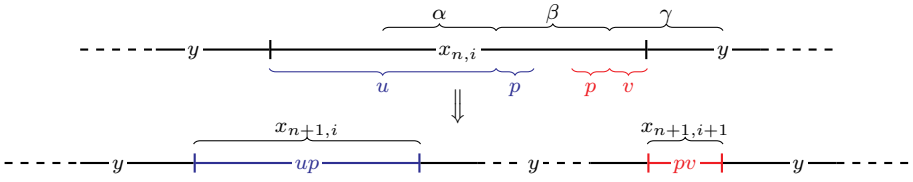

Figure 2: Obtaining the factorization of z_{n+1} (bottom) from the word z_n (top).

Next we will show that either

(i) $\beta = z_{n[k+|\alpha|;k+|\alpha\beta|]}$ is a proper factor of $x_{n,i} = u\beta v$ in the factorization of z_n (i. e., $u \neq \varepsilon$ and $v \neq \varepsilon$); or

(ii) $\alpha = \varepsilon$, $k = 0$, and $\beta = z_{n[0;|\beta|]}$ is a prefix of $x_{n,0}$ in the factorization of z_n; or

(iii) $\gamma = \varepsilon$, $k = |z_n| - \ell$, and $\beta = z_{n[|z_n|-|\beta|;|z_n|]}$ is a suffix of $x_{n,n}$ in the factorization of z_n;

because otherwise (a) or (b) is satisfied. Suppose that k in z_n is chosen by the algorithm such that neither (i), nor (ii), nor (iii) holds. Because p is a suffix of $x_{n,i}$ for $i = 0, \ldots, n-1$ and p is a prefix of $x_{n,i}$ for $i = 1, \ldots, n$, we see that $\beta = z_{n[k+|\alpha|;k+|\alpha\beta|]}$ has to overlap by at least $|p|$ letters with a factor $pyp = \beta^j$ in z_n; let h be the position in z_n such that $\beta^j = z_{n[h;h+|\beta^j|]}$ denotes the factor in z_n that overlaps by at least $|p|$ letters with $\beta = z_{n[k+|\alpha|;k+|\alpha\beta|]}$. Due to the properties of primitive roots, we have $k + |\alpha| - h \in |p| \cdot \mathbb{Z}$. Furthermore, if $k + |\alpha| \leq h + \left|\beta^{j/2}\right|$, then $z_{n[k+|\alpha|]}$ is succeeded by $p^{m \cdot j/2} = \beta^{j/2}$, hence, (a) holds. Otherwise, if $k + |\alpha| > h + \left|\beta^{j/2}\right|$, then $z_{n[k+|\alpha\beta|]}$ is preceded by $\beta^{j/2}$, hence, (b) holds. Therefore, the algorithm will always chose k such that either (i) or (ii) or (iii) holds.

Note that if k is chosen in the n-th cycle of the algorithm such that case (ii) holds, then for all $i \geq n$ we have $x_{i,0} = p$; therefore, k can never be chosen again such that case (ii) holds. Symmetrically, k can only once be chosen such that case (iii) holds.

Next, we show that case (i) can only occur a finite number of times. Consider that k is chosen in the n-th cycle such that case (i) holds, where $\beta = z_{n[k+|\alpha|;k+|\alpha\beta|]}$ is a proper factor of $x_{n,i} = u\beta v$. Because $|u|, |v| \geq 1$,

$$|x_{n+1,i}| = |u| + |p| = |x_{n,i}| - |\beta| - |v| + |p| \leq |x_{n,i}| - |v| < |x_{n,i}|$$

and, symmetrically, $|x_{n+1,i+1}| < |x_{n,i}|$. Thus, in each pumping step where (i) holds, we replace one of the factors $x_{n,i}$ by two strictly shorter factors $x_{n+1,i}$ and $x_{n+1,i+1}$. As we cannot pump in a factor $x_{n,i}$ if it is shorter than $|\beta| + 2$, eventually, all the factors will be too short and the pumping algorithm has to halt. □

2.2 Outline of the Main Proofs

In Sections 3 and 4 we prove that a regular language L is a Pixton or classical splicing language, respectively, if and only if it is generated by a splicing system whose size only depends on the size

of the syntactic monoid of L. Here, we outline the following proofs while ignoring most technical details. We focus on classical splicing (Section 4), but the proof outline for Pixton's variant of splicing (Section 3) is similar.

In Section 4.1 (resp. Section 3.1), we start by describing basic techniques to modify splicing rules; for example, if a rule $r = (u_1, v_1; u_2, v_2)$ belongs to a splicing system (I, R), we can extend the the second component v_1 of r to the right by any word x to obtain the new rule $s = (u_1, v_1x; u_2, v_2)$ and add this rule s to the splicing system (I, R) without changing the language $L(I, R) = L(I, R \cup \{s\})$ that is generated by the splicing system. If L is the language generated by a splicing system (I, R), then we can replace some of the four components in a rule by syntactically congruent words, with respect to L, and add this new rule to R. We can combine these two techniques as follows: when a word z is generated by splicing from two words w_1 and w_2 that are significantly longer than z, we can also find two shorter words $\tilde{w}_1$ a $\tilde{w}_2$ which generate z by splicing; see Figure 3.

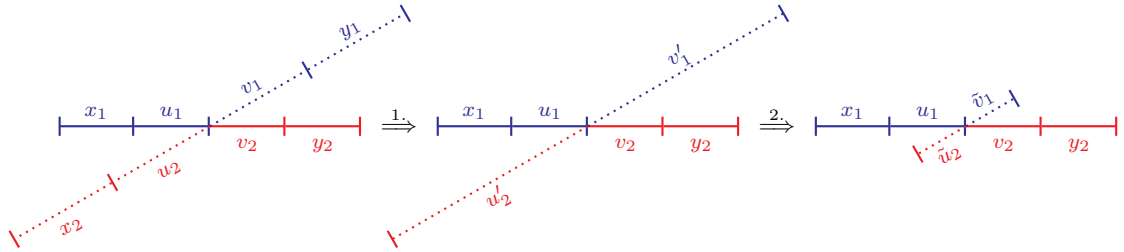

Figure 3: Modification of the splicing $(w_1, w_2) \vdash_r z$ with $w_1 = x_1u_1v_1y_1$, $w_2 = x_2u_2v_2y_2$, $z = x_1u_1v_2y_2$, and $r = (u_1, v_1; u_2, v_2)$: 1.) extend v_1 such that it covers a suffix of w_1 and extend u_2 such that it covers a prefix of w_2; 2.) replace v_1' by a shorter word $\tilde{v}_1 \in [v_1']_L$ and replace u_2' by a shorter word $\tilde{u}_2 \in [u_2']_L$. The new splicing is $(\tilde{w}_1, \tilde{w}_2) \vdash_s z$ where $\tilde{w}_1 = x_1u_1\tilde{v}_1 \sim_L w_1$, $\tilde{w}_2 = \tilde{u}_2v_2y_2 \sim_L w_2$ and $s = (u_1, \tilde{v}_1; \tilde{u}_2, v_2)$.

We continue by discussing series of splicings in Section 4.2 (resp. Section 3.2). The creation of a word by successive splicing in a splicing system (I, R) can be visualized by a binary tree, where $(w_1, w_2) \vdash w_3$ is interpreted as w_1 being the left child of w_3 and w_2 being the right child of w_3. Consider the creation tree of some word x_0zy_0 in a splicing system (I, R); here, z is considered to be relatively long compared to the prefix x and suffix y, and z is also longer than any of the four components of any rule in the splicing system. We are only interested in the nodes (or words) in the splicing tree which contain the factor z. All nodes in the tree that do not contain the factor z are considered to be leafs and those which contain z can be written as x_izy_i. This procedure yields a degenerated tree, as shown in Figure 4 on the left, which can also be interpreted as a series of splicings. There are two cases: either we find a first splicing $(w_{k+1}, w_{k+2}) \vdash x_kzy_k$ that *affects* z, i. e., a prefix of z belongs to w_{k+1} and the corresponding suffix belongs to w_{k+2} (this case is depicted in Figure 4); or the word x_kzy_k belongs to I which means that it is a leaf in the tree as well. It is not difficult to observe that the splicings which alter the prefixes x_i in this tree do not *interfere* with the splicings which alter the suffixes y_i and their order can be exchanged in order to reorganize the tree from one with a zig-zag shape (Figure 4 left) to one where all-right branches are followed by all-left branches (Figure 4 right).

Using the techniques employed in Section 4.1 (resp. Section 3.1), we can replace the words $w_1, \ldots, w_k$ on the leafs which only alter a prefix x_i or a suffix y_i in order to impose a length restriction for them; this length restriction depends on the lengths of x_0 and y_0 and the size of the syntactic monoid of $L(I, R)$. Ultimately, we will use this result to ensure that all the words $w_1, \ldots, w_k$ belong to a finite set $I' = L(I, R) \cap \Sigma^{\leq \ell}$ for some integer ℓ.

In Section 4.3 (resp. Section 3.3) we will prove that a regular language L is a splicing language only if it is generated by a splicing system (I_L, R_L) where we impose a length bound on all words in I_L and all four components of the rules in R_L. This is done in two steps: step 1, we let the set of initial words be arbitrarily large and just restrict the length of components of R_L; step 2, we impose the length restriction on words in I_L (for technical reasons, these two steps are swapped

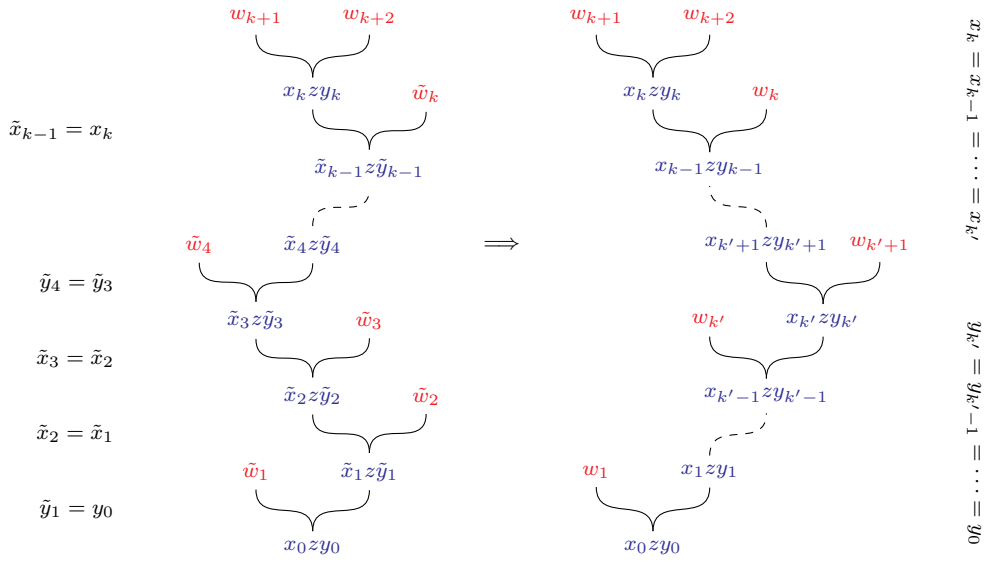

Figure 4: Tracing back the creation of $x_0 z y_0$ in the splicing tree: we go back to a point where the factor z is affected for the last time. The left side shows a general splicing tree, the right side shows a reorganized splicing tree where all modifications of y_i by splicings happen first, and then the modifications of x_i by splicing happen. The nodes w_{k+1}, w_{k+2}, $x_0 z y_0$, and $x_k z y_k$ are the same on the left and right side, but the other leafs w_i for $i = 1, \ldots, k$ appear in different places; the intermediate results on the non-leaf nodes change as well but they all contain the factor z.

in Section 3.3).

Step 1.) We assume, by contradiction, that $L = L(J, S)$ is a splicing language, but that $L(J, R_L) \subsetneq L$. There exists a length-lexicographically minimal word μ which is a component of a rule in S such that no rule in R_L contains the component μ. The length restriction on the components in R_L is chosen large enough such that we find a factor $\alpha\beta\gamma$ in μ, as described in the Pumping Lemma 2.1. Then, we pick a shortest word w from $L(J, S) \setminus L(J, R_L)$ and we apply the pumping argument from Section 2.1 which replaces all factors $\alpha\beta\gamma$ in w by $\alpha\beta^j\gamma$ for a large number j in order to obtain a word $\tilde{w}$ (the following arguments hold even if w does not contain a factor $\alpha\beta\gamma$). Next, we consider the generation of $\tilde{w}$ by splicing, as shown in Figure 4. If a rule with component μ is used during this splicing, then it has to overlap with a factor $\alpha\beta^j\gamma$ in $\tilde{w}$ which allows us to replace this rule by a rule where all four components are length-lexicographically shorter than μ. Now, since we know that $\tilde{w}$ can be generated from strictly shorter words and with rules from R_L, we reverse the pumping in order to obtain a splicing tree that generates w from strictly shorter words and rules in R_L. Thus, either w can be generated without using a rule with a component μ or w is not the shortest word in $L(J, S) \setminus L(J, R_L)$; both cases yield a contradiction.

Step 2.) After the first step of the proof we may assume that if L is a splicing language, then it is generated by a splicing system (J, R_L). This time, we prove by induction on the length of words in L, that every word $w \in L$ can be generated by splicing in (I_L, R_L). Clearly, this holds for all words which are short enough to belong to I_L. Then, we assume for a word $w \in L$ that all strictly shorter words belong to $L(I_L, R_L)$. We ensure that w is long enough to find a pumpable factor $\alpha\beta\gamma$ in w as described by the Pumping Lemma 2.1. This time we use a simple pumping argument and replace this factor $\alpha\beta\gamma$ in w by $\alpha\beta^j\gamma$ in order to obtain a word $\tilde{w}$; we choose j large enough to ensure that no word in J can contain the factor $\alpha\beta^j\gamma$. Therefore, we can trace back the generation of the word $\tilde{w}$ by splicing, as depicted in Figure 4, in order to obtain a splicing series that generates $\tilde{w}$ from strictly shorter words. After pumping $\alpha\beta^j\gamma$ back down to $\alpha\beta\gamma$, we obtain a series of splicings that generates w from strictly shorter words. Applying the induction hypothesis, we conclude that $w \in L(I_L, R_L)$, as desired.

3 Pixton's Variant of Splicing

In this section we use the definition of the splicing operation as it was introduced in [17]. A triplet of words $r = (u_1, u_2; v) \in (\Sigma^*)^3$ is called a *(splicing) rule*. The words u_1 and u_2 are called *left* and *right restriction site* of r, respectively, and v is the *bridge* of r. This splicing rule can be applied to two words $w_1 = x_1u_1y_1$ and $w_2 = x_2u_2y_2$, that each contain one of the restriction sites, in order to create the new word $z = x_1vy_2$, see Figure 5. This operation is called *splicing* and it is denoted by $(w_1, w_2) \vdash_r z$.

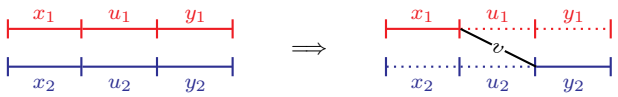

Figure 5: Splicing of the words $x_1u_1y_1$ and $x_2u_2y_2$ by the rule $r = (u_1, u_2; v)$.

For a rule r we define the *splicing operator* σ_r such that for a language L

$$\sigma_r(L) = \{z \in \Sigma^* \mid \exists w_1, w_2 \in L \colon (w_1, w_2) \vdash_r z\}$$

and for a set of splicing rules R, we let

$$\sigma_R(L) = \bigcup_{r \in R} \sigma_r(L).$$

The reflexive and transitive closure of the splicing operator σ_R^* is given by

$$\sigma_R^0(L) = L, \qquad \sigma_R^{i+1}(L) = \sigma_R^i(L) \cup \sigma_R(\sigma_R^i(L)), \qquad \sigma_R^*(L) = \bigcup_{i \geq 0} \sigma_R^i(L).$$

A finite set of axioms $I \subseteq \Sigma^*$ and a finite set of splicing rules $R \subseteq (\Sigma^*)^3$ form a *splicing system* (I, R). Every splicing system (I, R) generates a language $L(I, R) = \sigma_R^*(I)$. Note that $L(I, R)$ is the smallest language which is closed under the splicing operator σ_R and includes I. It is known that the language generated by a splicing system is regular; see [17]. A (regular) language L is called a *splicing language* if a splicing system (I, R) exists such that $L = L(I, R)$.

A rule r is said to *respect* a language L if $\sigma_r(L) \subseteq L$. It is easy to see that for any splicing system (I, R), every rule $r \in R$ respects the generated language $L(I, R)$. Moreover, a rule $r \notin R$ respects $L(I, R)$ if and only if $L(I, R \cup \{r\}) = L(I, R)$. We say a splicing $(w_1, w_2) \vdash_r z$ *respects* a language L if $w_1, w_2 \in L$ and r respects L; obviously, this implies $z \in L$, too.

Pixton introduced this variant of splicing in order to give a simple proof for the regularity of languages generated by splicing systems. As Pixton's variant of splicing is more general than the *classic splicing*, defined in the introduction and in Section 4, his proof of regularity also applies to classic splicing systems. For a moment, let us call a classic splicing rule a *quadruplet* and a Pixton splicing rule a *triplet*. Consider a quadruplet $r = (u_1, v_1; u_2, v_2)$. It is easy to observe that whenever we can use r in order to splice $w_1 = x_1u_1v_1y_1$ with $w_2 = x_2u_2v_2y_2$ to obtain the word $z = x_1u_1v_2y_2$, we can use the triplet $s = (u_1v_1, u_2v_2; u_1v_2)$ in order to splice $(w_1, w_2) \vdash_s z$ as well. However, for a triplet $s = (u_1, u_2; v)$ where v is not a concatenation of a prefix of u_1 and a suffix of u_2, there is no quadruplet r that can model the same splicings. Moreover, the class of classical splicing languages is strictly included in the class of Pixton splicing languages; for example, the language

$$L = cx^*ae + cx^*be + dcx^*bef$$

over the alphabet $\{a, b, c, d, e, f, x\}$ is a Pixton splicing language but not a classical splicing language [4]. For the rest of this section we focus on Pixton's splicing variant and by a rule we always mean a triplet.

The main result of this section states that if a regular language L is a splicing language, then it is created by a particular splicing system (I, R) which only depends on the syntactic monoid of L.

Theorem 3.1. *Let L be a splicing language and $m = |M_L|$. The splicing system (I, R) with $I = \Sigma^{<m^2+6m} \cap L$ and*

$$R = \left\{ r \in \Sigma^{<2m} \times \Sigma^{<2m} \times \Sigma^{<m^2+10m} \;\middle|\; r \text{ respects } L \right\}$$

generates the language $L = L(I, R)$.

As the language generated by the splicing system (I, R) is constructible, Theorem 3.1 implies that the problem whether or not a given regular language is a splicing language is decidable. A detailed discussion of the decidability result is given in Section 5.

Let L be a formal language. Clearly, every set of words $J \subseteq L$ and set of rules S where every rule in S respects L generates a subset $L(J, S) \subseteq L$. Therefore, in Theorem 3.1 the inclusion $L(I, R) \subseteq L$ is obvious. The rest of this section is devoted to the proof of the converse inclusion $L \subseteq L(I, R)$.

Example 3. Recall from Example 2 that the syntactic monoid of the regular language $L = a^+b^+$ contains $m = 5$ elements. Let I and R be as defined in Theorem 3.1 for the language L. We can modify the classic splicing system from Example 1 in order to obtain the Pixton splicing system (I', R') which generates $L = L(I', R')$ where $I' = \{ab\}$ and $R' = \{(ab, ab; aab), (ab, ab; abb)\}$. The splicing system (I, R), defined in Theorem 3.1, is much larger than (I', R') as it is composed of all initial words from $I = \left\{a^i b^j \;\middle|\; i, j \geq 1 \wedge i + j < 55\right\}$ and all rules from $\Sigma^{<10} \times \Sigma^{<10} \times \Sigma^{<75}$ that respect L. Therefore, $I' \subseteq I$ and $R' \subseteq R$. We conclude that $L = L(I', R') \subseteq L(I, R) \subseteq L$.

Consider a splicing language L. One of the main techniques we use in the proof is that, whenever a word z is created by a series of splicings from a set of words in L and a set of rules that respect L, then we can use a modified set of words from L, and a modified set of rules which respect L in order to obtain the same word z by splicing. If z is sufficiently long these words can be chosen such that they are all shorter than z and the restriction sites and bridges of the rules also satisfy certain length restrictions. Of course, our goal is to show that we can create z by splicing from a subset of I with rules which all satisfy the length bounds given by R (as defined in Theorem 3.1). In Section 3.1 we will present techniques to obtain rules that respect L from other rules respecting L and we show how we can modify a single splicing step, such that the words used for splicing are not significantly longer than the splicing result. In Section 3.2 we use these techniques to modify a series of splicings in the way described above (Lemma 3.8). Finally, in Section 3.3 we prove Theorem 3.1.

3.1 Rule Modifications

Let us start with the simple observation that we can extend the restriction sites and the bridge of a rule r such that the new rule respects all languages which are respected by r.

Lemma 3.2. *Let $r = (u_1, u_2; v)$ be a rule which respects a language L. For every word x, the rules $(xu_1, u_2; xv)$, $(u_1x, u_2; v)$, $(u_1, xu_2; v)$, and $(u_1, u_2x; vx)$ respect L as well.*

Proof. Let s be any of the four rules $(xu_1, u_2; xv)$, $(u_1x, u_2; v)$, $(u_1, xu_2; v)$, or $(u_1, u_2x; vx)$. In order to prove that s respects L we have to show that, for all $w_1, w_2 \in L$ and $z \in \Sigma^*$ such that $(w_1, w_2) \vdash_s z$, we have $z \in L$, too. Indeed, if $(w_1, w_2) \vdash_s z$, then $(w_1, w_2) \vdash_r z$ and as r respects L, we conclude $z \in L$. □

Henceforth, we will refer to the rules $(xu_1, u_2; xv)$ and $(u_1, u_2x; vx)$ as extensions of the bridge and to the rules $(u_1x, u_2; v)$ and $(u_1, xu_2; v)$ as extensions of the left and right restriction site, respectively.

For a language L, let us investigate the syntactic class of a rule $r = (u_1, u_2; v)$. The *syntactic class* (with respect to L) of r is the set of rules $[r]_L = [u_1]_L \times [u_2]_L \times [v]_L$ and two rules r and s are *syntactically congruent* (with respect to L), denoted by $r \sim_L s$, if $s \in [r]_L$. The next lemma is a direct consequence of the fact that $\sim_L$ is a syntactic congruence.

Lemma 3.3. *Let r be a rule which respects a language L. Every rule $s \in [r]_L$ respects L.*

Proof. Let $r = (u_1, u_2; v)$ and $s = (\tilde{u}_1, \tilde{u}_2; \tilde{v})$. Thus, $u_1 \sim_L \tilde{u}_1$, $u_2 \sim_L \tilde{u}_2$, and $v \sim_L \tilde{v}$. We will show that for all $\tilde{w}_1 = x_1\tilde{u}_1y_1 \in L$ and $\tilde{w}_2 = x_2\tilde{u}_2y_2 \in L$, we have $\tilde{z} = x_1\tilde{v}y_2 \in L$. Let $w_1 = x_1u_1y_1$, $w_2 = x_2u_2y_2$ and note that $w_1 \sim_L \tilde{w}_1$, $w_2 \sim_L \tilde{w}_2$; hence, $w_1, w_2 \in L$. Furthermore, $(w_1, w_2) \vdash_r x_1vy_2 = z \in L$ as r respects L and $\tilde{z} \in L$ as $z \sim_L \tilde{z}$. □

Consider a splicing $(x_1u_1y_1, x_2u_2y_2) \vdash_r x_1vy_2$ which respects a regular language L as shown in Figure 6 left side. The factors u_1y_1 and x_2u_2 may be relatively long but they do not occur as factors in the resulting word x_1vy_2. In particular, it is possible that two long words are spliced and the outcome is a relatively short word. Using Lemmas 3.2 and 3.3, we can find shorter words in L and a modified splicing rule which can be used to obtain x_1vy_2.

Figure 6: The factors u_1y_1 and x_2u_2 can be replaced by *short* words.

Lemma 3.4. *Let $r = (u_1, u_2; v)$ be a rule which respects a regular language L and $w_1 = x_1u_1y_1 \in L$, $w_2 = x_2u_2y_2 \in L$. There is a rule $s = (\tilde{u}_1, \tilde{u}_2; v)$ which respects L and words $\tilde{w}_1 = x_1\tilde{u}_1 \in L$, $\tilde{w}_2 = \tilde{u}_2y_2 \in L$ such that $|\tilde{u}_1|, |\tilde{u}_2| < |M_L|$. More precisely, $\tilde{u}_1 \in [u_1y_1]_L$ and $\tilde{u}_2 \in [x_2u_2]_L$. In particular, whenever $(w_1, w_2) \vdash_r x_1vy_2 = z$ respects L, then $(\tilde{w}_1, \tilde{w}_2) \vdash_s z$ respects L.*

Proof. By using Lemma 3.2 twice, the rule $(u_1y_1, x_2u_2; v)$ respects L. By Lemma 2.2, we can choose $\tilde{u}_1 \in [u_1y_1]_L$ and $\tilde{u}_2 \in [x_2u_2]_L$ as shortest words from the syntactic classes, respectively, such that $|\tilde{u}_1|, |\tilde{u}_2| < |M_L|$. Let $\tilde{w}_1 = x_1\tilde{u}_1 \in L$, $\tilde{w}_2 = \tilde{u}_2y_2 \in L$. Furthermore, by Lemma 3.3, $s = (\tilde{u}_1, \tilde{u}_2; v)$ respects L. □

Another way of modifying a splicing $(w_1, w_2) \vdash_r z$ is to extend the bridge of r to the left until it covers a prefix of w_1. Afterwards, we can use the same method we used in Lemma 3.4 and replace w_1 by a short word; see Figure 7. As the splicing operation is symmetric, we can also extend the bridge of r rightwards and replace w_2 by a short word, even though Lemma 3.5 does not explicitly state this.

Figure 7: The word $x_1u_1y_1$ can be replaced by a *short* word as long as we extend the bridge of the splicing rule accordingly.

Lemma 3.5. *Let $r = (u_1, u_2; v)$ be a rule which respects a regular language L and let $w_1 = x_1u_1y_1 \in L$. Every rule $s = (\tilde{w}, u_2; x_1v)$, where $\tilde{w} \in [w_1]_L \subseteq L$, respects L. In particular, there is a rule $s = (\tilde{w}, u_2; x_1v)$ such that $|\tilde{w}| < |M_L|$.*

Proof. By Lemma 3.2, we see that $(x_1u_1y_1, u_2; x_1v)$ respects L and, by Lemma 3.3, $s = (\tilde{w}, u_2; x_1v)$ respects L. If $\tilde{w} \in [w_1]_L$ is a shortest word from the set, then $|\tilde{w}| < |M_L|$, by Lemma 2.2. □

3.2 Series of Splicings

We are now investigating words which are created by a series of successive splicings which all respect a regular language L. Observe, that if a word is created by two (or more) successive splicings, but the bridges of the rules do not overlap in the generated word, then the order of these splicings is irrelevant. The notation in Remark 3.6 is the same as in the Figure 8.

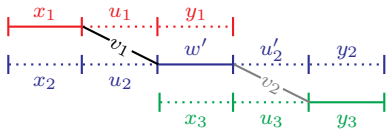

Figure 8: The word $x_1v_1w'v_2y_3$ can be created either by using the right splicing first or by using the left splicing first.

Remark 3.6. *Consider rules* $r = (u_1, u_2; v_1)$ *and* $s = (u_2', u_3; v_2)$ *and words* $w_1 = x_1u_1y_1$, $w_2 = x_2u_2w'u_2'y_2$, *and* $w_3 = x_3u_3y_3$. *The word* $z = x_1v_1w'v_2y_3$ *can be obtained by the splicings*

$$(w_1, w_2) \vdash_r x_1v_1w'u_2'y_2 = z', \qquad (z', w_3) \vdash_s z$$

as well as

$$(w_2, w_3) \vdash_s x_2u_2w'v_2y_3 = z'', \qquad (w_1, z'') \vdash_r z,$$

which makes the order of the splicing steps irrelevant.

Now, consider a word z which is created by two successive splicings from words $w_1 = x_1u_1y_1$, $w_2 = x_2u_2y_2$, and $w_3 = x_3u_3y_3$ as shown in Figure 9. If no factor of w_1 or of the bridge in the first splicing is a part of z, then we can find another splicing rule s such that $(w_3, w_2) \vdash_s z$ and the bridge of s is the bridge used in the second splicing.

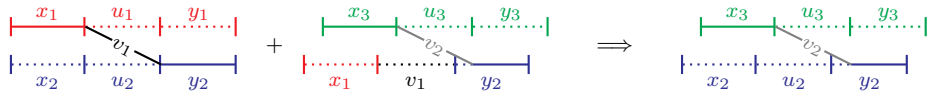

Figure 9: Two successive splicings can be replaced by one splicing in the case when the factor x_1 and the bridge v_1 do not contribute to the resulting word.

Lemma 3.7. *Let* L *be a language,* $w_i = x_iu_iy_i \in L$ *for* $i = 1, 2, 3$, *and* $r_1 = (u_1, u_2; v_1)$, $r_2 = (u_3, u_4; v_2)$ *be rules respecting* L. *If there are splicings*

$$(w_1, w_2) \vdash_{r_1} x_1v_1y_2 = w_4 = x_4u_4y_4, \qquad (w_3, w_4) \vdash_{r_2} x_3v_2y_4 = z$$

where y_4 *is a suffix of* y_2, *then there is a rule* $s = (u_3, \tilde{u}_2; v_2)$ *which respects* L *such that* $(w_3, w_2) \vdash_s z$.

Proof. First, we will prove that we may assume that the factors v_1 and u_4 match in w_4 by extending the bridge v_1 and the restriction sites u_1, u_2, u_4 (Lemma 3.2): let i, j, i', j' such that $w_{4[i,j]} = v_1$ and $w_{4[i',j']} = u_4$,

- if $i < i'$, extend u_4 in r_2 to the left by $i' - i$ letters,
- if $i > i'$, extend v_1 in r_1 to the left by $i - i'$ letters and extend u_1 accordingly, and
- extend v_1 in r_1 to the right by $j' - j$ letters and extend u_2 accordingly; note that $j' \geq j$ as y_4 is a suffix of y_2.

In addition to the factors u_1, u_2, u_4, v_1, we alter the factors x_1, x_4, y_2, accordingly; we call the newly obtained factors $\tilde{u}_1, \tilde{u}_2, \tilde{u}_4, \tilde{v}_1, \tilde{x}_1, \tilde{x}_4, \tilde{y}_2$, respectively. As a result of the extension we now have that $\tilde{x}_1 = \tilde{x}_4$, $\tilde{v}_1 = \tilde{u}_4$, and $\tilde{y}_2 = y_4$. The words

$$w_1 = \tilde{x}_1\tilde{u}_1y_1, \qquad w_2 = x_2\tilde{u}_2y_4, \qquad w_3 = x_3u_3y_3, \qquad w_4 = \tilde{x}_1\tilde{v}_1y_4$$

are the same as in the lemma, we just changed their factorizations. The rules $\tilde{r}_1 = (\tilde{u}_1, \tilde{u}_2; \tilde{v}_1)$ and $\tilde{r}_2 = (u_3, \tilde{v}_1; v_2)$ are the extended versions of the rules r_1 and r_2, respectively, which respect L. Furthermore,

$$(w_1, w_2) \vdash_{\tilde{r}_1} \tilde{x}_1\tilde{v}_1 y_4 = w_4, \qquad (w_3, w_4) \vdash_{\tilde{r}_2} x_3 v_2 y_4 = z.$$

Let $s = (u_3, \tilde{u}_2; v_2)$ and observe that $(w_3, w_2) \vdash_s x_3 v_2 y_4 = z$, as desired.

Next, let us prove that s respects L. If for all words $w_2' = x_2'\tilde{u}_2 y_2' \in L$ and $w_3' = x_3' u_3 y_3' \in L$ we have $x_3' v_2 y_2' \in L$, then s respects L. Indeed, we may splice

$$(w_1, w_2') \vdash_{\tilde{r}_1} \tilde{x}_1\tilde{v}_1 y_2', \qquad (w_3', \tilde{x}_1\tilde{v}_1 y_2') \vdash_{\tilde{r}_2} x_3' v_2 y_2'.$$

Because $\tilde{r}_1$ and $\tilde{r}_2$ respect L, the word $x_3' v_2 y_2'$ belongs to L. We conclude that s respects L. □

Consider a splicing system (J, S) and its generated language $L = L(J, S)$. Let n be the length of the longest word in J and let μ be the length-lexicographic largest word that is a component of a rule in S. Define $W_\mu = \{w \in \Sigma^* \mid w \leq_{\ell\ell} \mu\}$ as the set of all words that are at most as large as μ, in length-lexicographical order. Furthermore, let $I = \Sigma^{\leq n} \cap L$ be a set of axioms and let R be the set of rules

$$R = \left\{ r \in W_\mu^3 \;\middle|\; r \text{ respects } L \right\}.$$

It is not difficult to see that $J \subseteq I$, $S \subseteq R$, and $L = L(I, R)$. Whenever convenient, we may assume that a splicing language L is generated by a splicing system which is of the form of (I, R).

Consider the creation of some word $x_0 z y_0 \in L$ in a splicing system (I, R). We can trace back the generation of $x_0 z y_0$ to a word $x_k z y_k$ where the factor z is *affected* by splicing for the last time, i. e., the bridge in this splicing overlaps with the factor z in $x_k z y_k$; or the word $x_k z y_k$ belongs to I. This yields a degenerated splicing tree as discussed in Figure 4 in Section 2.2. Since the splicings which alter the prefixes x_i in this tree do not interfere with the splicings which alter the suffixes y_i, we can reorganize the tree as shown on the right in Figure 4, in accordance with Remark 3.6. The next lemma describes this creation of $x_0 z y_0 = z_0$ by k splicings in (I, R), and shows that we can choose the rules and words which are used to create z_0 from z_k such that the words and bridges of rules are not significantly longer than $\ell = \max\{|x_0|, |y_0|\}$.

Lemma 3.8. *Let L be a splicing language, let $m = |M_L|$, let $\ell, n \in \mathbb{N}$, such that for $I = \Sigma^{\leq n} \cap L$ and a rule set R we have $L = L(I, R)$. Let $z_0 = x_0 z y_0$ be a word with $|x_0|, |y_0| \leq \ell$.*

Suppose $z_0 = \tilde{z}_0 = \tilde{x}_0 z \tilde{y}_0$ (with $x_0 = \tilde{x}_0$, $y_0 = \tilde{y}_0$) is generated by a series of $\tilde{k}$ splicings from a word $\tilde{z}_{\tilde{k}} = \tilde{x}_{\tilde{k}} z \tilde{y}_{\tilde{k}}$ where either $\tilde{z}_{\tilde{k}} \in I$ or $\tilde{z}_{\tilde{k}}$ is created by a splicing $(\tilde{w}_{\tilde{k}+1}, \tilde{w}_{\tilde{k}+2}) \vdash_{\tilde{s}} \tilde{z}_{\tilde{k}}$ with $\tilde{w}_{\tilde{k}+1}, \tilde{w}_{\tilde{k}+2} \in L$, $\tilde{s} \in R$, and the bridge of $\tilde{s}$ overlaps with z in $\tilde{z}_{\tilde{k}}$. Furthermore, for $i = 1, \ldots, \tilde{k}$ the intermediate splicings are either

- *(i) $(\tilde{w}_i, \tilde{z}_i) \vdash_{\tilde{r}_i} \tilde{x}_{i-1} z \tilde{y}_{i-1} = \tilde{z}_{i-1}$, where $\tilde{w}_i \in L$, $\tilde{r}_i \in R$, $\tilde{y}_{i-1} = \tilde{y}_i$, and the bridge of $\tilde{r}_i$ is covered by the prefix $\tilde{x}_{i-1}$ or*
- *(ii) $(\tilde{z}_i, \tilde{w}_i) \vdash_{\tilde{r}_i} \tilde{x}_{i-1} z \tilde{y}_{i-1} = \tilde{z}_{i-1}$, where $\tilde{w}_i \in L$, $\tilde{r}_i \in R$, $\tilde{x}_{i-1} = \tilde{x}_i$, and the bridge of $\tilde{r}_i$ is covered by the suffix $\tilde{y}_{i-1}$.*

Then, there are also rules and words that generate $z_0 = \tilde{z}_0$ by a series of $k \leq \tilde{k}$ splicings from a word $z_k = x_k z y_k$ where either $z_k \in I$ or z_k is created by a splicing $(w_{k+1}, w_{k+2}) \vdash_s z_k$ that respects L and the bridge of s overlaps with z in z_k. Furthermore, there exists $k' \leq k$ such that

- *(i') for $i = 1, \ldots, k'$ the intermediate splicings are $(w_i, z_i) \vdash_{r_i} x_{i-1} z y_{i-1} = z_{i-1}$, where $w_i \in L$, r_i respects L, $y_{i-1} = y_i$, and the bridge of r_i is covered by the prefix x_{i-1}, and*
- *(ii') for $i = k'+1, \ldots, k$ the intermediate splicings are $(z_i, w_i) \vdash_{r_i} x_{i-1} z y_{i-1} = z_{i-1}$, where $w_i \in L$, r_i respects L, $x_{i-1} = x_i$, and the bridge of r_i is covered by the suffix y_{i-1}.*
- *(iii') For $i = 1, \ldots, k$ the following bounds apply: $|x_i|, |y_i| < \ell + 2m$, $|w_i| < m$, $r_i \in \Sigma^{<2m} \times \Sigma^{<2m} \times \Sigma^{<\ell+m}$. In particular, if $n \geq m$, then $w_1, \ldots, w_k \in I$. Moreover, if μ is the length-lexicographically largest component in $\tilde{s}$, then $s \in W_\mu^3$.*

Proof. Note that if $\tilde{k} = 0$, then the statements are trivially true. The fact that we can reorganize the splicing tree such that it looks like in Figure 4 on the right follows by Remark 3.6, that is, for some $\tilde{k}' < \tilde{k}$ all the splicing steps $i = 1, \dots, \tilde{k}'$ are of the form (i) and all the splicing steps $i = \tilde{k}' + 1, \dots, \tilde{k}$ are of the form (ii). If for $2 \leq i \leq \tilde{k}'$ we have that no factor of $\tilde{w}_i$ or the bridge in the i-th splicing step $(\tilde{w}_i, \tilde{z}_i) \vdash \tilde{z}_{i-1} = \tilde{x}_{i-1} z \tilde{y}_{i-1}$ occurs in $\tilde{x}_{i-2}$, then we can combine the i-th and $(i-1)$-th splicing step into one splicing step according to Lemma 3.7. Successively eliminating all these cases and the symmetric cases for $\tilde{k}' + 2 \leq i \leq \tilde{k}$, yields a shorter series of k splicing steps and an integer $k' \leq k$ such that: for $i = 1, \dots, k'$ the i-th splicing step is of the form (i) and a factor of $\tilde{w}_i$ or of the bridge in the i-th splicing appears in $x_0 = \tilde{x}_0$; and for $i = k' + 1, \dots, k$ the i-th splicing step is of the form (ii) and a factor of $\tilde{w}_i$ or of the bridge in the i-th splicing appears in $y_0 = \tilde{y}_0$. Note that this reorganization also implies $\tilde{x}_{k'+1} = \tilde{x}_{k'+2} = \cdots = \tilde{x}_k$ and $y_0 = \tilde{y}_0 = \tilde{y}_1 = \cdots = \tilde{y}_{k'}$.

We now have that $z_0 = \tilde{z}_0$ is generated by a series of k splicings from a word $\tilde{z}_k = \tilde{x}_k z \tilde{y}_k$ where either $\tilde{z}_k \in I$ or $\tilde{z}_k$ is created by a splicing $(\tilde{w}_{k+1}, \tilde{w}_{k+2}) \vdash_{\tilde{s}} \tilde{z}_k$ with $\tilde{w}_{k+1}, \tilde{w}_{k+2} \in L$, $\tilde{s} \in R$, and the bridge of $\tilde{s}$ overlaps with z in $\tilde{z}_k$. Furthermore, there exists $k' \leq k$ such that

(i) for $i = 1, \dots, k'$ the intermediate splicings are $(\tilde{w}_i, \tilde{z}_i) \vdash_{\tilde{r}_i} \tilde{x}_{i-1} z \tilde{y}_{i-1} = \tilde{z}_{i-1}$, where $\tilde{w}_i \in L$, $\tilde{r}_i$ respects L, $\tilde{y}_{i-1} = \tilde{y}_i$, the bridge of $\tilde{r}_i$ is covered by the prefix $\tilde{x}_{i-1}$, and a non-empty factor of $\tilde{w}_i$ or of the bridge of $\tilde{r}_i$ occurs in x_0; and

(ii) for $i = k'+1, \dots, k$ the intermediate splicings are $(\tilde{z}_i, \tilde{w}_i) \vdash_{\tilde{r}_i} \tilde{x}_{i-1} z \tilde{y}_{i-1} = \tilde{z}_{i-1}$, where $\tilde{w}_i \in L$, $\tilde{r}_i$ respects L, $\tilde{x}_{i-1} = \tilde{x}_i$, the bridge of $\tilde{r}_i$ is covered by the suffix $\tilde{y}_{i-1}$, and a non-empty factor of $\tilde{w}_i$ or of the bridge of $\tilde{r}_i$ occurs in y_0.

We will continue to modify and replace all words and rules which are marked by $\tilde{}$ in order to prove the length bounds which are stated in (iii').

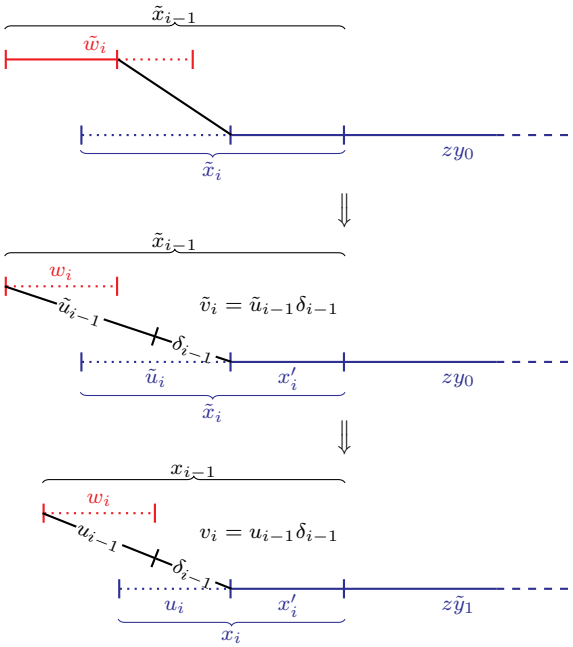

Figure 10: The i-th splicing step with $1 \leq i \leq k'$ in the proof of Lemma 3.8: the upper figure shows a general case where the words $\tilde{w}_i$ and $\tilde{x}_i z y_0$ are spliced in order to obtain the new word $\tilde{x}_{i-1} z y_0$; the middle figure shows the splicing after applying Lemmas 3.5 and 3.2; and the lower figure shows the splicing after we replace $\tilde{u}_i$ and $\tilde{u}_{i-1}$ by syntactically equivalent, short words u_i and u_{i-1}, respectively.

Let us consider the splicings of the form (i) which are the steps $i = 1, \dots, k'$. In order to prove

the length bounds stated in (iii'), we make several modifications in the series of splicings which generates $z_0 = x_0 z y_0$ by using the techniques that we established beforehand; these modifications are depicted in Figure 10. Let us start with the general case where $\tilde{w}_i$ and $\tilde{x}_i z y_0$ are spliced by the rule $\tilde{r}_i$ in order to obtain the new word $\tilde{x}_{i-1} z y_0$ (Figure 10 top). By applying Lemma 3.5, this splicing can be modified such that the bridge $\tilde{v}_i$ in the splicing becomes a prefix of $\tilde{x}_{i-1}$ and $\tilde{w}_i$ is replaced by $w_i \in [\tilde{w}_i]_L$ with $|w_i| < m$; furthermore, w_i is the left restriction site in this splicing. Let $\tilde{u}_i$ denote the right restriction site in this splicing, and by applying Lemma 3.2, extend $\tilde{u}_i$ such that it becomes a prefix of $\tilde{x}_i$. Therefore, we use the splicing rule $(w_i, \tilde{u}_i; \tilde{v}_i)$ in the i-th splicing step. We make these modification for all splicing steps $i = 1, \ldots, k'$ and we let x'_i such that $\tilde{x}_i = \tilde{u}_i x'_i$. Furthermore, we define $\tilde{u}_0 = \varepsilon$ and $x'_0 = x_0$. Since we established that a factor of $\tilde{w}_i$ or of the bridge in the i-th splicing step occurs in x_0 (due to Lemma 3.7), we now have that a factor of the bridge $\tilde{v}_i$ occurs in x_0 (as w_i does not occur in x_{i-1}). This implies that $\tilde{u}_{i-1}$ is a proper prefix of $\tilde{v}_i$ and we may write $\tilde{v}_i = \tilde{u}_{i-1}\delta_{i-1}$ for a suitable non-empty factor δ_{i-1} (Figure 10 middle). Note that we have $x'_{i-1} = \delta_{i-1} x'_i$ and $\tilde{x}_{i-1} = \tilde{u}_{i-1} x'_{i-1} = \tilde{v}_i x'_i$ for $i = 1, \ldots, k'$. As $|x_0| \leq \ell$, the length of x'_i is bounded by ℓ for all $i = 0, \ldots, k'$.

For $i = 1, \ldots, k'-1$ we replace $\tilde{u}_i$ by a shortest word $u_i \in [\tilde{u}_i]_L$; note that this does not change the fact that all rules respect L (Lemma 3.3). We also replace the prefix $\tilde{u}_i$ of $\tilde{x}_i$ and $\tilde{v}_{i+1}$ by this factor in order to obtain the words x_i and v_{i+1}, respectively (Figure 10 bottom). We did not change $\tilde{v}_1 = v_1$ since this is a prefix of $\tilde{x}_0 = x_0$ which we do not want to alter; yet, note that $|v_1| \leq |x_0| \leq \ell$. Therefore, $|x_i| < |x'_i| + m \leq \ell + m$ and $r_i = (w_i, u_i; v_i) \in \Sigma^{<m} \times \Sigma^{<m} \times \Sigma^{<\ell+m}$ for $i = 1, \ldots, k'-1$ (Lemma 2.2). We do not change $\tilde{u}_{k'}$ or $\tilde{x}_{k'}$ yet as this may affect the splicing stop $(\tilde{w}_k, \tilde{w}_{k+2}) \vdash_{\tilde{s}} \tilde{z}_k$ if it exists. Note that, for $i = 1, \ldots, k'-1$, we have actually proven a stronger bound than claimed in statement (iii') of Lemma 3.8.

Even though we have not proven the bound for $(w_{k'}, \tilde{u}_{k'}; v_{k'})$ yet, we have already established $(w_{k'}, \tilde{u}_{k'}; v_{k'}) \in \Sigma^{<m} \times \Sigma^* \times \Sigma^{<\ell+m}$. We will replace $\tilde{u}_{k'}$ by a word $u_{k'}$ of length less than $2m$ and we replace $\tilde{x}_{k'} = \tilde{x}_k$ by $x_k = x_{k'} = u_{k'} x'_{k'}$ accordingly in order to prove the claim for $i = k'$. If $\tilde{x}_k z \tilde{y}_k \in I$ we replace $\tilde{u}_{k'}$ by a shortest word $u_{k'} \in [\tilde{u}_{k'}]_L$ and the claim holds (Lemma 2.2). Otherwise (if $z_{k'} \notin I$), $(\tilde{w}_{k+1}, \tilde{w}_{k+2}) \vdash_{\tilde{s}} \tilde{x}_k z \tilde{y}_k$ where $\tilde{s} = (u, u', \tilde{v})$, $\tilde{w}_{k+1} = \tilde{x} u y'$, $w_{k+2} = x' u' \tilde{y}$, and $\tilde{u}_{k'} x'_{k'} z \tilde{y}_k = \tilde{x} \tilde{v} \tilde{y}$. Because we required that v overlaps with z in this factorization, $\tilde{u}_{k'}$ has to be a prefix of $\tilde{x}\tilde{v}$ In the case when $\tilde{v}$ does not overlap with the prefix $\tilde{u}_{k'}$, we replace the factor $\tilde{u}_{k'}$ by $u_{k'} \in [\tilde{u}_{k'}]_L$ with $|u_{k'}| < m$ (Lemma 2.2) and let $v = \tilde{v}$. If $\tilde{v}$ and the prefix $\tilde{u}_{k'}$ overlap, let $u_{k'} = \tilde{x}\tilde{\alpha}$ and $\tilde{v} = \tilde{\alpha}\beta$; that is, $\tilde{\alpha}$ is the overlap. Let $x \in [\tilde{x}]_L$ and $\alpha \in [\tilde{\alpha}]_L$ with $|x|, |\alpha| < m$; furthermore, let $u_{k'} = x\alpha$ and $v = \alpha\beta$. For both cases we obtain that $v \leq_{\ell\ell} \tilde{v}$ and hence $v \in W_\mu$. We let $w_{k+1} = xuy'$ and we obtain that $(w_{k+1}, \tilde{w}_{k+2}) \vdash_{s'} x_k z \tilde{y}_k$ with $s' = (u, u'; v) \in W_\mu^3$ as desired. Furthermore, $|u_{k'}| < 2m$ and $|x_k| < \ell + 2m$. This concludes the proof of statement (iii') of Lemma 3.8 for $i = 1, \ldots, k'$. Analogously, the statement can be proven for $i = k'+1, \ldots, k$. □

3.3 Proof of Theorem 3.1

Let L be a splicing language and $m = |M_L|$. Throughout this section, by $\sim$ we denote the equivalence relation $\sim_L$ and by $[\,\cdot\,]$ we denote the corresponding equivalence classes $[\,\cdot\,]_L$.

Recall that Theorem 3.1 claims that the splicing system (I, R) with $I = \Sigma^{<m^2+6m} \cap L$ and

$$R = \left\{ r \in \Sigma^{<2m} \times \Sigma^{<2m} \times \Sigma^{<m^2+10m} \;\middle|\; r \text{ respects } L \right\}$$

generates L. The proof is divided in two parts. In the first part, Lemma 3.9, we prove that L is generated by a splicing system (I, R') where all restriction sites of rules in R' are shorter than $2m$, but we do not care about the lengths of the bridges. The second part will then conclude the proof by showing that there are no rules in R' with bridges of length greater than or equal to $m^2 + 10m$ which are essential for the generation of the language L by splicing.

Lemma 3.9. *Let L, m, and I as above. There is $n \in \mathbb{N}$ such that the splicing system (I, R') with*

$$R' = \left\{ r \in \Sigma^{<2m} \times \Sigma^{<2m} \times \Sigma^{\leq n} \;\middle|\; r \text{ respects } L \right\}$$

generates the language $L = L(I, R')$.

Proof. As $I \subseteq L$ and every rule in R' respects L, it is clear that $L(I, R') \subseteq L$ for any n; we only need to prove the converse inclusion.

As L is a splicing language, $L = L(J, S)$ for some splicing system (J, S). Let n be larger than the length of every bridge of every rule in S and $n \geq 4m^2$. In order to prove $L \subseteq L(I, R')$ we use induction on the length of words in L. For all $w \in L$ with $|w| < m^2+6m$ we have $w \in I \subseteq L(I, R')$, by definition.

Consider $w \in L$ with $|w| \geq m^2 + 6m$. The induction hypothesis states that every word $w' \in L$ with $|w'| < |w|$ belongs to $L(I, R')$. Factorize

$$w = x_0\alpha\beta\gamma\delta y_0$$

such that $|x_0|, |y_0| = 3m$, $|\alpha\beta\gamma| = m^2$, $\beta \neq \varepsilon$, $\alpha \sim \alpha\beta$, $\gamma \sim \beta\gamma$ (see Pumping Lemma 2.1), and δ covers the remainder of the word w. For the correctness of the following steps in this proof it is important that the factors x_0, y_0, and $\alpha\beta\gamma$ each have a fixed length and only the length of δ is flexible.

The proof idea is as follows: We pump up the factor β in the word w to β^j in order to obtain a word $\tilde{w}$. We pick j large enough such that no word in J can contain the factor $z = \alpha\beta^j\gamma\delta$; in particular, $\tilde{w} \notin J$. Therefore, it has to be created by a series of splicings from other words in L and at some point during the creation of $\tilde{w}$ by splicing the bridge of the used splicing rule must overlap with z; the series of splicings is of the form as described in Lemma 3.8 and Figure 4. Next, we pump down the factor β^j in $\tilde{w}$ again in order to obtain the original word w. We adapt the series of splicings, that created $\tilde{w}$, in order to obtain series of splicings which creates w; then, we ensure that all words used in this series of splicings are shorter than w and all rules belong to R'. Using the induction hypothesis, this will conclude the proof.

Choose j sufficiently large ($j > n$ and J does not contain words of length j or more). We let $z = \alpha\beta^j\gamma\delta$ and investigate the creation of $x_0zy_0 \in L$. As z cannot be a factor of a word in J, every word in L which contains z is created by some splicing in (J, S). Thus, we can trace back the creation of x_0zy_0 by splicing to the point where the factor z is affected for the last time. Let $z_0 = x_0zy_0$, be created by k splicings from a word $z_k = x_kzy_k$ where x_kzy_k is created by a splicing $(w_{k+1}, w_{k+2}) \vdash_s z_k$ with $w_{k+1}, w_{k+2} \in L$, $s \in S$, and the bridge of s overlaps with z in z_k. Furthermore, for $i = 1, \ldots, k$ the intermediate splicings are either

(i) $(w_i, z_i) \vdash_{r_i} x_{i-1}zy_{i-1} = z_{i-1}$, where $w_i \in L$, $r_i \in S$, $y_{i-1} = y_i$, and the bridge of r_i is covered by the prefix x_{i-1} or

(ii) $(z_i, w_i) \vdash_{r_i} x_{i-1}zy_{i-1} = z_{i-1}$, where $w_i \in L$, $r_i \in S$, $x_{i-1} = x_i$, and the bridge of r_i is covered by the suffix y_{i-1}.

Following Lemma 3.8 (with $\ell = 3m$), we may assume that $w_1, \ldots, w_k \in I$, $r_1, \ldots, r_k \in \Sigma^{<2m} \times \Sigma^{<2m} \times \Sigma^{<4m}$, thus $r_1, \ldots, r_k \in R'$, and $|x_k|, |y_k| < 5m$. Furthermore, we may use the same words and rules in order to create $w = x_0\alpha\beta\gamma\delta y_0$ from $x_k\alpha\beta\gamma\delta y_k$ by splicing, i. e., if $x_k\alpha\beta\gamma\delta y_k$ belongs to $L(I, R')$, so does $w = z_0$ (as well as all intermediate words $x_i\alpha\beta\gamma\delta y_i$ for $i = 1, \ldots, k-1$).

Now, consider the first splicing $(w_{k+1}, w_{k+2}) \vdash_s z_k = x_kzy_k$. Applying restriction site extensions (Lemma 3.4), we may assume $s = (u_1, u_2; v)$ such that $w_{k+1} = xu_1$ and $w_{k+2} = u_2y$. Hence,

$$z_1 = xvy = x_k\alpha\beta^j\gamma\delta y_k$$

where x is a proper prefix of $x_k\alpha\beta^j\gamma\delta$ and y is a proper suffix of $\alpha\beta^j\gamma\delta y_k$ because we required that the bridge v of s overlaps with $\alpha\beta^j\gamma\delta$ in z_k.

We will now pump down the factor β^j to β in order to obtain the words $\tilde{x}$, $\tilde{v}$, $\tilde{y}$ from x, v, y, respectively, as follows:

1. If v overlaps with β^j but does neither cover α nor γ, extend v (Lemma 3.2) such that $v = \alpha\beta^j\gamma$. After possibly using this extension, the factor $\alpha\beta^j\gamma$ is covered by xv or vy (or both).

2. If $\alpha\beta^j$ or $\beta^j\gamma$ is covered by one of x, v, or y, then replace this factor by $\alpha\beta$ or $\beta\gamma$, respectively. Skip remaining steps.

3. If $\alpha\beta^j\gamma$ is covered by xv, we can factorize

$$x = x_1\alpha\beta^{j_1}\beta_1, \qquad v = \beta_2\beta^{j_2}\gamma v'$$

where $\beta_1\beta_2 = \beta$ and $j_1 + j_2 + 1 = j$. The results of pumping down are the words $\tilde{x} = x_1\alpha\beta_1$, $\tilde{v} = \beta_2\gamma v'$, and $\tilde{y} = y$. Skip last step.

4. The remaining case is that $\alpha\beta^j\gamma$ is covered by vy. We can factorize

$$v = v'\alpha\beta^{j_1}\beta_1, \qquad y = \beta_2\beta^{j_2}\gamma\delta y_1$$

where $\beta_1\beta_2 = \beta$ and $j_1 + j_2 + 1 = j$. The results of pumping down are the words $\tilde{x} = x$, $\tilde{v} = v'\alpha\beta_1$, $\tilde{y} = \beta_2\gamma\delta y_1$.

Let $\tilde{u}_1$ and $\tilde{u}_2$ be the restriction sites of s that may have been altered due to the extension of v and, by Lemma 3.4, assume $|\tilde{u}_1|, |\tilde{u}_2| < m$. If we used an extension for v, then $|\tilde{v}| \leq m^2$; otherwise $|\tilde{v}| \leq |v|$. No matter whether we used an extension or not, $t = (\tilde{u}_1, \tilde{u}_2; \tilde{v}) \in R'$ and $(\tilde{x}\tilde{u}_1, \tilde{u}_2\tilde{y}) \vdash_t x_k\alpha\beta\gamma\delta y_k$ as desired. Observe that $\tilde{x}$ is a prefix of $x_k\alpha\beta\gamma\delta$ and $\tilde{y}$ is a suffix of $\alpha\beta\gamma\delta y_k$ and recall that $|x_k|, |y_k| < 5m$. Therefore,

$$|\tilde{x}\tilde{u}_1| \leq |x_k| + |\alpha\beta\gamma\delta| + |\tilde{u}_k| < |\alpha\beta\gamma\delta| + 6m = |w|$$

and, symmetrically, $|\tilde{y}\tilde{u}_2| < |w|$. By induction hypothesis we obtain that $\tilde{x}\tilde{u}_1$ and $\tilde{u}_2\tilde{y}$ belong to $L(I, R')$. We conclude that $x_k\alpha\beta\gamma\delta y_k$ as well as $w = x_0\alpha\beta\gamma\delta y_0$ belong to $L(I, R')$. □

We are now prepared to prove the main result of this section.

Proof of Theorem 3.1. Recall that for a splicing language L with $m = |M_L|$, we intend to prove that the splicing system (I, R) with $I = \Sigma^{<m^2+6m} \cap L$ and

$$R = \left\{ r \in \Sigma^{<2m} \times \Sigma^{<2m} \times \Sigma^{<m^2+10m} \;\middle|\; r \text{ respects } L \right\}$$

generates the language $L = L(I, R)$. Obviously, $L(I, R) \subseteq L$. By Lemma 3.9, there is a finite set of rules $R' \subseteq \Sigma^{<2m} \times \Sigma^{<2m} \times \Sigma^*$ such that $L(I, R') = L$.

For a word μ we define $W_\mu = \{w \in \Sigma^* \mid w \leq_{\ell\ell} \mu\}$, as we did before. Define the set of rules where every bridge is length-lexicographically bounded by some word μ as

$$R_\mu = \left\{ r \in \Sigma^{<2m} \times \Sigma^{<2m} \times W_\mu \;\middle|\; r \text{ respects } L \right\}$$

and the language $L_\mu = L(I, R_\mu)$; clearly, $L_\mu \subseteq L$. For two words $\mu \leq_{\ell\ell} v$ we have $R_\mu \subseteq R_v$, and hence, $L_\mu \subseteq L_v$. Thus, if $L_\mu = L$ for some word μ, then for all words v with $\mu \leq_{\ell\ell} v$, we have $L_v = L$. As $L = L(I, R')$, there exists a word μ such that $L_\mu = L$. Now, we fix μ to be the smallest word, in the length-lexicographic order, such that $L_\mu = L$. Note that if $|\mu| < m^2 + 10m$, then $R_\mu \subseteq R$ and $L = L_\mu \subseteq L(I, R)$. For the sake of obtaining a contradiction assume $|\mu| \geq m^2 + 10m$. Let ν be the next-smaller word than μ, in the length-lexicographic order, and let $S = R_\nu$. Note that $L(I, S) \subsetneq L$ and $R_\mu \setminus S$ contains only rules whose bridges are μ.

Choose w from $L \setminus L(I, S)$ as a shortest word, i. e., for all $w' \in L$ with $|w'| < |w|$, we have $w' \in L(I, S)$. Factorize

$$w = x_0 z y_0$$

with $|x_0| = |y_0| = 3m$ and let z cover the remainder of word w. Factorize

$$\mu = \delta_1\alpha\beta\gamma\delta_2$$

with $|\delta_1|, |\delta_2| \geq 5m$, $|\alpha\beta\gamma| = m^2$, $\beta \neq \varepsilon$, $\alpha \sim \alpha\beta$, and $\gamma \sim \beta\gamma$; see Pumping Lemma 2.1.

The proof idea is as follows: We replace each factors $\alpha\beta\gamma$ in the factor z of word $w = x_0zy_0$ by $\alpha\beta^j\gamma$, for a large j, in order to obtain a word $\tilde{w} = x_0\tilde{z}y_0 \sim w$ as described in Lemma 2.3. As in the proof of Lemma 3.9, $\tilde{w}$ has to be created by a series of splicings in (I, R_μ) and at some point during the creation of $\tilde{w}$ by splicing the bridge of the used splicing rule must overlap with the factor $\tilde{z}$; the series of splicings is of the form as described in Lemma 3.8 and Figure 4. If a splicing rule from $R_\mu \setminus S$ is used during the creation of of $\tilde{w}$, then the bridge of this splicing has to overlap with a factor $\alpha\beta^j\gamma$ such that we can replace this splicing by using a different rule. Next, we pump down all the factors $\alpha\beta^j\gamma$ in $\tilde{w}$ again in order to obtain the original word w. We adapt the series of splicings, that created $\tilde{w}$, in order to obtain series of splicings which creates w; then, we ensure that all words used in this series of splicings are shorter than w and all rules belong to S. This will contradict, that w is a shortest word in $L(I, R_\mu) \setminus L(I, S)$.

Let j be a sufficiently large even number ($j > 4|\mu| + |z|$ will do). According to Lemma 2.3 and as $\alpha\beta\gamma \sim \alpha\beta^j\gamma$, there exists a word $\tilde{z}$, which is obtained by successively applying factor replacements $\alpha\beta\gamma \mapsto \alpha\beta^j\gamma$ in z, such that $\tilde{z} \sim z$ and for all factors $\tilde{z}_{[k;k']} = \alpha\beta\gamma$ in $\tilde{z}$

(a) $\alpha\beta^{j/2}$ is a factor of $\tilde{z}$ starting at position $\tilde{z}_{[k]}$ or

(b) $\beta^{j/2}\gamma$ is a factor of $\tilde{z}$ ending at position $\tilde{z}_{[k']}$.

In particular, if $\delta_1\alpha\beta\gamma\delta_2$ is a factor of $\tilde{z}$, then (a) $\gamma\delta_2$ is a prefix of a word in β^+ or (b) $\delta_1\alpha$ is a suffix of a word in β^+. Because $z \sim \tilde{z}$ we have $x\tilde{z}y \in L$.

Let us trace back the creation of $x_0\tilde{z}y_0 \in L$ by splicing in (I, R_μ) to a word $x_k\tilde{z}y_k$ where either $x_k\tilde{z}y_k \in I$ or where $x_k\tilde{z}y_k$ is created by a splicing that affects $\tilde{z}$. Let $z_0 = \tilde{w} = x_0\tilde{z}y_0$ be created by k splicings from a word $z_k = x_k\tilde{z}y_k$ where either $x_k\tilde{z}y_k \in I$ or $x_k\tilde{z}y_k$ is created by a splicing $(w_{k+1}, w_{k+2}) \vdash_s z_k$ with $w_{k+1}, w_{k+2} \in L$, $s \in R_\mu$, and the bridge of s overlaps with $\tilde{z}$. Furthermore, for $i = 1, \ldots, k$ the intermediate splicings are either

(i) $(w_i, z_i) \vdash_{r_i} x_{i-1}\tilde{z}y_{i-1} = z_{i-1}$, where $w_i \in L$, $r_i \in R_\mu$, $y_{i-1} = y_i$, and the bridge of r_i is covered by the prefix x_{i-1} or

(ii) $(z_i, w_i) \vdash_{r_i} x_{i-1}\tilde{z}y_{i-1} = z_{i-1}$, where $w_i \in L$, $r_i \in R_\mu$, $x_{i-1} = x_i$, and the bridge of r_i is covered by the suffix y_{i-1}.

Following Lemma 3.8 (with $\ell = 3m$), we may assume that $w_1, \ldots, w_k \in I$, $r_1, \ldots, r_k \in \Sigma^{<2m} \times \Sigma^{<2m} \times \Sigma^{<4m}$, thus $r_1, \ldots, r_k \in S$, and $|x_k|, |y_k| < 5m$. Moreover, we may use the same words and rules in order to create $w = x_0zy_0$ from x_kzy_k by splicing. As w does not belong to $L(I, S)$, the word x_kzy_k (or any other word x_izy_i for $i = 1, \ldots, k$) must not belong to $L(I, S)$ either. Note that $z_k \notin I$, because if $z_k = x_k\tilde{z}y_k$ were in I, then $x_kzy_k \in I$ as well, as z is at most as long as $\tilde{z}$.

Therefore, z_k is created by a splicing $(w_{k+1}, w_{k+2}) \vdash_s z_k$ where $s = (u_1, u_2; v)$, $w_{k+1} = xu_1$, and $w_{k+2} = u_2y$ where $|u_1|, |u_2| < m$, by Lemma 3.4. We have

$$z_k = x_k\tilde{z}y_k = xvy$$

where x is a proper prefix of $x_k\tilde{z}$ and y is a proper suffix of $\tilde{z}y_k$ because we required that the bridge v of s overlaps with $\tilde{z}$ in z_k. Recall that either $s \in S$ or $v = \mu$.

However, we will see next that if $v = \mu$, there is also a rule $\tilde{s} \in S$ and slightly modified words which can be used in order to create $x_k\tilde{z}y_k$ by splicing. If $v = \mu$, then $\mu = \delta_1\alpha\beta\gamma\delta_2$ is a factor of z_k. As $|\delta_1|, |\delta_2| \geq 5m > |x_k|, |y_k|$, the factor $\alpha\beta\gamma$ is covered by $\tilde{z}$ and, as such, (a) α is succeeded by $\beta^{j/2}$ or (b) γ is preceded by $\beta^{j/2}$. Due to symmetry, we only consider the former case, in which $\gamma\delta_2$ is a prefix of a word in β^+. Let us shorten the bridge v such that $\tilde{s} = (u_1, u_2; \delta_1\alpha\gamma\delta_2)$. Note that $\tilde{s} \in S$ (as $\alpha \sim \alpha\beta$ and by Lemma 3.3). Furthermore, as j is large enough, $y = \beta_2\beta^\ell y'$ where $\ell \geq |\gamma|$ and β_2 is the suffix of β such that $\gamma\delta_2\beta_2 \in \beta^+$. This implies that $\beta_2\gamma$ is a prefix of y, which allows us to add an additional β. Therefore, $(w_0, u_2\beta_2\beta^{\ell+1}y') \vdash_{\tilde{s}} z_k$ where $u_2\beta_2\beta^{\ell+1}y' \in L$. This observation justifies the assumption that $v \neq \mu$ and $s \in S$ which we will make for the remainder of the proof.

Recall that $\tilde{z}$ was obtained from z by successively pumping up factors $\alpha\beta\gamma$ to $\alpha\beta^j\gamma$. Now, we will pump down the factors $\alpha\beta^j\gamma$ to $\alpha\beta\gamma$ in $\tilde{z}$ again. At every position where we pumped up

before, we are now pumping down in reverse order, in order to obtain the factors $\tilde{x}$, $\tilde{v}$, $\tilde{y}$ from the factors x, v, y, respectively. The pumping in each step is done as in the proof of Lemma 3.9:

1. If v overlaps with β^j but does neither cover α nor γ, extend v (Lemma 3.2) such that $v = \alpha\beta^j\gamma$. After possibly using this extension, the factor $\alpha\beta^j\gamma$ is covered by xv or vy (or both).

2. If $\alpha\beta^j$ or $\beta^j\gamma$ is covered by one of x, v, or y, then replace this factor by $\alpha\beta$ or $\beta\gamma$, respectively. Skip remaining steps.

3. If $\alpha\beta^j\gamma$ is covered by xv, we can factorize

$$x = x'\alpha\beta^{j_1}\beta_1, \qquad v = \beta_2\beta^{j_2}\gamma v'$$

where $\beta_1\beta_2 = \beta$ and $j_1 + j_2 + 1 = j$. The results of pumping down are the words $x'\alpha\beta_1$, $\beta_2\gamma v'$, and y. Skip last step.

4. The remaining case is that $\alpha\beta^j\gamma$ is covered by vy. We can factorize

$$v = v'\alpha\beta^{j_1}\beta_1, \qquad y = \beta_2\beta^{j_2}\gamma y'$$

where $\beta_1\beta_2 = \beta$ and $j_1 + j_2 + 1 = j$. The results of pumping down are the words x, $v'\alpha\beta_1$, and $\beta_2\gamma y'$.

Let $\tilde{u}_1$ and $\tilde{u}_2$ be the restriction sites of s that may have been altered due to the extension of v and, by Lemma 3.4, assume $|\tilde{u}_1|, |\tilde{u}_2| < m$. If we used an extension for v in at least one of the steps, then $|\tilde{v}| \leq m^2$; otherwise $|\tilde{v}| \leq |v|$. No matter whether we used an extension or not, $t = (\tilde{u}_1, \tilde{u}_2; \tilde{v}) \in S$ and $(\tilde{x}\tilde{u}_1, \tilde{u}_2\tilde{y}) \vdash_t x_k z y_k$ as desired. Observe that $\tilde{x}$ is a prefix of $x_k z$ and $\tilde{y}$ is a suffix of zy_k and recall that $|x_1|, |y_1| < 5m$. Therefore,

$$|\tilde{x}\tilde{u}_1| \leq |x_k| + |\alpha\beta\gamma\delta| + |\tilde{u}_1| < |\alpha\beta\gamma\delta| + 6m = |w|$$

and, symmetrically, $|\tilde{y}\tilde{u}_2| < |w|$. Since all words which are shorter than w belong to $L(I, S)$, we have $\tilde{x}\tilde{u}_1, \tilde{u}_2\tilde{y} \in L(I, S)$. We conclude that $x_k z y_k$ as well as w belong to $L(I, S)$ — the desired contradiction. □

4 The Case of Classical Splicing

In this section, we consider the splicing operation as defined in [18]. This is the most commonly used definition for splicing in formal language theory. The notation we use has been employed in previous papers, see for example, [2, 9]. Throughout this section, a quadruplet of words $r = (u_1, v_1; u_2, v_2) \in (\Sigma^*)^4$ is called a *(splicing) rule*. The words u_1v_1 and u_2v_2 are called *left* and *right restriction site* of r, respectively, and the word u_1v_2 is called the *paste site*. This splicing rule can be applied to two words $w_1 = x_1u_1v_1y_1$ and $w_2 = x_2u_2v_2y_2$, that each contain one of the restriction sites, in order to create the new word $z = x_1u_1v_2y_2$, which contains the paste site; see Figure 11. This operation is called *splicing* and it is denoted by $(w_1, w_2) \vdash_r z$. The *splicing position* of this splicing is $z_{[|x_1u_1|]}$; that is the position between the factors x_1u_1 and v_2y_2 in z.

Just as in Section 3, for a rule r we define the *splicing operator* σ_r such that for a language L

$$\sigma_r(L) = \{z \in \Sigma^* \mid \exists w_1, w_2 \in L \colon (w_1, w_2) \vdash_r z\}$$

and for a set of splicing rules R, we let

$$\sigma_R(L) = \bigcup_{r \in R} \sigma_r(L).$$

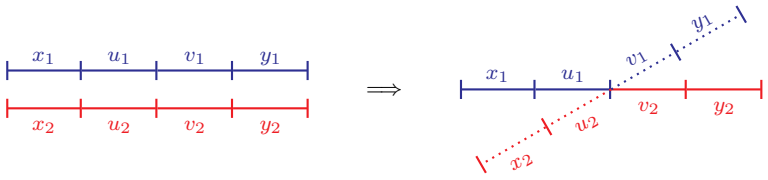

Figure 11: Splicing of the words $x_1u_1v_1y_1$ and $x_2u_2v_2y_2$ by the rule $r = (u_1, v_1; u_2, v_2)$.

The reflexive and transitive closure of the splicing operator σ_R^* is given by

$$\sigma_R^0(L) = L, \qquad \sigma_R^{i+1}(L) = \sigma_R^i(L) \cup \sigma_R(\sigma_R^i(L)), \qquad \sigma_R^*(L) = \bigcup_{i \geq 0} \sigma_R^i(L).$$

A finite set of axioms $I \subseteq \Sigma^*$ and a finite set of splicing rules $R \subseteq (\Sigma^*)^4$ form a *splicing system* (I, R). Every splicing system (I, R) generates a language $L(I, R) = \sigma_R^*(I)$. Note that $L(I, R)$ is the smallest language which is closed under the splicing operator σ_R and includes I. It is known that the language generated by a splicing system is regular; see [6, 17]. A (regular) language L is called a *splicing language* if a splicing system (I, R) exists such that $L = L(I, R)$.

A rule r is said to *respect* a language L if $\sigma_r(L) \subseteq L$. It is easy to see that for any splicing system (I, R), every rule $r \in R$ respects the generated language $L(I, R)$. Moreover, a rule $r \notin R$ respects $L(I, R)$ if and only if $L(I, R \cup \{r\}) = L(I, R)$. We say a splicing $(w_1, w_2) \vdash_r z$ *respects* a language L if $w_1, w_2 \in L$ and r respects L; obviously, this implies $z \in L$, too.

The main result of this section states that, if a regular language L is a splicing language, then it is generated by a particular splicing system (I, R) which only depends on the syntactic monoid of L.

Theorem 4.1. *Let L be a splicing language and $m = |M_L|$. The splicing system (I, R) with $I = \Sigma^{<m^2+6m} \cap L$ and*

$$R = \left\{ r \in \Sigma^{<m^2+10m} \times \Sigma^{<2m} \times \Sigma^{<2m} \times \Sigma^{<m^2+10m} \;\middle|\; r \text{ respects } L \right\}$$

generates the language $L = L(I, R)$.

As the language generated by the splicing system (I, R) is constructible [17], Theorem 4.1 implies that the problem whether or not a given regular language is a splicing language is decidable. A detailed discussion of the decidability result is given in Section 5.

Let L be a formal language. Clearly, every set of words $J \subseteq L$ and set of rules S where every rule in S respects L generates a subset $L(J, S) \subseteq L$. Therefore, in Theorem 4.1 the inclusion $L(I, R) \subseteq L$ is obvious. The rest of this section is devoted to the proof of the converse inclusion $L \subseteq L(I, R)$.

Example 4. Recall from Example 2 that the syntactic monoid of the regular language $L = a^+b^+$ contains $m = 5$ elements. Let I and R be as defined in Theorem 4.1 for the language L. We know from Example 1 that the splicing system (I', R') generates $L = L(I', R')$ where $I' = \{ab\}$ and $R' = \{(a, b; \varepsilon, ab), (ab, \varepsilon; a, b)\}$. The splicing system (I, R), defined in Theorem 4.1, is much larger than (I', R') as it is composed of all initial words from $I = \left\{a^ib^j \;\middle|\; i, j \geq 1 \wedge i + j < 55\right\}$ and all rules from $\Sigma^{<75} \times \Sigma^{<10} \times \Sigma^{<10} \times \Sigma^{<75}$ that respect L. Therefore, $I' \subseteq I$ and $R' \subseteq R$. We conclude that $L = L(I', R') \subseteq L(I, R) \subseteq L$.

The proof uses many ideas that have been employed in the Section 3. However, there are some challenges we encounter solely while considering the classic splicing variant. The additional complexity comes from having to handle the first and fourth components of rules, which occur in the restriction sites as well as in the paste site of the rule. In contrast, in Pixton splicing, the restriction sites of a rule do not necessarily occur in the bridge (which corresponds to the paste site

in classical splicing). The structure of this section is the same as Section 3. In Section 4.1 we will present techniques to obtain rules that respect a regular language L from other rules that respect L, and we show how we can modify a splicing step, such that the words used for splicing are not significantly longer than the splicing result; similar results can be found in [8,9]. In Section 4.2 we use these techniques to show that a long word $z \in L$ can be obtained by a series of splicings from a set shorter words from L and by using rules which satisfy certain length restrictions. Finally, in Section 4.3 we prove Theorem 4.1.

4.1 Rule Modifications

The first lemma states that we can extend the restriction sites of a rule r such that the extended rule respects all languages that are respected by r.

Lemma 4.2. *Let $r = (u_1, v_1; u_2, v_2)$ be a rule which respects a language L. For every word x, the rules $(xu_1, v_1; u_2, v_2)$, $(u_1, v_1x; u_2, v_2)$, $(u_1, v_1; xu_2, v_2)$, and $(u_1, v_1; u_2, v_2x)$ respect L as well.*

Proof. Let s be any of the rules $(xu_1, v_1; u_2, v_2)$, $(u_1, v_1x; u_2, v_2)$, $(u_1, v_1; xu_2, v_2)$, $(u_1, v_1; u_2, v_2x)$. In order to prove that s respects L, we have to show that, for all $w_1, w_2 \in L$ and $z \in \Sigma^*$ such that $(w_1, w_2) \vdash_s z$, we have $z \in L$, too. Indeed, if $(w_1, w_2) \vdash_s z$, then $(w_1, w_2) \vdash_r z$ and, as r respects L, we conclude $z \in L$. □

Henceforth, for a rule $r = (u_1, v_1; u_2, v_2)$, we will refer to the rules $(xu_1, v_1; u_2, v_2)$ and $(u_1, v_1x; u_2, v_2)$ as extensions of the left restriction site of r, and we refer to $(u_1, v_1; xu_2, v_2)$ and $(u_1, v_1; u_2, v_2x)$ as extensions of the right restriction site of r.

Next, for a language L, let us investigate the syntactic class of a rule $r = (u_1, v_1; u_2, v_2)$. The *syntactic class* (with respect to L) of r is the set of rules $[r]_L = [u_1]_L \times [v_1]_L \times [u_2]_L \times [v_2]_L$ and two rules r and s are *syntactically congruent* (with respect to L), denoted by $r \sim_L s$, if $s \in [r]_L$. The next lemma is a direct consequence of the fact that $\sim_L$ is a syntactic congruence.

Lemma 4.3. *Let r be a rule which respects a language L. Every rule $s \in [r]_L$ respects L.*

Proof. Let $r = (u_1, v_1; u_2, v_2)$ and $s = (\tilde{u}_1, \tilde{v}_1; \tilde{u}_2, \tilde{v}_2)$. Thus, $u_1 \sim_L \tilde{u}_1$, $u_2 \sim_L \tilde{u}_2$, $v_1 \sim \tilde{v}_1$, and $v_2 \sim \tilde{v}_2$. We will show that for all $\tilde{w}_1 = x_1\tilde{u}_1\tilde{v}_1y_1 \in L$ and $\tilde{w}_2 = x_2\tilde{u}_2\tilde{v}_2y_2 \in L$, we have $\tilde{z} = x_1\tilde{u}_1\tilde{v}_2y_2 \in L$. Let $w_1 = x_1u_1v_1y_1$, $w_2 = x_2u_2v_2y_2$ and note that $w_1 \sim_L \tilde{w}_1$, $w_2 \sim_L \tilde{w}_2$; hence, $w_1, w_2 \in L$. Furthermore, $(w_1, w_2) \vdash_r x_1u_1v_2y_2 = z \in L$ as r respects L, and $\tilde{z} \in L$ as $z \sim_L \tilde{z}$. □

Consider a splicing $(x_1u_1v_1y_1, x_2u_2v_2y_2) \vdash_r x_1u_1v_2y_2$ which respects a regular language L, as shown in Figure 12 on the left restriction site. The factors v_1y_1 and x_2u_2 may be relatively long but they do not occur as factors in the resulting word $x_1u_1v_2y_2$. In particular, it is possible that two long words are spliced and the outcome is a relatively short word. Using the Lemmas 4.2 and 4.3, we can find shorter words in L and a modified splicing rule which can be used to obtain $x_1u_1v_2y_2$.

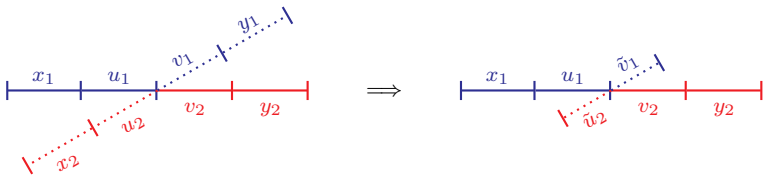

Figure 12: Replacing v_1y_1 and x_2u_2 by *short* words.

Lemma 4.4. *Let $r = (u_1, v_1; u_2, v_2)$ be a rule which respects a regular language L and $w_1 = x_1u_1v_1y_1 \in L$, $w_2 = x_2u_2v_2y_2 \in L$. There is a rule $s = (u_1, \tilde{v}_1; \tilde{u}_2, v_2)$ which respects L and words $\tilde{w}_1 = x_1u_1\tilde{v}_1 \in L$, $\tilde{w}_2 = \tilde{u}_2v_2y_2 \in L$ such that $|\tilde{v}_1|, |\tilde{u}_2| < |M_L|$. More precisely, $\tilde{v}_1 \in [v_1y_1]_L$ and $\tilde{u}_2 \in [x_2u_2]_L$. In particular, whenever $(w_1, w_2) \vdash_r x_1u_1v_2y_2 = z$, then $(\tilde{w}_1, \tilde{w}_2) \vdash_s z$ respects L.*

Proof. By using Lemma 4.2 twice, the rule $(u_1, v_1y_1; x_2u_2, v_2)$ respects L. By Lemma 2.2 we chose $\tilde{v}_1 \in [v_1y_1]_L$ and $\tilde{u}_2 \in [x_2u_2]_L$ as shortest words from the syntactic classes, respectively, such that $|\tilde{v}_1|, |\tilde{u}_2| \leq |M_L|$. Let $\tilde{w}_1 = x_1u_1\tilde{v}_1 \in L$, $\tilde{w}_2 = \tilde{u}_2v_2y_2 \in L$. Furthermore, by Lemma 4.3, $s = (u_1, \tilde{v}_1; \tilde{u}_2, v_2)$ respects L. □

4.2 Series of Splicings

Consider the creation of words by a series of splicings. Let us begin with a simple observation. In the case when a word is created by two (or more) successive splicings, but none of the restriction sites overlaps the position of the other splicing, the order of these splicings is irrelevant. Recall that the splicing position of a splicing $(w_1, w_2) \vdash_r z$ with $r = (u_1, v_1; u_2, v_2)$ is the position between the factors u_1 and v_2 in z. The notation in Remark 4.5 is the same as in the Figure 13.

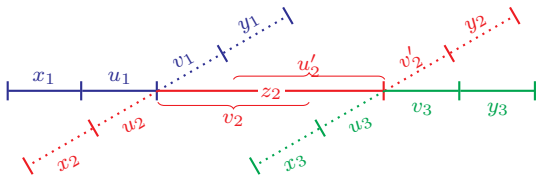

Figure 13: The word $x_1u_1z_2v_3y_3$ can be created either by using the right splicing first or by using the left splicing first.

Remark 4.5. *Let $w_1 = x_1u_1v_1y_1$, $w_2 = x_2u_2z_2v_2'y_2$, where v_2 is a prefix of z_2 and u_2' is a suffix of z_2, $w_3 = x_3u_3v_3y_3$ be words and $r_1 = (u_1, v_1; u_2, v_2)$, $r_2 = (u_2', v_2'; u_3, v_3)$ be rules. In order to create the word $z = x_1u_1z_2v_3y_3$ by splicing, we may use splicings*

$$(w_1, w_2) \vdash_{r_1} x_1u_1z_2v_2'y_2 = z', \qquad (z', w_3) \vdash_{r_2} z$$

as well as

$$(w_2, w_3) \vdash_{r_2} x_2u_2z_2v_3y_3 = z'', \qquad (w_1, z'') \vdash_{r_1} z.$$

Now, consider a word z which is created by two successive splicings from words $w_1 = x_1u_1v_1y_1$, $w_2 = x_2u_2v_2y_2$, and $w_3 = x_3u_3v_3y_3$ as shown in Figure 14. If no factor of w_1 is a part of z, then we can find another splicing rule s such that $(w_3, w_2) \vdash_s z$. This replacement will become crucial in the proof of Lemma 4.7.

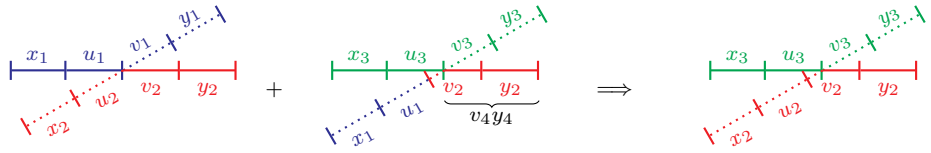

Figure 14: If no part of $x_1u_1v_1y_1$ is a factor of the splicing result, then the two splicings can be reduced to one splicing.

Lemma 4.6. *Let L be a language, $w_i = x_iu_iv_iy_i \in L$ for $i = 1, 2, 3$, and $r_1 = (u_1, v_1; u_2, v_2)$, $r_2 = (u_3, v_3; u_4, v_4)$ be rules respecting L. If there are splicings*

$$(w_1, w_2) \vdash_{r_1} x_1u_1v_2y_2 = w_4 = x_4u_4v_4y_4, \qquad (w_3, w_4) \vdash_{r_2} x_3u_3v_4y_4 = z$$

where v_4y_4 is a suffix of v_2y_2, then there is a rule $s = (u_3, v_3; u_2\delta, \tilde{v}_4)$ which respects L such that $(w_3, w_2) \vdash_s z$. Furthermore, $\tilde{v}_4 = v_4$ or $\tilde{v}_4 \leq_{\ell\ell} v_2$.

Proof. First, we will prove that we may assume that the factors u_1v_2 and u_4v_4 match in w_4 by extending the restriction sites u_1, v_2, u_4, v_4 (Lemma 4.2): let i, j, i', j' such that $w_{4[i;j]} = u_1v_2$ and $w_{4[i';j']} = u_4v_4$,

- if $i > i'$, extend u_1 in r_1 to the left by $i - i'$ letters,
- if $i < i'$, extend u_4 in r_2 to the left by $i' - i$ letters,
- if $j < j'$, extend v_2 in r_1 to the right by $j' - j$ letters, and
- if $j > j'$, extend v_4 in r_2 to the right by $j - j'$ letters.

In addition to the factors u_4, u_1, v_2, v_4, we alter the factors x_1, x_4, y_2, y_4, accordingly; we call the newly obtained factors $\tilde{u}_4, \tilde{u}_1, \tilde{v}_2, \tilde{v}_4, \tilde{x}_1, \tilde{x}_4, \tilde{y}_2, \tilde{y}_4$, respectively. As a result of the extension we now have $\tilde{u}_1\tilde{v}_2 = \tilde{u}_4\tilde{v}_4$, $\tilde{x}_1 = \tilde{x}_4$, and $\tilde{y}_2 = \tilde{y}_4$. The words

$$w_1 = \tilde{x}_4\tilde{u}_1v_1y_1, \qquad w_2 = x_2u_2\tilde{v}_2\tilde{y}_4, \qquad w_3 = x_3u_3v_3y_3, \qquad w_4 = \tilde{x}_4\tilde{u}_4\tilde{v}_4\tilde{y}_4$$

are the same as in the lemma, we just changed their factorization. The rules $\tilde{r}_1 = (\tilde{u}_1, v_1; u_2, \tilde{v}_2)$ and $\tilde{r}_2 = (u_3, v_3; \tilde{u}_4, \tilde{v}_4)$ are the extended versions of the rules r_1 and r_2, respectively, which respect L. Furthermore,

$$(w_1, w_2) \vdash_{\tilde{r}_1} \tilde{x}_4\tilde{u}_1\tilde{v}_2\tilde{y}_4 = w_4 = \tilde{x}_4\tilde{u}_4\tilde{v}_4\tilde{y}_4, \qquad (w_3, w_4) \vdash_{\tilde{r}_2} x_3u_3\tilde{v}_4\tilde{y}_4 = z.$$

As v_4y_4 is a suffix of v_2y_2 (before extension), $\tilde{v}_4$ is a suffix of $\tilde{v}_2$. Moreover, either $v_4 = \tilde{v}_4$ (v_4 was not extended) or $\tilde{v}_4 \leq_{\ell\ell} v_2 = \tilde{v}_2$ (v_2 was not extended). Let δ such that $\delta\tilde{v}_4 = \tilde{v}_2$, let $s = (u_3, v_3; u_2\delta, \tilde{v}_4)$, and observe that $(w_3, w_2) \vdash_s z$.

Next, let us prove that s respects L. If for all words $w_2' = x_2'u_2\delta\tilde{v}_4y_2' = x_2'u_2\tilde{v}_2y_2' \in L$ and $w_3' = x_3'u_3v_3y_3' \in L$ we have $x_3'u_3\tilde{v}_4y_2' \in L$, then s respects L. Indeed, we may splice

$$(w_1, w_2') \vdash_{\tilde{r}_1} \tilde{x}_4\tilde{u}_1\tilde{v}_2y_2' = w_4' = \tilde{x}_4\tilde{u}_4\tilde{v}_4y_2', \qquad (w_3, w_4) \vdash_{\tilde{r}_2} x_3'u_3\tilde{v}_4y_2' = z.$$

Because $\tilde{r}_1$ and $\tilde{r}_2$ respect L, the word $x_3'u_3\tilde{v}_4y_2'$ belongs to L. We conclude that s respects L. □

Consider a splicing system (J, S) and its generated language $L = L(J, S)$. Let n be the length of the longest word in J and let μ be the length-lexicographically largest word that is a component of a rule in S. Define $W_\mu = \{w \in \Sigma^* \mid w \leq_{\ell\ell} \mu\}$ as the set of words which are at most as large as μ, in length-lexicographic order. Furthermore, let $I = \Sigma^{\leq n} \cap L$ be a set of axioms and let R be the set of rules

$$R = \left\{ r \in W_\mu^4 \;\middle|\; r \text{ respects } L \right\}.$$

It is not difficult to see that $J \subseteq I$, $S \subseteq R$, and $L = L(I, R)$. Whenever convenient, we will assume that a splicing language L is generated by a splicing system which is of the form of (I, R).

Consider the creation of some word $x_0zy_0 \in L$ in the splicing system (I, R) where the length of the middle factor z is at least $|\mu|$. We can trace back the generation of x_0zy_0 to a word x_kzy_k where the factor z is *affected* by splicing for the last time, i. e., the splicing position lies in the factor z in x_kzy_k; or the word x_kzy_k belongs to I. This yields a degenerated splicing tree as discussed in Figure 4 in Section 2.2. Since the splicings which alter the prefixes x_i in this tree do not interfere with the splicings which alter the suffixes y_i (because $|z|$ is considered to be sufficiently long), we can reorganize the tree as shown on the right in Figure 4, in accordance with Remark 4.5. The next lemma describes this creation of x_0zy_0 by k splicings in (I, R), and shows that we can choose the rules and words which are used to create x_0zy_0 from x_kzy_k such that the words and the components of the rules satisfy certain length restrictions.

Lemma 4.7. *Let L be a splicing language, let $m = |M_L|$, let $\ell, n \in \mathbb{N}$, such that for $I = \Sigma^{\leq n} \cap L$ and a rule set R we have $L = L(I, R)$. Let μ denote the length-lexicographically largest component of all rules in R (therefore, $R \subseteq W_\mu^4$). Let $z_0 = x_0zy_0$ be a word with $|x_0|, |y_0| \leq \ell$, and $|z| \geq |\mu|$.*

Suppose $z_0 = \tilde{z}_0 = \tilde{x}_0 z \tilde{y}_0$ (with $x_0 = \tilde{x}_0$, $y_0 = \tilde{y}_0$) is generated by a series of $\tilde{k}$ splicings from a word $\tilde{z}_{\tilde{k}} = \tilde{x}_{\tilde{k}} z \tilde{y}_{\tilde{k}}$ where either $\tilde{z}_{\tilde{k}} \in I$ or $\tilde{z}_{\tilde{k}}$ is created by a splicing $(\tilde{w}_{\tilde{k}+1}, \tilde{w}_{\tilde{k}+2}) \vdash_{\tilde{s}} \tilde{z}_{\tilde{k}}$ with $\tilde{w}_{\tilde{k}+1}, \tilde{w}_{\tilde{k}+2} \in L$, $\tilde{s} \in R$, and the splicing position lies in the factor z in $\tilde{z}_{\tilde{k}}$. Furthermore, for $i = 1, \ldots, \tilde{k}$ the intermediate splicings are either

(i) $(\tilde{w}_i, \tilde{z}_i) \vdash_{\tilde{r}_i} \tilde{x}_{i-1} z \tilde{y}_{i-1} = \tilde{z}_{i-1}$, where $\tilde{w}_i \in L$, $\tilde{r}_i \in R$, $\tilde{y}_{i-1} = \tilde{y}_i$, and the splicing position lies at the left of the factor z or

(ii) $(\tilde{z}_i, \tilde{w}_i) \vdash_{\tilde{r}_i} \tilde{x}_{i-1} z \tilde{y}_{i-1} = \tilde{z}_{i-1}$, where $\tilde{w}_i \in L$, $\tilde{r}_i \in R$, $\tilde{x}_{i-1} = \tilde{x}_i$, and the splicing position lies at the right of the factor z.

Then, there are also rules and words that generate $z_0 = \tilde{z}_0$ by a series of $k \leq \tilde{k}$ splicings from a word $z_k = x_k z y_k$ where either $z_k \in I$ or z_k is created by a splicing $(w_{k+1}, w_{k+2}) \vdash_s z_k$ and the splicing position lies in the factor z in $z_{\tilde{k}}$. Furthermore, there exists $k' \leq k$ such that

(i') for $i = 1, \ldots, k'$ the intermediate splicings are $(w_i, z_i) \vdash_{r_i} x_{i-1} z y_{i-1} = z_{i-1}$, where $w_i \in L$, r_i respects L, $y_{i-1} = y_i$, $r_i \in \Sigma^{<\ell+m} \times \Sigma^{<2m} \times \Sigma^{<2m} \times W_\mu$, and the splicing position lies at the left of the factor z;

(ii') for $i = k'+1, \ldots, k$ the intermediate splicings are $(z_i, w_i) \vdash_{r_i} x_{i-1} z y_{i-1} = z_{i-1}$, where $w_i \in L$, r_i respects L, $x_{i-1} = x_i$, $r_i \in W_\mu \times \Sigma^{<2m} \times \Sigma^{<2m} \times \Sigma^{<\ell+m}$, and the splicing position lies at the right of the factor z; and

(iii') for $i = 1, \ldots, k$ the following length bounds apply: $|x_i|, |y_i| < \ell + 2m$, $|w_i| < 2m + \ell$, $r_i \in \Sigma^{<2m} \times \Sigma^{<2m} \times \Sigma^{<\ell+m}$. In particular, if $n \geq 2m + \ell$, then $w_1, \ldots, w_k \in I$. Moreover, $s \in W_\mu^4$.

Proof. Note that if $\tilde{k} = 0$, then the statements are trivially true. The fact that we can reorganize the splicing tree such that it looks like in Figure 4 on the right follows by Remark 4.5 and because $|z| \geq |\mu|$; that is, for some $\tilde{k}' < \tilde{k}$ all the splicing steps $i = 1, \ldots, \tilde{k}'$ are of the form (i) and all the splicing steps $i = \tilde{k}' + 1, \ldots, \tilde{k}$ are of the form (ii).

If for $2 \leq i \leq \tilde{k}'$ we have that no factor of $\tilde{w}_i$ occurs in $\tilde{x}_{i-2}$ (the splicing position in $(\tilde{w}_i, \tilde{z}_i) \vdash \tilde{z}_{i-1}$ lies to the left of the splicing in $(\tilde{w}_{i-1}, \tilde{z}_{i-1}) \vdash \tilde{z}_{i-2}$), then we can combine the i-th and $(i-1)$-th splicing step into one splicing step according to Lemma 4.6. Successively eliminating all these cases and the symmetric cases for $\tilde{k}' + 2 \leq i \leq \tilde{k}$, yields a possibly shorter series of k splicing steps and an integer $k' \leq k$ such that: for $i = 1, \ldots, k'$ the i-th splicing step is of the form (i) and a factor of $\tilde{w}_i$ appears in $x_0 = \tilde{x}_0$; and for $i = k'+1, \ldots, k$ the i-th splicing step is of the form (ii) and a factor of $\tilde{w}_i$ appears in $y_0 = \tilde{y}_0$. Note that this reorganization also implies $\tilde{x}_{k'} = \tilde{x}_{k'+1} = \cdots = \tilde{x}_k$ and $y_0 = \tilde{y}_0 = \tilde{y}_1 = \cdots = \tilde{y}_{k'}$.

We now have that $z_0 = \tilde{z}_0$ is generated by a series of k splicings from a word $\tilde{z}_k = \tilde{x}_k z \tilde{y}_k$ where either $\tilde{z}_k \in I$ or $\tilde{z}_k$ is created by a splicing $(\tilde{w}_{k+1}, \tilde{w}_{k+2}) \vdash_{\tilde{s}} \tilde{z}_k$ with $\tilde{w}_{k+1}, \tilde{w}_{k+2} \in L$, $\tilde{s} \in R$, and the bridge of $\tilde{s}$ overlaps with z in $\tilde{z}_k$. Furthermore, there exists $k' \leq k$ such that

(i) for $i = 1, \ldots, k'$ the intermediate splicings are $(\tilde{w}_i, \tilde{z}_i) \vdash_{\tilde{r}_i} \tilde{x}_{i-1} z \tilde{y}_{i-1} = \tilde{z}_{i-1}$, where $\tilde{w}_i \in L$, $\tilde{r}_i$ respects L, $\tilde{y}_{i-1} = \tilde{y}_i$, the bridge of $\tilde{r}_i$ is covered by the prefix $\tilde{x}_{i-1}$, and a non-empty factor of $\tilde{w}_i$ occurs in x_0; and

(ii) for $i = k'+1, \ldots, k$ the intermediate splicings are $(\tilde{z}_i, \tilde{w}_i) \vdash_{\tilde{r}_i} \tilde{x}_{i-1} z \tilde{y}_{i-1} = \tilde{z}_{i-1}$, where $\tilde{w}_i \in L$, $\tilde{r}_i$ respects L, $\tilde{x}_{i-1} = \tilde{x}_i$, the bridge of $\tilde{r}_i$ is covered by the suffix $\tilde{y}_{i-1}$, and a non-empty factor of $\tilde{w}_i$ occurs in y_0.

We will continue to modify and replace all words and rules which are marked by $\tilde{}$ in order to prove the length bounds which are stated in (iii').

Let us consider the splicings of the form (i) which are the steps $i = 1, \ldots, k'$. In order to prove statement (iii') we apply several modifications in the series of splicings which generates $x_0 z y_0$ by using the techniques that we established beforehand. These modifications are depicted in Figure 15; the notational convention during this proof is that all words (or factors) which are

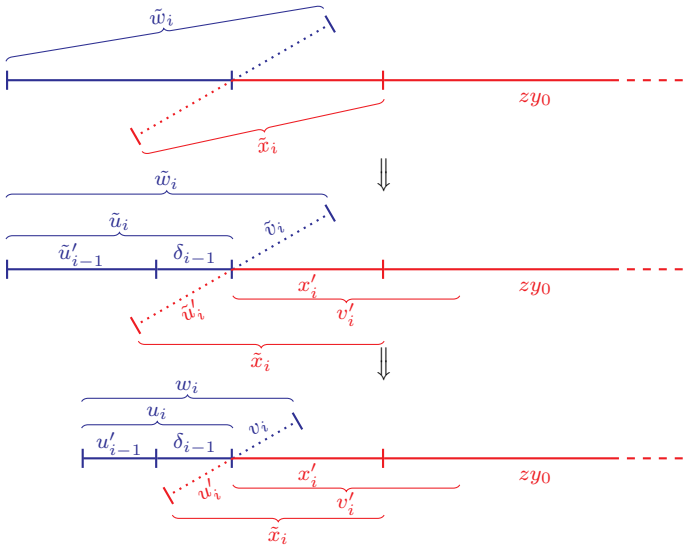

Figure 15: The i-th splicing step with $i \leq k'$ in the proof of Lemma 4.7: the upper figure shows a general case where the words $\tilde{w}_i$ and $\tilde{x}_i z y_0$ are spliced in order to obtain the new word $\tilde{x}_{i-1} z y_0$; the middle figure shows the splicing after applying Lemma 4.2; and the lower figure shows the splicing after we replace $\tilde{u}'_i$, $\tilde{u}'_{i+1}$, and $\tilde{v}_1$ by syntactically equivalent, short words u'_i, u'_{i+1}, and v_1, respectively.

marked by $\tilde{}$ will be replaced during the proof. Let us start with the general case where $\tilde{w}_i$ and $\tilde{x}_i z \tilde{y}_1$ are spliced in order to obtain the new word $\tilde{x}_{i-1} z \tilde{y}_0$ (Figure 15 top). By applying extension (Lemma 4.2) to the first, second, and third component of the rule in this splicing step, we may assume that a rule $(\tilde{u}_i, \tilde{v}_i; \tilde{u}'_i, v'_i)$ is used where $\tilde{w}_i = \tilde{u}_i \tilde{v}_i$ and $\tilde{u}'_i$ covers a prefix of $\tilde{x}_i$. We apply the same modifications for all splicing steps $i = 1, \ldots, k'$ and we let x'_i such that $\tilde{x}_i = \tilde{u}_i x'_i$. The forth component v'_i of the rule is a prefix of $x'_i z$ and remains unaltered throughout this proof, therefore $v'_i \in W_\mu$ (note that this is in accordance with Lemma 4.6 which may have been used to alter this rule). Furthermore, we let $\tilde{u}_0 = \varepsilon$ and $x'_0 = \tilde{x}_0 = x_0$. As stated above, the position in the $(i-1)$-th splicing step $(\tilde{w}_{i-1}, \tilde{x}_{i-1} z y_0) \vdash \tilde{x}_{i-2} z \tilde{y}_1$ lies to the left of the position in the i-th splicing step $(\tilde{w}_i, \tilde{x}_i z y_0) \vdash \tilde{x}_{i-1} z y_0$. Therefore, x'_i is a proper suffix of x'_{i-1} and we may write $x'_{i-1} = \delta_{i-1} x'_i$ for some word δ_{i-1} (Figure 15 middle). Note that we also have $\tilde{u}_i = \tilde{u}'_{i-1} \delta_{i-1}$.

Next, we replace most factors which do not actually appear in $x_0 z y_0$ by shortest words from their respective syntactic classes. We replace $\tilde{u}'_i$ by $u'_i \in [\tilde{u}'_i]_L$ with $|u'_i| < m$ for $i = 1, \ldots, k'-1$; and $\tilde{v}_i$ by $v_i \in [\tilde{v}_i]_L$ with $|v_i| < m$ for $i = 1, \ldots, k'$ (Lemma 2.2); this does not change the fact that all rules respect L (Lemma 4.3). Note that we did not replace $\tilde{u}'_k$ yet as this factor is a prefix of the word $\tilde{x}_k z \tilde{y}_k = \tilde{z}_k$ which is part of the splicing $(w_0, w'_0) \vdash_s \tilde{z}_k$ if it exists. We also have to replace $\tilde{u}_i$ by $u_i = u'_{i-1} \delta_{i-1}$ for $i = 2, \ldots, k'-1$ and $\tilde{u}_1$ by $u_1 = \delta_0$; and replace $\tilde{x}_i$ by $x_i = u'_i x'_i$ for $i = 1, \ldots, k'-1$ (Figure 15 bottom). As δ_{i-1} is an infix of x_0, we have $|\delta_{i-1}| \leq \ell$, $|u_i| < m + \ell$, and $|w_i| < 2m + \ell$ for $i = 1, \ldots, k'$. As x'_i is an infix of x_0 as well, we have $|x'_i| \leq \ell$ and $|x_i| < m + \ell$ for $i = 1, \ldots, k'-1$. We let $r_i = (u_i, v_i; u'_i, v'_i)$ be the rule which is used now in the i-th splicing step and we obtain that $r_i \in \Sigma^{<\ell+m} \times \Sigma^{<m} \times \Sigma^{<m} \times W_\mu$ for $i = 1, \ldots, k'-1$. This means, for $i = 1, \ldots, k'-1$ we have proven even stronger length bounds than stated in statements (i') and (iii') of Lemma 4.7.

The rule which is used in the k'-th splicing step is $(u_{k'}, v_{k'}; \tilde{u}'_{k'}, v'_{k'}) \in \Sigma^{<\ell+m} \times \Sigma^{<m} \times \Sigma^* \times W_\mu$. We will replace the third component $\tilde{u}'_1$ by a word u'_1 which is shorter than $2m$, and we will replace $\tilde{x}_{k'} = \tilde{x}_k$ by $x_k = u_{k'} x'_{k'}$ accordingly in order to prove the remaining length bounds of statements (i') and (iii'). If $\tilde{x}_k z \tilde{y}_k \in I$, we let $u'_{k'} \in [\tilde{u}'_{k'}]_L$ with $|u'_{k'}| < m$ (Lemma 2.2) and we are done. Otherwise, we have $(\tilde{w}_{k+1}, \tilde{w}_{k+2}) \vdash_{\tilde{s}} \tilde{x}_k z \tilde{y}_k$ with $\tilde{s} = (\tilde{u}_{k+1}, v_{k+1}; u_{k+2}, \tilde{v}_{k+2})$ such that the splicing position lies in the factor z in $\tilde{z}_k$. This implies that $\tilde{u}'_{k'}$ is a prefix of $\tilde{w}_0$ and $\tilde{u}'_{k'}$ may overlap

with $\tilde{u}_{k+1}$ but not with v_{k+1}. In case when $\tilde{u}'_{k'}$ does not overlap with $\tilde{u}_{k+1}$, let $u'_{k'} \in [\tilde{u}'_{k'}]_L$ with $|u'_{k'}| < m$ (Lemma 2.2) and let $u_{k+1} = \tilde{u}_{k+1}$. In case when $\tilde{u}'_{k'}$ overlaps with $\tilde{u}_{k+1}$, let $\tilde{u}'_{k'} = \tilde{\alpha}\tilde{\beta}$ and $\tilde{u}_{k+1} = \tilde{\beta}\gamma$ such that $\tilde{\beta}$ is the overlap. Let $\alpha \in [\tilde{\alpha}]_L$ and $\beta \in [\tilde{\beta}]_L$ with $|\alpha|, |\beta| < m$ (Lemma 2.2); furthermore, let $u'_{k'} = \alpha\beta$, $u_{k+1} = \beta\gamma$. For both cases we obtain that $u_{k+1} \leq_{\ell\ell} \tilde{u}_{k+1}$ and hence $u_{k+1} \in W_\mu$. We let w_{k+1} be the word that is obtained by replacing the prefix $\tilde{u}'_{k'}$ in $\tilde{w}_{k+1}$ by $u'_{k'}$. Clearly, we have that $(w_{k+1}, \tilde{w}_{k+2}) \vdash_s x_1 z \tilde{y}_1$ with $s = (u_{k+1}, v_{k+1}; u_{k+2}, \tilde{v}_{k+2}) \in R$ as desired. Furthermore, $|u'_{k'}| < 2m$ and $|x_{k'}| < \ell + 2m$. This concludes the proof of statements (i') and (iii') of Lemma 4.7 for $i = 1, \ldots, k'$. Statements (ii') and (iii') of Lemma 4.7 for $i = k'+1, \ldots, k$ can be proven analogously. □

4.3 Proof of Theorem 4.1

Let L be a splicing language and $m = |M_L|$. Throughout this section, by $\sim$ we denote the equivalence relation $\sim_L$ and by $[\cdot]$ we denote the corresponding equivalence classes $[\cdot]_L$.

Recall that Theorem 4.1 claims that the splicing system (I, R) with $I = \Sigma^{<m^2+6m} \cap L$ and

$$R = \left\{ r \in \Sigma^{<m^2+10m} \times \Sigma^{<2m} \times \Sigma^{<2m} \times \Sigma^{<m^2+10m} \;\middle|\; r \text{ respects } L \right\}$$

generates L. The proof is divided in two parts. In the first part, Lemma 4.8, we prove that the set of rules can be chosen as $\left\{ r \in (\Sigma^{<m^2+10m})^4 \;\middle|\; r \text{ respects } L \right\}$ for some finite set of axioms. The second part concludes the proof of Theorem 4.1, by employing the length bound $2m$ for the second and third component of rules and by proving that the set of axioms can be chosen as $I = \Sigma^{<m^2+6m} \cap L$.

Lemma 4.8. *Let L and m as above. There exists $n \in \mathbb{N}$ such that the splicing system (I, R) with $I = \Sigma^{\leq n} \cap L$ and*

$$R = \left\{ r \in (\Sigma^{<m^2+10m})^4 \;\middle|\; r \textit{ respects } L \right\}$$

generates the same language $L = L(I, R)$.

Proof. As every word in I belongs to L and every rule in R respects L, the inclusion $L(I, R) \subseteq L$ holds (for any n).

Since L is a splicing language, there exists a splicing system (I', R') which generates L. Let n' be a number larger than any word in I' and larger than any component of a rule in R' and let $n = n' + 6m$. Let $I = \Sigma^{\leq n} \cap L$ as in the claim and observe that $L(I, R') = L$.

For a word μ we define $W_\mu = \{w \in \Sigma^* \mid w \leq_{\ell\ell} \mu\}$, as we did before. We let the set of rules where every component is length-lexicographically bounded by some word μ be

$$R_\mu = \left\{ r \in W_\mu^4 \;\middle|\; r \text{ respects } L \right\},$$

and we let $L_\mu = L(I, R_\mu)$. Observe that $L_\mu \subseteq L$ for all words μ. For two words v, μ with $\mu \leq_{\ell\ell} v$ we see that $R_\mu \subseteq R_v$, and hence, $L_\mu \subseteq L_v$. Thus, if $L_\mu = L$ for some word μ, then for all words v with $\mu \leq_{\ell\ell} v$, we have $L_v = L$. Because $L = L(I, R')$ is a splicing language, there exists a word μ such that $L_\mu = L$. Now, we fix μ to be the smallest word, in the length-lexicographic order, such that $L_\mu = L$. Note that if $|\mu| < m^2 + 10m$, then $R_\mu \subseteq R$ and $L = L_\mu \subseteq L(I, R)$, which in turn, would prove our claim. For the sake of obtaining a contradiction assume $|\mu| \geq m^2 + 10m$. Let ν be the next-smaller word than μ, in the length-lexicographic order, and let $S = R_\nu$. Note that $L(I, S) \subsetneq L$ and $R_\mu \setminus S$ is non-empty, containing only rules which have a component that is equal to μ.

Choose w from $L \setminus L(I, S)$ as a shortest word, i. e., for all $w' \in L$ with $|w'| < |w|$, we have $w' \in L(I, S)$. Factorize

$$w = x_0 z y_0$$

with $|x_0| = |y_0| = 3m$ and let z cover the remainder of w; note that $|z| \geq |\mu|$, otherwise $w \in I \subseteq L(I, S)$. Factorize

$$\mu = \delta_1 \alpha \beta \gamma \delta_2$$

such that $|\delta_1|, |\delta_2| \geq 5m$, $|\alpha\beta\gamma| = m^2$, $\beta \neq \varepsilon$, $\alpha \sim \alpha\beta$, and $\gamma \sim \beta\gamma$, see Pumping Lemma 2.1.

The proof idea is as follows: we replace each factors $\alpha\beta\gamma$ in the factor z of word $w = x_0zy_0$ by $\alpha\beta^j\gamma$, for a large j, in order to obtain a word $\tilde{w} = x_0\tilde{z}y_0 \sim w$, as described in Lemma 2.3. The word $\tilde{w}$ has to be created by a series of splicings in (I, R_μ) and at some point during the creation of $\tilde{w}$ by splicing the splicing has to affect the factor $\tilde{z}$; the series of splicings is of the form as described in Lemma 4.7 and Figure 4. If a splicing rule from $R_\mu \setminus S$ is used during the creation of of $\tilde{w}$, then the component μ of this rule has to overlap with a factor $\alpha\beta^j\gamma$ such that we can replace this splicing by using a different rule. Next, we pump down all the factors $\alpha\beta^j\gamma$ in $\tilde{w}$ again in order to obtain the original word w. We adapt the series of splicings, that created $\tilde{w}$, in order to obtain series of splicings which creates w; then, we ensure that all words used in this series of splicings are shorter than w and all rules belong to S. This will contradict, that w is a shortest word in $L(I, R_\mu) \setminus L(I, S)$.

Let j be a sufficiently large even number ($j = 4|\mu| + 2|z|$ will suffice). According to Lemma 2.3 and as $\alpha\beta\gamma \sim_L \alpha\beta^j\gamma$, there exists a word $\tilde{z}$, which is obtained by successively applying factor replacements $\alpha\beta\gamma \mapsto \alpha\beta^j\gamma$ in z, such that $\tilde{z} \sim_L z$ and for all factors $\tilde{z}_{[k;k']} = \alpha\beta\gamma$ in $\tilde{z}$

(a) $\alpha\beta^{j/2}$ is a factor of $\tilde{z}$ starting at position $\tilde{z}_{[k]}$ or

(b) $\beta^{j/2}\gamma$ is a factor of $\tilde{z}$ ending at position $\tilde{z}_{[k']}$.

In particular, if $\delta_1\alpha\beta\gamma\delta_2$ is a factor of $\tilde{z}$ (a) $\gamma\delta_2$ is a prefix of a word in β^+ or (b) $\delta_1\alpha$ is a suffix of a word in β^+. Because $z \sim \tilde{z}$, we have $x\tilde{z}y \in L$.

Let us trace back the creation of $x_0\tilde{z}y_0 \in L$ by splicing in (I, R_μ) to a word $x_k\tilde{z}y_k$ where either $x_k\tilde{z}y_k \in I$ or where $x_k\tilde{z}y_k$ is created by a splicing that affects $\tilde{z}$, i. e., the splicing position lies in the factor $\tilde{z}$. Let $z_0 = x_0\tilde{z}y_0$ be created by k splicings from a word $z_k = x_k\tilde{z}y_k$ where either $z_k \in I$ or z_k is created by a splicing $(w_{k+1}, w_{k+2}) \vdash_s z_k$ with $w_{k+1}, w_{k+2} \in L$, $s \in R_\mu$, and the splicing position lies in the factor $\tilde{z}$. Furthermore, for $i = 1, \ldots, k$ the intermediate splicings are either

(i) $(w_i, z_i) \vdash_{r_i} x_{i-1}\tilde{z}y_{i-1} = z_{i-1}$, where $w_i \in L$, $r_i \in R_\mu$, $y_{i-1} = y_i$, and the splicing position lies at the left of the factor $\tilde{z}$ or

(ii) $(z_i, w_i) \vdash_{r_i} x_{i-1}\tilde{z}y_{i-1} = z_{i-1}$, where $w_i \in L$, $r_i \in R_\mu$, $x_{i-1} = x_i$, and the splicing position lies at the right of the factor $\tilde{z}$.

Note that $|\tilde{z}| \geq |z| \geq |\mu|$ and, therefore, we can apply Lemma 4.7 (with $\ell = 3m$). Thus, we may assume that $w_i \in I$ and $|x_i|, |y_i| < 5m$ for $i = 1, \ldots, k$.

Consider a rule r_i in a splicing of the form (i). By Lemma 4.7, $r_i \in \Sigma^{<4m} \times \Sigma^{<2m} \times \Sigma^{<2m} \times W_\mu$. Suppose the fourth component of r_i covers a prefix of the left-most factor $\alpha\beta^{j/2}$ in $\tilde{z}$ which is longer than α (as j is very large, it cannot fully cover $\alpha\beta^{j/2}$). We may write $r_i = (u_1, v_1; u_2, v'v'')$ where v'' is the prefix of $\alpha\beta^{j/2}$. By extension (Lemma 4.2) the rule $(u_1, v_1; u_2, v'\alpha\beta^{j/2})$ respects L, and so does the rule $\tilde{r}_i = (u_1, v_1; u_2, v'\alpha)$ because $\alpha \sim \alpha\beta^{j/2}$ (Lemma 4.3). The thusly obtained rule $\tilde{r}_i$ can be used in place of r_i; furthermore, as the fourth component got shorter, we have $\tilde{r}_i \in S$ and the rule does not overlap with the factor $\beta^{j/2}$ anymore. For convenience, we assume that every r_i is of the form of its corresponding $\tilde{r}_i$ from here on.

After we symmetrically treated rules of form (ii), these new rules $r_1, \ldots, r_k$ and the words $w_1, \ldots, w_k$ can be used in order to create $w = x_0zy_0$ from x_kzy_k by splicing. In order to see this, observe that, even though the factors $\alpha\beta\gamma$ in z, which we pumped up before, may overlap with each other, the left-most (and right-most) position where we replaced β by β^j is preceded by the factor α (resp. succeeded by the factor γ) in $\tilde{z}$.

Next, we show that all the rules $r_1, \ldots, r_k$ belong to S. By contradiction, suppose $r_i \notin S$ for some i and, by symmetry, suppose this i-th splicing is of the form (i). Thus, the fourth component of r_i has to be $\mu = \delta_1\alpha\beta\gamma\delta_2$. As $|\delta_1| \geq 5m > |x_i|$, the factor $\alpha\beta\gamma$ in μ is covered by $\tilde{z}$. Let ℓ such that $\alpha\beta\gamma = \tilde{z}_{[\ell;\ell+m^2]}$ is this factor in $\tilde{z}$. The properties of $\tilde{z}$ ensure that (a) $\alpha\beta^{j/2}$ is a factor of $\tilde{z}$ starting at position $\tilde{z}_{[\ell]}$ or (b) $\beta^{j/2}\gamma$ is a factor of $\tilde{z}$ ending at position $\tilde{z}_{[\ell+m^2]}$. As $j/2$ is very large and the splicing position of $(w_i, z_i) \vdash_{r_i} z_{i-1}$ is too close to the left end of z_{i-1}, case (b) is not possible. Thus, case (a) holds, the fourth component of r_i overlaps in more than $|\alpha|$ letters

with the left-most factor $\alpha\beta^{j/2}$ in $\tilde{z}$, and we used the replacement above which ensured $r_i \in S$ — contradiction.

Let us summarize: if $x_k z y_k$ was in $L(I,S)$, then $w = x_0 z y_0 \in L(I,S)$ (as well as all intermediate $x_i z y_i \in L(I,S)$ for $i = 1,\ldots,k-1$), which would contradict the choice of w. If $z_k = x_k \tilde{z} y_k \in I$, then $x_k z y_k$, which is at most as long as z_k, would belong to I and we are done. We only have to consider the case when $(w_{k+1}, w_{k+2}) \vdash_s z_k = x_k \tilde{z} y_k$ and the splicing position lies in the factor $\tilde{z}$. We will show that, from this splicing, we derive another splicing $(\tilde{w}_{k+1}, \tilde{w}_{k+2}) \vdash_t x_k z y_k$ which respects $L(I,S)$ and, therefore, yields the contradiction.

Due to Lemma 4.4 we may assume that $s = (u, v_1; u_2, v)$, $w_{k+1} = xuv_1$ and $w_{k+2} = u_2 vy$ where $|v_1|, |u_2| < m$. We have

$$z_k = x_k \tilde{z} y_k = xuvy$$

where xu is a proper prefix of $x_k\tilde{z}$ and vy is a proper suffix of $\tilde{z}y_k$ because we required that the splicing position lies in $\tilde{z}$ in z_k.

We will see next that if $s \notin S$, then we can use a rule $\tilde{s} \in S$ and maybe slightly modified words in order to obtain z_k by splicing. If $s \notin S$, then $u = \mu$ or $v = \mu$. Suppose $u = \mu = \delta_1\alpha\beta\gamma\delta_2$ by symmetry. Thus, the factor $\alpha\beta\gamma$ of μ is covered by the factor $\tilde{z}$ in z_k as $|\delta_1| \geq 5m > |x_k|$. Choose ℓ such that $\alpha\beta\gamma = \tilde{z}_{[\ell;\ell+m^2]}$ is this factor. Recall that (a) $\alpha\beta^{j/2}$ is a factor of $\tilde{z}$ starting at position $\tilde{z}_{[\ell]}$ or (b) $\beta^{j/2}\gamma$ is a factor of $\tilde{z}$ ending at position $\tilde{z}_{[\ell+m^2]}$. If (b) holds, $\delta_1\alpha$ is a suffix of a word in β^+ and we may write $\delta_1\alpha = \beta_2\beta^\ell$ where $\ell \geq 0$ and β_2 is a suffix of β. Replace u by $\beta_2\gamma\delta_1 \sim u$ and use this new rule $\tilde{s} = (\beta_2\gamma\delta_1, v_1; u_2, v)$ in order to splice $(w_{k+1}, w_{k+2}) \vdash_{\tilde{s}} z_k$. Note that the first component is now shorter than μ. Otherwise ((a) holds), $\gamma\delta_2 v$ is a prefix of a word in β^+. As j is very large and γ is a prefix of a word in β^+, we may extend v (Lemma 4.2) such that we can write $\beta\gamma\delta_2 = \beta^{\ell_1}\beta_1$ and $v = \beta_2\beta^{\ell_2}\gamma$ where $\ell_1 \geq 1$, $\ell_2 \geq 0$, , and $\beta_1\beta_2 = \beta$. Now, we pump down one of the β in the first component and β^{ℓ_2} in the fourth component and we let $\tilde{s} = (\delta_1\alpha\beta^{\ell_1-1}\beta_1, v_1; u_2, \beta_2\gamma) \sim s$. As all components are shorter than μ, we see that $\tilde{s} \in S$ and

$$(x\delta_1\alpha\beta^{\ell_1-1}\beta_1 v_1, u_2\beta_2\beta^{\ell_2+1}\gamma y) \vdash_{\tilde{s}} z_k,$$

that is, we have shifted one of the occurrences of β from w_{k+1} to w_{k+2}. Note that $\beta_2\gamma$ is a prefix of $\beta_2\beta^{\ell_2+1}\gamma$. Treating the fourth component analogously justifies the assumption that $s \in S$.

Recall that $\tilde{z} = xuvy$ was obtained from z by successively pumping up factors $\alpha\beta\gamma$ to $\alpha\beta^j\gamma$. Now, we will pump down the factors $\alpha\beta^j\gamma$ to $\alpha\beta\gamma$ in $\tilde{z}$ again. At every position where we pumped up before, we are now pumping down in reverse order, in order to obtain the factors $\tilde{x}, \tilde{u}, \tilde{v}, \tilde{y}$ from the words x, u, v, y, respectively.

For each pumping step do:

1. If u is covered by the factor $\alpha\beta^j\gamma$ (which we pump down in this step), extend u to the left such that it becomes a prefix of $\alpha\beta^j\gamma$. Symmetrically, if v is covered by the factor $\alpha\beta^j\gamma$, extend v to the right such that it becomes a suffix of $\alpha\beta^j\gamma$ (Lemma 4.2). After this extension the factor $\alpha\beta^j\gamma$ is covered by xu, uv, or vy.

2. If $\alpha\beta^j$ or $\beta^j\gamma$ is covered by one of x, u, v, or y, then replace this factor by $\alpha\beta$ or $\beta\gamma$, respectively. Skip next step.

3. If $\alpha\beta^j\gamma$ is covered by xu (the cases when $\alpha\beta^j\gamma$ is covered by uv or vy can be treated analogously), we can factorize $x = x'\alpha\beta^{j_1}\beta_1$ and $u = \beta_2\beta^{j_2}\gamma u'$ where $\beta_1\beta_2 = \beta$ and $j_1 + j_2 + 1 = j$. The results of pumping down are the words $x'\alpha\beta_1$ and $\beta_2\gamma u'$, respectively.

Observe that, after reversing all pumping steps, $\tilde{x}\tilde{u} \sim xu$, $\tilde{v}\tilde{y} \sim vy$, $\tilde{x}\tilde{u}\tilde{v}\tilde{y} = x_k z y_k$, and the rule $t = (\tilde{u}, v_1; u_2, \tilde{v})$ respects L. Furthermore, if we used extension for u (or v) in one of the steps, then $|\tilde{u}| \leq m^2$ (resp. $|\tilde{v}| \leq m^2$); no matter whether we used extension or not, we have $t \in S$. Note that

$$|\tilde{x}\tilde{u}v_1| < |x_k z| + |v_1| \leq |z| + 6m = |w|$$

Since w was chosen as the shortest word from $L \setminus L(I,S)$, we have $\tilde{w}_{k+1} = \tilde{x}\tilde{u}v_1 \in L(I,S)$ and, symmetrically, $\tilde{w}_{k+2} = u_2\tilde{v}\tilde{y} \in L(I,S)$. Because $(\tilde{w}_{k+1}, \tilde{w}_{k+2}) \vdash_t x_k z y_k$, we conclude that $x_k z y_k$ as well as w belong to $L(I,S)$ — the desired contradiction. □

Now, we can prove our main result of this section.

Proof of Theorem 4.1. Recall that for a splicing language L with $m = |M_L|$ we intend to prove that the splicing system (I, R) with $I = \Sigma^{<m^2+6m} \cap L$ and

$$R = \left\{ r \in \Sigma^{<m^2+10m} \times \Sigma^{<2m} \times \Sigma^{<2m} \times \Sigma^{<m^2+10m} \;\middle|\; r \text{ respects } L \right\}$$

generates the language $L = L(I, R)$.

Obviously, $L(I, R) \subseteq L$. By Lemma 4.8, we may assume that L is generated by a splicing system (J, S) where

$$S = \left\{ r \in (\Sigma^{<m^2+10m})^4 \;\middle|\; r \text{ respects } L \right\}.$$

In order to prove $L \subseteq L(I, R)$, we use induction on the length of words in L. For $w \in L$ with $|w| < m^2 + 6m$, we have $w \in I \subseteq L(I, R)$ by definition. Consider $w \in L$ with $|w| \geq m^2 + 6m$. The induction hypothesis states that every word $w' \in L$ with $|w'| < |w|$ belongs to $L(I, R)$. Factorize

$$w = x_0 \alpha \beta \gamma \delta y_0$$

such that $|x_0| = |y_0| = 3m$, $|\alpha\beta\gamma| = m^2$, $\beta \neq \varepsilon$, $\alpha \sim \alpha\beta$, $\gamma \sim \beta\gamma$ (see Pumping Lemma 2.1), and δ covers the remainder of the word w. For the correctness of the following steps in this proof it is important that the factors x_0, y_0, and $\alpha\beta\gamma$ each have a fixed length and only the length of δ is flexible.

The proof idea is somewhat similar as in the proof of Lemma 4.8: We pump up the factor β in the word w to β^j in order to obtain a word $\tilde{w}$. We pick j large enough such that no word in J can contain the factor $z = \alpha\beta^j\gamma\delta$; in particular, $\tilde{w} \notin J$. Therefore, it has to be created by a series of splicings from other words in L and at some point during the creation of $\tilde{w}$ by splicing position lies in the factor z of $\tilde{w}$; the series of splicings is of the form as described in Lemma 4.7 and Figure 4. Next, we pump down the factor β^j in $\tilde{w}$ again in order to obtain the original word w. We adapt the series of splicings, that created $\tilde{w}$, in order to obtain series of splicings which creates w; then, we ensure that all words used in this series of splicings are shorter than w and all rules belong to R. Using the induction hypothesis, this will conclude the proof.

Choose j sufficiently large ($j > |w| + m^2 + 10m$ and j is greater than the maximal length of the words in J). We let $z = \alpha\beta^j\gamma\delta$ and investigate the creation of $x_0zy_0 \in L$ by splicing in (J, S). As z cannot be a factor of a word in J, we can trace back the creation of x_0zy_0 by splicing to the point where the factor z is affected for the last time. Let $z_0 = x_0zy_0$ be created by k splicings from a word $z_k = x_kzy_k$ which is created by a splicing $(w_{k+1}, w_{k+2}) \vdash_s z_k$ with $w_{k+1}, w_{k+2} \in L$, $s \in S$, and the splicing position lies in the factor z. Furthermore, for $i = 1, \ldots, k$ the intermediate splicings are either

(i) $(w_i, z_i) \vdash_{r_i} x_{i-1}zy_{i-1} = z_{i-1}$, where $w_i \in L$, $r_i \in S$, $y_{i-1} = y_i$, and the splicing position lies at the left of the factor z or

(ii) $(z_i, w_i) \vdash_{r_i} x_{i-1}zy_{i-1} = z_{i-1}$, where $w_i \in L$, $r_i \in S$, $x_{i-1} = x_i$, and the splicing position lies at the right of the factor z.

As $|z| \geq m^2 + 10m$ we can apply Lemma 4.7. Thus, we may assume $w_1, \ldots, w_k \in I$, $r_1, \ldots, r_k \in R$, and $|x_k|, |y_k| < 5m$.

Consider a rule r_i in a splicing of the form (i). Suppose the fourth component of r_i covers a prefix of the factor $\alpha\beta^j$ in z which is longer than $\alpha\beta$ (as j is large, it cannot fully cover $\alpha\beta^j$). We may write $r_i = (u_1, v_1; u_2, v'v'')$ where v'' is the prefix of $\alpha\beta^{j/2}$. By extension (Lemma 4.2) the rule $(u_1, v_1; u_2, v'\alpha\beta^j)$ respects L, and so does the rule $\tilde{r}_i = (u_1, v_1; u_2, v'\alpha)$ because $\alpha \sim \alpha\beta^{j/2}$ (Lemma 4.3). The thusly obtained rule $\tilde{r}_i$ can be used in place of r_i. For convenience, we assume that every r_i is of the form of its corresponding $\tilde{r}_i$ from here on. Moreover, after we symmetrically treated rules of form (ii), these new rules $r_1, \ldots, r_k$ and the words $w_1, \ldots, w_k$ can be used in order to create $w = x_0\alpha\beta\gamma\delta y_0$ from $x_k\alpha\beta\gamma\delta y_k$ by splicing. Thus, if $x_k\alpha\beta\gamma\delta y_k$ belongs to $L(I, R)$, so does $w = x_0\alpha\beta\gamma\delta y_0$.

Now, consider the first splicing $(w_{k+1}, w_{k+2}) \vdash_s z_k = x_k z y_k$. Due to Lemma 4.4 we may assume that $s = (u, v_1; u_2, v)$, $w_{k+1} = xuv_1$ and $w_{k+2} = u_2 vy$ where $|v_1|, |u_2| < m$. We have

$$z_k = xuvy = x_k z y_k = x_k \alpha \beta^j \gamma \delta y_k$$

where xu is a proper prefix of $x_k z$ and vy is a proper suffix of zy_k because we required that the splicing position lies in z in z_k.

Next, we will pump down the factor $\alpha\beta^j\gamma$ to $\alpha\beta\gamma$ in z again in order to obtain the words $\tilde{x}, \tilde{u}, \tilde{v}, \tilde{y}$ from the word x, u, v, y, respectively. The pumping is done as in the proof of Lemma 4.8: For each pumping step do:

1. If u is covered by the factor $\alpha\beta^j\gamma$ (which we pump down in this step), extend u to the left such that it becomes a prefix of $\alpha\beta^j\gamma$. Symmetrically, if v is covered by the factor $\alpha\beta^j\gamma$, extend v to the right such that it becomes a suffix of $\alpha\beta^j\gamma$ (Lemma 4.2). After this extension the factor $\alpha\beta^j\gamma$ is covered by xu, uv, or vy.

2. If $\alpha\beta^j$ or $\beta^j\gamma$ is covered by one of x, u, v, or y, then replace this factor by $\alpha\beta$ or $\beta\gamma$, respectively. Skip next step.

3. If $\alpha\beta^j\gamma$ is covered by xu (the cases when $\alpha\beta^j\gamma$ is covered by uv or vy can be treated analogously), we can factorize $x = x'\alpha\beta^{j_1}\beta_1$ and $u = \beta_2\beta^{j_2}\gamma u'$ where $\beta_1\beta_2 = \beta$ and $j_1 + j_2 + 1 = j$. The results of pumping down are the words $x'\alpha\beta_1$ and $\beta_2\gamma u'$, respectively.

Observe that, $\tilde{x}\tilde{u} \sim xu$, $\tilde{v}\tilde{y} \sim vy$, $\tilde{x}\tilde{u}\tilde{v}\tilde{y} = x_k\alpha\beta\gamma\delta y_k$, and the rule $t = (\tilde{u}, v_1; u_2, \tilde{v})$ respects L. Furthermore, if we used extension for u (or v), then $|\tilde{u}| \leq m^2$ (resp. $|\tilde{v}| \leq m^2$). No matter whether we used extension or not, $t \in R$. As

$$|\tilde{x}\tilde{u}v_1| < |x_k z| + |v_1| \leq |z| + 6m = |w|$$

we have $\tilde{w}_{k+1} = \tilde{x}\tilde{u}v_1 \in L(I, S)$. Symmetrically, $u_2\tilde{v}\tilde{y} < |w|$ and $\tilde{w}_{k+2} = u_2\tilde{v}\tilde{y} \in L(I, S)$. We conclude that $(\tilde{w}_{k+1}, \tilde{w}_{k+2}) \vdash_t x_k\alpha\beta\gamma\delta y_k \in L(I, R)$ and, therefore, $w = x_0\alpha\beta\gamma\delta y_0 \in L(I, R)$ as well. □

5 Decidability

The main question we intended to answer when starting our investigation was, whether or not it is decidable if a given regular language L is a splicing language. If we can decide whether or not a splicing rule respects a regular language and if we can construct a (non-deterministic) finite automaton (NFA) accepting the language generated by a given splicing system, then we can decide whether or not L is a classic splicing language (Pixton splicing language) as follows. We compute the splicing system (I, R) as given in Theorem 4.1 (resp. Theorem 3.1) and we compute a finite automaton accepting the splicing language $L(I, R)$. Theorem 4.1 (resp. Theorem 3.1) implies that L is a splicing language if and only if $L = L(I, R)$. Recall that equivalence of regular languages is decidable, for example, by constructing and comparing the minimal deterministic finite automata of both languages.

It is known from [8, 13] that it is decidable whether or not a classic splicing rule respects a regular language. Furthermore, there is an effective construction of a finite automaton which accepts the language generated by a Pixton splicing system [17]. As mentioned earlier, Pixton splicing systems are more general than classic splicing systems, which means the latter result applies to classic splicing systems, too. Such a construction for classic splicing systems is also given in [12].

Let us prove that it is decidable whether or not a Pixton splicing rule r respects a regular language L. Actually, we will decide whether or not the set $[r]_L$ respects L, which is equivalent by Lemma 4.3. The proof can easily be adapted in order to prove that it is decidable whether a classic splicing rule respects L.

Lemma 5.1. *Let L be a regular language and let r be a Pixton splicing rule. It is decidable whether r respects L or not.*

Proof. Let $\sim$ denote the equivalence relation $\sim_L$ and $[\,\cdot\,]$ denote the corresponding equivalence classes $[\,\cdot\,]_L$.

Let $r = (u_1, u_2; v)$. We define the two sets $S_1, S_2 \subseteq M_L$ as

$$S_1 = \{X \in M_L \mid \exists Y \colon X[u_1]Y \subseteq L\}, \qquad S_2 = \{Y \in M_L \mid \exists X \colon X[u_2]Y \subseteq L\},$$

i. e., $[x_1]$ belongs to S_1 if and only if $x_1u_1y_1 \in L$ for some word y_1 and $[y_2]$ belongs to S_2 if and only if $x_2u_2y_2 \in L$ for some word x_2. We claim that r respects L if and only if $X[v]Y \subseteq L$ for all $X \in S_1$ and $Y \in S_2$, which is a property that can easily be decided.

Firstly, suppose r respects L. For $X \in S_1$ and $Y \in S_2$ choose words $x_1 \in X$ and $y_2 \in Y$. By definition of S_1 and S_2, there is y_1 and x_2 such that $x_1u_1y_1 \in L$, $x_1u_1y_1 \in L$, and as r respects L, $x_1vy_2 \in L$. This implies $X[v]Y \subseteq L$.

Vice verse, suppose $X[v]Y \subseteq L$ for all $X \in S_1$ and $Y \in S_2$. For all $x_iu_iy_i \in L$ with $i = 1, 2$, we have $[x_1] \in S_1$ and $[y_2] \in S_2$. Therefore, $x_1vy_2 \in [x_1][v][y_2] \subseteq L$ and r respects L. □

These observations lead to the following decidability results.

Corollary 5.2.

i.) For a given regular language L, it is decidable whether or not L is a classic splicing language. Moreover, if L is a classic splicing language, a splicing system (I, R) generating L can be effectively constructed.

ii.) For a given regular language L, it is decidable whether or not L is a Pixton splicing language. Moreover, if L is a Pixton splicing language, a splicing system (I, R) generating L can be effectively constructed.

Final Remarks

It has been known since 1991 that the class $\mathcal{S}$ of languages that can be generated by a splicing system is a proper subclass of the class of regular languages. However, to date, no other natural characterization for the class $\mathcal{S}$ exists. The problem of deciding whether or not a regular language is generated by a splicing system is a fundamental problem in this context and has remained unsolved. To the best of our knowledge, the problem was first stated in the literature in 1998 [11]. In this paper we give a positive answer to this open problem.

Some remarks are in order regarding thecomplexity of the decision algorithm. Let L be a regular language given as syntactic monoid M_L and (I, R) be the splicing system described in Theorem 4.1 (resp. Theorem 3.1). An automaton which accepts $L(I, R)$, created as described in Section 5, has a state set of size in $2^{\mathcal{O}(m^2)}$, where $m = |M_L|$. Deciding the equivalence of two regular languages, given as NFAs, is known to be PSPACE-complete [20]; hence, the naive approach to decide whether $L = L(I, R)$ or not uses double exponential time $2^{2^{\mathcal{O}(m^2)}}$. As there may be an exponential gap between an NFA accepting L and the syntactic monoid M_L, the complexity, when considering an NFA as input, becomes triple exponential. Improving the complexity of the algorithm is subject of future research.

Finally, let us note that the related problem of characterizing the class splicing languages intrinsically remains open. Such a characterization of splicing languages will lead to a better understanding of the language class of splicing languages and has the potential to improve the complexity of the decision algorithm.

References

[1] P. Bonizzoni. Constants and label-equivalence: A decision procedure for reflexive regular splicing languages. *Theor. Comput. Sci.*, 411(6):865–877, 2010.

[2] P. Bonizzoni, C. de Felice, and R. Zizza. The structure of reflexive regular splicing languages via Schützenberger constants. *Theor. Comput. Sci.*, 334(1-3):71–98, 2005.

[3] P. Bonizzoni, C. de Felice, and R. Zizza. A characterization of (regular) circular languages generated by monotone complete splicing systems. *Theor. Comput. Sci.*, 411(48):4149–4161, 2010.

[4] P. Bonizzoni, C. Ferretti, G. Mauri, and R. Zizza. Separating some splicing models. *Inf. Process. Lett.*, 79(6):255–259, 2001.

[5] P. Bonizzoni and N. Jonoska. Regular splicing languages must have a constant. In G. Mauri and A. Leporati, editors, *Developments in Language Theory*, volume 6795 of *Lecture Notes in Computer Science*, pages 82–92. Springer Berlin / Heidelberg, 2011.

[6] K. Culik II and T. Harju. Splicing semigroups of dominoes and DNA. *Discrete Applied Mathematics*, 31(3):261–277, 1991.

[7] R. Gatterdam. Splicing systems and regularity. *International Journal of Computer Mathematics*, 31(1-2):63–67, 1989.

[8] E. Goode. *Constants and Splicing Systems*. PhD thesis, Binghamton University, 1999.

[9] E. Goode and D. Pixton. Recognizing splicing languages: Syntactic monoids and simultaneous pumping. *Discrete Applied Mathematics*, 155(8):989–1006, 2007.

[10] T. Head. Formal language theory and DNA: an analysis of the generative capacity of specific recombinant behaviors. *Bulletin of Mathematical Biology*, 49(6):737–759, 1987.

[11] T. Head. Splicing languages generated with one sided context. In G. Păun, editor, *Computing With Bio-molecules: Theory and Experiments*, pages 269–282. Springer Verlag, 1998.

[12] T. Head and D. Pixton. Splicing and regularity. In Z. Ésik, C. Martín-Vide, and V. Mitrana, editors, *Recent Advances in Formal Languages and Applications*, volume 25 of *Studies in Computational Intelligence*, pages 119–147. Springer, 2006.

[13] T. Head, D. Pixton, and E. Goode. Splicing systems: Regularity and below. In M. Hagiya and A. Ohuchi, editors, *DNA*, volume 2568 of *Lecture Notes in Computer Science*, pages 262–268. Springer, 2002.

[14] J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979.

[15] L. Kari and S. Kopecki. Deciding whether a regular language is generated by a splicing system. In D. Stefanovic and A. Turberfield, editors, *DNA*, volume 7433 of *Lecture Notes in Computer Science*, pages 98–109. Springer, 2012.

[16] S. Kim. An algorithm for identifying spliced languages. In T. Jiang and D. Lee, editors, *COCOON*, volume 1276 of *Lecture Notes in Computer Science*, pages 403–411. Springer, 1997.

[17] D. Pixton. Regularity of splicing languages. *Discrete Applied Mathematics*, 69(1-2):101–124, 1996.

[18] G. Păun. On the splicing operation. *Discrete Applied Mathematics*, 70(1):57 – 79, 1996.

[19] M. Schützenberger. Sur certaines opérations de fermeture dans le langages rationnels. *Symposia Mathematica*, 15:245–253, 1975.

[20] L. Stockmeyer and A. Meyer. Word problems requiring exponential time: Preliminary report. In A. Aho, A. Borodin, R. Constable, R. Floyd, M. Harrison, R. Karp, and H. Strong, editors, *STOC*, pages 1–9. ACM, 1973.