Journal of Computer and System Sciences 84 (2017) 263–287

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

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

ARTICLE INFO

Article history:
Received 31 August 2012
Received in revised form 26 January 2015
Accepted 5 September 2016
Available online 13 October 2016

Keywords:
Splicing systems
Decidability

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

$$5'\text{—}\alpha\text{—}\mathsf{T}\overset{\downarrow}{\mathsf{C}}\,\mathsf{G}\,\mathsf{A}\text{—}\beta\text{—}3' \qquad 5'\text{—}\gamma\text{—}\mathsf{G}\overset{\downarrow}{\mathsf{C}}\,\mathsf{G}\,\mathsf{C}\text{—}\delta\text{—}3'$$
$$3'\text{—}\overline{\alpha}\text{—}\mathsf{A}\,\mathsf{G}\,\mathsf{C}\overset{\downarrow}{\mathsf{T}}\text{—}\overline{\beta}\text{—}5' \quad \text{and} \quad 3'\text{—}\overline{\gamma}\text{—}\mathsf{C}\,\mathsf{G}\,\mathsf{C}\overset{\downarrow}{\mathsf{G}}\text{—}\overline{\delta}\text{—}5'$$

* Corresponding author.
E-mail addresses: lila@csd.uwo.ca (L. Kari), steffen.kopecki@gmail.com (S. Kopecki).

http://dx.doi.org/10.1016/j.jcss.2016.10.001
0022-0000/

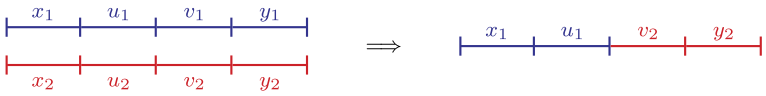

Fig. 1. Splicing of the words $x_1u_1v_1y_1$ and $x_2u_2v_2y_2$ by the rule $r = (u_1, v_1; u_2, v_2)$.

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

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

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 Fig. 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 (T, CGA; G, CGC); the rules (TC, GA; GC, GC) or (TCG, A; GCG, 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.

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