Journal of Physics: Conference Series

PAPER • **OPEN ACCESS**

The Properties of Semi-Simple Splicing System Over Alternating Group, A_3

To cite this article: Mathuri Selvarajoo *et al* 2021 *J. Phys.: Conf. Ser.* **1770** 012001

The Properties of Semi-Simple Splicing System Over Alternating Group, A_3

Mathuri Selvarajoo[a]*, Fong Wan Heng[b], Nor Haniza Sarmin [b] and Sherzod Turaev[c]

[a]Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia
[b]Department of Mathematical Science, Faculty of Science, Universiti Teknologi Malaysia, Johor Bharu, Johor, Malaysia
[c]Faculty of Engineering and Natural Sciences, International University of Sarajevo, Hrasnicka cesta 15, 71210, llidza, Sarajevo, Bosnia i Herzegovina

mathuri@tmsk.uitm.edu.my

Abstract. Splicing system was introduced by Head in 1987 in order to explore the recombinant behaviour of deoxyribonucleic acid (DNA) strands in the presence of restriction enzymes and ligases. Restriction enzymes cut the DNA strands into a left - pattern and right-pattern while the ligases recombine the left-pattern of the first string with the right-pattern of the second string and vice-versa. Semi-simple splicing system is a variant of splicing system where the splicing of DNA strands will be at two distinct sites. Splicing languages generated from the splicing system are classified based on their computational power, according to the Chomsky Hierarchy. Previous researchers found that the splicing system of finite set of strings and rules generates only regular languages which has the lowest computational power. Hence, some restriction has been introduced to increase the computational power of splicing languages generated. In this research, an element of alternating group of order three is associated with the initial strings of semi-simple splicing system to generate non-regular languages which has higher computational power. Some lemmas and theorems are proven to show that associating the alternating group to the initial strings could increase the computational power of the splicing languages

1. Introduction

The model of recombinant behaviour of the DNA was first introduced by Head in 1987 [1]. DNA stored the hereditary information in the form of a code of four nitrogenous bases, Adenine (A), Guanine (G), Thymine (T) and Cytosine (C). The combination of a sugar group, a phosphate group, and a nitrogenous base form a nucleotide. Nucleotides bind together to form a strand. The sugar molecule of a nucleotide binds to phosphate molecule of another nucleotide by covalent bonds thus, forming an alternating sugar-phosphate backbone. The combination of two nucleotides forms a double-helical structure [2]. The splicing operation deals with the process of cutting the DNA string in the presence of restriction enzymes to form two cleavage patterns. Ligases help in recombining two cleavage pattern of different strings and form a new string.

The model of splicing has been studied by Paun in 1996 where the splicing rule was being introduced in the form of $r=a\#b\$c\#d$ [3]. The symbol of # and $ is not an alphabet but symbolize the cut and paste operation. Pixton also studied the splicing system and introduced Pixton splicing operation in the same year which focuses on the splicing operation when the rules in triple [4]. Later, Yusof-Goode splicing was introduced where the rules are symmetric and reflexive [5]. The splicing languages generated from splicing system are classified according to Chomsky hierarchy [6].

Splicing systems are classified based on distinct splicing rules. Each restriction enzymes cut the strings at a different site, thus differentiate each of the splicing rules. [7] has introduced simple splicing system with the splicing rules of *r=a#1$a#1* where 1 represent any strings. The simple splicing system of finite sets of strings and rules generates regular splicing language. Another variant of splicing that been introduced was semi-simple splicing where the rules slightly different from simple splicing [8]. The splicing rules for simple splicing system are *r=a#1$b#1*. The splicing system of finite sets of strings and finite sets of rules generate regular language.

Some restriction has been introduced to the splicing system to increase the computational power of the splicing languages generated especially for finite splicing [9-11]. Permutation element has been used to associate the initial strings of splicing system. The strings generated with identity element is considered valid.

This research study the semi-simple splicing system over alternating group of order three where alternating element of order three are used to associates the initial strings of splicing system. The definition of a semi-simple splicing system is defined, and the splicing languages generated from the splicing system are classified according to the Chomsky Hierarchy.

2. Preliminaries

The definitions of splicing systems and other related terms are given in this section. The definition of splicing system as introduced by Head is given followed by definitions of other types of splicing system.

Definition 1 [12] Splicing System

The splicing system *S=(A, I, B, C)* consists of a finite alphabet *A*, finite set I of initial strings in *A**, and finite sets *B* and *C* of triples *(c,x,d)* where *c,x,* and *d* is in *A**. Each triple in *B* or *C* is called a pattern. For each triple string, *cxd* is called site and *x* is crossing. *B* is the left pattern and *C* is the right pattern. *L* is the splicing language exists in a splicing system *S* for which *L=L(S).*

Definition 2 [12] Paun Splicing System

The splicing scheme $\sigma = (A, R)$ where A is the alphabet and R is splicing rules in the form of $R = A^*\#A^*\$A^*\#A^*$. Let $x, y \in A^*$ and the two symbols # and $ is not in A^*. The initial strings will be spliced using the rules $r \in R$.

Let two initial strings of $U = u_1u_2v_1v_2$ and $V = u'_1u'_2v'_1v'_2$ where strings $U, V \in A^*$ and $u, v, u', v' \in A$. U and V are spliced by the rules of $u_2\#v_1\$u'_2\#v'_1$ where the first string spliced between u_2 and v_1 while the second string spliced between u'_2 and v'_1. The new strings produced are Z_1 and Z_2 where $Z_1 = u_1u_2v'_1v'_2$ and $Z_2 = u'_1u'_2v_1v_2$.

Definition 3 [12] Pixton Splicing System

Splicing system in a pair of $\mathcal{Y} = (R, L_0)$, where R is the splicing scheme and L_0 is the set of strings or initial languages. The Splicing scheme can be denoted as (A, r). The $r \in R$ is denoted in a triple in the form of $(\alpha, \alpha'; \beta)$. The rule r is applied to two initial strings $\rho_1 = \varphi\alpha\omega$ and $\rho_2 = \varphi'\alpha'\omega'$ generated a new string $\rho_3 = \varphi\beta\omega'$.

Definition 4 [12] Yusof-Goode Splicing System

The splicing system $S = (A, I, R)$, where A is the alphabet of a, c, g, and t, I represent the initial string of DNA and R is the restriction enzymes. One of the theorems regarding this model is the rules are symmetric and reflexive which summarized as $(b, 1, c: a, 1, b)$.

The definition of permutation group and alternating group also stated as alternating group is used as a restriction to the splicing system.

Definition 5 [13] Permutation Group
Let A be a finite set of $\{1, 2, 3, 4, \ldots, n\}$. The group of all permutations of A or permutation group A is the symmetric group on n letters and is denoted by S_n.

Definition 6 [14] Even Permutation
A permutation of a finite set is even if it can be expressed as a product of even number of transpositions. A subgroup of S_n with even permutations of the finite element of n is the **Alternating Group** of A_n.

The splicing language generated from the splicing system is determined by using Chomsky grammar. The definition of Chomsky grammar and family of languages are given in the following.

Definition 7 [15] Chomsky Grammar
The grammar $\mathcal{G}$ defined in quadruple $\mathcal{G} = (\Sigma, N, S, R)$ where Σ is the set of terminals, N is the set of non-terminals, S is the start symbol and R is a set of rules.

Definition 8 [15] Regular Grammar
The rule for regular grammar is in the form of either $A \rightarrow a$ or $A \rightarrow aB$. The letter A and B denote the non-terminal symbol while a is a terminal symbol. Regular grammar generates a regular language.

Definition 9 [15] Context-Free Grammar
Context-free grammar generates the context-free language. The rules in a context-free grammar in the form of $A \rightarrow \beta$ where A is a single and non-terminal symbol and β is a string of symbols. Context-free grammar generates a context-free language.

Definition 10 [15] Context-Sensitive Grammar
Grammar in form of $A \rightarrow B$ for each of the left pattern rule (A) never exceeds the length of right pattern (B) or in short, $length(A) \leq length(B)$ is known as context-sensitive grammar. Context-sensitive grammar generates a context-free language.

Definition 11 [15] Recursively Enumerable Grammar
Recursively enumerable language defined from unrestricted grammar. All classes of languages represented by some of the formal theories are called recursively enumerable. Recursively enumerable grammar generates recursively enumerable language.

The regular, context-free, context-sensitive, and recursively enumerable language are denoted as REG, CF, CS, and RE respectively. The languages are being classified according to the Chomsky hierarchy where REG $\subset$ CF $\subset$ CS $\subset$ RE. The Chomsky hierarchy is illustrated in the Figure below.

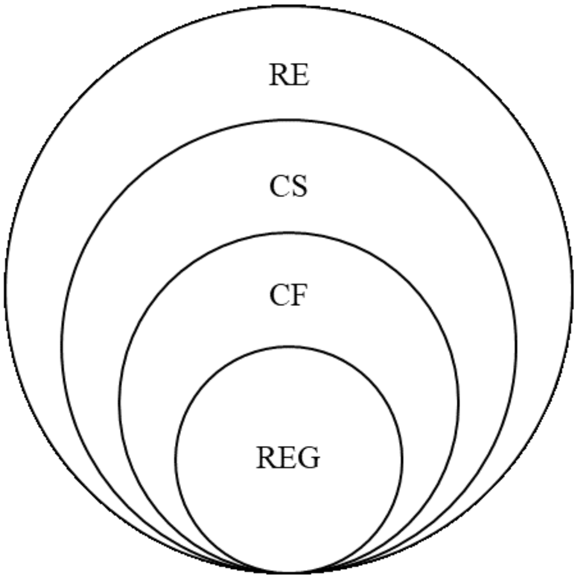

Fig. 1. Chomsky Hierarchy [14]

SSS(F_1, F_2) denotes the languages generated from splicing system S=(A, I, R) where $F_1 \in A$ and $F_2 \in R$ where $F_1, F_2 \in$ {FIN, REG, CF, LIN, CS, RE}.

Theorem 1: The intersection between the row of F_1 and column of F_2 shows the language generated which is either the family F_1 or F_2 under splicing operation [17].

F_2 \ F_1	FIN	REG	CF	LIN	CS	RE
FIN	REG	RE	RE	RE	RE	RE
REG	REG	RE	RE	RE	RE	RE
CF	LIN,CF	RE	RE	RE	RE	RE
LIN	CF	RE	RE	RE	RE	RE
CS	RE	RE	RE	RE	RE	RE
RE	RE	RE	RE	RE	RE	RE

Table 1. Languages generated from F_1 and F_2 [15]

The splicing languages generated from F_1 and F_2 shown in Table 1 for the family of languages $F_1, F_2 \in$ {FIN, REG, CF, LIN, CS, RE}.

3. Semi-Simple Splicing System Over Alternating Group

A restriction has been introduced to increase the computational power of the languages generated from finite sets of strings and rules. Alternating group is used as a restriction to increase the computational power of the semi-simple splicing languages generated. Every initial string in splicing system is associated with an element of alternating group. The definition of this restricted splicing system is given.

Definition 12: Semi-Simple Splicing Systems Over Alternating Group, A_3
A semi-simple splicing system over alternating group is defined as $\kappa_{ss} = (V, A, R, A_n)$, where
V is an alphabet,
A is a finite subset of V,
$R = a\#1\$b\#1$ is a set of splicing rules where $a, b \in V$,
A_n is an alternating group.

Languages L generated by semi-simple splicing system over alternating group κ_{ss} is defined by $L(\kappa_{ss})$ =

$\{w \in V \mid [w, (\gamma)] \in \sigma (A)\}$. The definition of the threshold language of semi-simple splicing system over alternating group is given as follow.

Definition 13: Threshold language of semi-simple splicing system over alternating group
Let $L_s(\kappa_{ss}')$ be a language generated from semi-simple splicing system over alternating group, $\kappa_{ss}' = (V, A, R, A_n)$. The threshold (cut-points) are considered as sub-segments and discrete subsets of integer n. The threshold languages with respect to the thresholds is defined as:

$$L_s(\kappa', \star) = \{w \in T^* : (w, \gamma(w)) \in \sigma_s^*(A) \wedge \gamma(w) \star \ \omega\},$$

where $\star \in \{\in, \notin\}$ is called threshold mode.

An element of an alternating group is associated with the initial strings A and newly generated strings are considered valid if the computation of the associated alternating element is identity, (1). The simplified notation $A_n SSS(F_1, F_2)$ indicates the language generated from semi-simple splicing system over alternating group with axioms and rules in the type of (F_1, F_2) where $F_1, F_2 \in \{FIN, REG, CF, LIN, CS, RE\}$. Thus a lemma arises from the definition of the splicing system.

Lemma: $SSS(FIN, F) \subseteq A_n SSS(FIN, F)$ for all $F \in \{FIN, REG, CF, LIN, CS, RE\}$.

Proof:
Let $\kappa_{ss} = (V, A, R)$ be the splicing system generates the language $L(\kappa_{ss}) \in SSS(FIN, F)$ where F is the family of language, $F \in \{FIN, REG, CF, LIN, CS, RE\}$ and $A = x_1, x_2, \dots, x_n$ for $n \geq 1$. The splicing system over alternating group defined as $\kappa'_{ss} = (V, A', R, A_n)$ where the set of axioms is defined as

$$A' = \{[x_i, (\gamma(x_i))] | x_i \in A, 1 \leq i \leq n, \gamma \in A_n\}$$

for all $1 \leq i \leq n$. Thus

$$\sum_{i=1}^{n} \gamma = (1)$$

We defined the threshold language generated by κ_{ss} as $L_S(\kappa'_{ss}, \gamma)$ then it is not difficult to see that

$$L(\kappa_{ss}) = L_S(\kappa'_{ss}, \gamma) = A_n SSS(F).$$

4. Results
In this section, the results of languages generated by semi-simple splicing system are discussed. The examples for semi-simple splicing system with alternating group of order three are given to study the splicing languages generated for finite sets of strings and rules of splicing system.

Example 1: Semi-Simple Splicing System Over Alternating Group, A_3 on two identical string

Let the splicing system in the form of $\kappa_{ss} = (V, A, R, A_3)$ where
V is an alphabet
$A = \{a,b\}$
$R = \{r = b\#1\$a\#1\}$
$A_3 = \{aba, (123)\}$
The rule $r=b\#1\$a\#1$ splice the same initial strings *aba(123)* and form new strings. The process of the splicing operation is shown below.

Step 1: aba, the generated string is $w = \{[ab, ba], (123)(123)\} = \{ab^2a, (132)\}$.
Step 2: ab^2a and aba, the generated string is $w = \{[ab^2, ba], (132)(123)\} = \{ab^3a, (1)\}$.
Step 3: ab^3a and aba, the generated string is $w = \{[ab^3, ba], (1)(123)\} = \{ab^4a, (123)\}$.
Step 4: ab^4a and aba, the generated string is $w = \{[ab^4, ba], (123)(123)\} = \{ab^5a, (132)\}$.
Step 5: ab^5a and aba, the generated string is $w = \{[ab^5, ba], (132)(123)\} = \{ab^6a, (1)\}$.
Step 6: ab^6a and aba, the generated string is $w = \{[ab^6, ba], (1)(123)\} = \{ab^7a, (123)\}$.
Step 7: ab^7a and aba, the generated string is $w = \{[ab^7, ba], (123)(123)\} = \{ab^8a, (132)\}$.
Step 8: ab^8a and aba, the generated string is $w = \{[ab^8, ba], (132)(123)\} = \{ab^9a, (1)\}$.

The process of splicing continues and strings with identity element is considered valid. The newly generated strings with identity element are $\{[ab^3a, (1)], [ab^6a, (1)], [ab^9a, (1)], \ldots\}=\{ab^{3n}a, (1)\}$ where $n \geq 1$. A context-sensitive grammar generated from the splicing operation, thus the language generated is $L(\kappa_{ss}) = \{ab^{3n}a, n \geq 1\}$. The context-sensitive grammar generated from the splicing operation is shown.

$$G=([S,B],S,R)$$
$$S \rightarrow aBa$$
$$S \rightarrow bbbB/bbb$$

The derivation of language $L(\kappa_{ss})=\{ab^{3n}a, n \geq 1\}$ using the context-sensitive grammar above is in the following.

For $n = 1$, $S \rightarrow aBa \rightarrow abbba$
For $n = 2$, $S \rightarrow aBa \rightarrow abbbBa \rightarrow abbbbbba$
For $n = 3$, $S \rightarrow aBa \rightarrow abbbBa \rightarrow abbbbbbBa \rightarrow abbbbbbbbba$

The context-sensitive grammar generates splicing language $L(\kappa_{ss}) = \{ab^{3n}a, n \geq 1\}$ where the language generated is not a regular language. Example 1 generates language of $L(\kappa_{ss}) = CS - REG$ where regular grammar is not inclusive in context-sensitive grammar. Next is the example of semi-simple splicing system over alternating group of order three using two different initial strings.

Example 2: Semi-Simple Splicing System Over Alternating Group, A_3 using two different strings

Let the splicing system in the form of $\kappa_{ss} = (V, A, R, A_3)$ where
V is an alphabet
$A = \{a,b\}$
$R = \{r = a\#1\$b\#1\}$
$A_3 = \{[abb, (123)], [bab, (132)]\}$
The splicing operation on strings *abb(123)* and *bab(132)* using the rule, $r = a\#1\$b\#1$ generates new string w are shown below.

Step 1: abb and bab, the generated string is $w = \{[a, ab], (123)(132)\} = \{a^2b, (1)\}$.
Step 2: a^2b and aba, the generated string is $w = \{[a^2, ab], (1)(132)\} = \{a^3b, (132)\}$.
Step 3: a^3b and aba, the generated string is $w = \{[a^3, ab], (132)(132)\} = \{a^4b, (123)\}$.
Step 4: a^4b and aba, the generated string is $w = \{[a^4, ab], (123)(132)\} = \{a^5b, (1)\}$.
Step 5: a^5b and aba, the generated string is $w = \{[a^5, ab], (1)(132)\} = \{a^6b, (132)\}$.

Step 6: a^6b and aba, the generated string is $w = \{[a^6, ab], (132)(132)\} = \{a^7b, (123)\}$.
Step 7: a^7b and aba, the generated string is $w = \{[a^7, ab], (123)(132)\} = \{a^8b, (1)\}$.

The step continues and the strings generated with identity permutation are considered valid. The new strings with identity element generated are $\{[a^2b, (1)], [a^5b, (1)], [a^8b, (1)], \ldots\} = \{a^{3n-1}b, (1)\}$ when $n \geq 1$. The grammar of the splicing operation is a context-free grammar and the splicing language generated from this grammar is $L(\kappa_{ss}) = \{ab^{3n-1}a, n \geq 1\}$. The context-free grammar of the semi-simple splicing is

$$G = \{[S, A], S, R\}$$
$$S \rightarrow Ab$$
$$S \rightarrow aaA | aaaA | \lambda$$

The derivation of the splicing language $L(\kappa_{ss}) = \{ab^{3n-1}a, n \geq 1\}$ using the context-free grammar above as follows.

For $n = 1$, $S \rightarrow Ab \rightarrow aaAb \rightarrow aa\lambda b \rightarrow aab$
For $n = 2$, $S \rightarrow Ab \rightarrow aaAb \rightarrow aaaaaAb \rightarrow aaaaa\lambda b \rightarrow aaaaab$
For $n = 3$, $S \rightarrow Ab \rightarrow aaAb \rightarrow aaaaaAb \rightarrow aaaaaaaaAb \rightarrow aaaaaaaa\lambda b$

Semi-simple splicing system associated with an element of alternating generates splicing language $L(\kappa_{ss}) = \{ab^{3n-1}a, n \geq 1\}$ for a finite set of n where the language is classified as a context-free language in Chomsky hierarchy. The language generated is not identified as a regular language. Thus, the language generated is $L(\kappa_{ss})$ = CF – REG.

From the two examples shown, the splicing languages generated for semi-simple splicing system are context-sensitive and context-free language which is not regular. Since, the computational power of the splicing languages generated is higher than the computational power of regular language, associating an element of permutation increase the computational power of the splicing languages generated. Thus, the theorem for semi-simple splicing system over alternating group arise.

Theorem 1: REG = *SSS*(FIN) ⊆ A_3*SSS*(FIN)
Proof: From table 1, regular language is generated from finite semi-simple splicing system. Lemma shows that the language generated by finite semi-simple splicing system is a subset of the language generated by A_3*SSS*(FIN). When the threshold is defined as $\in A_n$, A_3*SSS*(FIN) generate regular language.

Theorem 2: *SSS*(FIN) ⊆ A_3*SSS*(FIN)

Proof: The hypothesis states that there is a language generated in A_3*SSS*(FIN) which is not generated in *SSS*(FIN). From Table 1 of splicing system, it proves that the language generated by finite semi-simple splicing system, *SSS*(FIN) is Regular. Since, the language $L(\kappa_{ss}, \in (1)) = \{ab^{3n-1}a, n \geq 1\}$ in Example 2 belong to A_3*SSS*(FIN) but not to *SSS*(FIN), the proper inclusion is obtained.

Theorem 3: A_3*SSS*(FIN) – CF ≠ Ø

Proof by contradiction: Suppose that the set of languages generated from semi-simple splicing system over alternating group without Context-Free Language are empty. The set is empty when there are no languages generated by A_3*SSS*(FIN) has higher computational power than Context-Free Languages. From Example 2, A_3*SSS*(FIN) generate a Context-Sensitive Language. Therefore, it contradicts that the set of languages generated from A_3*SSS*(FIN) contain no language with higher computational power than Context-Free Language. Hence, A_3*SSS*(FIN) is not an empty set.

5. Conclusion

This paper study the semi-simple splicing system when a restriction of an alternating element is associated with the initial strings. The splicing languages generated for finite splicing is a regular language. A restriction is used to increase the computational power of the languages generated for finite splicing. This paper uses an element of alternating group as a restriction to semi-simple splicing system. The languages generated shows that alternating group as a restriction to the splicing system could increase the computational power of the splicing languages to non-regular language.

6. Acknowledgements

This research is supported by FRGS-RACER (600-IRMI/FRGS-RACER 5/3 (050/2019)), Ministry of Higher Education and Universiti Teknologi MARA Malaysia.

References

[1] Head, T. (1987). Formal language theory and DNA: An analysis of the generative capacity of specific recombinant behaviors. *Bulletin of Mathematical Biology*, *49*(6), 737–759.
[2] Watson, J., & Crick, F. (1953, April). A structure for deoxyribose nucleic acid. *Nature*, *171*, 737.
[3] Păun, G. (1996). On the splicing operation. *Discrete Applied Mathematics*, *70*(1), 57–79.
[4] Pixton, D. (1996). Regularity of splicing languages. *Discrete Applied Mathematics*, *69*(1–2), 101–124.
[5] Yusof, Y., Sarmin, N. H., Goode, T. E., Mahmud, M., & Heng, F. W. (2011). An extension of DNA splicing system. *Proceedings - 2011 6th International Conference on Bio-Inspired Computing: Theories and Applications, BIC-TA 2011*, 246–248.
[6] Chomsky, N. (1956). Three Models For The Description of Language. *IRE Transactions on Information Theory*, *2*(3), 113–124.
[7] Mateescu, A., Păun, G., Rozenberg, G., & Salomaa, A. (1998). Simple splicing systems. *Discrete Applied Mathematics*, *84*(1–3), 145–163.
[8] Laun, T. E. G. (1999). Constants And Splicing Systems. *(Doctoral Dissertation, State University of New York at Binghamton, Mathematical Sciences Department*, (September 2000).
[9] Turaev, S., Selvarajoo, M., Selamat, M. H., Sarmin, N. H., & Fong, W. H. (2013). Probabilistic Splicing Systems. In *Advanced Methods for Computational Collective Intelligence* (pp. 259–268). Berlin, Heidelberg: Springer.
[10] Selvarajoo, M., Wan Heng, F., Haniza Sarmin, N., & Turaev, S. (2016). Computational Power of Probablistic Simple One-Sided Sticker languages. *Jurnal of Mathematics & Computing Sciences*, *2*(2), 1–9.
[11] Selvarajoo, M., Wan Heng, F., Haniza Sarmin, N., & Turaev, S. (2017). Probabilistic simple sticker system. *AIP Conference Proceedings*, *1830*(1), 020057.
[12] Yusof, Y., Sarmin, N. H., Fong, W. H., Goode, T. E., & Ahmad, M. A. (2013). An analysis of four variants of splicing system. *AIP Conference Proceedings*, *1522*(1), 888–895.
[13] Hamzah, N. Z. A., Sebry, N. ., Fong, W. H., Sarmin, N. H., & Turaev, S. (2012). Splicing Systems over Permutation Groups of Length Two. *Malaysian Journal of Fundamental and Applied Sciences*, *8*(2), 83–88.
[14] Fraleigh, J. B. (2003). *A First Course In Abstract Algebra* (7th ed.). Pearson Education Limited.
[15] Jäger, G., & Rogers, J. (2012). Formal language theory: Refining the Chomsky hierarchy. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *367*(1598), 1956–1970.
[16] Yusof, Y., Sarmin, N. H., Fong, W. H., & Karimi, F. (2010). Some Relations on Different Types of Splicing Systems. *Journal of Fundamental Sciences*, *6*(2), 143–147.
[17] Selvarajoo, M., Wan Heng, F., Haniza Sarmin, N., & Turaev, S. (2013). Probabilistic semi-simple splicing system and its characteristics. *Jurnal Teknologi (Sciences and Engineering)*, *62*(3), 21–26.