Journal of Mathematical and Computational Science
Vol. 2, No 2, December 2016

Computational Power of Probabilistic Simple One-Sided Sticker Languages

Mathuri Selvarajoo[1] , Fong Wan Heng[2], Nor Haniza Sarmin[2], Sherzod Turaev[3]

[1]Department of Mathematical Sciences, Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Kelantan, Malaysia.
[2]Department of Mathematical Sciences, Faculty of Science, Universiti Teknologi Malaysia, Johor Bharu, Malaysia.
[3]Department of Computer Science, Kulliyah of Information and Communication Technology, International Islamic University Malaysia, Kuala Lumpur, Malaysia.

mathuri644@kelantan.uitm.edu.my[1*]
[*]Corresponding author

Abstract: A DNA model of computing using the recombinant behaviour of DNA molecules known as sticker system was introduced by Kari in 1994. Sticker system is an abstract computational model which uses the Watson-Crick complementarity principle of DNA molecules. It starts from incomplete double-stranded sequences and using sticking operations iteratively which bring about complete double-stranded sequences. Sticker system and its variants including simple one-sided sticker system with finite sets of axioms and sticker rules, generate only regular languages. Therefore, different types of restrictions have been introduced in order to increase the computational power of the languages generated by sticker systems. In recent times, sticker system with probability as restriction, known as probabilistic sticker system is introduced. In this system, probabilities are initially associated with the axioms, and the probability of newly generated string is computed by multiplying the probabilities of all occurrences of the initial strings used in the computation of the new string. In this paper, the computational power of probabilistic simple one-sided sticker systems is investigated. We prove that the probability restriction on simple one-sided sticker system can increase the computational power of the languages generated.

Keywords: DNA computing; probability; regular language; simple one-sided sticker system; sticker system

1 Introduction

One of the early theoretical proposals for DNA-based computation was given by Head in 1987, known as the splicing systems [1]. There is in fact another mathematical model of DNA computing, known as the sticker systems. Adleman used the model of sticker system in his experiment of computing a Hamiltonian path in a graph by using DNA molecules [2]. The structure of DNA is double-helix (helicoidal) which is composed of four nucleotides: A (adenine), C (cytosine), G (guanine), and T (thymine) are paired as A with T and C with G according to Watson-Crick complementarity [3]. The double-stranded DNA sequences [A/T], [C/G], [G/C], and [T/A] are also represented as alphabets over four alphabets a, c, g and t respectively.

The concept of sticker system as a language generating model based on sticker operations was first proposed by Kari [3]. The axioms and strings generated by a sticker system are considered as encoded models of single and double-stranded DNA molecules. Moreover, sticker operations have advantages over splicing operations used in splicing systems because the sticker operations require no strands extension and use no enzymes [4]. In sticker systems, the initial sequences of DNA are prolonged to the left and to the right, producing computations of possible arbitrary length and the process stop when a complete double-stranded sequence is obtained and no sticky ends exist [3].

When forming new complete double-stranded sequences, the initial strands called axioms and well-started sequences are utilized and prolonged either to the left or to the right by the process of the sticker operation μ [5]. Since sticker systems with finite sets of axioms and sticker rules generate only regular languages without restrictions [6], monoids [7] and permutation groups [8] have been associated to generate more powerful languages than the regular languages.

Recently, a sticker system with restriction called probabilistic sticker system has been introduced in [9]. In the probability system, probabilities (real numbers in the range [0, 1]) are associated with the axioms, and the probability $p(z)$ of the string z generated from two strings x and y is calculated from the probability $p(x)$ and $p(y)$ according to the multiplication operation $*$ defined on the probabilities, i.e., $p(z) = p(x) * p(y)$. Then, the language generated by a probabilistic sticker system consists of all strings generated by the sticker system whose probabilities are greater than (or smaller than, or equal to) some previously chosen cut-points.

In this paper, a variant of sticker system known as simple one-sided sticker system is discussed. This system differs from the usual sticker system as restriction is imposed on the shape of the attaching sequence. Then, probability is considered in simple one-sided sticker system to form probabilistic simple one-sided sticker system. The necessary definitions from formal language theory, DNA computing and sticker systems are recalled in Section 2. The main results of this paper which involve the computational power of the languages generated by probabilistic simple one-sided sticker systems are discussed in Section 3. Discussion on the results and possible directions for future research will be discussed in Section 4.

2 Preliminaries

In this section, some fundamentals are recalled: some basic notions of the theories of formal languages, sticker systems and probabilistic sticker systems. The reader is referred to [3, 5, 9-10] for more detailed information. 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 $\varnothing$. The sets of integers, positive rational numbers and real numbers are denoted by $\mathbb{Z}, \mathbb{Q}^+$ and $\mathbb{R}$ respectively.

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 [10]:

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

Definition 1 [10]: Deterministic Finite Acceptor
A *deterministic finite accepter* or *DFA* is defined by the quintuple $M = (Q, \Sigma, \delta, q_0, F)$, where Q is a finite set of internal states, Σ is a finite set of symbols called the input alphabet, $\delta : Q \times \Sigma \to Q$ is a transition function, $q_0 \in Q$ is the initial state, and $F \subseteq Q$ is a set of final states.

Definition 2 [10]: Nondeterministic Finite Acceptor
A *nondeterministic finite accepter* or *NFA* is defined by the quintuple $M = (Q, \Sigma, \delta, q_0, F)$ where Q is a finite set of internal states, Σ is a finite set of symbols called the input alphabet, $\delta : Q \times (\Sigma \cup \{\lambda\}) \to 2^Q$ is a transition function, $q_0 \in Q$ is the initial state, and $F \subseteq Q$ is a set of final states.

Definition 3 [10]: Regular Language
A language is called *regular* if and only if there exists some deterministic or nondeterministic finite accepter M such that $L = L(M)$.

The concept of *sticker systems* was first considered in [3] and extended to *bidirectional sticker systems* in [11]. The sticker operation starts from incomplete double-stranded sequence and by iterative sticking operation, the strands are prolonged in order to obtain a complete double-stranded sequence.

The possible incomplete double-stranded sequences are defined below: let V be an alphabet (a finite set of abstract symbols) endowed with a symmetric relation ρ (of complementarity), i.e., $\rho \subseteq V \times V$. The symbol V^* is the set of all strings, including the empty string denoted by $\#$, composed of elements of V, and $V^+ = V^* - \{\#\}$. We write $\begin{pmatrix} V^* \\ V^* \end{pmatrix}$ instead of $V^* \times V^*$. We denote

$$\begin{bmatrix} V \\ V \end{bmatrix}_\rho = \left\{ \begin{pmatrix} a \\ b \end{pmatrix} \middle| \, a, b \in V, (a,b) \in \rho \right\}, \text{ and}$$

$$WK_\rho(V) = \begin{bmatrix} V \\ V \end{bmatrix}_\rho^*,$$

where the set $WK_\rho(V)$ is called the *Watson-Crick domain* associated to the alphabet V and complementarity relation ρ. We also denote

$$W_\rho(V) = L_\rho(V) \cup R_\rho(V) \cup LR_\rho(V),$$

where

$$L_\rho(V) = \left(\begin{pmatrix} \# \\ V^* \end{pmatrix} \cup \begin{pmatrix} V^* \\ \# \end{pmatrix} \right) \begin{bmatrix} V \\ V \end{bmatrix}_\rho^*,$$

$$R_\rho(V) = \begin{bmatrix} V \\ V \end{bmatrix}_\rho^* \left(\begin{pmatrix} \# \\ V^* \end{pmatrix} \cup \begin{pmatrix} V^* \\ \# \end{pmatrix} \right),$$

$$LR_\rho(V) = \left(\begin{pmatrix} \# \\ V^* \end{pmatrix} \cup \begin{pmatrix} V^* \\ \# \end{pmatrix} \right) \begin{bmatrix} V \\ V \end{bmatrix}_\rho^+ \left(\begin{pmatrix} \# \\ V^* \end{pmatrix} \cup \begin{pmatrix} V^* \\ \# \end{pmatrix} \right).$$

Any element of $W_\rho(V)$ which contains at least a position $\begin{bmatrix} a \\ b \end{bmatrix}, a \neq \#, b \neq \#,$ is called a *well-started* double-stranded sequence.

Let $x = x_1 x_2 x_3 \in W_\rho(V)$ where $x_2 \in WK_\rho(V) - \left\{ \begin{bmatrix} \# \\ \# \end{bmatrix} \right\}$ is a well-started double-stranded sequence and $y \in W_\rho(V)$ is a double-stranded sequence. The sticker operation denoted by $\mu(x, y)$, is defined as follows:

1. $x_3 = \begin{pmatrix} u \\ \# \end{pmatrix}, y = \begin{pmatrix} \# \\ v \end{pmatrix} y'$, for $u, v \in V^*$ such that $\begin{bmatrix} u \\ v \end{bmatrix} \in WK_\rho(V)$ and $y' \in R_\rho(V)$; then
   $\mu(x, y) = x_1 x_2 \begin{bmatrix} u \\ v \end{bmatrix} y'$;
2. $x_3 = \begin{pmatrix} \# \\ v \end{pmatrix}, y = \begin{pmatrix} u \\ \# \end{pmatrix} y'$, for $u, v \in V^*$ such that $\begin{bmatrix} u \\ v \end{bmatrix} \in WK_\rho(V)$ and $y' \in R_\rho(V)$; then
   $\mu(x, y) = x_1 x_2 \begin{bmatrix} u \\ v \end{bmatrix} y'$;

3. $x_3 = \begin{pmatrix} u_1 \\ \# \end{pmatrix}$, $y = \begin{pmatrix} u_2 \\ \# \end{pmatrix}$, for $u_1, u_2 \in V^*$; then $\mu(x,y) = x_1 x_2 \begin{pmatrix} u_1 u_2 \\ \# \end{pmatrix}$;

4. $x_3 = \begin{pmatrix} u_1 u_2 \\ \# \end{pmatrix}$, $y = \begin{pmatrix} \# \\ v \end{pmatrix}$, for $u_1, u_2 \in V^*$; such that $\begin{bmatrix} u_1 \\ v \end{bmatrix} \in WK_\rho(V)$; then

   $\mu(x,y) = x_1 x_2 \begin{bmatrix} u_1 \\ v \end{bmatrix} \begin{pmatrix} u_2 \\ \# \end{pmatrix}$;

5. $x_3 = \begin{pmatrix} u \\ \# \end{pmatrix}$, $y = \begin{pmatrix} \# \\ v_1 v_2 \end{pmatrix}$, for $u, v_1, v_2 \in V^*$ such that $\begin{bmatrix} u \\ v_1 \end{bmatrix} \in WK_\rho(V)$; then

   $\mu(x,y) = x_1 x_2 \begin{bmatrix} u \\ v_1 \end{bmatrix} \begin{pmatrix} \# \\ v_2 \end{pmatrix}$;

6. $x_3 = \begin{pmatrix} \# \\ v_1 \end{pmatrix}$, $y = \begin{pmatrix} \# \\ v_2 \end{pmatrix}$, for $v_1, v_2 \in V^*$; then $\mu(x,y) = x_1 x_2 \begin{pmatrix} \# \\ v_1 v_2 \end{pmatrix}$;

7. $x_3 = \begin{pmatrix} \# \\ v_1 v_2 \end{pmatrix}$, $y = \begin{pmatrix} u \\ \# \end{pmatrix}$, for $u, v_1, v_2 \in V^*$ such that $\begin{bmatrix} u \\ v_1 \end{bmatrix} \in WK_\rho(V)$; then

   $\mu(x,y) = x_1 x_2 \begin{bmatrix} u \\ v_1 \end{bmatrix} \begin{pmatrix} \# \\ v_2 \end{pmatrix}$;

8. $x_3 = \begin{pmatrix} \# \\ v \end{pmatrix}$, $y = \begin{pmatrix} u_1 u_2 \\ \# \end{pmatrix}$, for $u_1, u_2, v \in V^*$ such that $\begin{bmatrix} u_1 \\ v \end{bmatrix} \in WK_\rho(V)$; then

   $\mu(x,y) = x_1 x_2 \begin{bmatrix} u_1 \\ v \end{bmatrix} \begin{pmatrix} u_2 \\ \# \end{pmatrix}$,

where $\mu(y,x)$ $\gamma = (V, \rho, A, D)$, where V is an alphabet, $\rho \subseteq V \times V$ is the symmetric relation in V , $A \in LR_\rho(V)$ is a finite set of axioms, and D is a finite subset of $W_\rho(V) \times W_\rho(V)$ called dominoes.

For a given sticker system $\gamma = (V, \rho, A, D)$ and two sequences $x, y \in LR_\rho(V)$, we write $x \Rightarrow y$ if and only if $\mu(u, \mu(x,v))$ for some $(u,v) \in D$. Note that, $\mu(u, \mu(x,v)) = \mu(\mu(u,x), v)$. By $\Rightarrow^*$, we denote the reflexive and transitive closure of the relation $\Rightarrow$. A sequence $x_1 \Rightarrow x_2 \Rightarrow \cdots \Rightarrow x_k$, where $x_1 \in A$, is called a *computation* in γ with length $k-1$. If $x_k \in WK_\rho(V)$, the above computation is considered as *complete*.

The "*language of molecules*", denoted by $LM(\gamma)$, is the set of all double-stranded strings over V produced at the end of complete computations in γ i.e.,

$$LM(\gamma) = \{ w \in WK_\rho(V) \mid x \Rightarrow^* w, x \in A \}.$$

The *sticker language* generated by γ is defined by

$$L(\gamma) = \left\{ w \in V^* \mid \begin{bmatrix} w \\ w' \end{bmatrix} \in LM(\gamma) \text{ for some } w' \in V^* \right\}.$$

A sticker system $\gamma_{sos} = (V, \rho, A, D)$ is called *simple one-sided* if for all pairs $(u,v) \in D$ we have either $u,v \in \begin{pmatrix} \# \\ V \end{pmatrix}^*$ or $u,v \in \begin{pmatrix} V \\ \# \end{pmatrix}^*$, and for each $(u,v) \in D$ we have $u,v \in \begin{pmatrix} \# \\ V \end{pmatrix}^*$ or $u,v \in \begin{pmatrix} V \\ \# \end{pmatrix}^*$. The language generated by these systems is called simple one-sided sticker language (*SOSSL*).

Next, some important definitions of probabilistic sticker system are explained.

Definition 4 [9]: A Probabilistic Sticker System

A *probabilistic sticker system* (*pSS*) is a 5-tuple $\gamma' = (V, \rho, A_p, D_p, p)$, where V is an alphabet, $\rho \subseteq V \times V$ is the symmetric relation, A_p is a finite subset of axioms $(W_\rho(V) \times p)$, all the pairs in D_p are finite sets of pairs $[(B_d, B_u) \times p]$ where B_d and B_u are finite subsets of lower and upper stickers of the forms $\begin{pmatrix} \# \\ V \end{pmatrix}^+$ or $\begin{pmatrix} V \\ \# \end{pmatrix}^+$ respectively and $p : V^* \to [0,1]$ is a probability function such that

$$\sum_{(x, p(x)) \in \{A_p, D_p\}} p(x) = 1.$$

Definition 5 [9]: A Probabilistic Sticker Operation

A probabilistic sticker operation is defined as follows: for $(x, p(x)) \in A_p$ and $(u, p(u)), (v, p(v)) \in D_p$,

$$[(x, p(x))] \Rightarrow^* [(y, p(y))]$$

if and only if

i) $(y, p(y)) = \mu[(u, p(u)), \mu((x, p(x)), (v, p(v)))]$ and $p(y) = p(u) \cdot [p(x) \cdot p(v)]$ or

ii) $(y, p(y)) = \mu[\mu((x, p(x)), (u, p(u))), (v, p(v))]$ and $p(y) = [p(x) \cdot p(u)] \cdot p(v)$.

Definition 6 [9]: Probabilistic Sticker Languages

1. The *probabilistic molecular sticker language* (*pMSL*) generated by a probabilistic sticker system γ' is defined as

$$pMSL(\gamma') = \{(y, p(y)) \in WK_p(V) \times [0,1] \mid (x, p(x)) \Rightarrow^* (y, p(y)) \text{ for } (x, p(x)) \in A_p\}.$$

2. The *probabilistic sticker language* (*pSL*) generated by a probabilistic sticker system γ' is defined by

$$pSL(\gamma') = \{(w, p(y)) \mid (y, p(y)) \in pSLM(\gamma) \text{ and } y = \begin{bmatrix} w \\ w' \end{bmatrix} \text{ where } w, w' \in V^*\}.$$

In the next section, the main results of this paper on the probabilistic simple one-sided sticker system are discussed.

3 Results

In this section, some results regarding probabilistic simple one-sided sticker systems are discussed and proved. Here, *SOSSL* and *pSOSSL* denote the families of languages generated by simple one-sided sticker system and probabilistic simple one-sided sticker system respectively.

Definition 7: Probabilistic Simple One-Sided Sticker System

A probabilistic simple one-sided sticker system **(***pSOSSS***)** is a construct of 5-tuple

$$\gamma'_{sos} = (V, \rho, A_p, D_p, p),$$

where V is an alphabet, $\rho \subseteq V \times V$ is the symmetric relation in V, A_p is a finite subset of axioms $(W_\rho(V) \times p)$ and all the pairs in D_p are finite sets of pairs $[(B_d, B_u) \times p]$ where B_d and B_u are finite subsets of lower and upper stickers of the forms $\begin{pmatrix} \# \\ V \end{pmatrix}^+$ or $\begin{pmatrix} V \\ \# \end{pmatrix}^+$ respectively. Each pair attached to A_p is the finite subset of lower or upper sticker in the forms $\begin{pmatrix} \# \\ V \end{pmatrix}^+$ or $\begin{pmatrix} V \\ \# \end{pmatrix}^+$ respectively and $p : V^* \rightarrow [0,1]$ is a probability function such that

$$\sum_{(x, p(x)) \in \{A_p, D_p\}} p(x) = 1.$$

Definition 8: Probabilistic Simple One-Sided Sticker Operation

A probabilistic simple one-sided sticker operation is defined as follows: for $(x, p(x)) \in A_p$ and $(u, p(u)), (v, p(v)) \in D_p$,

$$[(x, p(x))] \Rightarrow^* [(y, p(y))]$$

if and only if

i) $(y, p(y)) = \mu[(u, p(u)), \mu((x, p(x)), (v, p(v)))]$ and $p(y) = p(u) \cdot [p(x) \cdot p(v)]$ or

ii) $(y, p(y)) = \mu[\mu((x, p(x)), (u, p(u))), (v, p(v))]$ and $p(y) = [p(x) \cdot p(u)] \cdot p(v)$.

Definition 9: Probabilistic Simple One-Sided Sticker Language (*pSOSSL***)**

The language generated by the probabilistic simple one-sided sticker system γ' is defined as

$$pSOSSL(\gamma') = \{ y \in WK_p(V) \mid (x, p(x) \overset{*}{\Rightarrow} (y, p(y)) \text{ for } (x, p(x)) \in A_p \}.$$

The probability extension for simple one-sided sticker systems can be used as special subsets of the languages generated by simple one-sided sticker systems according to some thresholds (cut-points) which are sub-segments, discrete subsets of $[0,1]$ and real numbers in $[0,1]$. We define the following two types of *threshold languages* with respect to thresholds $\alpha \in [0,1]$ and $\beta \subseteq [0,1]$:

$pSOSSL(\gamma', *\alpha) = \{y \mid (y, p(y)) \in pSOSSL(\gamma') \text{ and } p(y) * \alpha\}$,
$pSOSSL(\gamma', \bullet\beta) = \{y \mid (y, p(y)) \in pSOSSL(\gamma') \text{ and } p(y) \bullet \beta\}$, where $* \in \{=, \neq, \geq, >, \leq, <\}$ and $\bullet \in \{\in, \notin\}$ are the *threshold modes*.

The following lemma and propositions show that the languages generated by probabilistic simple one-sided sticker systems have a higher computational power than the languages generated by the usual simple one-sided sticker systems.

Lemma 1: $SOSSL \subseteq pSOSSL$.

Proof:

Consider a simple one-sided sticker system $\gamma_{sos} = (V, \rho, A, D)$. Then the language generated by the sticker system γ_{sos} is

$$SOSSL(\gamma) = \left\{ z \in WK(V) \mid x \overset{*}{\Rightarrow} z, x \in A \right\}.$$

Let $\gamma'_{sos} = (V, \rho, A_p, D_p, p)$ be a probabilistic simple one-sided sticker system where $A_p = \{(x_i, p(x_i)) \mid x_i \in A, 1 \leq i \leq n)\}$, $D_p = \{(u_i, p(u_i)), (v_i, p(v_i)) \mid u_i, v_i \in D, 1 \leq i \leq n\}$ and $p(\theta_i) = 1/n$ for all $1 \leq i \leq n, \theta \in \{x, u, v\}$, then

$$\sum_{i=1}^{n} p(\theta_i) = 1.$$

The language generated by the probabilistic simple one-sided sticker system γ'_{sos}:

$$pSOSSL(\gamma', *\alpha) = \{z \in WK_p(V) \mid [x, p(x)] \overset{*}{\Rightarrow} [z, p(z)] \text{ for } [x, p(x)] \in A_p \} \text{ where}$$
$$p(z) = p(x) \cdot p(\tau_1) \cdot p(\tau_2) \cdots p(\tau_n) \text{ for } \tau_1, \tau_2, \ldots, \tau_n \in D_p.$$

We define the threshold language generated by γ' as $pSOSSL(\gamma', > 0)$, then it is not difficult to see that

$$SOSSL(\gamma) = pSOSSL(\gamma', > 0).$$

Proposition 1:

For any probabilistic simple one-sided sticker system γ'_{sos} the threshold language $L(\gamma'_{sos}, = 0)$ is the empty set, i.e. $L(\gamma'_{sos}, = 0) = \varnothing$.

Proposition 2:

For any probabilistic simple one-sided sticker system γ'_{sos}, the threshold language $L(\gamma'_{sos}, = \eta)$ where $\eta > 0$ is finite.

Next, some important theorems on the computational power of the languages generated by simple one-sided sticker system and probabilistic simple one-sided sticker system are proven.

Theorem 1: $pSOSSL \subseteq \mathbf{CS}$.

Proof:

Since the erasing rule does not exist in simple one-sided sticker systems and probabilistic simple one-sided sticker system, hence $SOSSL \neq \mathbf{RE}$ and $pSOSSL \neq \mathbf{RE}$. Therefore, the theorem holds.

Theorem 2: $SOSSL \subset \mathbf{CF}$.

Proof:

From Corollary 1 in [5], it is stated that $SOSSL \subset SSL$ where SSL are the languages generated by simple sticker systems. Moreover, Theorem 5 in [5] proves that $SSL \subset \mathbf{CF}$. Hence, the theorem is proven.

Theorem 3: $SOSSL \subset pSOSSL$.

Proof:

From Lemma 1, it is proven that $SOSSL \subseteq pSOSSL$. Theorem 2 shows that $SOSSL \subset \mathbf{CF}$, while Theorem 1 proves that $pSOSSL \subseteq \mathbf{CS}$. Hence, the theorem holds.

The following lemma and theorem show the relation of the languages generated by probabilistic simple sticker systems and probabilistic simple one-sided sticker systems.

Lemma 2: $pSOSSL \subseteq pSSL$.

Proof:

From the definition of probabilistic simple sticker language ($pSSL$) in [9] and $pSOSSL$, it is clear that all languages generated by probabilistic simple one-sided sticker system can be generated by probabilistic simple sticker system. Hence, the lemma is proven.

Theorem 4: $SOSSL \subset pSOSSL \subseteq pSSL$.

Proof:

Theorem 3 proves that $SOSSL \subset pSOSSL$ and Lemma 2 shows that $pSOSSL \subseteq pSSL$. Hence, the theorem holds.

4 Conclusion

In this paper, we introduce probabilistic simple one-sided splicing system by associating probabilities to the axioms and dominoes of simple one-sided sticker system. The computational power of the languages generated by probabilistic simple one-sided sticker systems are discussed, and it has been found that the computational power of the languages generated are up to context-sensitive. The strictness of the inclusion $SOSSL \subset pSOSSL$ has also been shown. However, the studies on the restrictions such as primitive and of delay variants of probabilistic simple one-sided sticker systems remain open.

Acknowledgments

The second and third authors would also like to thank MOE and Research Management Center (RMC), Universiti Teknologi Malaysia for the financial funding through Fundamental Research Grant Scheme (FRGS) Vote No. 4F590. The fourth author acknowledges the financial support by MOE FRGS13-066-0307.

References

[1] Head, T. Formal language theory and DNA: An analysis of the generative capacity of specific recombination behaviors. BULLETIN OF MATHEMATICAL BIOLOGY. 1987. 49(6): 737-759.

[2] Adleman, L. Molecular computations of solutions to combinatorial problems. SCIENCE. 1994. 266: 1021-1024.

[3] Kari, L., Paun, G., Rozenberg, G., Salomaa, A. and Yu, S. DNA computing, sticker systems and universality. ACTA INFORMATICA. 35: 401-420.

[4] Kari, L., Seki, S. and Sosik, P. DNA Computing: Foundations and Implications for Computer Science. New York: Springer-Verlag. 2010.

[5] Paun, G. and Rozenberg, G. Sticker systems, THEORETICAL OF COMPUTER SCIENCE. 1998. 204; 183-203.

[6] Xu, J., Dong, Y. and Wei, X. Sticker DNA computer model, Part 1: Theory. CHINESE SCIENCE BULLETIN. 2004. 49(8). 772-780.

[7] Mohd Sebry, N. A., Hamzah, N. Z. A., Sarmin, N. H., Fong, W. H. and Turaev S. Sticker system over monoid. MALAYSIAN JOURNAL OF FUNDAMENTAL AND APPLIED SCIENCES. 2012. 8(3). 127 – 132.

[8] Mohd Sebry, N. A., Hamzah, N. Z. A., Sarmin, N. H., Fong, W. H. and Turaev S. Splicing systems over some permutation groups. MALAYSIAN JOURNAL OF FUNDAMENTAL AND APPLIED SCIENCES. 2012. 8(2). 83 – 88.

[9] Selvarajoo, M., Fong, W. H., Sarmin, N. H. and Turaev S. Probabilistic sticker systems. MALAYSIAN JOURNAL OF FUNDAMENTAL AND APPLIED SCIENCES. 2013. 9(3). 150 – 155.

[10] Linz, P. An Introduction to Formal Languages and Automata. United States: Jones and Bartlett. 2012.

[11] Freund, R., Paun, G., Rozenberg, G. and Salomaa, A. Bidirectional sticker systems. In Proceedings of Annual Pacific Conference of Biocomputing. 1998. 536-546.

[12] Alhazov, A. and Ferretti, M. Computing by Observing Bio-Systems: The Case of Sticker Systems. New York: Springer-Verlag. 2004.