Acta Cybernetica **23** (2018) 981–993.

On Derivation Languages of a Class of Splicing Systems

Kalpana Mahalingam[ab], Prithwineel Paul[b], and Erkki Mäkinen[c]

Abstract

Derivation languages are language theoretical tools that describe halting derivation processes of a generating device. We consider two types of derivation languages, namely Szilard and control languages for splicing systems where iterated splicing is done in non-uniform way defined by Mitrana, Petre and Rogojin in 2010. The families of Szilard (rules and labels are mapped in a one to one manner) and control (more than one rule can share the same label) languages generated by splicing systems of this type are then compared with the family of languages in the Chomsky hierarchy. We show that context-free languages can be generated as Szilard and control languages and any non-empty context-free language is a morphic image of the Szilard language of this type of system with finite set of rules and axioms. Moreover, we show that these systems with finite set of axioms and regular set of rules are capable of generating any recursively enumerable language as a control language.

Keywords: Splicing systems, Szilard languages, Control languages

1 Introduction

The information regarding terminal derivation processes of a generative device is well studied in the literature. Each rule of the generative system in question is labeled and the given sequence of labels is considered as the output of the computation. The set of all such words constitute a language. When the labelling is done in a one to one fashion, the set of all labeled sequences is called a Szilard language. Szilard languages have been defined for a variety of generative mechanisms (for Chomsky grammars [8, 9, 14, 12], for regulated rewritings [5, 18] and for grammar systems [6, 10], to name a few) and their closure and decidability properties and

[a]corresponding author

[b]Department of Mathematics,Indian Institute of Technology, Madras, Chennai - 36. E-mail: `kmahalingam@iitm.ac.in, prithwineelpaul@gmail.com`

[c]Faculty of Natural Sciences/Computer Science, University of Tampere, Finland. E-mail: `em@sis.uta.fi`

DOI: 10.14232/actacyb.23.4.2018.1

complexity ([2, 3]) have been studied extensively. The study of the derivation process has also been extended in the context of P systems ([15, 16, 21, 20]). Since P systems are parallel computing devices and several rules can be used in a single computation step, one to one mapping of the labels may lead to complications (see [1]). To overcome this, all rules that are used in a computation step are labeled with the same symbol or some of them are labeled with the empty symbol λ. The set of all sequences of labels that lead to a halting computation is called the control language. The characterization of such control languages in terms of Chomsky hierarchy has been discussed for various P systems [15, 16, 21, 20]. Note that we use the terms control word and control language in the sense of [15, 16, 21, 20] which differs from their original use [19].

In this paper, we extend the study of derivation languages to a particular type of splicing systems, namely to $EGenSS$'s defined in [11]. In $EGenSS$, iterated splicing is done in non-uniform way. More specifically, at any step splicing is done between a string generated in the previous step and axioms. Splicing systems were introduced by Head [7] as a theoretical model to study the recombinant behaviour of DNA molecules. Splicing operation between two strings is defined to be a cut and paste operation where the both strings are cut at particular sites and the first component of the first string is pasted with the second component of the second string, and the second component of the first string is pasted with the first component of the second string to obtain two new strings. It is well known that if in a splicing system, the set of axioms and the set of rules are finite, the system cannot generate beyond regular languages [4, 13]. Different versions of such finite splicing systems are capable of generating recursively enumerable languages [13]. One such version is the concept of extended H system. Extended H systems can be thought of as the set of all DNA strings satisfying a particular property. However, it is not clear when this desired set of strings is obtained. In order to understand this, a labelling of rules is done. The sequence of labels of the applied rules that leads to a terminal derivation is included in the language of the system. In this paper we consider the derivation languages of a variant of splicing system defined in [11].

The paper is organized as follows: Section 2 presents some basic notations. Section 3 defines the Szilard language of splicing systems and shows that there exist regular and context-free languages that are Szilard languages. It is well known that $\{aa\}$ and $\{a^{4n} \mid n \geq 1\}$ cannot be a Szilard language of a Chomsky grammar ([12]). However, we show that $\{aa\}$ can be the Szilard language of $EGenSS$ with finite set of axioms and rules where splicing is done in non-uniform way. The language $\{a^{4n} \mid n \geq 1\}$ cannot be Szilard language of this type of system with finite set of axioms and rules but it can be a Szilard language if the system contains regular set of axioms and finite set of rules. Also we will show that every non-empty context-free language is a morphic image of the Szilard language of an $EGenSS$ with finite set of axioms and rules. In Section 4, we define the control language of a splicing system. We show that both the families of regular and context-free languages are proper subsets of the family of control languages generated by the $EGenSS$'s with finite set of axioms and rules. Also we show that any recursively enumerable language is a control language of this type of splicing system with finite set of axioms and regular set of rules when the rules can also be labeled with λ. We end the paper with a few concluding remarks.

2 Preliminaries

For basic notations and results of formal language theory we refer the reader to [13, 17, 19]. Let V be an alphabet and let V^* denote the set of all strings over V. The empty string is denoted by λ. If F is a family of languages, then $F \setminus \{\lambda\}$ denotes the λ-free family of languages. By FIN, REG, CF, CS, RE we denote the families of finite, regular, context-free, context-sensitive and recursively enumerable languages, respectively.

A word u is a prefix (resp. suffix) of a word v if v is of the form $v = uw$, $w \in V^*$ (resp. $v = wu$). The set of all prefixes (resp. suffixes) of v is denoted as $pref(v)$ (resp. $suff(v)$). The length of a string w is denoted by $|w|$.

A morphism is a mapping from $h : \Sigma^* \to \Delta^*$ such that $h(xy) = h(x)h(y)$ where $x, y \in \Sigma^*$. A morphism $h : \Sigma^* \to \Delta^*$ is called non-erasing, if $h(x) \neq \lambda$ for all $x \in \Sigma$.

A splicing rule over V is a string of the form $r = u_1\#u_2\$u_3\#u_4$, where $u_i \in V^*$, $1 \leq i \leq 4$ and $\#, \$ \notin V$. The maximum of $|u_i|, 1 \leq i \leq 4$, is the radius of the splicing rule r.

An extended generating H system is a 4-tuple $H = (V, T, A, R)$, where V is the alphabet, $T \subseteq V$ is the terminal alphabet, $A \subseteq V^*$ is the set of axioms, and $R \subseteq V^*\#V^*\$V^*\#V^*; \#, \$ \notin V$ is the set of splicing rules. For a splicing rule $r = u_1\#u_2\$u_3\#u_4$ and an ordered pair of words $x, y \in V^*$, denote, $\sigma_r(x, y) = \{u = x_1u_1u_4y_2, v = y_1u_3u_2x_2$ where $x = x_1u_1u_2x_2,\ y = y_1u_3u_4y_2$, for some $x_1, x_2, y_1, y_2 \in V^*\}$. We also write $(x, y) \vdash_r (u, v)$, where u and v are referred to as the first and the second components obtained when r is applied to x and y. Let R be a set of splicing rules and L a language, then $\sigma_R(L)$ is defined as

$$\sigma_R(L) = \bigcup_{r \in R} \bigcup_{w_1, w_2 \in L} \sigma_r(w_1, w_2).$$

If L_1, L_2 are any two languages, then $\sigma_R(L_1, L_2)$ is denoted as

$$\sigma_R(L_1, L_2) = \bigcup_{x_1 \in L_1} \bigcup_{x_2 \in L_2} \sigma_R(x_1, x_2),$$

where

$$\sigma_R(x_1, x_2) = \bigcup_{r \in R} \sigma_r(x_1, x_2).$$

A non-uniform variant for extended generating splicing system is defined in [11]. The system is an extended generating H system, $H = (V, T, A, R)$ with the additional requirement that splicing at any step occurs between a generated word in the previous step and an axiom:

$$\tau_R^0(A) = A,\ \tau_R^{i+1}(A) = \sigma_R(\tau_R^i(A), A), i \geq 0\ ,\ \tau_R^*(A) = \textstyle\bigcup_{i \geq 0} \tau_R^i(A).$$

The system is denoted as $EGenSS\ H$. The language generated by an $EGenSS\ H$ is defined as $L_n(H) = T^* \cap \tau_R^*(A)$. The family of languages generated by $EGenSS$'s in non-uniform way is denoted by $\mathscr{L}_n(EGenSS)$.

The class of languages generated by non-uniform extended generating splicing systems with finite set of axioms and finite set of rules equals REG [11].

3 Szilard language associated with splicing systems

In this section we extend the concept of Szilard languages to splicing systems. We define Szilard languages of $EGenSS$'s and compare them with the family of languages in the Chomsky hierarchy. We also show that the language $\{aa\}$ which is not the Szilard language of a Chomsky grammar [12] is indeed the Szilard language of an $EGenSS$.

A labeled extended generating H system is a construct of the form $\gamma = (V_1, T_1, A_1, R_1, Lab)$, where $H = (V_1, T_1, A_1, R_1)$ is an extended generating splicing system as defined in Section 2, and Lab, $Lab \cap V_1 = \emptyset$ is a set of labels that are used to uniquely name the rules. Since the splicing in the system works in the non-uniform manner, we call this type of splicing systems non-uniform labeled extended generating splicing systems. A derivation in the splicing system is terminal if it obeys one of the following two patterns:
(1) $(x_0, y_0) \vdash^{a_1} (x_1, y_1^0), (x_1, y_1) \vdash^{a_2} (x_2, y_2^0), (x_2, y_2) \vdash^{a_3} (x_3, y_3^0), \cdots$
$(x_{n-1}, y_{n-1}) \vdash^{a_n} (x_n, y_n^0)$, or
(2) $(y_0, x_0) \vdash^{a_1} (x_1, y_1^0), (y_1, x_1) \vdash^{a_2} (x_2, y_2^0), (y_2, x_2) \vdash^{a_3} (x_3, y_3^0), \cdots$
$(y_{n-1}, x_{n-1}) \vdash^{a_n} (x_n, y_n^0)$,
where $x_i \in V_1^*$, $y_i \in A_1$, for $0 \leq i \leq n-1$, $x_n \in T_1^*$, $y_i^0 \in V_1^*$, and $a_i \in Lab$, for $1 \leq i \leq n$.

The set of all such label sequences $a_1 a_2 \cdots a_n$ of the applied rules that leads to a terminal derivation constitute $SZ(\gamma)$, the Szilard language of the non-uniform labeled extended generating H system γ. The family of Szilard languages generated by the non-uniform labeled extended generating splicing systems is denoted by $\mathscr{S}\mathscr{Z}_{LEGenSS_n}(FL_1, FL_2)$, with axioms from the family FL_1 and rules from the family FL_2 .

In the following we show that there exist an infinite regular and a non-regular context-free language which is the Szilard language of a finite labeled $EGenSS$.

Theorem 1. $REG \cap \mathscr{S}\mathscr{Z}_{LEGenSS_n}(FIN, FIN) \neq \emptyset$.

Proof. We construct a labeled H system such that $SZ(\gamma) = \{a^n \mid n \geq 1\}$. Let $\gamma = (V_1, T_1, A_1, R_1, Lab)$ be an labeled $EGenSS$ where $V_1 = \{X, S_1, Y, Z\}, T_1 = \{X, Y\}, A_1 = \{XS_1Y, ZS_1Y, ZY\}$, $R_1 = \{a : \#S_1Y\$Z\#\}$ and $Lab = \{a\}$. If the strings XS_1Y and ZS_1Y are spliced, then

$$(X \mid S_1Y,\ Z \mid S_1Y) \vdash^a (XS_1Y,\ ZS_1Y).$$

The rule can be applied iteratively to XS_1Y and ZS_1Y to obtain XS_1Y and ZS_1Y. A terminal derivation is obtained if XS_1Y is spliced with ZY:

$$(X \mid S_1Y,\ Z \mid Y) \vdash^a (XY,\ ZS_1Y).$$

Any other possibility does not lead to a terminal derivation and, hence, $SZ(\gamma) = \{a^n \mid n \geq 1\}$.

$\square$

It was shown in [12] that the language $\{aa\}$ cannot be a Szilard language of any type-0 grammar. We show that there exists a splicing system γ such that $SZ(\gamma) = \{aa\}$.

Theorem 2. *The language $\{aa\}$ is a Szilard language of a finite labeled EGenSS.*

Proof. We construct a labeled splicing system $\gamma = (V_1, T_1, A_1, R_1, Lab)$ such that $SZ(\gamma) = \{aa\}$. We define, $V_1 = \{X_1^1, Y_0^1, u_1^1, u_3^1, \alpha, \beta, \beta_1, X_0^2, Y_0^2\}$, $T_1 = \{X_1^1, u_1^1, \beta, \beta_1, Y_0^2\}$, $A_1 = \{X_1^1u_1^1\alpha u_1^1\alpha u_1^1\beta X_0^2, Y_0^1u_3^1\beta\beta_1Y_0^2\}$, $Lab = \{a\}$ and
$R_1 = \{a : u_1^1\#\alpha u_1^1\beta\$u_3^1\#\beta\beta_1\}$. There is a terminal derivation

$$(X_1^1u_1^1\alpha u_1^1 \mid \alpha u_1^1\beta X_0^2\ ,\ Y_0^1u_3^1 \mid \beta\beta_1Y_0^2) \vdash^a (X_1^1u_1^1\alpha u_1^1\beta\beta_1Y_0^2\ ,\ Y_0^1u_3^1\alpha u_1^1\beta X_0^2)$$
$$(X_1^1u_1^1 \mid \alpha u_1^1\beta\beta_1Y_0^2\ ,\ Y_0^1u_3^1 \mid \beta\beta_1Y_0^2) \vdash^a (X_1^1u_1^1\beta\beta_1Y_0^2\ ,\ Y_0^1u_3^1\alpha u_1^1\beta\beta_1Y_0^2).$$

It is easy to verify that no other derivation is possible and, hence, $SZ(\gamma) = \{aa\}$.

$\square$

In the following we show that there exists a regular language that cannot be a Szilard language of any finite labeled *EGenSS*.

Theorem 3. $REG \setminus \mathscr{SZ}_{LEGenSS_n}(FIN, FIN) \neq \emptyset$.

Proof. Let $L = \{a^{4n} \mid n \geq 1\}$. To derive a contradiction, suppose a finite labeled *EGenSS*, $\gamma = (V_1, T_1, A_1, R_1, Lab)$ such that $SZ(\gamma) = \{a^{4n} \mid n \geq 1\}$. The system contains only one rule, $a : u_1^1\#u_2^1\$u_3^1\#u_4^1$, where $u_1^1, u_2^1, u_3^1, u_4^1 \in V_1^*$. Hence, there exists a terminal derivation with label sequence a^4 as follows:
$(x_0, y_0) = (x_0^1u_1^1 \mid u_2^1x_0^2\ ,\ y_0^1u_3^1 \mid u_4^1y_0^2) \vdash^a (x_0^1u_1^1u_4^1y_0^2\ ,\ y_0^1u_3^1u_2^1x_0^2)$
$(x_1, y_1) = (x_1^1u_1^1 \mid u_2^1x_1^2\ ,\ y_1^1u_3^1 \mid u_4^1y_1^2) \vdash^a (x_1^1u_1^1u_4^1y_1^2\ ,\ y_1^1u_3^1u_2^1x_1^2)$
$(x_2, y_2) = (x_2^1u_1^1 \mid u_2^1x_2^2\ ,\ y_2^1u_3^1 \mid u_4^1y_2^2) \vdash^a (x_2^1u_1^1u_4^1y_2^2\ ,\ y_2^1u_3^1u_2^1x_2^2)$
$(x_3, y_3) = (x_3^1u_1^1 \mid u_2^1x_3^2\ ,\ y_3^1u_3^1 \mid u_4^1y_3^2) \vdash^a (x_3^1u_1^1u_4^1y_3^2\ ,\ y_3^1u_3^1u_2^1x_3^2)$,
where $x_3^1u_1^1u_4^1y_3^2 \in T_1^*, x_i^j, y_i^j \in V_1^*$, $0 \leq i \leq 3$, $1 \leq j \leq 2$ such that $x_0^1u_1^1u_4^1y_0^2 = x_1^1u_1^1u_2^1x_1^2$, $x_1^1u_1^1u_4^1y_1^2 = x_2^1u_1^1u_2^1x_2^2$ and $x_2^1u_1^1u_4^1y_2^2 = x_3^1u_1^1u_2^1x_3^2$. Then we have the following cases:

1. $x_2^1 = x_3^1u_1^1u_2^1\alpha_1$, $\alpha_1 \in pref(x_3^2)$
2. $x_2^1 = x_3^1u_1^1\alpha_1$, $\alpha_1 \in pref(u_2^1)$
3. $x_2^1 = x_3^1\alpha_1$, $\alpha_1 \in pref(u_1^1)$
4. $x_2^1 = \alpha_1$, $\alpha_1 \in pref(x_3^1)$.

If $x_2^1 = x_3^1u_1^1u_2^1\alpha_1$, then $x_2 = x_3^1u_1^1u_2^1\alpha_1u_1^1u_2^1x_2^2$ and $(x_2, y_3) \vdash^a (x_4, y_4)$ where $x_4 \in T_1^*$ generating a^3. The cases $x_2^1 = x_3^1\alpha_1$, $\alpha_1 \in pref(u_1^1)$ and $x_2^1 = \alpha_1$, $\alpha_1 \in pref(x_3^1)$ lead to the same contradiction. If $x_2^1 = x_3^1u_1^1\alpha_1$ where $\alpha_1 \in pref(u_2^1)$, then $x_2 = x_2^1u_1^1\alpha_1u_1^1u_2^1x_2^2$. Note that $\alpha_1 \notin T_1^*$, otherwise, the rule a can be applied to x_2 and y_3 which leads to a terminal derivation generating a^3.

Now from $x_0^1u_1^1u_4^1y_2^2 = x_1^1u_1^1u_2^1x_1^2$, we have the following possibilities:
$x_0^1 \in pref(x_1^1), x_0^1 \in x_1^1\ pref(u_1^1), x_0^1 \in x_1^1u_1^1\ pref(u_2^1)$ and $x_0^1 \in x_1^1u_1^1u_2^1\ pref(x_1^2)$. Similarly, from $x_1^1u_1^1u_4^1y_1^2 = x_2^1u_1^1u_2^1x_2^2$, we obtain $x_1^1 \in pref(x_2^1)$, $x_1^1 \in x_2^1\ pref(u_1^1)$, $x_1^1 \in x_2^1u_1^1\ pref(u_2^1)$ and $x_1^1 \in x_2^1u_1^1u_2^1\ pref(x_2^2)$. If $x_0^1 = x_1^1$ or $x_1^1 = x_2^1$, then the system will generate strings $a^i \notin L$.

All other cases mentioned will either increase in the length of u_1^1 or u_2^1 or both or increase in the number of independent occurrences of prefixes of u_1^1 and prefixes of u_2^1. Since L is infinite and x_0 is a finite string, this is not possible and, hence, L is not a Szilard language of any finite splicing system.

□

In the following we construct a labeled *EGenSS* with regular set of axioms such that $\{a^{4n} \mid n \geq 1\} \in \mathscr{SZ}_{LEGenSS_n}(REG, FIN)$.

Example 1. We construct a labeled *EGenSS*, $\gamma = (V_1, T_1, A_1, R_1, Lab)$ such that $SZ(\gamma) = \{a^{4n} \mid n \geq 1\}$. Let $V_1 = \{X, u_1^1, \beta, Y, Z\}, T_1 = \{X, \beta, Y\}, A_1 = \{Xu_1^{4n}\beta Y \mid n \geq 1\} \cup \{Z\beta Y\}, R_1 = \{a : \#u_1\beta Y\$Z\#\beta Y\}$, and $Lab = \{a\}$. It is clear that any derivation of the above system reaches a terminal derivation only after applying the rule a four times. Since the set A_1 is regular, $\{a^{4n} \mid n \geq 1\} \in \mathscr{SZ}_{LEGenSS_n}(REG, FIN)$.

Next, we show that there exists a context-free language that is the Szilard language of a finite labeled *EGenSS*. From [11] we know that this type of splicing systems cannot generate non-regular languages.

Theorem 4. *$CF \cap \mathscr{SZ}_{LEGenSS_n}(FIN, FIN) \neq \emptyset$.*

Proof. Let $\gamma = (V_1, T_1, A_1, R_1, Lab)$ be a labeled *EGenSS* where $V_1 = \{X_1, Y, Y_1, a, Z, A_1, A_2\}$, $T_1 = \{X_1, Y_1\}$, $A_1 = \{X_1Y, ZaA_1Y, ZA_2Y_1\}$, and $Lab = \{1, 2, 3, 4, 5\}$. The system contains the rules $R_1 = \{\mathbf{1} : X_1\#Y\$Z\#aA_1Y$, $\mathbf{2} : a\#A_1Y\$Z\#aA_1Y$, $\mathbf{3} : aa\#A_1Y\$Z\#A_2Y_1$, $\mathbf{4} : a\#aA_2Y_1\$Z\#A_2Y_1$, $\mathbf{5} : X_1\#aA_2Y_1\$ZA_2\#Y_1\}$.

Initially, the rule 1 can be applied to the strings X_1Y and ZaA_1Y. After applying rule 1, the string X_1aA_1Y is produced. Then rule 2 can be applied iteratively $(n-1)$ times, to generate a string of the form $X_1a^nA_1Y$. Then rule 3 can be applied to obtain the string $X_1a^nA_2$. After application of rule 3, only rules 4 or 5 are applicable. If rule 4 is applied $(n-1)$ times, XaA_2Y is produced. Finally, rule 5 is applied to obtain the terminal string X_1Y_1. If a derivation does not start with 1, it does not lead to a terminal derivation. Thus, $SZ(\gamma) = \{12^n34^n5 \mid n \geq 1\}$. □

It was shown in [9] that context-free languages can be represented as a morphic images of Szilard languages associated with the left most derivations of context-free grammars. However, the class of all languages obtained by taking the morphic image of Szilard languages of the context-free grammars in general are incomparable with context-free languages [9]. In the following we show a result similar to that in [9], that each context-free languages can be expressed as a morphic image of the Szilard language of a finite labeled *EGenSS*.

Theorem 5. *Every non-empty context free language is a morphic image of the Szilard language of a finite labeled EGenSS.*

Proof. Let L be a non-empty context-free language and let $G = (N, T, P, S)$ be a grammar in Greibach normal form such that $L = L(G)$. The rules in P are of the form, $D_i \to a\alpha$ and $D_i \to a$, where $\alpha \in N^+, D_i \in N, a \in T$ and $N = \{D_1, D_2, \ldots, D_n\}$. We show that there exists a finite splicing system γ such that $L = h(SZ(\gamma))$ where h is a non-erasing morphism from Lab^* to T^*.

We construct a labeled splicing system $\gamma = (V_1, T_1, A_1, R_1, Lab)$ such that $L = h(SZ(\gamma))$ where

- $V_1 = \{Y\} \cup N \cup \Delta_1$, for $\Delta_1 = \{Y_a \mid D_i \to a\alpha, \alpha \in N^+, D_i \in N, a \in T\} \cup \{Y_a \mid D_i \to a \in P, D_i \in N, a \in T\}$;
- $T_1 = \{Y\}$;
- $A_1 = \{YSY\} \cup \Delta_2 \cup \Delta_3$, where
  $\Delta_2 = \{Y\alpha Y_a \mid D_i \to a\alpha, D_i \in N, a \in T, \alpha \in N^+\}$ and
  $\Delta_3 = \{YY_a \mid D_i \to a, D_i \in N, a \in T\}$.
- $R_1 = \{(a^i_{\alpha_k} : Y\alpha_k\#Y_a\$YD_i\#) \mid D_i \to a\alpha_k \in P\} \cup \{(a^i : Y\#Y_a\$YD_i\#) \mid D_i \to a \in P\}$ that is, for every rule of the form $D_i \to a\alpha_k$ in G, where $a \in T, \alpha_k \in N^+$, k a positive integer, a splicing rule $(a^i_{\alpha_k} : Y\alpha_k\#Y_a\$YD_i\#)$ is constructed. Similarly, if there exists a rule $D_i \to a$, then a splicing rule $(a^i : Y\#Y_a\$YD_i\#)$ is constructed.
- $Lab = \{a^i_{\alpha_k} \mid D_i \to a\alpha_k \in P\} \cup \{a^i \mid D_i \to a \in P\}$.

Finally, we define the morphism $h : Lab^* \to T^*$ such that $h(a^i_{\alpha_k}) = h(a^i) = a$ where $a_{\alpha^i_k}, a^i \in Lab$ and $a \in T$.

We first prove that $L(G) \subseteq h(SZ(\gamma))$. Any computation in G starts from S and after sequential application of the rules in P, a string over T is generated. The splicing rules simulating the rules $D_i \to a\alpha_k$, $D_i \to a, D_j \to a\alpha_k$, and $D_j \to a$ in P are labeled with $a^i_{\alpha_k}, a^i, a^j_{\alpha_k}$, and a^j, respectively. Suppose a terminal string, say, w is generated in G. If the corresponding labeled rules are also applied in γ, a terminal derivation can be obtained. If the labels of the applied splicing rules are concatenated, a string over Lab, say, w_1 is generated. But if the morphism h is applied w, each occurrence of $a^i_{\alpha_k}, a^i, a^j_{\alpha_k}$, and a^j is replaced by a. Hence, if $w \in L(G)$, we have $w = h(w_1) \in h(SZ(\gamma))$ where $w_1 \in SZ(\gamma)$.

Next we prove the inclusion $h(SZ(\gamma) \subseteq L(G)$. Let $w = h(w_1)$ where $w_1 \in SZ(\gamma)$. Let $w_1 = a_1a_2 \ldots a_n \in SZ(\gamma)$, i.e., there exists a terminal derivation in γ with which w_1 is generated. In G, computations starts from S. If the rules in

G are applied in the same sequence as the (simulated) labeled rules are applied in γ, a terminal string is generated. So, a terminal string in G and a terminal derivation in γ is obtained at the same time. Again, $h(a^i_{\alpha_k}) = h(a^i) = a$, and hence, $h(w_1) = w \in L(G)$. So, we can conclude $h(SZ(\gamma)) \subseteq L(G)$. $\square$

4 Control languages of splicing systems

In the previous section we discussed the Szilard languages associated with splicing systems. In this section we define control languages associated with splicing systems and compare the family of control languages generated by the labeled *EGenSS* with the family of languages in the Chomsky hierarchy. Control languages have already been discussed for several variants of, for example, tissue P systems, spiking neural P systems, and P systems with isotonic array grammars ([15, 16, 21, 20]) to name a few. We extend the concept of control languages to splicing systems and show that all non-empty regular and context-free languages are indeed control languages of finite labeled *EGenSS*.

We conisder a labeled extended generating H system $\gamma = (V_1, T_1, A_1, R_1, Lab)$, working in non-uniform manner, where V_1, T_1, A_1, R_1, and Lab are as defined in Section 3 except that multiple rules in R_1 can be assigned with the same label. Also a single rule cannot be mapped with different labels. The rules can also be labeled with the empty string λ. The concatenation of the labels of the applied splicing rules in any terminal derivation will form a string over Lab. It is called a control word of the labeled *EGenSS*. The set of all control words constitute the control language of the labeled *EGenSS* γ. It is denoted by $CTL(\gamma)$.

The family of control languages generated by any labeled extended generating splicing system $\gamma = (V_1, T_1, A_1, R_1, Lab)$ with $card(A) \leq n$ and $rad(R) \leq m$, where $n, m \geq 1$, is denoted by $\mathscr{RL}_{CTL}([n],[m])$. When no restriction on the number n of axioms or on the maximal radius m are considered but n and m are still finite, they are simply replaced with FIN. If empty labels are allowed then the family is denoted by $\mathscr{RL}_{CTL_\lambda}([n],[m])$. If the system contains axioms from F_1 and rules from F_2, for some families of languages F_1 and F_2, then the family of control languages generated by the systems is denoted by $\mathscr{RL}_{CTL}(F_1, F_2)$. When the system contains λ-labeled rules, we denote it by $\mathscr{RL}_{CTL_\lambda}(F_1, F_2)$.

$\mathscr{L}_n(EGenSS)$ with finite set of axioms and finite set of rules [11] with no restriction on the radius of the splicing rules equals the class of regular languages. In the next theorem, we show that the class of non-empty regular languages are contained in $\mathscr{RL}_{CTL}(FIN,[1])$.

Theorem 6. *$(REG \setminus \{\lambda\}) \subseteq \mathscr{RL}_{CTL}(FIN,[1])$.*

Proof. Let L be a λ-free regular language. Then there exists a regular grammar $G = (N, T, P, S)$ such that $L = L(G)$. Suppose the non-terminals N of G are $D_i, 1 \leq i \leq n$, where $D_1 = S$ is the start symbol. We now construct a finite labeled $EGenSS$ γ such that $L = L(G) = CTL(\gamma)$. The rules in P are of the form $D_i \to aD_i$, $D_i \to aD_j (i \neq j)$, and $D_i \to a$, $D_i, D_j \in N$, and $a \in T$.

Let $\gamma = (V_1, T_1, A_1, R_1, Lab)$ be a labeled $EGenSS$, where

- $V_1 = \{X, Y, Y_1, D_1, D_2, \ldots, D_n\} \cup \Delta_1$, for $\Delta_1 = \{Y_a \mid D_i \to aD_i\} \cup \{Y_a \mid D_i \to aD_j (i \neq j)\} \cup \{Y_a \mid D_i \to a \in P, a \in T\}$;
- $T_1 = \{Y\} \cup \Delta_1$;
- $A_1 = \{XD_1Y\} \cup \Delta_2 \cup \Delta_3 \cup \Delta_4$, where
    1. $\Delta_2 = \{YD_jY_a \mid D_i \to aD_j \in P,\ a \in T\}$,
    2. $\Delta_3 = \{YD_iY_a \mid D_i \to aD_i \in P,\ a \in T\}$,
    3. $\Delta_4 = \{Y_aY_1 \mid D_i \to a \in P,\ a \in T\}$;
    (The set $(\Delta_2 \cap \Delta_3)$ may or may not be disjoint.)
- The rules in R_1 are of the form
  $(a : D_i\#Y_a\$D_i\#Y)$, for $D_i \to aD_i, a \in T$,
  $(a : D_j\#Y_a\$D_i\#Y)$, for $D_i \to aD_j, i \neq j, a \in T$,
  $(a : Y_a\#Y_1\$D_i\#Y)$, for $D_i \to a, a \in T$, where $1 \leq i, j \leq n$;
- $Lab = T$.

Every rule in G is simulated by a corresponding splicing rule with the required label a that corresponds to the grammar rule under consideration. Thus, every $w \in L(G)$ can be simulated by a terminal derivation in γ and vice versa. The sequence of splicing rules reach a terminal derivation only when the rule $(a : Y_a\#Y_1\$D_i\#Y)$ corresponding to the rule $D_i \to a, a \in T$, is applied. Thus, $L(G) = CTL(\gamma)$.

□

In the next theorem we show that, every non-empty context-free language can be a control language of a finite labeled $EGenSS$.

Theorem 7. $(CF \setminus \{\lambda\}) \subseteq \mathscr{RL}_{CTL}(FIN, FIN)$.

Proof. Let L be any non-empty context-free language such that $\lambda \notin L$. Then let $G = (N, T, P, S)$ be a context-free grammar in Greibach normal form such that $L = L(G)$. We construct a finite labeled $EGenSS$, $\gamma = (V_1, T_1, A_1, R_1, Lab)$ such that $L = L(G) = CTL(\gamma)$. Let $\gamma = (V_1, T_1, A_1, R_1, Lab)$ be a labeled splicing system where:

- $V_1 = \{X, Y, Y_1\} \cup N \cup \Delta_1$, for $\Delta_1 = \{Y_a \mid D \to a\alpha, \alpha \in N^+, a \in T\} \cup \{Y_a \mid D \to a \in P, a \in T, , D \in N\}$;
- $T_1 = \{Y\} \cup \Delta_1$;
- $A_1 = \{XSY\} \cup \Delta_2 \cup \Delta_3$, where
    1. $\Delta_2 = \{Y\alpha Y_a \mid D \to a\alpha \in P, a \in T, \alpha \in N^+, D \in N\}$,

2. $\Delta_3 = \{YY_a, Y_aY_1 \mid D \to a \in P, a \in T, D \in N\}$;

- R_1 contains the following rules :
  For $D \to a\alpha \in P$, we have, $\{(a : Y\alpha\#Y_a\$XD\#Y), (a : Y\alpha\#Y_a\$YD\#Y)\} \cup$
  $\{(a : Y\alpha\#Y_a\$YD\#\beta_1\beta_2 \dots \beta_iY) \mid \beta_i \in N, 1 \leq i \leq (n-1)\} \cup$
  $\{(a : Y\alpha\#Y_a\$YD\#\beta_1\beta_2 \dots \beta_n) \mid \beta_i \in N\}$
  For $D \to a \in P$, we have, $\{(a : Y\#Y_a\$YD\#\beta_1\beta_2 \dots \beta_iY) \mid \beta_i \in N, 1 \leq i \leq (n-1)\} \cup$ $\{(a : Y_a\#Y_1\$XD\#Y)\}$ $\cup \{(a : Y\#Y_a\$YD\#\beta_1\beta_2 \dots \beta_n) \mid \beta_i \in N\} \cup \{(a : Y_a\#Y_1\$YD\#Y)\}$
  where $n = Max\{|\alpha| \mid D \to a\alpha \in P\}$;
- $Lab = T$.

Corresponding to each rule of the form $D \to a\alpha \in P$ there exist rules in γ labeled with a, $(a : Y\alpha\#Y_a\$XD\#Y)$, $(a : Y\alpha\#Y_a\$YD\#\beta_1Y)$, $(a : Y\alpha\#Y_a\$YD\#\beta_1\beta_2Y)$, $(a : Y\alpha\#Y_a\$YD\#\beta_1\beta_2\beta_3Y)$, ..., and $(a : Y\alpha\#Y_a\$YD\#\beta_1\beta_2 \dots \beta_n)$. These rules can be applied to the pairs of strings XDY, $Y\alpha Y_a$ and $YDQY$, $Y\alpha Y_a$, where $Q \in N^*$, respectively. At first, $Y\alpha Y_a$ and XSY are spliced and $Y\alpha Y$ and XSY_a are produced. No rule is applicable to XSY_a, but $Y\alpha Y$ can be spliced further with the rules in the system. If XSY and Y_aY_1 are spliced together, it will produce XSY_1 and Y_aY. Strings of the form $YDQY$, where $Q \in N^*$, can be spliced with the strings $Y\alpha Y_a$ and YY_a to obtain YDY_a and $Y\alpha QY$ or YQY. After the application of the rule $(a : Y_a\#Y_1\$YD\#Y)$ to YDY and Y_aY_1, the strings YDY_1 and Y_aY are produced. The string Y_aY is a terminal string and the strings of labels of the rules applied are in the control language. The above construction of γ simulates the rules of P in R. The splicing rules in γ are applied in the same sequence as the rules are applied in the derivation $S \Rightarrow^* x$, for $x \in L(G)$. Thus $x \in L(G)$ iff there exist a terminal derivation in γ generating x. Whenever the rules $D \to a\alpha$, and $D \to a$ are applied to a non-terminal in G, the corresponding splicing rule labeled with a is applied in the system γ and vice versa. Thus, $L(G) = CTL(\gamma)$. □

In the following we show that there exists a context-sensitive language that cannot be the control language of any finite labeled *EGenSS*.

Theorem 8. *$CS \setminus \mathscr{RL}_{CTL\lambda}(FIN, FIN) \neq \emptyset$.*

Proof. Let $L = \{a^{2^n} \mid n \geq 0\}$ be a context-sensitive language. Assume that there exists a finite labeled *EGenSS*, $\gamma = (V_1, T_1, A_1, R_1, Lab)$ such that $CTL(\gamma) = \{a^{2^n} \mid n \geq 0\}$.
Since $a \in CTL(\gamma)$, there exist an 'a' labeled rule (say r_1) and $x_0, y_0 \in A_1$ such that $(x_0, y_0) \vdash^a_{r_1} (x_t, y'), x_t \in T_1^*$. Since $x_t \in T_1^*$, x_t cannot be spliced further and hence it is not possible to generate a^2, from the strings x_0, and y_0 and just by using the rule r_1. Therefore there exists an 'a' labeled rule r_2 such that $r_1 \neq r_2$ and $(x_0, y_0) \vdash^a_{r_2} (x_1, y')$, $(x_1, y_1) \vdash^a_{r_1} (x_t, y'')$, $x_t \in T_1^*$. Thus to generate a^{2^n}, for some n starting with x_0, y_0, there must exist 'a' labeled rules $r_1, r_2, \cdots r_k$ such that $k \leq 2^n$. Since the number of rules in the system are finite, some of these rules are repeated recursively which will end up in generating strings of the type a^i such that $i \neq 2^n$. Hence $CTL(\gamma) \neq L$. □

The following theorem shows that the family of control languages generated by the labeled *EGenSS* with rules from a regular set where some of the rules are labeled with λ is equal to the family of recursively enumerable languages.

Theorem 9. $\mathscr{RL}_{CTL_\lambda}(FIN, REG) = RE$.

Proof. The inclusion $\mathscr{RL}_{CTL_\lambda}(FIN, REG) \subseteq RE$ follows from the Church-Turing Thesis. We have to prove only the inclusion $RE \subseteq \mathscr{RL}_{CTL_\lambda}(FIN, REG)$. Let $G = (N, T, P, S)$ be a type-0-grammar in Kuroda normal form. We construct a labeled $EGenSS$, $\gamma = (V_1, T_1, A_1, R_1, Lab)$ such that $CTL_\lambda(\gamma) = L(G)$. Let $\gamma = (V_1, T_1, A_1, R_1, Lab)$ is a labeled $EGenSS$ with $U = N \cup T \cup \{E\}$, where

- $V_1 = N \cup \{E, X, X^{'}, Y, Z\} \cup \{Y_\alpha \mid \alpha \in U\}$;
- $T_1 = \{X, Y, E\}$;
- $A_1 = \{XESY, XZ, ZY\} \cup \{ZY_\alpha \mid \alpha \in U\} \cup \{X^{'}\alpha Z \mid \alpha \in N \cup E\} \cup \{ZBCY \mid A \to BC \in P\} \cup \{ZCDY \mid AB \to CD \in P\} \cup \{ZY_aY \mid A \to a \in P\}$;
- R_1 contains the following rules:
  1. $(\lambda : Xw\#AY\$Z\#BCY)$, for $A \to BC \in P, w \in (N \cup \{E\})^*$
  2. $(\lambda : Xw\#ABY\$Z\#CDY)$, for $AB \to CD \in P, w \in (N \cup \{E\})^*$
  3. $(a : XwE\#AY\$ZY_a\#Y)$, for $A \to a \in P, w \in N^*$
  4. $(\lambda : XwE\#AY\$Z\#Y)$, for $A \to \lambda \in P, w \in N^*$
  5. $(\lambda : Xw\#\alpha Y\$Z\#Y_\alpha)$, for $\alpha \in N \cup E, w \in (N \cup \{E\})^*$
  6. $(\lambda : X^{'}\alpha\#Z\$X\#wY_\alpha)$, for $\alpha \in N \cup E, w \in (N \cup \{E\})^*$
  7. $(\lambda : X^{'}w\#Y_\alpha\$Z\#Y)$, for $\alpha \in N \cup E, w \in (N \cup \{E\})^*$
  8. $(\lambda : X\#Z\$X^{'}\#wY)$, for $w \in (N \cup \{E\})^*$;
- $Lab = T \cup \{\lambda\}$.

The above system is constructed in the same manner as in any standard proof of extended H system with finite set of axioms and regular set of rules that can generate RE languages. The splicing rules are labeled with the terminal symbol a that simulates the rule $A \to a$ in G and the rest of the rules are labeled with λ. The grammar G is in Kuroda normal form and any element $w \in L(G)$ can be generated by the application of the recursive rules $A \to BC$ and $AB \to CD$ rules in P in any manner and then by application of the terminating rules $A \to \lambda$ and $A \to a$ in the leftmost manner. The splicing rules (1) and (2) simulate the non-terminal recursive rules and the splicing rules in (3) and (4) simulate the terminating rules $A \to a$ and $A \to \lambda$, respectively. The splicing rules (3) and (4) are applicable only to the non-terminal symbol present between E and the right hand marker Y and by doing this the left-most derivation is simulated, since the terminating rules are applied in leftmost manner. The rules in γ from (1) to (4) simulate the rules in P. Rules from (5) to (8) are used to rotate the string inside the markers X and Y.

Note that the λ-labeled splicing rules in (1) and (2) can be applied any number of times. Also, the rotation rules from (5) to (8) are λ-labeled and they can also be applied any number of times. But the rules in (3) and (4) are applicable to the non-terminals in between E and Y. The a-rule in (3) eliminates one non-terminal (adjacent to E) from the string inside the markers X and Y in γ. It simulates the application of the rule $A \to a$ to the left most non-terminal in any derivation of G . The λ-labeled rule in (4) also works in the same manner. Thus, $w \in L(G)$ iff there exists a derivation $S \Rightarrow^* w$ and the system γ generates the string $XEY \in T_1^*$ in the first component of a step, i.e., a terminal derivation is obtained. Thus, $w \in L(G)$ iff $w \in CTL_\lambda(\gamma)$. □

5 Conclusion

We have defined the derivation languages of non-uniform variant of generating splicing systems and have compared them with the families of languages in the Chomsky hierarchy. We have shown that infinite regular and non-regular context-free languages can be Szilard languages of finite splicing systems. and that every non-empty context-free language is a morphic image of the Szilard language of a finite splicing system. Also we showed that the family of infinite regular and non-regular context-free languages are properly contained in the family of control languages of finite splicing systems. We also have shown that if the set of axioms are finite and the set of rules are regular and λ labeled rules are allowed, any recursively enumerable language can be generated as a control language of a non-uniform labeled extended generating splicing system. It will also be interesting to explore power of the derivation languages of other variants of splicing system.

References

[1] Ciobanu, G., Păun, G., Stefanescu, G. : Sevilla carpets associated with P systems. in BWMC 2003, Tarragona Univ., TR 26/03 (2003)

[2] Cojocaru, L., Mäkinen, E., Tiplea, F. L. : Classes of Szilard languages in $\mathscr{NC}$. In: 11th International Symposium on Symposium on Symbolic and Numeric Algorithms for Scientific Computing, pp. 299–306 (2009)

[3] Cojocaru, L., Mäkinen, E. : On some derivation mechanisms and the complexity of their Szilard languages. Theoretical Computer Science. 537, 87–96 (2014)

[4] Culik, K. II, Harju, T. : Splicing semigroups of dominoes and DNA. Discrete Appllied Matematics. 31, 261–277 (1991)

[5] Dassow, J., Păun, G. : Regulated rewriting in formal language theory. Springer, Berlin (1989)

[6] Dassow, J., Mitrana, V., Păun, G. : Szilard languages associated to co-operating distributed grammar systems. Stud. Cercet. Mat. 45, 403–413 (1993)

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

[8] Mäkinen, E. : On context-free and Szilard languages. BIT. 24, 164–170 (1984)

[9] Mäkinen, E. : On homomorphic images of Szilard languages. International Journal of Computer Mathematics. 18, 239–245 (1986)

[10] Mihalache, V. : Szilard languages associated to parallel communicating grammar systems. Developments in Language Theory II, At the Crossroads of Mathematics, Computer Science and Biology, Magdeburg, Germany, July 1995, World Scientific, Singapore, 247–256 (1996)

[11] Mitrana, V., Petre, I., Rogojin, V. : Accepting splicing systems. Theoretical Computer Science. 411, 2414–2422 (2010)

[12] Păun, G. : On some families of Szilard languages. BULL. MATH. de la Soc. Sci. Math. de la R. S. de Roumanie Tome 27(75), 259–265 (1983)

[13] Păun, Gh., Rozenberg, G., Salomaa, A. : DNA Computing: New computing paradigms, Springer-Verlag, Berlin (1998)

[14] Penttonen, M. : On derivation language corresponding to context-free grammars. Acta Informatica. 3, 285–291 (1974)

[15] Ramanujan, A., Krithivasan, K. : Control words of transition P systems. BIC-TA 2012, Advances in Intelligent Systems and Computing. 145–155 (2012)

[16] Ramanujan, A., Krithivasan, K. : Control languages associated with spiking neural P systems. Romanian Journal of Information Science and Technology. 15(4), 301–318 (2012)

[17] Rozenberg, G., Salomaa, A. (Eds.) : Handbook of formal languages, vol. I-III, Springer-Verlag, Berlin (1997)

[18] Salomaa, A. : Matrix grammars with a left most restriction. Information Control. 20(2), 143–149 (1972)

[19] Salomaa, A. : Formal languages, Academic Press, New York (1973)

[20] Sureshkumar, W., Rama, R. : Chomsky hierarchy control on isotonic array P systems. International Journal of Pattern Recognition and Artificial Intelligence. 30(2), 10.1142/S021800141650004X (2016)

[21] Zhang, X., Liu, Y., Luo, B., Pan, L. : Computational power of tissue P systems for generating control languages. Information Sciences. 278, 285–297 (2014)

Received 23rd March 2017