International Conference Mathematical and Computational Biology 2011
International Journal of Modern Physics: Conference Series
Vol. 9 (2012) 89–94
© World Scientific Publishing Company
DOI: 10.1142/S2010194512005144

THE CHARACTERIZATIONS OF DIFFERENT SPLICING SYSTEMS

FARIBA KARIMI

Department of Mathematics, Faculty of Science, Universiti Teknologi Malaysia,
81310 UTM Johor Bahru, Malaysia
fk.karimi@gmail.com

NOR HANIZA SARMIN

Department of Mathematics, Faculty of Science, Universiti Teknologi Malaysia,
81310 UTM Johor Bahru, Malaysia
nhs@utm.my

FONG WAN HENG

Ibnu Sina Institute for Fundamental Science Studies, Universiti Teknologi Malaysia,
81310 UTM Johor Bahru, Malaysia
fwh@ibnusina.utm.my

The concept of splicing system was first introduced by Head in 1987 to model the biological process of DNA recombination mathematically. This model was made on the basis of formal language theory which is a branch of applied discrete mathematics and theoretical computer science. In fact, splicing system treats DNA molecule and the recombinant behavior by restriction enzymes and ligases in the form of words and splicing rules respectively. The notion of splicing systems was taken into account from different points of view by many mathematicians. Several modified definitions have been introduced by many researchers. In this paper, some properties of different kinds of splicing systems are presented and their relationships are investigated. Furthermore, these results are illustrated by some examples.

Keywords: Splicing systems; DNA recombination; formal language theory.

1. Introduction

In 1987, Head introduced a mathematical model for investigating the recombinant behavior of DNA molecules in the presence of restriction enzymes and ligases[1]. The DNA molecule, which is the genetic material of organisms, is a chain of nucleotides. The nucleotides differ by their chemical bases that are adenine (A), guanine (G), cytosine (C), and thymine (T). In[2] double stranded DNAs the bases pair up with each other: A with T and C with G. So, DNAs can be considered as sequences over four alphabets, A = {[A/T], [T/A], [C/G], [G/C]} and sometimes it is denoted by A = {a, t, c, g} where a = [A/T], t = [T/A], c = [C/G] and g = [G/C]. A restriction enzyme is an enzyme that cuts double stranded or single stranded DNA at specific recognition nucleotide sequences, known as restriction sites. After cutting DNAs with restriction enzymes, some fragments with sticky ends will be produced. If there is another kind of enzyme which is called DNA ligase existing in the system[3], the fragments with complementary ends can join together and generate new DNAs. In the splicing system model, the initial DNAs are associated with strings over the alphabet A = {a, t, c, g} and the restriction sites of enzymes are associated with some rules that determine their patterns. The language generated by the splicing system illustrates the recombinant DNAs. After this concept is introduced by Head, many other mathematicians developed it and considered it from different points of view[4-7]. Some important types of splicing systems are permanent, persistent, uniform and null-context are discussed here and their relations with each other are investigated.

2. Basic Definitions and Notations

In this section, some main concepts and notations that will be used in this paper will be introduced.

The theoretical basis of splicing system is studied under the framework of formal language theory that is mainly the study of finite sets of strings called languages[8].

Definiton 2.1.[8] A finite, nonempty set A of symbols is called an *alphabet*. Any finite sequence of symbols from alphabet is called a *string*.

The *empty string* which is a string with no symbol at all is usually denoted by 1.

If A is an alphabet, we use A^* to denote the set of strings obtained by concatenating zero or more symbols from A.

Any subset of A^* is called a *language* over A.

Definition 2.2.[1] A *splicing system* $S = (A, I, B, C)$ consists of a finite alphabet A, a finite set I of initial strings in A^*, and finite sets B and C of triples (c, x, d) with c, x and d in A^*. Each such triple in B or C is called a pattern. For each such triple the string cxd is called a site and the string x is called a crossing. Patterns in B are called left patterns and patterns in C are called right patterns. The language $L = L(S)$ generated by S consists of the strings in I and all strings that can be obtained by adjoining the words $ucxfq$ and $pexdv$ to L whenever $ucxdv$ and $pexfq$ are in L and (c, x, d) and (e, x, f) are patterns of the same hand. A language L is a *splicing language* if there exists a splicing system S for which $L = L(S)$.

Definition 2.3.[1] Let $S = (A, I, B, C)$ be a splicing system. Then S is *persistent* if for each pair of strings $ucxdv$ and $pexfq$, in A^* with (c, x, d) and (e, x, f) patterns of the same hand:

If *y* is a subsegment of *ucx* (respectively *xfq)* that is the crossing of a site in *ucxdv* (respectively *pexfq)* then this same subsegment *y* of *ucxfq* contains an occurrence of the crossing of a site in *ucxfq.*

Definition 2.4.[5] Let S = (A, I, B, C) be a splicing system. Then S is *permanent* if for each pair of strings *ucxdv,* and *pexfq,* in A^* with (c, x, d) and (e, x, f) patterns of the same hand: If *y* is a subsegment of *ucx* (respectively *xfq)* that is the crossing of a site in *ucxdv* (respectively *pexfq)* then this same subsegment *y* of *ucxfq* is an occurrence of the crossing of a site in *ucxfq.*

Definition 2.5.[1] A *null context* splicing system is a splicing system S = (A, I, B, C) for which each cleavage pattern in B and each in C has the form (1, x, 1).

Definition 2.6.[1] A *uniform* splicing system is a null context splicing system S = (A, I, X, X) for which there is a positive integer P such that $X = A^P$. A language L is a *uniform splicing language* if there is a uniform splicing system S for which $L = L(S)$.

Definition 2.7.[1] With respect to a language over A, a string c in A^* is a *constant* if, whenever *ucv* and *pcq* are in the language, *ucq* and *pcv* are also in the language.

Definition 2.8. [4] A language L is *strictly locally testable* if there is a positive integer k for which every factor of L of length k is a constant.

3. Main Results

According to Ref. 1 and Ref. 9, some of the relationships among different kinds of splicing systems and languages are known. In this section, the relations will be generalized by considering the permanent splicing systems and its relation with other types of splicing languages. Also, some examples are provided to demonstrate the results.

Persistent, uniform and strictly locally testable languages are equivalent. This is stated in the following theorem.

Theorem 3.1[1]**.** The following conditions on a language L over an alphabet A are equivalent:

1) L is a persistent splicing language.
2) L is a strictly locally testable language.
3) The set of constants for L contains A^p for some p.
4) L is a uniform splicing language.

Based on the definition, a set of permanent splicing systems is a subset of persistent splicing systems. However, the relation between the language associated with permanent splicing system and other types splicing languages are not investigated in the above theorem.

In the following theorems the placement of permanent splicing languages in the hierarchy of splicing languages is determined.

Theorem 3.2. Every null-context splicing system is permanent.

Proof. Suppose S = (A, I, B, C) is a null-context splicing system. Then according to definition, $B=\{(1,x_i,1):1\leq i\leq n\}$ and $C=\{(1,y_i,1):1\leq i\leq m\}$. To show that S is permanent, the patterns with the same crossings and the same hand should be considered. So suppose that $u1x_i1v$ and $p1x_i1q$ are two arbitrary strings from A^* such that $(1,x_i,1)\in B$.

If y is a subsegment of $u1x_i$ (respectively x_i1q) that is the crossing of a site in $u1x_i1v$ (respectively $p1x_i1q$) then according to B there exists $1\leq k\leq n$ such that $y=x_k$ and $(1,x_k,1)\in B$. Now x_k is a factor of $u1x_i1q$ and it is the crossing of the site $(1,x_k,1)$. In the case that $(1,x_i,1)\in C$ similarly it can be shown that the condition is satisfied. So S is persistent.

Theorem 3.3. Every uniform splicing system is permanent.

Proof. By Definition 2.6, a uniform splicing system is a null-context splicing system. Thus, the proof follows from Theorem 3.2.

From Theorem 3.3, the following result can be directly concluded.

Theorem 3.4. Every uniform splicing language is permanent.

Although persistent and permanent splicing systems are not equivalent, and permanent splicing systems are proper subsets of persistent splicing languages, it will be shown in the next theorem that the languages produced by them are equivalent.

Theorem 3.5. A language is persistent if and only if it is permanent.

Proof. Since every permanent splicing system is persistent, it is obvious that a permanent splicing language is persistent. For the converse, suppose that L is a persistent splicing language. According to Theorem 3.2, L is a uniform splicing language. By Theorem 3.3, L is a permanent splicing language.

Thus, Theorem 3.1 can be generalized as the following.

Theorem 3.6. The following conditions on a language L over an alphabet A are equivalent:

1) L is a permanent splicing language.
2) L is a persistent splicing language.
3) L is a strictly locally testable language.
4) The set of constants for L contains A^p for some p.
5) L is a uniform splicing language.

Therefore, persistent, permanent, uniform and strictly locally testable languages are equivalent. However, uniform, null-context, permanent and persistent splicing systems are proper subsets in the following manner:

$$\text{Uniform} \subset \text{null-context} \subset \text{permanent} \subset \text{persistent splicing systems}$$

Although those splicing languages are equivalent, they can be produced by different and distinct splicing systems.

An example is given in the following to illustrate a language that is generated by different splicing systems.

Example 3.1. Let L be the language $L = \{ttgatct, aagatca, ttgatca, aagatct\}$ over the alphabet $A = \{a, c, g, t\}$. It will be shown that L is a permanent splicing language that can be generated by three different kinds of splicing systems: S_1 that is permanent but not null-context, S_2 that is permanent and null-context and S_3 that is not permanent.

Let S_1 be the following splicing system:
$S_1 = \{A, I = \{ttgatct, aagatca\}, B = \{(t, gatc, a), (a, gatc, t), (t, gatc, t), (a, gatc, a)\}, C = \varnothing\}$.
We show that $L(S_1) = L$ while S_1 is permanent but not null-context.

In fact, if the strings $ttgatct$ and $aagatca$ are spliced with the patterns $(t, gatc, t)$ and $(t, gatc, a)$ the strings $ttgatca$ and $aagatct$ will be produced. So, $L(S_1) = L$.

Also, S_1 is permanent. Indeed, if $uegatcfv$ and $phgatckq$ are two arbitrary strings in A^*, with $(e, gatc, f)$ and $(h, gatc, k) \in B$ and $e, f, h, k \in A^*$, then according to B, e, f, h and k have to be in $\{a,t\}$. Suppose that y is a subsegment of $uegatc$ (respectively $gatckq$) and simultaneously crossing of a site in $uegatcfv$ (respectively $phgatckq$), then $y = gatc$. Now, it should be shown that y is crossing of a site in the string $uegatckq$. There are four sites in B, in the forms of tya, ayt, tyt and aya. If y is a substring of ue, since it is known that y is crossing of a site in $uegatcfv$, then y necessarily should be in u. Also, that site should be in ue and consequently in $uegatckq$. If y is the string exactly after ue and before fv then it is crossing of the site eyf. Therefore, eyf should be in one of the forms, tya, ayt, tyt or aya. On the other hand, since k is either a or t, so eyk also should be in B. Therefore, y is crossing of the site eyk in $uegatckq$. Thus S_1 is permanent.

Let S_2 be a splicing system associated with the restriction enzyme BfacI that is $S_2 = \{\{a, c, g, t\}, \{ttgatca, aagatct\}, \{(1, gatc, 1)\}, \varnothing\}$. It is clear that S_2 is null-context and consequently permanent. If the strings $ttgatca$ and $aatgatct$ are spliced with the restriction enzyme BfacI, the strings *ttgatct* and *aatgatca* will be produced. So, $L(S_2) = L$.

Let S_3 be a splicing system associated with the restriction enzymes {BclI, BglII} that is $S_3 = \{\{a, c, g, t\}, \{ttgatca, aagatct\}, \{(t, gatc, a), (a, gatc, t)\}, \varnothing\}$. It is clear that $L(S_3) = L$. Next, we show that S_3 is not permanent. Indeed, if we consider the strings *ttgatca* and *aatgatct* and $y = gatc$ as a substring of *ttgatc* , then *y* is the crossing of the site $(t, gatc, a)$ in *ttgatca* while it is not a crossing of any site in *ttgatct* . So according to Definition 2.4 S_3 cannot be permanent. However, the language generated by S_3 is permanent.

4. Conclusion

In this paper, some properties of different splicing systems are investigated. Although permanent splicing system is not equivalent to persistent, null-context and uniform splicing systems, the languages associated with them are equivalent. Also, it is shown that a splicing language can be generated with different types of splicing systems.

Acknowledgments

We would like to thank the Ministry of Higher Education (MOHE) and Research Management Centre (RMC), UTM for the FRGS Vote No. 78482.

References

1. T. Head, Formal language theory and DNA: an analysis of the generative capacity of specific recombinant behaviors, *Bulletin of Mathematical Biology*, vol. 49, 737-759 (1987).
2. R. H. Tamarin, *Principle of Genetics*, Seventh Edition. (The McGraw-Hill Companies, USA, 2001).
3. Gh. Paun, G. Rozenberg and A. Salomaa, *DNA Computing - New Computing Paradigms*, (Springer-Verlag, 1998).
4. T. Head, Splicing representations of strictly locally testable languages, *Discrete Applied Mathematics*, vol. 87, 139-147 (1998).
5. R. W. Gatterdam, Algorithms for splicing systems, *SIAM J. Comput.*, 21, pp. 507–520 (1992).
6. Victor Mitrana, Ion Petre and Vladimir Rogojin, Accepting Splicing Systems, *Theoretical Computer Science*, 411, 2414-2422 (2010).
7. P. Bonizzoni, C. D. Felice, G. Mauri and R. Zizza, Regular Languages Generated by Reflexive Finite Splicing Systems, In: Esik, Z. and Fulop, Z. (Eds). DLT 2003, LNCS 2710. 134-145 (2003).
8. P. Linz, *An introduction to formal languages and automata*, 3rd. ed. (Jones and Bartlett Publishers, Inc, USA, 2001).
9. W. H. Fong, Modelling of splicing systems using formal language theory, Ph.D. Thesis. Universiti Teknologi Malaysia (2008).