Deciding whether a Regular Language is Generated by a Splicing System*

Lila Kari and Steffen Kopecki

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

Abstract. 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 an operation on formal languages, called *splicing*, which models DNA recombination, a cut-and-paste operation on DNA strands under the action of restriction enzymes and ligases. A splicing system consists of a set of *axioms* or *initial words* and a set of *(splicing) rules*. The most commonly used definition for a splicing rule is a quadruple 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 [17].

Splicing as a language-theoretic word operation is meant to abstract the action of two compatible restriction enzymes and the ligase enzyme on two DNA strands. The first enzyme recognizes the subword u_1v_1, called its *restriction site*, in any DNA string and cuts the string containing this subword 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 in some sense "compatible", the enzyme ligase aids then the recombination (catenation) of the first segment of one cut string with the second segment of another cut string.

* This research was supported by the Natural Sciences and Engineering Research Council of Canada Discovery Grant R2824A01 and UWO Faculty of Science grant to L. K.

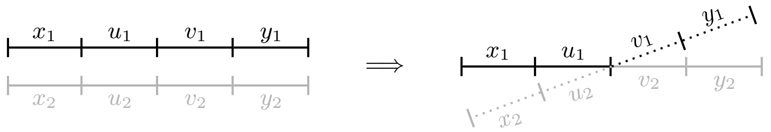

Fig. 1. Splicing of the words $x_1u_1v_1y_1$ and $x_2u_2v_2y_2$ by the rule $r = (u_1, v_1; u_2, v_2)$. The splicing result is the word $x_1u_1v_2y_2$.

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. 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 generate regular languages, only; see also [12,16]. 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 a 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]; e. g., the language $b(aa)^*$ is generated by the axioms $\{b, baa\}$ and the rule $(baa, \varepsilon; b, \varepsilon)$, where ε is the empty word.

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* has been 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, too. A word v is a Schützenberger constant of a language L if $x_1vy_1 \in L$ and $x_2vy_2 \in L$ imply $x_1vy_2 \in L$ [18]. 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, e. g., 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 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 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.

In this paper we settle the problem by proving that for a given regular language L, it is indeed decidable whether L is generated by a splicing system (which is not necessarily reflexive), Corollary 1. More precisely, if the language L is generated by a splicing system, then it is generated by one particular splicing system whose size is a function of the size of the syntactic monoid of L, Theorem 1. If m is the size of the syntactic monoid of L, then all axioms and all components of rules have a length in $\mathcal{O}(m^2)$. By results from [12, 13], we can construct a finite automaton which accepts the language generated by this splicing system, compare it with a finite automaton which accepts L, and, thus, decide whether L is generated by a splicing system.

Due to page limitations the proofs of most of the lemmas have been omitted in this version of the paper. The missing proofs can be found in the arXiv version [15].

2 Notation and Definitions

We assume the reader to be familiar with the fundamental concepts of language theory, see [14]. Let Σ be an *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 consider languages over the alphabet Σ, only. We consider the letters of Σ to be ordered and for words $u, v \in \Sigma^*$ we denote the *(strict) length-lexicographical order* by $u \leq_{\ell\ell} v$ (resp., $u <_{\ell\ell} v$); i. e., $u \leq_{\ell\ell} v$ if either $|u| \leq |v|$, or $|u| = |v|$ and u is equal or less than v in lexicographic order. For a length bound $m \in \mathbb{N}$ 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 $\Sigma^{<m} = \bigcup_{i<m} \Sigma^i$. Let $w \in \Sigma^*$ be a word. 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*.

Every language L induces a *syntactic congruence* $\sim_L$ over words such that $u \sim_L v$ if and only if for all words x, y we have $xuy \in L \iff xvy \in L$. 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 $M_L = \Sigma^*/{\sim_L} = \{[u]_L \mid u \in \Sigma^*\}$. 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 1. *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 $\gamma \sim_L \beta\gamma$.*

Lemma 2. *Let L be a regular language. Every element $X \in M_L$ contains a word $x \in X$ with $|x| < |M_L|$.*

3 Splicing Systems and Regular Languages

We consider the splicing operation as defined in [17]. This is the most commonly used definition for splicing in formal language theory. The notation we use has been employed in previous papers, see e. g., [2, 9]. A quadruple 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 side* of r, respectively. 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 sides, in order to create the new word $z = x_1u_1v_2y_2$, see again Figure 1. This operation is called *splicing* and it is denoted by $(w_1, w_2) \vdash_r z$. The *splicing position* of this splicing is the position between the factors x_1u_1 and v_2y_2 in z.

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

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

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, 16]. 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 $L(I, R)$ and a rule $r \notin R$ respects $L(I, R)$ if and only if $L(I, R \cup \{r\}) = L(I, R)$. Furthermore, we say a splicing step $(w_1, w_2) \vdash_r z$ *respects* a language L if $w_1, w_2 \in L$ and r respects L; obviously, this implies $z \in L$, too.

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

Theorem 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 $L = L(I, R)$.

The structure of this section is the following. In Section 3.1 we will present techniques to obtain rules that respect a regular language 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; similar results can be found in [8, 9]. In Section 3.2 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 3.3 we prove Theorem 1.

3.1 Rule Modifications

Our first lemma tells us that we can extend the sides of a rule r such that the extended rule respects all languages that are respected by r.

Lemma 3. *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.*

Henceforth, we will refer to the rules $(xu_1, v_1; u_2, v_2)$, $(u_1, v_1x; u_2, v_2)$ as extensions of the left side and to $(u_1, v_1; xu_2, v_2)$, $(u_1, v_1; u_2, v_2x)$ as extensions of the right side.

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 $s \in [r]_L$.

Lemma 4. *Let r be a rule which respects a language L. Every rule $s \in [r]_L$ respects L.*

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 2 on the left side. 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 3 and 4, we can find shorter words in L and a modified splicing rule which can be used to obtain $x_1u_1v_2y_2$.

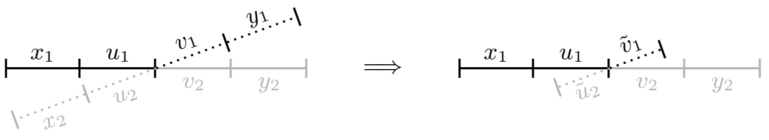

Fig. 2. Replacing v_1y_1 and x_2u_2 by *short* words.

Lemma 5. *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 there is a splicing $(\tilde{w}_1, \tilde{w}_2) \vdash_s z$ which respects L where $\tilde{w}_1$, $\tilde{w}_2$, and s have the properties described above.

3.2 Series of Splicings

Let us consider the creation of words by a series of splicings. We begin with a simple observation. In case when a word is created by two (or more) successive splicings, but the sides of the splicings do not cover the splicing position of the other splicing, then 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 1 is the same as in the Figure 3.

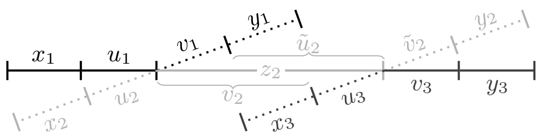

Fig. 3. The word $x_1u_1z_2v_3y_3$ can be created by either using the right splicing first or by using the left splicing first.

Remark 1. Let $w_1 = x_1u_1v_1y_1$, $w_2 = x_2u_2z_2\tilde{v}_2y_2$, where v_2 is a prefix of z_2 and $\tilde{u}_2$ is a suffix of z_2, $w_3 = x_3u_3v_3y_3$ be words and $r_1 = (u_1, v_1; u_2, v_2)$, $r_2 = (\tilde{u}_2, \tilde{v}_2; u_3, v_3)$ be rules. In order to create the word $z = x_1u_1z_2v_3y_3$ by splicing, we may use splicings

$$\begin{aligned}
&(w_1, w_2) \vdash_{r_1} x_1u_1z_2\tilde{v}_2y_2 = z', && (z', w_3) \vdash_{r_2} z \qquad \text{or}\\
&(w_2, w_3) \vdash_{r_2} x_2u_2z_2v_3y_3 = z'', && (w_1, z'') \vdash_{r_1} z.
\end{aligned}$$

Consider a splicing system (J, S) and the 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

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

be a set of rules. 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).

Let $xzy \in L$ be a word where the length of the middle factor z is at least $|\mu|$. The creation of xzy by splicing in (I, R) can be traced back to a word $x_1zy_1 = z_1$ where either $z_1 \in I$ or where z_1 is created by a splicing that affects the factor z, i. e., the splicing position lies in the factor z. The next lemma describes this creation of $xzy = z_{k+1}$ by k splicings in (I, R), and shows that we can choose the rules and words which are used to create z_{k+1} from z_1 such that the words and bridges of rules are not significantly longer than $\ell = \max\{|x|, |y|\}$.

Lemma 6. *Let L be a splicing language, let $\ell, n \in \mathbb{N}$, let $m = |M_L|$, and let μ be a word with $|\mu| \geq \ell + 2m$ such that for $I = \Sigma^{\leq n} \cap L$ and $R = \left\{r \in W_\mu^4 \;\middle|\; r \text{ respects } L\right\}$ we have $L = L(I, R)$.*

Let $z_{k+1} = x_{k+1}zy_{k+1}$ with $|z| \geq |\mu|$ and $|x_{k+1}|, |y_{k+1}| \leq \ell$ be a word that is created by k splicings from a word $z_1 = x_1zy_1$ where either $z_1 \in I$ or z_1 is created by a splicing $(\tilde{w}_1, \tilde{w}_2) \vdash_s z_1$ where $\tilde{w}_1, \tilde{w}_2 \in L$, s respects L, and the splicing position lies in the factor z. Furthermore, for $i = 1, \ldots, k$ the intermediate splicings are either

(i) $(w_i, z_i) \vdash_{r_i} x_{i+1}zy_{i+1} = z_{i+1}$, $w_i \in L$, $r_i \in R$, $y_{i+1} = y_i$, *and the splicing position lies on the left of the factor z or*

(ii) $(z_i, w_i) \vdash_{r_i} x_{i+1}zy_{i+1} = z_{i+1}$, $w_i \in L$, $r_i \in R$, $x_{i+1} = x_i$, *and the splicing position lies on the right of the factor z.*

There are rules and words creating z_{k+1}, as above, satisfying in addition:

1. *There is $k' \leq k$ such that for $i = 1, \ldots, k'$ all splicings are of the form (i) and for $i = k'+1, \ldots, k$ all splicings are of the form (ii).*
2. *For $i = 1, \ldots, k'$ the following bounds apply: $|x_i| < \ell + 2m$, $|w_i| < \ell + 2m$, $r_i \in \Sigma^{<\ell+m} \times \Sigma^{<2m} \times \Sigma^{<2m} \times W_\mu$, and $x_{k'+1} = x_{k'+2} = \cdots = x_{k+1}$.*
3. *For $i = k'+1, \ldots, k$ the following bounds apply: $|y_i| < \ell+2m$, $|w_i| < \ell+2m$, $r_i \in W_\mu \times \Sigma^{<2m} \times \Sigma^{<2m} \times \Sigma^{<\ell+m}$, and $y_1 = y_2 = \cdots = y_{k'+1}$.*

In particular, if $n \geq \ell + 2m$, then $w_1, \ldots, w_k \in I$.

The first statement follows by the fact that $|z| \geq |\mu|$ and by Remark 1. The proof of the other two statements requires a much more complicated analysis of the creation of the word z_{k+1} by splicing which is omitted in this version of the paper.

3.3 Proof of Theorem 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 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 7, we proof 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 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 7. *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*

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

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).

Let (I', R') be a splicing system that generates $L = L(I', R')$ and let n such that $n - 6m$ is larger than any word in I' and larger than any component of a rule in R'. As in the claim, let $I = \Sigma^{\leq n} \cap L$.

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

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

and the language $L_\mu = L(I, R_\mu)$; clearly, $L_\mu \subseteq L$. For two words $\mu \leq_{\ell\ell} v$ we 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 $L_v = L$. As $L = L(I', R')$, there exists a word μ such that $L_\mu = L$ and $|\mu| + 6m \leq n$. Let b be the lexicographically largest letter in Σ. For $\nu = b^{m^2+10m-1}$ the set R_ν contains exactly the rules that respect L and where every component has a length of less than $m^2 + 10m$; therefore, $R_\nu = R$ and if $L_\nu = L$, the claim holds. For the sake of contradiction assume $L_\nu \neq L$ and let μ be the smallest word, in the length-lexicographic order, such that $L_\mu = L$; hence, $|\mu| \geq m^2 + 10m$. Let μ' be the next-smaller word than μ, in the length-lexicographic order, and let $S = R_{\mu'}$. Note that $L(I, S) \subsetneq L$ and $R_\mu \setminus S$ contains 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 $\tilde{w} \in L$ with $|\tilde{w}| < |w|$, we have $\tilde{w} \in L(I, S)$. Factorize $w = xzy$ with $|x| = |y| = 3m$, n. b., $|z| \geq |\mu|$, otherwise $w \in I$. 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 $\gamma \sim \beta\gamma$ (Lemma 1).

We will show that there is a series of splicings which creates w from a set of shorter words and by using splicing rules from S. This yields a contradiction to the choice of w. In order to find this series of splicings we investigate the creation of a word $x\tilde{z}y$ where $\tilde{z}$ is derived by using a pumping argument on all factors $\alpha\beta\gamma$ in z.

Let j be a sufficiently large even number ($j > 4|\mu| + |z|$ will suffice). Let $\tilde{z}$ be the word that we obtain by replacing all factors $\alpha\beta\gamma$ by $\alpha\beta^j\gamma$ in z by the following pumping algorithm:

```
1. $\tilde{z} := z$;
2. if there is a factor $\alpha\beta\gamma$ of $\tilde{z}$ such that neither
   (a) the factor $\alpha\beta\gamma$ is a prefix of a factor $\alpha\beta^{j/2}$ in $\tilde{z}$ nor
   (b) the factor $\alpha\beta\gamma$ is a suffix of a factor $\beta^{j/2}\gamma$ in $\tilde{z}$,
   then replace this factor by $\alpha\beta^j\gamma$;
3. repeat step 2 until there is no such factor $\alpha\beta\gamma$ left.
```

A proof that the algorithm will terminate, hence $\tilde{z}$ is well defined, can be found in the arXiv version [15]. The new word $\tilde{z}$ may still contain the factor $\alpha\beta\gamma$, but if it does, then (a) or (b) holds. By induction and as $\alpha\beta\gamma \sim \alpha\beta^j\gamma$, it is easy to see that $\tilde{z} \sim z$.

Let us trace back the creation of $x\tilde{z}y \in L$ by splicing in (I, R_μ) to a word $x_1\tilde{z}y_1$ where either $x_1\tilde{z}y_1 \in I$ or where $x_1\tilde{z}y_1$ is created by a splicing that affects $\tilde{z}$, i. e., the splicing position lies within the factor $\tilde{z}$. Let $z_{k+1} = x_{k+1}\tilde{z}y_{k+1}$, where $x_{k+1} = x$ and $y_{k+1} = y$, be created by k splicings from a word $z_1 = x_1\tilde{z}y_1$ where either $x_1\tilde{z}y_1 \in I$ or $x_1\tilde{z}y_1$ is created by a splicing $(\tilde{w}_1, \tilde{w}_2) \vdash_s z_1$ with $\tilde{w}_1, \tilde{w}_2 \in L$, $s \in R_\mu$, and the splicing position lies in the factor $\tilde{z}$. Furthermore, for $i = 1, \ldots, k$ the intermediate splicings are either

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

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

Note that $|\tilde{z}| \geq |z| \geq |\mu|$ and, therefore, we can apply Lemma 6 (with $\ell = 3m$). Thus, $w_i \in I$ 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 6, $r_i \in \Sigma^{<4m} \times \Sigma^{<2m} \times \Sigma^{<2m} \times W_\mu$. 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 $\alpha\beta^{j/2}$). By extension (Lemma 3), we may write $r_i = (u_1, v_1; u_2, \tilde{v}\alpha\beta^e)$ for some $e \geq 1$. By Lemma 4 and as $\alpha \sim \alpha\beta$, we may replace this rule by $(u_1, v_1; u_2, \tilde{v}\alpha)$. Note that, as the fourth component got shorter, now $r_i \in S$.

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 $w = x_{k+1}zy_{k+1}$ from x_1zy_1 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 $\tilde{z}$.

Furthermore, the rules $r_1, \ldots, r_k$ all belong to S. By contradiction, suppose $r_i \notin S$ for some i and, by symmetry, suppose the 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 $|\delta_1| \geq 5m > |x_i|$, $\alpha\beta\gamma$ is a factor of $\tilde{z}$. The pumping algorithm ensured that (a) the prefix α is succeeded by $\beta^{j/2}$ or (b) the suffix γ is preceded by $\beta^{j/2}$. As $j/2$ is very large and the splicing position is too close to the left end of z_i, case (b) is not possible. Thus, 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 $r_i \in S$ — the contradiction.

Let us summarize: If x_1zy_1 was in $L(I, S)$, then $w \in L(I, S)$ as well, which would contradict the choice of w. If $z_1 = x_1\tilde{z}y_1 \in I$, then x_1zy_1, which is at most as long as z_1, would belong to I and we are done. We only have to consider the case when $(\tilde{w}_1, \tilde{w}_2) \vdash_s z_1 = x_1\tilde{z}y_1$ and the splicing position lies within the factor $\tilde{z}$. We will show that, from this splicing, we derive another splicing $(\hat{w}_1, \hat{w}_2) \vdash_t x_1zy_1$ which respects $L(I, S)$ and, therefore, yields the contradiction.

Let $s = (u, v_1; u_2, v)$, $\tilde{w}_1 = xuv_1$ and $\tilde{w}_2 = u_2vy$ where $|v_1|, |u_2| < m$, by Lemma 5 (here, x and y are newly chosen words). We have

$$z_1 = x_1\tilde{z}y_1 = xuvy$$

where xu is a proper prefix of $x_1\tilde{z}$ and vy is a proper suffix of $\tilde{z}y_1$.

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_1 by splicing. If $s \notin S$, then $u = \mu$ or $v = \mu$. Suppose $u = \mu = \delta_1\alpha\beta\gamma\delta_2$. Thus, $\alpha\beta\gamma$ is a factor of $\tilde{z}$, as $|\delta_1| \geq 5m > |x_1|$, and, as such, either (a) α is succeeded by $\beta^{j/2}$ or (b) γ is preceded by $\beta^{j/2}$. If (b) holds, $\delta_1\alpha$ is a suffix of a word in β^+. 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 $\beta_2\gamma\delta_1$ and use this new rule $\tilde{s}$ in order to splice $(\tilde{w}_1, \tilde{w}_2) \vdash_{\tilde{s}} z_1$. Note that the first component is now shorter than μ. Otherwise, (a) holds and $\gamma\delta_2v$ 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 3) 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 both components are shorter than μ, we see that $\tilde{s} \in S$ and

$$(x\delta_1\alpha\beta^{\ell_1-1}\beta_1 v_1, u_2\beta_2\beta^{\ell_2+1}\gamma y) \vdash_{\tilde{s}} z_1,$$

i. e., we have shifted one of the occurrences of β from $\tilde{w}_1$ to $\tilde{w}_2$. 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 $s \in S$.

Next, 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 words $\hat{x}, \hat{u}, \hat{v}, \hat{y}$ from the words x, u, v, y, respectively. For each pumping step:

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 3). Observe that extension ensures that the factor $\alpha\beta^j\gamma$ is covered by either xu, uv, or vy. If $\alpha\beta^j$ or $\beta^j\gamma$ is fully covered by one of x, u, v, or y, then replace this factor by $\alpha\beta$ or $\beta\gamma$, respectively. Otherwise, let us show how to pump when $\alpha\beta^j\gamma$ is covered by xu. The cases when $\alpha\beta^j\gamma$ is covered by uv or vy can be treated analogously. We can factorize

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

where $\beta_1\beta_2 = \beta$ and $j_1 + j_2 + 1 = j$. The pumping result are the words $\tilde{x}\alpha\beta_1$ and $\beta_2\gamma\tilde{u}$, respectively.

Observe that, after reversing all pumping steps, $\hat{x}\hat{u} \sim xu$, $\hat{v}\hat{y} \sim vy$, $\hat{x}\hat{u}\hat{v}\hat{y} = x_1zy_1$, and the rule $t = (\hat{u}, v_1; u_2, \hat{v})$ respects L. Furthermore, if we used extension for u (or v) in one of the steps, then $|\hat{u}| \leq m^2$ (resp., $|\hat{v}| \leq m^2$); in any case $t \in S$. Recall that w was chosen as the shortest word from $L \setminus L(I, S)$. As $|\hat{x}\hat{u}v_1|, |u_2\hat{v}\hat{y}| < |z| + 6m = |w|$, the words $\hat{x}\hat{u}v_1$ and $u_2\hat{v}\hat{y}$ belong to $L(I, S)$, and as $(\hat{x}\hat{u}v_1, u_2\hat{v}\hat{y}) \vdash_t x_1zy_1$, we conclude that x_1zy_1 as well as w belong to $L(I, S)$ — the desired contradiction. □

Now, let us outline how the proof of Theorem 1 can be concluded. The full proof can be found in the arXiv version [15].

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 $L = L(I, R)$. By Lemma 7, 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 $|w| < m^2 + 6m$, by definition, $w \in I \subseteq L(I, R)$.

For $w \in L$ with $|w| \geq m^2 + 6m$, the induction hypothesis states that every word $\tilde{w} \in L$ with $|\tilde{w}| < |w|$ belongs to $L(I, R)$. Factorize $w = x\alpha\beta\gamma\delta y$ such that $|x|, |y| = 3m$, $|\alpha\beta\gamma| = m^2$, $\beta \neq \varepsilon$, $\alpha \sim \alpha\beta$, and $\gamma \sim \beta\gamma$.

The proof idea is similar to the idea in the proof of Lemma 7, but this time we are using induction instead of a proof-by-contradiction. We use a pumping argument on β in order to obtain a very long word $\tilde{w}$. This word has to be created by a series of splicings in (J, S). Due to Lemma 6 these splicings can be modified in order to create $\tilde{w}$ by splicing from a set of strictly shorter words and with rules from R. Just like in the proof of Lemma 7, almost the same words and rules can be used in order to create w from a set of strictly shorter words and with rules from R. Then, the induction hypothesis yields $w \in L(I, R)$.

4 Conclusion and Final Remarks

The main question we intended to answer when starting our investigation was, if it is decidable whether a given regular language L is a splicing language. If we can decide whether a splicing rule respects a regular language and if we can construct a (non-deterministic) finite automaton accepting the language generated by a given splicing system, then we can decide whether L is a splicing language as follows. We compute the splicing system (I, R) as given in Theorem 1, we compute a finite automaton accepting the splicing language $L(I, R)$, and we test whether $L(I, R)$ equals to L. Recall that Theorem 1 implies that L is a splicing language if and only if $L = L(I, R)$ and that equivalence of regular languages is decidable [14]. It is known from [8, 13] that it is decidable whether 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 splicing system [12, 16]. These observations lead to the following decidability result.

Corollary 1. *For a given regular language L, it is decidable whether or not L is a splicing language. Moreover, if L is a splicing language, a splicing system (I, R) generating L can be effectively constructed.*

Another variant of splicing has been defined by Pixton in [16]. Pixton's variant of splicing can be seen as more general than the classical splicing, which we investigated in this paper, because every classical splicing rule can easily be translated into a Pixton splicing rule, but not the other way around. Actually, the class of classical splicing languages is strictly included in the class of Pixton splicing languages [4]. In the online version of our paper [15] we also prove that if a regular language L is a Pixton splicing language, then it is generated by one particular Pixton splicing system whose size is a function of the size of the syntactic monoid of L. A decidability result, analogous to Corollary 1, follows immediately.

As final remarks, note that 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 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 solved this long standing open problem.

References

1. P. Bonizzoni. Constants and label-equivalence: A decision procedure for reflexive regular splicing languages. *TCS*, 411(6):865–877, 2010.
2. P. Bonizzoni, C. de Felice, and R. Zizza. The structure of reflexive regular splicing languages via Schützenberger constants. *TCS*, 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. *TCS*, 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 *LNCS*, pages 82–92. Springer Berlin / Heidelberg, 2011.
6. K. Culik II and T. Harju. Splicing semigroups of dominoes and DNA. *Discrete Applied Math.*, 31(3):261–277, 1991.
7. R. W. 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 Math.*, 155(8):989–1006, 2007.
10. T. Head. Formal language theory and DNA: an analysis of the generative capacity of specific recombinant behaviors. *Bull. of Math. Bio.*, 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 *LNCS*, pages 262–268. Springer, 2002.
14. J. E. Hopcroft and J. D. 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. *CoRR*, abs/1112.4897, 2011.
16. D. Pixton. Regularity of splicing languages. *Discrete Applied Math.*, 69(1-2):101–124, 1996.
17. G. Păun. On the splicing operation. *Discrete Applied Math.*, 70(1):57 – 79, 1996.
18. M. P. Schützenberger. Sur certaines opérations de fermeture dans le langages rationnels. *Symposia Mathematica*, 15:245–253, 1975.