Fuzzy Splicing Systems

Fariba Karimi[1], Sherzod Turaev[2], Nor Haniza Sarmin[3], and Wan Heng Fong[4]

[1] Royal Society Wolfson Biocomputation Research Lab,
Biocomputation School of Computer Science,
University of Hertfordshire,
Hatfield, Hertfordshire AL10 9AB, UK
fk.karimi@gmail.com
[2] Department of Computer Science,
Kulliyyah of Information and Communication Technology,
International Islamic University Malaysia,
53100 Kuala Lumpur, Malaysia
sherzod@iium.edu.my
[3] Department of Mathematical Sciences, Faculty of Science,
Universiti Teknologi Malaysia,
81310 UTM Johor Bahru, Johor, Malaysia
nhs@utm.my
[4] Ibnu Sina Institute for Fundamental Science Studies,
Universiti Teknologi Malaysia,
81310 UTM Johor Bahru, Johor, Malaysia
fwh@ibnusina.utm.my

Abstract. In this paper we introduce a new variant of splicing systems, called *fuzzy splicing systems*, and establish some basic properties of language families generated by this type of splicing systems. We study the "fuzzy effect" on splicing operations, and show that the "fuzzification" of splicing systems can increase and decrease the computational power of splicing systems with finite components with respect to fuzzy operations and cut-points chosen for threshold languages.

1 Introduction

Though computers have gained such a dominant position in our life, they have many drawbacks: there are numerous intractable problems, which cannot be solved with their help. *DNA computing* appears as a challenge to develop new types of algorithms and to design new types of computers which differ from classical notions of algorithms and computers in fundamental way. DNA computing models use *Watson-Crick complementary* of DNA molecules that are double stranded structures composed of four nucleotides A (*adenine*), C (*cytosine*), G (*guanine*) and T (*thymine*) always presenting in pairs A–T and C–G. Another feature of DNA molecules is the *massive parallelism* of DNA strands, which allows constructing many copies of DNA strands and carrying out operations on the encoded information simultaneously. The use of these two fundamental features of DNA molecules has already illustrated that DNA based computers can

D. Hwang et al. (Eds.): ICCCI 2014, LNAI 8733, pp. 20–29, 2014.

solve many computationally intractable problems: Hamiltonian Path Problem [1], the Satisfiability Problems [2,3], etc.

A concept of *splicing system*, one of the early theoretical proposals for DNA based computation, was introduced by Head [4] using a *splicing operation* – a formal model for DNA recombination under the influence of restriction enzymes. This process works as follows: two DNA molecules are cut at specific subsequences and the first part of one molecule is connected to the second part of the other molecule, and vice versa. This process can be formalized as an operation on *strings*, described by a so-called *splicing rule*, which are the basis of a computational model called a *splicing system*. A system starts from a given set of strings (*axioms*) and produces a *language* by iterated splicing according to a given set of splicing rules. Because of practical reasons, the case when the components of splicing systems are finite is of special interest. But splicing systems with finite sets of axioms and rules generate only regular languages (see [5]). Consequently, several restrictions in the use of rules have been considered (for instance, see [6]), which increase the computational power up to the Turing equivalent languages.

The treatment of splicing systems as language-generating devices allows using concepts, methods and techniques of formal language theory to study the properties of splicing systems. One can easily adapt many extension and restriction mechanisms associated with grammars and automata for splicing systems. In this paper we focus on the study of "fuzzified" splicing systems, whose grammar and automata counterparts have widely been investigated recent years (for details, see the monograph [7]). The concept of fuzzy splicing systems is introduced as follows: we associate the truth values from the closed interval $[0, 1]$ with each axiom, and calculate the truth value of a string w resulted from strings u and v applying a fuzzy operation over their truth values. We select a subset of the language generated by a fuzzy splicing system according to some cut-points in $[0, 1]$, which is called a *threshold language*. We show that some threshold languages with the selection of appropriate cut-points can generate non-regular languages.

This paper is organized as follows. Section 2 contains some necessary definitions and notations from the theories of formal languages and splicing systems. In Section 3, the concepts of fuzzy splicing systems and threshold languages generated by fuzzy splicing systems are introduced. Section 4 shows the power of fuzzy splicing systems: some fuzzy splicing systems of finite components can generate context-free and context-sensitive languages. Section 5 discusses some open problems and possible topics for future research in this direction.

2 Preliminaries

In this section we recall some prerequisites, by giving basic notions and notations of the theories of formal languages and splicing systems which are used in sequel. The reader is referred to [8,6,9] for further information.

Throughout the paper we use the following general notations. The symbol $\in$ denotes the membership of an element to a set while the negation of set

membership is denoted by $\notin$. The inclusion is denoted by $\subseteq$ and the strict (proper) inclusion is denoted by $\subset$. The empty set is denoted by $\emptyset$. The cardinality of a set X is denoted by $|X|$. The families of recursively enumerable, context-sensitive, context-free, linear, regular and finite languages are denoted by **RE**, **CS**, **CF**, **LIN**, **REG** and **FIN**, respectively. For these language families, the next strict inclusions, named *Chomsky hierarchy* (see [9]), hold:

Theorem 1. $\mathbf{FIN} \subset \mathbf{REG} \subset \mathbf{LIN} \subset \mathbf{CF} \subset \mathbf{CS} \subset \mathbf{RE}$.

Further, we briefly cite some basic definitions and results of iterative splicing systems which are needed in the next section.

Let V be an alphabet, and $\#, \$ \notin V$ be two special symbols. A *splicing rule* over V is a string of the form

$$r = u_1 \# u_2 \$ u_3 \# u_4, \text{ where } u_1, u_2, u_3, u_4 \in V^*.$$

For such a rule $r \in R$ and strings $x, y, z \in V^*$, we write

$$(x, y) \vdash_r z \text{ if and only if } x = x_1 u_1 u_2 x_2,\ y = y_1 u_3 u_4 y_2, \text{ and } z = x_1 u_1 u_4 y_2,$$

for some $x_1, x_2, y_1, y_2 \in V^*$.

The string z is said to be obtained by splicing x, y, as indicated by the rule r; the strings $u_1 u_2$ and $u_3 u_4$ are called the *sites* of the splicing. We call x the *first term* and y the *second term* of the splicing operation.

An *H scheme* (a *splicing scheme*) is a pair $\sigma = (V, R)$, where V is an alphabet and $R \subseteq V^* \# V^* \$ V^* \# V^*$ is a set of splicing rules. For a given H scheme $\sigma = (V, R)$ and a language $L \subseteq V^*$, we write

$$\sigma(L) = \{z \in V^* \mid (x, y) \vdash_r z, \text{ for some } x, y \in L, r \in R\},$$

and we define

$$\sigma^*(L) = \bigcup_{i \geq 0} \sigma^i(L)$$

by

$$\begin{aligned} \sigma^0(L) &= L, \\ \sigma^{i+1}(L) &= \sigma^i(L) \cup \sigma(\sigma^i(L)), i \geq 0. \end{aligned}$$

An *extended H system* is a construct $\gamma = (V, T, A, R)$, where V is an alphabet, $T \subseteq V$ is the *terminal* alphabet, $A \subseteq V^*$ is the set of *axioms*, and $R \subseteq V^* \# V^* \$ V^* \# V^*$ is the set of *splicing rules*. The system is said to be *non-extended* when $T = V$. The language generated by γ is defined by

$$L(\gamma) = \sigma^*(A) \cap T^*.$$

$\mathbf{EH}(F_1, F_2)$ denotes the family of languages generated by extended H systems $\gamma = (V, T, A, R)$ with $A \in F_1$ and $R \in F_2$ where

$$F_1, F_2 \in \{\mathbf{FIN}, \mathbf{REG}, \mathbf{CF}, \mathbf{LIN}, \mathbf{CS}, \mathbf{RE}\}.$$

Theorem 2 ([6]). *The relations in the following table hold, where at the intersection of the row marked with F_1 with the column marked with F_2 there appear either the family $\mathbf{EH}(F_1, F_2)$ or two families F_3, F_4 such that $F_3 \subset \mathbf{EH}(F_1, F_2) \subseteq F_4$.*

	FIN	**REG**	**LIN**	**CF**	**CS**	**RE**
FIN	**REG**	**RE**	**RE**	**RE**	**RE**	**RE**
REG	**REG**	**RE**	**RE**	**RE**	**RE**	**RE**
LIN	**LIN, CF**	**RE**	**RE**	**RE**	**RE**	**RE**
CF	**CF**	**RE**	**RE**	**RE**	**RE**	**RE**
CS	**RE**	**RE**	**RE**	**RE**	**RE**	**RE**
RE	**RE**	**RE**	**RE**	**RE**	**RE**	**RE**

3 Main Results

In this section, we introduce the concept of *fuzzy splicing system*, initially assigning the truth values (i.e., the fuzzy membership values) from the closed interval $[0, 1]$ to the axioms of splicing systems. Then, we calculate the truth value of every generated string z from strings x and y using a fuzzy operation over their truth values.

Definition 1. *A fuzzy extended splicing system (a fuzzy H system) is a 6-tuple $\gamma = (V, T, A, R, \mu, \odot)$ where V, T, R are defined as for a usual extended H system, $\mu : V^* \to [0, 1]$ is a (fuzzy) membership function, A is a subset of $V^* \times [0, 1]$ and $\odot$ is a fuzzy operation over $[0, 1]$.*

A fuzzy splicing operation is defined as follows.

Definition 2. *For $(x, \mu(x)), (y, \mu(y)), (z, \mu(z)) \in V^* \times [0, 1]$ and $r \in R$,*

$$[(x, \mu(x)), (y, \mu(y))] \vdash_r (z, \mu(z))$$

if and only if $(x, y) \vdash_r z$ and $\mu(z) = \mu(x) \odot \mu(y)$.

Then, for a fuzzy splicing system $\gamma = (V, T, A, R, \mu, \odot)$, we define the fuzzy set of strings obtained by splicing strings in A according to splicing rules in R and the fuzzy operation $\odot$.

Definition 3. *Let $\gamma = (V, T, A, R, \mu, \odot)$ be a fuzzy splicing system. Then*

$$\sigma_f(A) = \{(z, \mu(z)) : (x, y) \vdash_r z \wedge \mu(z) = \mu(x) \odot \mu(y) \text{ for some } (x, \mu(x)), (y, \mu(y)) \in A \text{ and } r \in R\}.$$

Further, for a fuzzy splicing system $\gamma = (V, T, A, R, \mu, \odot)$, we define the closure of A under splicing with respect to rules in R and the fuzzy splicing operation $\odot$.

Definition 4. *Let $\gamma = (V, T, A, R, \mu, \odot)$ be a fuzzy splicing system. Then*

$$\sigma_f^*(A) = \bigcup_{i \geq 0} \sigma_f^i(A)$$

where $\sigma_f^0(A) = A$ and $\sigma_f^i(A) = \sigma_f^{i-1}(A) \cup \sigma_f(\sigma_f^{i-1}(A))$ for $i \geq 1$.

Definition 5. *The fuzzy language generated by a fuzzy splicing system $\gamma = (V, T, A, R, \mu, \odot)$ is defined as $L_f(\gamma) = \{(z, \mu(z)) \in \sigma_f^*(A) : z \in T^*\}$.*

We also define the "crisp" languages generated by fuzzy splicing systems.

Definition 6. *The crisp language generated by a fuzzy splicing system $\gamma = (V, T, A, R, \mu, \odot)$ is defined as $L_c(\gamma) = \{z : (z, \mu(z)) \in L_f(\gamma)\}$.*

Remark 1. It is clear that for every fuzzy splicing system $\gamma = (V, T, A, R, \mu, \odot)$, $L(\gamma') = L_c(\gamma)$ where $\gamma' = (V, T, A', R)$ with $A' = \{x : (x, \mu(x)) \in A\}$.

Example 1. We consider the fuzzy splicing system γ with multiplication operation as following,

$$\gamma = (\{a, b\}, \{a, b\}, \{(aa, 1/2), (aba, 1/3)\},$$
$$\{r_1 = a\#\lambda\$\lambda\#b, r_2 = a\#\lambda\$\lambda\#a, r_3 = b\#\lambda\$\lambda\#b, r_4 = b\#\lambda\$\lambda\#b\}).$$

One can easily show that $L_c(\gamma) = a\{a, b\}^*a$. Let us analyze the truth values of the strings obtained by splicing the strings in A:

$$\sigma^1(A) = \{(aa, 1/2), (aa, 1/4), (aa, 1/6), (aba, 1/3), (aba, 1/6), (aba, 1/9),$$
$$(aaba, 1/6), (aaba, 1/9), (aaa, 1/4), (aaa, 1/6), (a^3ba, 1/6),$$
$$(ababa, 1/9), (abaaba, 1/9), (abba, 1/9)\}.$$

We can see that string aa resulted from different strings has different truth values $1/2, 1/4, 1/6$, and strings aba, $aaba$ and $aaba$ have also different truth values. In order to overcome the ambiguity of truth values of strings, we can consider another fuzzy operation.

Another approach for the elimination of ambiguity is to define threshold languages, i.e., the selection of the "successful" subset of the crispy language generated by a fuzzy splicing system with respect to some cut-points. In fact, the fuzzy membership value of each string in the successful subset must satisfy the selected threshold mode. Hereby, we consider two interpretation of the threshold modes: in *strong interpretation* all fuzzy membership values of a string must satisfy the threshold condition and in *weak interpretation* at least one fuzzy membership value of a string must satisfy the threshold condition.

Further, we give formal definitions of threshold languages with respect to cut-points and relations of fuzzy membership values to these cut-points. We consider numbers α, subintervals and discrete subsets Ω (i.e., finite or countable subsets) of $[0, 1]$ as cut-points, and $=, \neq, <, >, \leq, \geq, \in, \notin$ as relations, which are called *threshold modes*.

Definition 7. *Let $\gamma = (V, T, A, R, \mu, \odot)$ be a fuzzy extended splicing system. Then, strong threshold languages generated by γ are defined as*

$$L_s(\gamma, *\alpha) = \{z : (z, \mu(z)) \in L_f(\gamma) \text{ and for all } \mu(z), \mu(z) * \alpha\},$$
$$L_s(\gamma, \star\Omega) = \{z : (z, \mu(z)) \in L_f(\gamma) \text{ and for all } \mu(z), \mu(z) \star \Omega\}$$

where $* \in \{=, \neq, >, \geq, <, \leq\}$ and $\star \in \{\in, \notin\}$.

Definition 8. *Let $\gamma = (V, T, A, R, \mu, \odot)$ be a fuzzy extended splicing system. Then, weak threshold languages generated by γ are defined as*

$$L_w(\gamma, *\alpha) = \{z : (z, \mu(z)) \in L_f(\gamma) \text{ and for some } \mu(z), \mu(z) * \alpha\},$$
$$L_w(\gamma, \star\Omega) = \{z : (z, \mu(z)) \in L_f(\gamma) \text{ and for some } \mu(z), \mu(z) \star \Omega\}$$

where $* \in \{=, \neq, >, \geq, <, \leq\}$ and $\star \in \{\in, \notin\}$.

We denote the family of strong and weak threshold languages generated by fuzzy extended H systems of type (F_1, F_2) by $sf\mathbf{EH}(F_1, F_2)$ and $wf\mathbf{EH}(F_1, F_2)$, respectively, where $F_1, F_2 \in \{\mathbf{FIN}, \mathbf{REG}, \mathbf{CF}, \mathbf{LIN}, \mathbf{CS}, \mathbf{RE}\}$.

Lemma 1. *For all families $F_1, F_2 \in \{\mathbf{FIN}, \mathbf{REG}, \mathbf{CF}, \mathbf{LIN}, \mathbf{CS}, \mathbf{RE}\}$,*

$$\mathbf{EH}(F_1, F_2) \subseteq xf\mathbf{EH}(F_1, F_2)$$

where $x \in \{s, w\}$.

Proof. Let $\gamma = (V, T, A, R)$ be an extended splicing system generating the language $L(\gamma) \in \mathbf{EH}(FIN, F)$ where $F \in \{\mathbf{FIN}, \mathbf{REG}, \mathbf{CF}, \mathbf{LIN}, \mathbf{CS}, \mathbf{RE}\}$. Let $A = \{x_1, x_2, ..., x_n\}$, $n \geq 1$. We associate the fuzzy splicing system γ' with γ where $\gamma' = (V, T, A', R, \mu, \odot)$, $A' = \{(x_i, 1) : x_i \in A, 1 \leq i \leq n\}$ and $\odot$ is a fuzzy operation (e.g., the multiplication operation, *max* or *min*) with the identity element 1. Then, it is not difficult to see that $L(\gamma) = L(\gamma')$. □

Lemma 2. *Let $\gamma = (V, T, A, R, \mu, \times)$ be a fuzzy extended splicing system with multiplication operation $\times$, where $0 < \mu(x) < 1$ for all $x \in A$. Let the sets A and R are finite. Then, for $x \in \{s, w\}$, $\alpha \in [0, 1]$ and $I \subseteq [0, 1]$,*

1. *$L_x(\gamma, > \alpha)$ is a finite language.*
2. *$L_x(\gamma, \leq \alpha)$ is a regular language.*
3. *$L_x(\gamma, \in I)$ is a regular language.*

Proof. Case 1. Let $\gamma = (V, T, A, R, \mu, \times)$ be a fuzzy splicing system where

$$A = \{(x_1, \mu_1), (x_2, \mu_2), \ldots, (x_n, \mu_n)\}$$

and $0 < \mu_i < 1$ for all $1 \leq i \leq n$. Then, it is clear that

$$\prod_{j=1}^{k} \mu_{i_j} > \prod_{j=1}^{k+1} \mu_{i_j}, \ \mu_{i_j} \in \{\mu_1, \ldots, \mu_n\}, \ 1 \leq j \leq m.$$

Hence, there exists $m \in \mathbb{N}$ such that

$$\prod_{j=1}^{m} \mu_{i_j} < \alpha,\ \mu_{i_j} \in \{\mu_1, \ldots, \mu_n\},\ 1 \leq j \leq m.$$

Thus, a finite number of $\mu(x)$s, $x \in L_f(\gamma)$, can satisfy the inequality $\mu(x) > \alpha$.

Case 2. It is clear that for $x \in \{s, w\}$, $L_c(\gamma) = L_x(\gamma, > \alpha) \cup L_x(\gamma, \leq \alpha)$. Since $L_c(\gamma)$ is regular and $L_x(\gamma, > \alpha)$ is finite then $L_x(\gamma, \leq \alpha)$ is regular.

Case 3. Let $I = (\alpha_1, \alpha_2)$. Then $L_x(\gamma, \in I) = L_x(\gamma, > \alpha_1) \cap L_x(\gamma, < \alpha_2)$, $x \in \{s, w\}$. From (i) and (ii), it follows that $L_x(\gamma, \in I)$ is regular. □

Lemma 3. *Let $\gamma = (V, T, A, R, \mu, \odot)$ be a fuzzy splicing system and $L_w(\gamma, *\alpha)$ be a threshold language where $\odot \in \{min, max\}$, $* \in \{>, <, =\}$ and $\alpha \in [0, 1]$. Let the sets A and R are finite. Then,*

1. *$L_w(\gamma, *\alpha)$ is a regular language.*
2. *If α is large enough then $L_w(\gamma, > \alpha) = \emptyset$ and $L_w(\gamma, \leq \alpha) = L_c(\gamma)$.*
3. *If α is small enough then $L_w(\gamma, > \alpha) = L_c(\gamma)$ and $L_w(\gamma, \leq \alpha) = \emptyset$.*
4. *If I is a subsegment of $[0, 1]$ then $L_w(\gamma, \in I)$ is regular.*

Proof. Let $\gamma = (V, T, A, R, \mu, \odot)$ be a fuzzy splicing system with

$$A = \{(x_1, \mu_1), (x_2, \mu_2), \ldots, (x_n, \mu_n)\}.$$

We denote by A' the crispy part of the set of axioms, i.e., $A' = \{x : (x, \mu(x)) \in A\}$.

Case 1. Consider *max* as the fuzzy operation and $>$ as the threshold mode. Then, the set $\sigma_f^*(A)$ can be represented as $\sigma_f^*(A) = \sigma_{f,1}^*(A) \cup \sigma_{f,2}^*(A)$ where

$$\sigma_{f,1}^*(A) = \{(x, \mu(x) \in \sigma_f^*(A) : \mu(x) > \alpha\}$$

and

$$\sigma_{f,2}^*(A) = \{(x, \mu(x) \in \sigma_f^*(A) : \mu(x) \leq \alpha\}.$$

Let $\sigma_{f,i}^0 = A_i$ and $A_i' = \{x : (x, \mu(x)) \in A_i\}$, $i = 1, 2$. Obviously, $A = A_1 \cup A_2$ and $A' = A_1' \cup A_2'$. Let $\sigma_{c,i}^*(A) = \{x : (x, \mu(x)) \in \sigma_{f,i}^*(A)\}, i = 1, 2$.

We construct the splicing system $\gamma' = (V, T, A_2', R)$ where $L(\gamma') = \sigma^*(A_2') \cap T^*$ is regular. Moreover, we show that $\sigma_{c,2}^*(A) = \sigma^*(A_2')$.

First, $\sigma^*(A_2') \subseteq \sigma_{c,2}^*(A)$ since $A_2 \subseteq A$. On the other hand, $\sigma_{c,2}^*(A) \subseteq \sigma^*(A_2')$. Let $x \notin \sigma^*(A_2')$. Then, there is an axiom $(x_1, \mu(x_1)) \in A_1$ such that

$$\begin{aligned}
&((x_1, \mu(x_1)), (x_2, \mu(x_2))) \vdash (z_1, \mu(z_1)),\\
&((z_1, \mu(z_1)), (z_2, \mu(z_2))) \vdash (z_3, \mu(z_3)),\\
&\qquad\vdots\\
&((z_k, \mu(z_k)), (z_{k+1}, \mu(z_{k+1}))) \vdash (x, \mu(x))
\end{aligned}$$

where $(x_2, \mu(x_2)) \in A$ and $(z_i, \mu(z_i)) \in \sigma_f^*(A)$. Then,

$$max\{\mu(x_1), \mu(x_2)\} = \mu(z_1) > \alpha,$$

$$\vdots$$

$$max\{\mu(z_k), \mu(z_{k+1})\} = \mu(x) > \alpha.$$

Consequently, $(x, \mu(x)) \notin \sigma_{f,2}^*(A)$, i.e., $x \notin \sigma_{c,2}^*(A)$. Thus, $\sigma_{c,2}^*(A) = \sigma^*(A_2')$. It follows that the language $L_w(\gamma, \leq \alpha) = \sigma_{c,2}^*(A) \cap T^*$ is regular.

In its turn, $\sigma_{c,1}^*(A) = \sigma_c^*(A) - \sigma_{c,2}^*(A)$, and the language $L_w(\gamma, > \alpha) = L_c(\gamma) - L_w(\gamma, \leq \alpha)$ is also regular.

Similarly, if the fuzzy operation is min, it can also be proved that $L_w(\gamma, > \alpha)$ and $L_w(\gamma, \leq \alpha)$ are regular.

Case 2. We choose $\alpha > max\{\mu_1, \mu_2, \ldots, \mu_n\}$.

Case 3. We choose $\alpha < min\{\mu_1, \mu_2, \ldots, \mu_n\}$.

Case 4. $L_w(\gamma, \in I) = L_w(\gamma, > \alpha_1) \cap L_w(\gamma, < \alpha_2)$ where $I = (\alpha_1, \alpha_2)$. From (i), $L_w(\gamma, > \alpha_1)$ and $L_w(\gamma, < \alpha_2)$ are regular. Therefore, their intersection is also regular. □

Remark 2. It should be noted that the arguments of the proof in Lemma 3.13 cannot be used for the strong case; because $L_s(\gamma, \leq \alpha) \subseteq \sigma_{c,2}^*(A) \cap T^*$, and it is not necessary the equality holds.

From the lemmas above we obtain the following theorem.

Theorem 3. *Every fuzzy splicing system with the fuzzy operation: multiplication, max or min, and the cut-point: any number in* $[0, 1]$ *or any subinterval of* $[0, 1]$ *generates a regular language.*

Although the threshold languages with numbers and subsegments of $[0, 1]$ are regular, the generative power of fuzzy splicing systems can be increased using discrete subsets of $[0, 1]$, i.e., functions whose codomains are subintervals of $[0, 1]$ as cut-points. The following examples show that, with this restriction, the generative capacity of fuzzy splicing systems can be increased up to context-sensitive languages.

Example 2. Let

$$\gamma = (\{a, b, c, d\}, \{a, b\}, \{(cad, 1/3), (dbc, 1/2)\},$$
$$\{r_1 = a\#d\$c\#ad, r_2 = db\#c\$a\#b, r_3 = a\#d\$d\#b\})$$

be a fuzzy splicing system with multiplication operation.

Then, by applying rule r_1 to axiom cad, we obtain strings ca^nd, $n \geq 1$, with $\mu(ca^nd) = 1/3^n$. Similarly, by applying rule r_2 to axiom dbc, we obtain strings db^mc, $m \geq 1$, with $\mu(db^mc) = 1/2^m$. The application of rule r_3 to these strings results in ca^nb^mc with $\mu(ca^nb^mc) = 1/3^n \cdot 1/2^m$.

Then $L_c(\gamma) = \{a^n b^m : n, m \geq 1\} \in \mathbf{REG}$ and

$$L_w(\gamma, = 1/5) = \emptyset \in \mathbf{FIN},$$
$$L_w(\gamma, > 1/3) = \{b\} \in \mathbf{FIN},$$
$$L_w(\gamma, \in \{1/6^n : n \geq 1\}) = \{a^n b^n : n \geq 1\} \in \mathbf{CF} - \mathbf{REG}.$$

One can see that the last threshold language generated by the fuzzy splicing system is not regular. However, if we consider *min* or *max* as fuzzy operations with the splicing system above, then the threshold languages are not more than regular. In this case for the generated strings $ca^n b^m c$ we have

$$\mu(ca^n b^m c) = \begin{cases} 1/3, n > 0, \\ 1/2, n = 0. \end{cases}$$

Therefore,

$$L_w(\gamma, \in \{1/6^n : n \geq 1\}) = \emptyset \in \mathbf{FIN},$$
$$L_w(\gamma, > 1/3) = \{b^n : n \geq 1\} \in \mathbf{REG}.$$

Example 3. Consider the following fuzzy splicing system with the multiplication operation

$$\gamma = (\{a, b, c, w, x, y, z\}, \{a, b, c\}, \{(xay, 1/3), (ybz, 1/5), (zcw, 1/7)\},$$
$$\{r_1 = xa\#y\$x\#a, r_2 = yb\#z\$y\#b, r_3 = zc\#w\$z\#c,$$
$$r_4 = a\#y\$y\#b, r_5 = b\#z\$z\#c\}).$$

By rule r_1 to the initial string xay, we obtain $(xa^k y, 1/3^k), k \geq 1$, by rule r_2 to the initial string ybz, we get $(yb^m z, 1/5^m), m \geq 1$, by rule r_3 to the initial string zcw, we have $(zc^n w, 1/7^n), n \geq 1$. The rules r_4 and r_5 the strings above result in

$$[(xa^k y, 1/3^k), (yb^m z, 1/5^m)] \vdash_{r_4} (xa^k b^m z, 1/3^k 5^m)$$

and

$$[(xa^k b^m z, 1/3^k 5^m), (zc^n w, 1/7^n)] \vdash_{r_5} (xa^k b^m c^n w, 1/3^k 5^m 7^n).$$

Then, the fuzzy language generated by γ is

$$L_f(\gamma) = \{(a^k b^m c^n, 1/3^k 5^m 7^n) : k, m, n \geq 1\}.$$

Further, we consider the following threshold languages:

$$L_w(\gamma, > 0) = \{a^k b^m c^n : k, m, n \geq 1\} \in \mathbf{REG},$$
$$L_w(\gamma, > 1/105^5) = \{a^k b^m c^n : 1 \leq k, m, n \leq 5\} \in \mathbf{REG},$$
$$L_w(\gamma, \in \{1/105^n : n \geq 1\}) = \{a^n b^n c^n : n \geq 1\} \in \mathbf{CS} - \mathbf{CF}.$$

4 Conclusions

In this paper, we have introduced the concept of fuzzy splicing system and established their preliminary properties. When fuzzy splicing systems are considered with multiplication, *max* or *min* operations and subintervals of $[0, 1]$, they cannot increase the generative power of splicing systems. The regularity of fuzzy splicing systems under strong interpretation remains open. If we choose discrete sets from $[0, 1]$, the power can be increased up to some context-sensitive languages. On the one hand, fuzzy splicing systems allow modeling molecular uncertainty processes appearing in molecular biology, systems biology and medicine. On the other hand, the study of fuzzy splicing systems in particular and the fuzzy variants of other theoretical models of DNA computing makes a significant contributions to formal language and automata theories.

Acknowledgement. This work has been supported through the Research University Grant (RUG) **07J41**, Universiti Teknologi Malaysia and Fundamental Research Grant Scheme **FRGS13-066-0307**, International Islamic University Malaysia, Ministry of Education, Malaysia.

References

1. Adleman, L.: Molecular computation of solutions to combinatorial problems. Science 266, 1021–1024 (1994)
2. Boneh, D., Dunworth, C., Lipton, R., Sgall, J.: On the computational power of DNA. Discrete Applied Mathematics. Special Issue on Computational Molecular Biology 71, 79–94 (1996)
3. Lipton, R.: Using DNA to solve NP–complete problems. Science 268, 542–545 (1995)
4. Head, T.: Formal language theory and DNA: An analysis of the generative capacity of specific recombination behaviors. Bull. Math. Biology 49, 737–759 (1987)
5. Pixton, D.: Regularity of splicing languages. Discrete Applied Mathematics 69, 101–124 (1996)
6. Păun, G., Rozenberg, G., Salomaa, A.: DNA computing. New computing paradigms. Springer-Verlag (1998)
7. Mordeson, J., Malik, D.: Fuzzy Automata and Languages. Theory and Applications. Chapman & Hall/CRC (2002)
8. Dassow, J., Păun, G.: Regulated rewriting in formal language theory. Springer-Verlag, Berlin (1989)
9. Rozenberg, G., Salomaa, A.: Handbook of formal languages, vol. 1-3. Springer, Heidelberg (1997)