Weighted Splicing Systems

S. Turaev[1], Y.S. Gan[2], M. Othman[1], N.H. Sarmin[2,3], and W.H. Fong[3]

[1] Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
43400 UPM Serdang, Selangor, Malaysia
{sherzod,mothman}@fsktm.upm.edu.my
[2] Department of Mathematical Sciences, Faculty of Science
Universiti Teknologi Malaysia
81310 UTM Johor Bahru, Johor, Malaysia
ysgn88@gmail.com, nhs@utm.my
[3] 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 *weighted splicing systems*, and establish some basic properties of language families generated by this type of splicing systems. We show that a simple extension of splicing systems with weights can increase the computational power of splicing systems with finite components.

1 Introduction

DNA computing is one of the most exciting new developments of computer science from both theoretical and practical points of view. DNA computing models use *Watson-Crick complementary* of DNA molecules (sequences), which are double stranded structures composed of four nucleotides A (*adenine*), C (*cytosine*), G (*guanine*), and T (*thymine*). According to Watson-Crick complementary, adenine always bonds with thymine, and guanine with cytosine. This feature of DNA molecules makes possible only to check the information encoded on a single strand and makes far-reaching conclusions since the information on the other strand can be decoded according to the complementary. 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 solve many computationally intractable problems: Hamiltonian path problem (Adleman, [1]), the satisfiability problem for arbitrary contact networks (Lipton, [2]), the satisfiability problem for Boolean circuits (Boneh et al., [3]), etc.

One of the early theoretical proposals for DNA based computation was made by Head [4] who used the *splicing operation* – a formal model of the cutting and recombination of DNA molecules under the influence of restriction enzymes. This

Z. Li et al. (Eds.): ISICA 2012, CCIS 316, pp. 416–424, 2012.

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* (or *H 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.

Since splicing systems with finite sets of axioms and rules generate only regular languages (see [5]), several restrictions in the use of rules have been considered (see [6]), which increase the computational power up to the recursively enumerable languages. This is important from the point of view of DNA computing: splicing systems with restrictions can be considered as theoretical models of *universal programmable DNA based computers*.

In this paper we define another restriction of splicing systems, called *weighted splicing systems*, associating weights with the axioms, and calculating the weight $w(z)$ of the string z generated from two strings x and y from their weights $w(x)$ and $w(y)$ according to the operation $\odot$ defined on weights, i.e., $w(z) = w(x) \odot w(y)$. Then we consider several types of threshold languages generated by weighted splicing systems considering different weighting spaces and cut-points. We show that the selection of weighting spaces and cut-points effect to the generative power of splicing systems.

We should mention that weighted grammars and automata have been widely investigated in formal language theory since they were introduced in different forms in the 1960's (for instance, see[7-12]). On the one hand, the study of weighted grammars and automata shows that weights can increase the generative power of usual grammars and automata, and on the other hand, they help to construct more accurate models for stochastic phenomena and processes appearing in many applications of formal language theory. For instance, the use of weights makes possible to develop more efficient parsing and tagging algorithms for the natural and programming language processing.

This paper is organized as follows. Section 2 contains some necessary definitions and notations from the theories of formal languages and splicing systems. The concepts of weighted splicing systems and threshold languages generated by weighted splicing systems are introduced in Section 3. Section 4 illustrates the power of weighted splicing systems: it shows that some weighted splicing systems of finite components can generate even non-context-free 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 [6,13,14] 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 $\subset$. $\emptyset$ denotes the empty set. The sets of integers and positive rational numbers are denoted by $\mathbb{Z}$ and $\mathbb{Q}_+$, respectively. 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*, hold:

Theorem 1 ([??]). FIN $\subset$ **REG** $\subset$ **LIN** $\subset$ **CF** $\subset$ **CS** $\subset$ **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 $x_1, x_2, y_1, y_2 \in V^*$.

z is said to be obtained by splicing x, y, as indicated by the rule r; $u_1 u_2$ 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* 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

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

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*. When $T = V$, the system is said to be *non-extended*. 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* $F_3 \subset \mathbf{EH}(F_1, F_2) \subseteq F_4$.

F1 \ F2	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 Definitions

In this section we introduce the notion of weighted splicing systems which is specified with a weighting space and operations over weights closed in the weighting space.

Definition 1. *A weighted splicing system is a 7-tuple* $\gamma = (V, T, A, R, \omega, M, \odot)$ *where V, T, R are defined as for a usual extended H system, M is a weighting space, $\omega : V^* \to M$ is a weight function, $\odot$ is the operation over the weights $\omega(x)$, $x \in V^*$, and A is a subset of $V^* \times M$.*

Further, we define a weighted splicing operation and the language generated by a weighted splicing system.

Definition 2. *For* $(x, \omega(x)), (y, \omega(y)), (z, \omega(z)) \in V^* \times M$ *and* $r \in R$,

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

iff $(x, y) \vdash_r z$ *and* $\omega(z) = \omega(x) \odot \omega(y)$. *Then the language generated by the weighted splicing system γ is defined as*

$$L_\omega(\gamma) = \{z \in T^* \mid (z, \omega(z)) \in \sigma^*(A)\}.$$

Remark 1. As a weighting space one can consider different sets and (algebraic) structures, for instance, the sets of integers, rational numbers, real numbers, the sets of Cartesian products of the sets of numbers, the set of matrices with integer entries, the set of lattices, groups, etc. Then, the operations over weights of strings are defined with respect to the chosen weighting space. In this paper we consider as weighting spaces the sets of integers, positive rational numbers, the set of Cartesian products of integers and the set of 2×2 matrices with integer entries.

Definition 3. *Let $L_\omega(\gamma)$ be the language generated by a weighted splicing system $\gamma = (V, T, A, R, \omega, M, \odot)$. A threshold language $L_\omega(\gamma, *\tau)$ with respect to a threshold (cut-point) $\tau \in M$ is a subset of $L_\omega(\gamma)$ defined by*

$$L_\omega(\gamma, *\tau) = \{z \in T^* \mid (z, \omega(z)) \in \sigma^*(A) \text{ and } \omega(z) * \tau\}$$

where $* \in \{=, >, <\}$ is called the mode of $L_\omega(\gamma, *\tau)$.

Remark 2. We can also consider as a threshold a subset of M. Then, the mode for such a threshold is defined as a membership to the threshold set, i.e., for a threshold set $A \subseteq M$, the modes are $\in$ and $\notin$.

The family of threshold languages generated by weighted splicing systems of type (F_1, F_2) is denoted by $w\mathbf{EH}(F_1, F_2)$ where

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

From the definition, the next lemma follows immediately.

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

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

Proof. For any splicing system γ, we define the weighted splicing system by choosing $\{0\}$ as the weighting space, the usual addition as the weighting operation, and associating 0 with each axiom. Then $L_\omega(\gamma', = 0) = L(\gamma)$. □

4 Examples and Results

In this section we consider examples of weighted splicing systems with different weighting spaces, and show that for the same finite sets of axioms and splicing rules, the selection of weighting spaces effect to the generative power of the weighted splicing system, i.e., the same splicing system with different weighting spaces can generated regular, context-free and context-sensitive languages.

Example 1. Let us consider the weighted splicing system

$$\gamma_1 = (\{a, b, x, y\}, \{a, b, x\}, \{(xay, \tau_1), (ybx, \tau_2)\},$$
$$\{r_1 = a\#y\$x\#ay, r_2 = yb\#x\$y\#b, r_3 = a\#y\$y\#b\}, \omega, M, \odot).$$

It is not difficult to see that the application of the rule r_1 iteratively to the axiom xay and the generated strings results in the strings $xa^k y$ for all $k \geq 1$, and the application of the rule r_2 iteratively to the axiom yax and the generated strings results in the strings $yb^m x$ for all $m \geq 1$. Further, applying the rule r_3 to the strings $xa^k y$, $k \geq 1$, and $yb^m x$, $m \geq 1$, we obtain the strings $xa^k b^m x$. Thus,

$$L_\omega(\gamma_1) = \{xa^k b^m x \mid (xa^k b^m x, \omega(xa^k b^m x)) \in \sigma^*(A), k, m \geq 1\},$$

where $A = \{(xay, \tau_1), (ybx, \tau_2)\}$.

First, we choose the set $\mathbb{Z}$ of all integers as the weighting space M, the addition $+$ of integers as the operation $\odot$, and $\tau_1 = 1, \tau_2 = -1$. Then, it is clear that

$$L_\omega(\gamma_1) = \{xa^k b^m x \mid (xa^k b^m x, k-m) \in \sigma^*(A),\ k, m \geq 1\}$$

and

$$\begin{aligned} L_\omega(\gamma_1, = 0) &= \{xa^n b^n x \mid n \geq 1\}, \\ L_\omega(\gamma_1, > 0) &= \{xa^k b^m x \mid k > m \geq 1\}, \\ L_\omega(\gamma_1, < 0) &= \{xa^k b^m x \mid m > k \geq 1\}. \end{aligned}$$

These three languages are context-free and not regular.

Second, we choose the set $\mathbb{Z} \times \mathbb{Z}$ as the weighting space M. We define the operation $\odot$ as the componentwise addition of pairs from $\mathbb{Z} \times \mathbb{Z}$, i.e., for any two $(x_1, x_2), (y_1, y_2) \in \mathbb{Z} \times \mathbb{Z}$, the sum is defined by $(x_1 + y_1, x_2 + y_2)$ and the ordering relation is also defined by componentwise, i.e., $(x_1, x_2) > (y_1, y_2)$ if and only if $x_1 > y_1$ and $x_2 > y_2$. Let $\tau_1 = (1, 0), \tau_2 = (-1, 0)$. Then with the cut-point $(0, 0)$, we generate the same languages above.

Third, the same languages above are also generated, when we consider the set of all 2×2 matrices with integer entries as the weighting space M, the componentwise addition as the operation $\odot$, the componentwise ordering as the ordering relation, $\tau_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ and $\tau_2 = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$ as the weights, and $\tau = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$ as the cut-point.

Example 2. Consider a weighted splicing system

$$\begin{aligned} \gamma_2 = (\{a, b, c, w, x, y\}, \{a, b, w\}, \{(wax, \tau_1), (xby, \tau_2), (ycw, \tau_3)\}, \\ \{r_1 = a\#x\$w\#ax, r_2 = b\#y\$x\#by, r_3 = c\#w\$y\#cw, \\ r_4 = a\#x\$x\#b, r_5 = b\#y\$y\#c\}, \omega, M, \odot). \end{aligned}$$

One can see that for all $k, m, n \geq 1$,

$$(wa^k x, wax) \vdash_{r_1} wa^{k+1} x,\ (xb^m y, xby) \vdash_{r_2} xb^{m+1} y,\ (yc^k w, ycw) \vdash_{r_3} yc^{n+1} w.$$

Further,

$$(wa^k x, xb^m y) \vdash_{r_4} wa^k b^m y, k, m \geq 1,$$

and

$$(wa^k b^m y, yc^k w) \vdash_{r_5} wa^k b^m c^n w, k, m, n \geq 1.$$

Then, the language generated by the weighted splicing system γ_2 is

$$L_\omega(\gamma_2) = \{wa^k b^m c^n w \mid (wa^k b^m c^n w, \omega(wa^k b^m c^n w)) \in \sigma^*(A),\ k, m, n \geq 1\},$$

where $A = \{(wax, \tau_1), (xby, \tau_2), (ycw, \tau_3)\}$.

Next, we define different threshold languages with different weighting spaces and operations.

First, let $M = \mathbb{Q}_+$, the operation $\odot$ be the usual multiplication, and $\tau_1 = 3^{-1}$, $\tau_2 = 5^{-1}$, $\tau_3 = 15$. Then,

$$L_\omega(\gamma_2) = \{wa^k b^m c^n w \mid (wa^k b^m c^n w, 3^{n-k}5^{n-m}) \in \sigma^*(A),\ k, m, n \geq 1\}.$$

We choose $\tau = 1$ as a cut-point, and define the following threshold languages:

$$L_\omega(\gamma_2, = 1) = \{wa^n b^n c^n w \mid n \geq 1\} \in \mathbf{CS} - \mathbf{CF},$$
$$L_\omega(\gamma_2, > 1) = \{wa^k b^m c^n w \mid n > k, m \geq 1\} \in \mathbf{CF} - \mathbf{REG},$$
$$L_\omega(\gamma_2, < 1) = \{wa^k b^m c^n w \mid k, m > n \geq 1\} \in \mathbf{CF} - \mathbf{REG}.$$

Second, let $M = \mathbb{Z} \times \mathbb{Z}$, the operation $\odot$ is defined as in Example 1, and $\tau_1 = (1, 0)$, $\tau_2 = (-1, 1)$, $\tau_3 = (0, -1)$. Then,

$$L_\omega(\gamma_2) = \{wa^k b^m c^n w \mid (wa^k b^m c^n w, (k - m, m - n)) \in \sigma^*(A),\ k, m, n \geq 1\}.$$

Consequently,

$$L_\omega(\gamma_2, = (0, 0)) = \{wa^n b^n c^n w \mid n \geq 1\} \in \mathbf{CS} - \mathbf{CF},$$
$$L_\omega(\gamma_2, > (0, 0)) = \{wa^k b^m c^n w \mid k > m > n \geq 1\} \in \mathbf{CS} - \mathbf{CF},$$
$$L_\omega(\gamma_2, < (0, 0)) = \{wa^k b^m c^n w \mid n > m > k \geq 1\} \in \mathbf{CS} - \mathbf{CF}.$$

Third, the same languages above can be generated by γ_2 if we choose the set of all 2×2 matrices with integer entries as the weighting space, the componentwise addition as the operation $\odot$,

$$\tau_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},\ \tau_2 = \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix} \text{ and } \tau_3 = \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix},$$

and $\tau = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$ as the cut-point for the threshold languages.

Remark 3. Example 2 shows that the use of a Cartesian product of integers and matrices as weighting spaces make possible for splicing systems with finite components to generate non-context-free languages. For instance, one can easily construct a weighted splicing system which generates the language

$$\{wa_1^n a_2^n \cdots a_k^n w \mid n \geq 1\}$$

for any $k \geq 2$, if the Cartesian product $\underbrace{\mathbf{Z} \times \mathbf{Z} \times \cdots \times \mathbf{Z}}_{k}$ is used as the weighting space.

Combining the results of Theorem 2, Lemma 1 and Examples 1 and 2 above, we obtain the following results:

Theorem 3. *For* $F_1 \in \{\mathbf{LIN}, \mathbf{CF}\}$,

$$w\mathbf{EN}(\mathbf{FIN}, \mathbf{FIN}) - \mathbf{EN}(F_1, \mathbf{FIN}) \neq \emptyset.$$

Theorem 4.

$$\mathbf{REG} \subset w\mathbf{EN}(\mathbf{FIN}, \mathbf{FIN}) \subseteq \mathbf{RE}.$$

5 Conclusion

This paper introduces a new definition of weighted splicing systems and establishes some new facts. We have shown that even a simple extension of splicing systems with weights increases the generative power of splicing systems with finite components: in some cases they can generate non-context-free languages. The problem of the incomparability of the family of linear and context-free languages with the family of threshold languages generated by weighted splicing systems with finite components (the inverse inequality of that in Theorem 3) and the strictness of the second inclusion in Theorem 4 remain open. Most probably, linear languages as well as simple matrix languages (see [14]) can be generated by weighted splicing systems if matrices are used as weighting spaces but recursively enumerable languages may not be generated since the simulation of the context-sensitivity property of phrase-structure grammars is impossible with weights.

Acknowledgment. The first and third authors would like to acknowledge MOHE and UPM for the research grants with Vote No FRGS/1/11/SG/UPM/01/1 and RUGS 05-01-10-0896RU/F1 respectively. The second author is indebted to MOHE for his MyBrain15 Scholarship, and the fourth and fifth authors have been supported by UTM Research University Grant Vote No. 02J65.

References

1. Adleman, L.: Molecular computation of solutions to combinatorial problems. Science (266), 1021–1024 (1994)
2. Lipton, R.: Using DNA to solve NP Ccomplete problems. Science (268), 542–545 (1995)
3. Boneh, D., Dunworth, C., Lipton, R., Sgall, J.: On the computational power of DNA. Special Issue on Computational Molecular Biology (71), 79–94 (1996)
4. Head, T.: Formal language theory and DNA: An analysis of the generative capacity of specific recombination behaviors. Bull. Math. Biology (49) (1987)
5. Pixton, D.: Regularity of splicing languages. Discrete Applied Mathematics (69), 101–124 (1996)
6. Păaun, G., Rozenberg, G., Salomaa, A.: DNA computing. In: New Computing Paradigms. Springer (1998)
7. Salomaa, A.: Probabilistic and weighted grammars. Information and Control 15, 529–544 (1969)

8. Fu, K.S., Li, T.: On stochastic automata and languages. International Journal of Information Science (1969)
9. Mizumoto, M., Toyoda, J., Tanaka, K.: Examples of formal grammars with weights. Information Processing Letters 2, 74–78 (1973)
10. Mizumoto, M., Toyoda, J., Tanaka, K.: Various kinds of automata with weights. Journal of Computer and System Sciences 10, 219–236 (1975)
11. Alexandrakis, A., Bozapalidis, S.: Weighted grammars and kleenes theorem. Information Processing Letters 24, 1–4 (1987)
12. Droste, M., Kuich, W., Vogler, H. (eds.): Handbook of Weighted Automata. Springer (2009)
13. Rozenberg, G., Salomaa, A. (eds.): Handbook of formal languages, vol. 1-3. Springer (1997)
14. Dassow, J., Páun, G.: Regulated rewriting in formal language theory. Springer, Berlin (1989)