SEMARAK ILMU PUBLISHING
202103268166(003316878-P)

Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

Classification of *n*-th Order Limit Language in Formal Language Classes

Siti Hajar Mohd Khairuddin[1], Muhammad Azrin Ahmad[1,*], Mohd Sham Mohamad[1]

[1] Pusat Sains Matematik, Universiti Malaysia Pahang Al-Sultan Abdullah, Lebuh Persiaran Tun Khalil Yaakob, 26300 Gambang, Kuantan, Pahang, Malaysia

ARTICLE INFO

Article history:
Received 25 September 2023
Received in revised form 17 November 2023
Accepted 27 February 2024
Available online 5 May 2024

Keywords:
Chomsky hierarchy; Formal language theory; *n*-th order limit language; DNA splicing system; Formal language classes

ABSTRACT

The study of splicing systems and their language has grown rapidly since Paun developed a splicing system known as a regular splicing scheme that produces a regular language. Since then, the researchers have been eager to classify the splicing language into certain classes in the Chomsky hierarchy, such as context-free language, context-sensitive language and recursive enumerable language. Previously, the study on the *n*-th order limit language was conducted from the biological perspective to the limit language produced. Still, no research has been done from the generation of language point of view. This research presents a generalization on the type of classes of the formal language, the *n*-th order limit language. The cases to obtain the *n*-th order limit language are revisited and used to obtain the types of language classes according to the Chomsky hierarchy produced by the *n*-th order limit language.

1. Introduction

Deoxyribonucleic acid (DNA) is a complex molecule that contains the genetic information responsible for the development, functioning, and reproduction of all living organisms [1]. Its structure consists of four different nitrogenous bases denoted by A, G, C and T, representing adenine, guanine, cytosine, and thymine, respectively [2]. The application of DNA knowledge involves the relation between formal language theory and informational molecular [3]. On the other hand, formal language theory is a branch of mathematics and computer science that studies the properties of formal languages, which are the sets of strings of symbols that follow certain rules or grammatical structures. Formal languages play a crucial role in computer science since they represent the basis of programming languages and are used to define the syntax and semantics of computer programs [4].

While seemingly unrelated, DNA and formal language theory share some surprising connections. One of the examples of studies relating to formal language theory and molecular biology is the splicing system [18]. In a splicing system, strings of symbols are transformed according to a set of rules, or splicing rules, that specify which portions of the string can be removed or joined together.

* *Corresponding author.*
E-mail address: azrin@umpsa.edu.my

https://doi.org/10.37934/araset.44.2.110

Besides, splicing systems are a class of formal language systems originally introduced in the context of molecular biology perspectives involving cutting and pasting the DNA molecule in the presence of restriction enzymes and ligase. The splicing system includes Head [3], Paun [5], Pixton [6], Goode- Pixton [7], Yusof-Goode [8] and fuzzy splicing systems [9].

In this research, the splicing language is discussed from the aspect of language generation. The splicing language is generated by certain types of grammar, according to Chomsky. Previously, Paun has researched the splicing system that produces a regular language called a regular splicing scheme. It focuses on the type of language produced after splicing, in which many further language theoretical problems can be formulated for the splicing operations. Other than that, the regular splicing scheme has been developed, and the language produced by this kind of splicing system is a regular language. Then, the research was developed, and an extended H-system was established [5]. The writing notation is changed to input several variables in the splicing notation. Since the research focused on the theoretical part of the splicing operation, in the same year, Paun developed the test tube system [10]. Test tube systems not only consider the theoretical part but also cover the biological part. In addition, the test tube system is proven to produce recursively enumerable language. Lastly, Paun has developed a P-system, which does not focus on the DNA itself but the membrane of the cells [11].

In the next section, the preliminaries of this research are presented.

2. Preliminaries

In this section, some basic definitions related to formal language theory and the *n*-th order limit language are given.

2.1 Definition 1: Alphabets [12]

The alphabet, denoted by the symbol, Σ is a fundamental concept in formal language theory as it serves as the basic set of symbols used to construct strings in a language. It provides the foundation for defining the grammar rules and syntax of a language.

2.2 Definition 2: Strings [12]

A string is a finite sequence of symbols (or characters) chosen from a specified alphabet. It denoted as $|w|$.

2.3 Definition 3: Language [13]

Language, L is defined as a set of possibly finite set of strings over some finite alphabet. We define Σ^* as the set of all possible strings over some alphabet Σ. Thus $L \subseteq \Sigma^*$. The set of all possible languages over some alphabet Σ is the set of all possible subsets of Σ^*.

Next, the *n*-th order limit language is presented below.

2.4 Definition 4: n-th Order Limit Language [7]

Let L_{n-1} be the set of second-order limit language of L , the set L_n of *n*-th order limit language of L to be the set of the first-order limit language of L_{n-1}. We obtain L_n from $L_{n-1}$by deleting the language that are transient in L_{n-1}.

The original concept is then modified by Khairuddin *et al.,* who note that the rules used in the splicing system define the order of the limit language, but rules must all be of the same length [14,15]. Goode and Pixton previously said that previous orders of the limit language are different from the current order limit language in order to define the *n*-th order limit language [19]. The improve version of the definition is given as follows.

2.5 Definition 5: n-th Order Limit Language from Rules Perspective [17]

The *n*-th order limit language is defined by the number of rules that act on each crossing sites, x, where the set of rules are different from each other. Note that the rules must have the same length of crossing sites. A splicing language is called *n*-th order limit language, denoted by $L_n(S)$ if the set of string produce in $L_n(S)$ is distinct from the set of n strings of $L_1(S), L_2(S), \dots, L_n(S)$ such that $\bigcap_{i=1}^{n} L_i = \emptyset$ and $L_1(S) \not\subset L_2(S) \not\subset \cdots \not\subset L_n(S)$.

2.6 Definition 6: Grammar [12]

Grammar is the way to characterize a language L, a way to list out the strings of Σ^* are in L and which are not. Grammar $G = (V, T, S, P)$ where V is non-terminal, T is the terminal symbols, S is the start symbols and P is the production. According to Chomsky, grammar is classified into 4 types of grammar which are type 0, type 1, type 2 and type 3 which indicates unrestricted grammar, context-sensitive grammar, context-free and regular grammar respectively. Figure 1 below shows the hierarchy of the grammar.

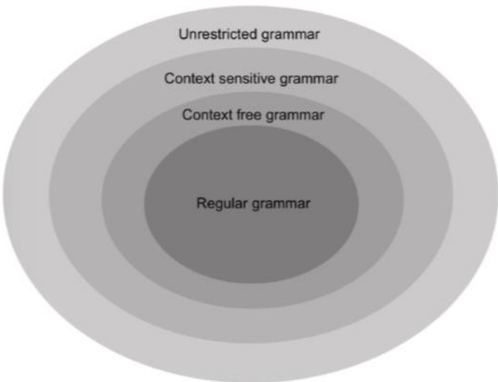

Fig. 1. Chomsky Classification of Grammars

2.6.1 Type-0 (unrestricted grammar)

$\alpha \to \beta, \alpha, \beta \in (V \cup T)^*$.

Based on this statement, α and β belong to any combination of terminal and non-terminal with no restrictions.

2.6.2 Type-1 (context-sensitive grammar)

$\alpha \to \beta, |\alpha| \leq |\beta|$.

Based on this statement, the length of α should be less than or equal the length of β , in which α and β belongs to any combination of terminal and non-terminal, respectively.

2.6.3 Type-2 (context-free grammar)

$\alpha \rightarrow \beta, \alpha \in V$.

This statement indicates that α belongs to only one non-terminal.

2.6.4 Type-3 (regular grammar)

$\alpha \rightarrow \beta, \alpha \in V$.

Based on this statement, α belongs to only one non-terminal and β belongs to the terminal only, the terminal followed by non-terminal for the right linear grammar cases and non-terminal followed by the terminal for the left linear grammar cases.

2.7 Definition 7: Chomsky Hierarchy [16,20]

The Chomsky hierarchy is a classification of formal grammar and languages, named after the linguist Noam Chomsky. The hierarchy consists of four levels, each of which describes a subset of all possible languages:

Table 1
Type of Languages

Type of Language	Explanation
Regular, *REG*	These are the languages that can be described by regular expressions, regular grammar or finite-state machines.
Context-free, *CF*	These are languages that can be described by context-free grammars, which are more powerful than regular expressions. Context-free languages are used to describe many programming languages and natural languages.
Context-sensitive, *CS*	These are languages that can be described by context-sensitive grammars, which are more powerful than context-free grammars. Context-sensitive languages are used to describe more complex languages, such as natural languages.
Recursively enumerable, *RE*	These are languages that can be described by Turing machines, which are the most powerful type of formal language. Recursively enumerable languages are used to describe any computable language, but they can also describe many languages that are not computable.

The following language is generated by a specific grammar as tabulated in Table 2.

Table 2
The Type Finite Languages Generated by Grammar

Type of Language	Explanation
REG	Regular grammar, Finite state automaton and Regular expression
CF	Context-free grammar
CS	Context-sensitive
RE	Recursively enumerable grammar

The following is the strict inclusions hold

$FIN \subset REG \subset CF \subset CS \subset RE$

Therefore, the statement implies that the set of finite languages (FIN) is a proper subset of the set of regular languages (REG) if every element in a set of finite languages (FIN) is also in a set of regular languages (REG). Still, the set of finite languages (FIN) does not equal the set of regular languages (REG). The cycle continues for CF, CS and RE languages.

2.8 Definition 8: Extended H-system [11]

Let extended Head splicing system be $\gamma = (V, T, A, R)$ which consist of V is a set of alphabets, T is a terminal symbol, A is a set of axioms and R is a set of rules. An example is given below to illustrate the extended H-system.

Example $\gamma = (\{a, b, c, d, e\}\{a, b, c, d\}, \{cabd, caebd\}, \{c\#ca\#ebd, ce\#\$b\#d\})$

When the first rule applies to the axioms, the language produced is shown below.

$(c|a^n b^n d, ca|ebd) \rightarrow (cebd, ca(a^{n+1} b^n d)$

Then, the resulting axiom from the first splicing will splice by using the second rule. It is presented below.

$(ce|bd, ca^{n+1} bn|d) \rightarrow (ced, ca^{n+1} b^{n+1} d)$

The language produced by the extended H splicing system is $L(\gamma) = \{ca^n b^n d | n \geq 1\}$. In the next section, the relation of formal language theory and the *n*-th order limit language is presented.

3. Results

The generalization of the *n*-th order limit language is previously proved by the cases given in [15]. In this section, two theorems will be presented for the *n*-th order limit language.

The theorem shows what type of language is produced by the *n*-th order limit language. According to Chomsky, there are few types of language in the family of finite languages. There is regular, linear, context-free, context- sensitive and recursive enumerable language.

i. **Theorem 1** If $n = 1$ in the *n*-th order limit language consisting of a finite set of axioms and a finite set of rules, then the type of language classes is regular. It can also be represented as $\forall Ln(\gamma), EH(FIN, FIN) \in REG \Leftrightarrow n = 1$.
ii. **Proof:** Case 1 is considered to prove this theorem. Since only a rule is considered in the splicing process, this case is used [15]. The following pattern can be obtained such as $L(S) = \{\theta \ldots \varphi, \theta \ldots \theta', \varphi' \ldots \varphi\}$. In this study, another method of writing is proposed such as:

$$L(S) = \{\theta \ldots \varphi, \theta \ldots \theta', \varphi' \ldots \varphi\} \Rightarrow L(\gamma) = \{g \ldots h, g \ldots g, h \ldots h\}$$

Crossing sites of the axiom is represent as $(\ldots)$ in the above case. Then, the language can also be written as follows.

$L(\gamma) = \{gh, gg, hh\}$, where $g \in \theta, \theta'$ and $h \in \varphi, \varphi'$

3.1 Case 1

Let $\gamma = (\{a, b\}, \{a, b\}, \{ab\}, \{a\# b\})$be an extended H splicing system that contains one axiom and a rule.

After applying the rule, the axiom will be separated into two parts.

$a|b$

Then, the splicing language produced from the splicing system above can be rewritten as follow.

$L(\gamma) = \sigma * (A) \cap T = \{ab, aa, bb\}$

Then, since a rule is resorted in the extended H splicing system, the first-order limit language is presented below.

$L(\gamma) = \{aa, bb\}$

The language produced is regular because it results from regular grammar, as presented below.

The grammar $G_1 = (\{S, A, B\}\{a, b\}, \{S\}, P)$ which P given as $P = \{S \to a|Aa|b|Bb, A \to a \; and \; B \to b\}$. The derivation as follows.

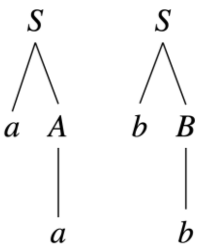

Fig. 2. The Derivation Tree of G_1

The derivation of G_1 can be rewritten as below.

$S \Rightarrow aA \Rightarrow aa \; or \; S \Rightarrow bB \Rightarrow bb$

From here, the language generated by $G_1 = \{aa, bb\}$ is a regular language generated by regular grammar since it obeys the definition of regular grammar. In conclusion, Theorem 1 shows the *n*-th order limit language produced by the extended H system with a finite number of initial strings, in which a rule will produce regular language. ■

Then, Theorem 2 is presented for Case 2 until Case 6.

i. **Theorem 2** If $n \geq 2$ in the *n*-th order limit language consists of a finite set of axioms and a finite set of rules, then the type of language classes is context-free. It can also be represented as $\forall Ln(S), EH(FIN, FIN) \in CF - REG \iff n \geq 2$.

ii. **Proof**: Case 2 until Case 6 are considered to prove this theorem. Let $\gamma = (V, T, A, R)$ be an extended H splicing system containing finite rules and axioms.

3.2 Case 2

Let $\gamma = (\{c, x_1, d, g, x_2, h\}, \{c, h\}, \{cx_1dgx_2h\}, \{c\#x_1d\$g\#x_2h\})$ be an extended H splicing system containing one axiom and two rules. The language generated from the splicing system is shown below.

After applying the rule, the axiom will be separated into three parts.

$c|x_1dg|x_2h$

The following axiom is obtained by $c \rightarrow a, x_2dg \rightarrow b$ and $x_2h \rightarrow c$. Then, the splicing language produced from the above splicing system can be rewritten as follows.

$$L(\gamma) = \sigma^*(A) \cap T = \begin{Bmatrix} ac, aa, cc, abc, aba, cbc, abbc, abba, cbbc, \\ abbbc, abbba, cbbbc, ab^nc, ab^na, cb^n\ c \end{Bmatrix} \quad (1)$$

From Eq. (1), the splicing language is presented in the form of limit language. Since two rules are used in the splicing system, the generalization of the second-order limit language is presented below.

$L2(\gamma) = \{ab^nc, ab^na, cb^nc\}$

The language produced is context-free because it is generated from context-free grammar. Based on the definition of context-free language, some context-free language is also a regular language. To obey the definition of the *n*-th order limit language, the regular language is discarded from the limit language produced. It is because based on the *n*-th order limit language, the word that is transient from the previous order of limit language is deleted. Thus, the words or axioms that have been deleted are regular axioms, which can be said as regular language. Then, the grammar is presented below.

The grammar $G_2 = (\{S, X, a, b, c\}\{a, b, c\}, \{S\}, P)$, where P is given by $P = \{S \rightarrow aXc|aXa|cXc|$ and $X \rightarrow b|\varepsilon\}$. This is presented by the derivation tree as follows.

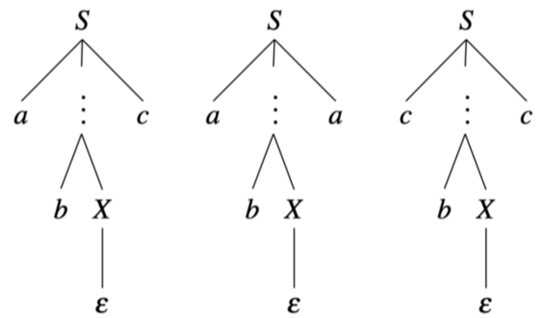

Fig. 3. The Derivation Tree of G_2

The derivation of G_2 can be rewritten as below.
$S \Rightarrow aXbc \Rightarrow abbc, S \Rightarrow aXba \Rightarrow abba\ or\ S \Rightarrow cXbc \Rightarrow cbbc$

From here, the language generated by grammar, $G_2 = \{ab^*c, ab^*a, cb^*c\}$, is a context-free language since it is generated by a regular and context-free grammar.

Next, Case 3 is presented.

3.3 Case 3

Let $\gamma = (\{c, x_1, d, g, x_2, h, k, x_3, l\}, \{c, l\}, \{cx_1dgx_2hkx_3l\}, \{c\# \, x_1d\$g\#x_2h\$k\#x_3l\})$ be an extended H splicing system containing one axiom and three rules. The language generated from the splicing system is shown below.

After applying the rule, the axiom will be separated into four parts.

$c|x_1dg|x_2hk|x_3l$

The following axiom is named by the following $c \rightarrow a, x_1dg \rightarrow b_1, x_2hk \rightarrow b_2$ and $x_3l \rightarrow c$. Then, the language obtained from the splicing system above can be rewritten as follows.

$$L(\gamma) = \sigma^*(A) \cap T = \left\{ \begin{matrix} ac, aa, cc, ab_1c, ab_1a, cb_1c, ab_2c, ab_2a, \\ cb_2czab_1b_2c, ab_1b_2a, cb_1b_2c, \\ ab_1b_2b_1c, ab_1b_2b_1a, cb_1\, b_2\, b_1\, c, \\ ab_1^n\, b_2^n\, c, ab_1^n\, b_2^n\, a, cb_1^n\, b_2^n\, c \end{matrix} \right\} \tag{2}$$

From Eq. (2), the splicing language is presented in the form of limit language. Since three rules are used in the splicing system, the generalization of the third-order limit language is presented below.

$L3(\gamma) = \{ab_1^n\, b_2^n\, c, ab_1^n\, b_2^n\, a, cb_1^n\, b_2^n\, c\}$.

The language produced is context-free because the grammar generated in the language is regular and context-free grammar as presented below.

The grammar $G_3 = (\{S, X, a, b_1, b_2, c\}\{a, b_1, b_2, c\}, \{S\}, P)$, where P is given as $P = \{S \rightarrow aXc|aXa|cXc$ and $X \rightarrow b_1|b_2|\varepsilon\}$, given by the derivation tree below.

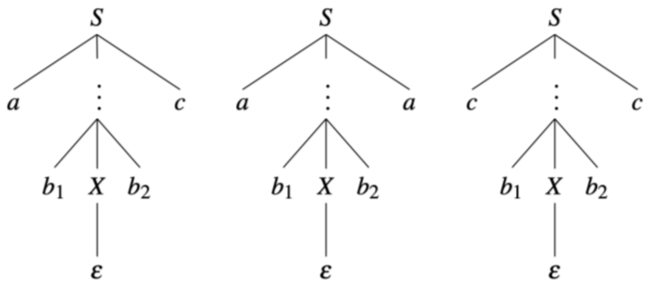

Fig. 4. The Derivation Tree of $G3$

The derivation of G_3 can be rewritten as below.

$S \Rightarrow aXb_1c \Rightarrow ab_2b_1c, S \Rightarrow aXb_2a \Rightarrow ab_1b_2a \; or \; S \Rightarrow cXb_1c \Rightarrow cb_1b_2c$

From here, the language generated by grammar, $G_3 = \{ab_1^*b_2^*c, ab_1^*b_2^*a, cbb_1^*b_2^*c\}$ is a context-free language since it is generated by a context-free grammar. We can see that Case 2 and Case 3 produce context-free language. The criteria of context-free language are obeyed for both cases since the length of α should be less than or equal to the length of β as well as α and β belongs to any combination of terminal and non-terminal.

The grammar pattern produced is similar for Case 4 until Case 6. Thus, it is evident that the only part that changes in the language is the combination of the string in $b_1, b_2, \ldots, b_n$. In conclusion, Theorem 2 shows that all the *n*-th order limit language produced from the extended H system using a finite number of initial strings and a finite set of rules will produce context-free language. ∎

4. Conclusions

Two theorems are presented to prove the type of language classes produced by the *n*-th order limit language. In Theorem 1, regular language is produced by splicing one axiom and one rule. Hence, a regular language is obtained for all cases of first-order limit language. However, in Theorem 2, context-free language is produced from the splicing that includes more than two rules. Moreover, second-order to *n*-th order limit language produced context-free language. As a conclusion, $L_1(\gamma) \in REG$ and $\forall L_i(\gamma)$, where $i = 2, \ldots, n \in CF - REG$.

Acknowledgement
The authors would like to thank the Ministry of Higher Education for providing financial support under the Fundamental Research Grant Scheme (FRGS), FRGS/1/2022/STG06/UMP/02/4 (University reference RDU220131).

References

[1] Watson, James D. "The Double Helix: A Personal Account of the Discovery of the Structure of DNA." (2001). https://doi.org/10.1063/1.3035117
[2] Watson, James D., and Francis HC Crick. "Molecular structure of nucleic acids: a structure for deoxyribose nucleic acid." *Nature* 171, no. 4356 (1953): 737-738. https://doi.org/10.1038/171737a0
[3] Head, Tom. "Formal language theory and DNA: an analysis of the generative capacity of specific recombinant behaviors." *Bulletin of mathematical biology* 49 (1987): 737-759. https://doi.org/10.1016/S0092-8240(87)90018-8
[4] Ruohonen, K. (2007). "Formal Languages." In Advanced Textbooks in Control and Signal Processing, 33-40. https://doi.org/10.1007/978-1-84628-877-7_3
[5] Păun, Gheorghe, Grzegorz Rozenberg, and Arto Salomaa. "Computing by splicing." *Theoretical Computer Science* 168, no. 2 (1996): 321-336. https://doi.org/10.1016/S0304-3975(96)00082-5
[6] Pixton, Dennis. "Regularity of splicing languages." *Discrete Applied Mathematics* 69, no. 1-2 (1996): 101-124. https://doi.org/10.1016/0166-218X(95)00079-7
[7] Goode, Elizabeth, and Dennis Pixton. "Splicing to the Limit." In *Aspects of Molecular Computing: Essays Dedicated to Tom Head, on the Occasion of His 70th Birthday*, pp. 189-201. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003. https://doi.org/10.1007/978-3-540-24635-0_13
[8] Yusof, Yuhani. "Dna splicing system inspired by bio molecular operations." PhD diss., Universiti Teknologi Malaysia, 2012.
[9] Karimi, Fariba, Sherzod Turaev, Nor Haniza Sarmin, and Wan Heng Fong. "Fuzzy splicing systems." In *Computational Collective Intelligence. Technologies and Applications: 6th International Conference, ICCCI 2014, Seoul, Korea, September 24-26, 2014. Proceedings 6*, pp. 20-29. Springer International Publishing, 2014. https://doi.org/10.1007/978-3-319-11289-3_3
[10] Varjú, E. Csuhaj, Lila Kari, and Gh Paun. "Test tube distributed systems based on splicing." *Computing and Informatics* 28, no. 1 (2012). https://doi.org/10.1007/978-3-540-24635-0_10
[11] Păun, Gheorghe. "Regular extended H systems are computationally universal." *Journal of Automata, Languages and Combinatorics* 1, no. 1 (1996): 27-36. https://doi.org/10.1515/9783110752304-002
[12] Peter, Linz. "An introduction to formal languages and automata." (2001). https://doi.org/10.1142/9789812817495_0002
[13] Rozenberg, Grzegorz, and Arto Salomaa, eds. *Handbook of Formal Languages: Volume 3 Beyond Words*. Springer Science & Business Media, 2012. https://doi.org/10.1007/978-3-642-59126-6

[14] Mohd Khairuddin, S. H. "Modelling of *n*-th Order Limit Language Using Automata in DNA Splicing System." Master's thesis, Universiti Malaysia Pahang, 2022.

[15] M. Khairuddin, S. H., Ahmad, M. A. and Adzhar, N. An Introduction to n-th Order Limit Language." Engineering Letters, 30 (2022).

[16] Seuren, Pieter AM. *From Whorf to Montague: Explorations in the theory of language*. OUP Oxford, 2013. https://doi.org/10.1093/acprof:oso/9780199682195.001.0001

[17] Mohd Khairuddin, S. H., Ahmad, M. A. and Adzhar, N."Various Regulations on the Existence of *n*-th Order Limit Language." *Jurnal Teknologi* 84, no. 3 (2022): 17-23. https://doi.org/10.11113/jurnalteknologi.v84.17845

[18] Jamil, Anwardi, Amirul Ridzuan Abu Bakar, and Nurul Husna Khairuddin. "Structural Elucidation of the Masp1 and Masp2 Protein from Nephila Pilipes Web Via Bioinformatics Approaches." *Journal of Advanced Research in Applied Sciences and Engineering Technology* 30, no. 1 (2023): 132-144. https://doi.org/10.37934/araset.30.1.132144

[19] Mohd Khairuddin, S. H., Ahmad, M. A. and Adzhar, N. "Splicing System in Automata Theory: A Review." In *Journal of Physics: Conference Series*, vol. 1366, no. 1, p. 012066. IOP Publishing, 2019. https://doi.org/10.1088/1742-6596/1366/1/012066

[20] Mateescu, Alexandru, and Arto Salomaa. "Formal languages: an introduction and a synopsis." *Handbook of Formal Languages: Volume 1 Word, Language, Grammar* (1997): 1-39. https://doi.org/10.1007/978-3-642-59136-5_1