Closure properties of bonded sequential insertion-deletion systems

To cite this article: Wan Heng Fong *et al* 2021 *J. Phys.: Conf. Ser.* **1988** 012075

Closure properties of bonded sequential insertion-deletion systems

Wan Heng Fong[1], Ahmad Firdaus Yosman[2], Hazzirah Izzati Mat Hassim[3]

[1,2,3] Department of Mathematical Sciences, Faculty of Science, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia

E-mail: fwh@utm.my[1], afirdaus49@graduate.utm.my[2], hazzirah@utm.my[3]

Abstract. Through the years, formal language theory has evolved through continual interdisciplinary work in theoretical computer science, discrete mathematics and molecular biology. The combination of these areas resulted in the birth of DNA computing. Here, language generating devices that usually considered any set of letters have taken on extra restrictions or modified constructs to simulate the behavior of recombinant DNA. A type of these devices is an insertion-deletion system, where the operations of insertion and deletion of a word have been combined in a single construct. Upon appending integers to both sides of the letters in a word, bonded insertion-deletion systems were introduced to accurately depict chemical bonds in chemical compounds. Previously, it has been shown that bonded sequential insertion-deletion systems could generate up to recursively enumerable languages. However, the closure properties of these systems have yet to be determined. In this paper, it is shown that bonded sequential insertion-deletion systems are closed under union, concatenation, concatenation closure, λ-free concatenation closure, substitution and intersection with regular languages. Hence, the family of languages generated by bonded sequential insertion-deletion systems is shown to be a full abstract family of languages.

1. Introduction

In formal language theory, language generating devices are constructed by subjecting sets of elements to predetermined rules of production. This way, a desired form of strings of elements called words can be generated to form a language. A set of languages that share the same properties is called a family. Usually, the family of languages are classified according to their generative power, which goes from unrestricted (Type-0) to regular (Type-3) [1]. This classification according to generative power creates the Chomsky hierarchy, giving an overview of the computational capability of each family of languages.

On the other hand, families can also be classified according to their closure properties. If all the languages in a family are closed under a specific operation, we say that the whole family is closed under that operation. Otherwise, the family is not closed even if some of the languages in that family are. The study of closure properties in formal language theory is similar to that of set theory, which is one of the key components to this field, after all. The closure properties of the families in the aforementioned Chomsky hierarchy have been presented in [2]. Besides that, the closure properties of developmental systems have also been explored in [3, 4] while the closure properties of some DNA-inspired language generating devices have been presented in [5–7].

In this research, the closure properties of the family of bonded sequential insertion-deletion systems under certain operations are determined.

This paper is organized as follows: Section 1 provides the introduction to the paper with a brief background of the research. Section 2 provides the preliminaries pertaining to the research, including some literature review and important notations, definitions, and theorems. The main results of the research are presented in Section 3, along with discussions. Lastly, Section 4 serves as the conclusion to the paper.

2. Preliminaries

In this section, only a few important notations, definitions, and theorems are presented. For supplementary and required reading, the reader may refer to [2, 8, 9] among others.

To begin with, an inclusion of a set A within a set B is denoted by $A \subseteq B$, and the proper inclusion by $A \subset B$. A set of symbols or letters is called an *alphabet*, Σ, while the set of words generated by operations on those letters is denoted by Σ^*. A *language* L over an alphabet Σ is a subset of the set Σ^*. Furthermore, a *family* of languages $\mathcal{L}$ is a set of languages. In addition, the empty word is denoted by λ. For any word w, the length of the word is denoted by $|w|$, where the length of the empty word is 0.

As mentioned previously in Section 1, the Chomsky hierarchy describes the inclusion of one family within another, depicting the generative power of the formal grammars that generate the families. The Chomsky hierarchy is presented in Theorem 2.1.

Theorem 2.1 [9] (The Chomsky hierarchy) The following strict inclusions hold:

$$\mathcal{L}(\text{REG}) \subset \mathcal{L}(\text{CF}) \subset \mathcal{L}(\text{CS}) \subset \mathcal{L}(\text{RE}).$$

Note that $\mathcal{L}(\text{REG}), \mathcal{L}(\text{CF}), \mathcal{L}(\text{CS}), \mathcal{L}(\text{RE})$ denote the families of regular languages, context-free languages, context sensitive languages and recursively enumerable languages, respectively.

In this research, we would like to determine the closure properties of $\mathcal{L}(\text{bSINSDEL})$ under specific operations. By doing so, we are able to determine whether it is a full abstract family of languages (full AFL). To that end, we first require the definition of closure, provided in Definition 2.1.

Definition 2.1 [2] A family of languages $\mathcal{L}$ is *closed* under a certain operation Ω if and only if the language L' obtained whenever Ω is applied to an $L \in \mathcal{L}$ belongs to the same family, such that $L' \in \mathcal{L}$.

Next, the formal definitions of the operations involved in this research are presented in the following.

Definition 2.2 [2] Let L_1 and L_2 be two languages.

(i) The *union* of L_1 and L_2 is defined as

$$L_1 \cup L_2 = \{P \mid P \in L_1 \text{ or } P \in L_2\}.$$

(ii) The *intersection* of L_1 and L_2 is defined as

$$L_1 \cap L_2 = \{P \mid P \in L_1 \text{ and } P \in L_2\}.$$

(iii) The *concatenation* of L_1 and L_2 is defined as

$$L_1L_2 = \{P_1P_2 \mid P_1 \in L_1 \text{ and } P_2 \in L_2\}.$$

Consequently, $L^i, i \geq 0$ is the language obtained by concatenating i copies of L.

(iv) The *concatenation closure* of L, denoted by L^* is defined as the union of all powers of L, such that

$$L^* = \bigcup_{i=0}^{\infty} L^i, \text{ where } L^0 = \{\lambda\}.$$

Additionally, the $\lambda-$*free concatenation closure* of L, L^+ is the union of all positive powers of L, such that

$$L^+ = \bigcup_{i=1}^{\infty} L^i.$$

Definition 2.3 [2] Let Σ be an alphabet and Σ^* the set of all words over the alphabet Σ. For each letter a of an alphabet Σ, let $\sigma(a)$ be a language over an alphabet Σ_a. Furthermore, let

$$\sigma(\lambda) = \lambda, \quad \sigma(PQ) = \sigma(P)\sigma(Q), \quad \text{for } P, Q \in \Sigma^*.$$

The mapping σ from Σ^* into $2^{\Sigma'}$, where Σ' is the union of all alphabets of Σ_a, is called a *substitution*. For a language L over Σ,

$$\sigma(L) = \{Q \mid Q \in \sigma(P) \text{ for some } P \in L\}.$$

From there, a *homomorphism* is defined as a substitution where each $\sigma(a)$ consists of only one word.

Now, we shall provide the definition for AFL and full AFL.

Definition 2.4 [2] A family of languages $\mathcal{L}$ is termed an *abstract family of langauges* (AFL) if and only if it contains a nonempty language, and is closed under union, λ-free concatenation closure, λ-free homomorphism, inverse homomorphism, and intersection with regular languages. An AFL is termed *full* if and only if it is closed under arbitrary homomorphism.

The systems involved in this research involve the bonding alphabet, which is made up of letters that have been appended with integers on each of its sides. These integers serve as *bonds* so that words can only be formed if the bonds between two letters fit. The bonding alphabet was introduced by Fong *et al.* in [10], where the concept of bonded systems was first brought to light. These systems, which now consist of a slew of variants for both insertion and deletion systems [6, 11–13], were introduced to model the formation/deformation of chemical bonds of DNA molecules during DNA recombination.

The mechanism of the bonded alphabet is explained in more detail in Definition 2.5.

Definition 2.5 [10] Let $\mathbb{Z}$ be the set of integers as well as

$$\mathbb{Z}_0^- = \{\ldots, -2, -1, 0\} \quad \text{and} \quad \mathbb{Z}_0^+ = \{0, 1, 2, \ldots\}.$$

Let Σ be an alphabet. Then, the set $\mathcal{B}_\Sigma = \mathbb{Z}_0^+ \times \Sigma \times \mathbb{Z}_0^-$ is a *bonding alphabet* over Σ. An element $(i, a, -j)$ of $\mathcal{B}_\Sigma$ is called a *letter* a with left bond i and right bond $-j$. To simplify the presentation, $[_i a_{-j}]$ is written instead of $(i, a, -j)$ for a letter a from $\mathcal{B}_\Sigma$.
Let

$$w = [_{i_0} a_{1 i_1}][_{i_2} a_{2 i_3}][_{i_4} a_{3 i_5}] \cdots [_{i_{2n-2}} a_{n i_{2n-1}}]$$

be a non-empty sequence of letters from $\mathcal{B}_\Sigma$, where n is an integer. The sequence w is said to be *well-formed* if all bonds fit, i. e., $i_{2j-1} + i_{2j} = 0$, for $1 \leq j \leq n - 1$. If additionally, $i_0 + i_{2n-1} = 0$ holds, then w is said to be a *balanced word.* If $i_0 + i_{2n-1} \neq 0$, then the word is said to be *unbalanced.* Moreover, a word is *neutral* if $i_0 = i_{2n-1} = 0$.
For a well-formed word

$$w = [_{i_0} a_{1 -i_1}][_{i_1} a_{2 -i_2}][_{i_2} a_{3 -i_3}] \cdots [_{i_{n-1}} a_{n -i_n}],$$

the word w is said to have the left bond i_0 and the right bond $-i_n$ as the *outer bonds* and $i_1, \ldots, i_{n-1}$ as *inner bonds.* If the inner bonds are not of interest, then the word is written as

$$[_{i_0} a_1 a_2 a_3 \cdots a_{n -i_n}].$$

The set of all well-formed words built from letters of $\mathcal{B}_\Sigma$ including the empty word is referred to as $\mathcal{B}_\Sigma^*$ and the set of all balanced words built from letters of $\mathcal{B}_\Sigma$ including the empty word is referred to as $\mathcal{B}_\Sigma^{\circledast}$. By definition, $\mathcal{B}_\Sigma^{\circledast} \subset \mathcal{B}_\Sigma^*$.
The *empty word* is the neutral element of both structures $\mathcal{B}_\Sigma^*$ and $\mathcal{B}_\Sigma^{\circledast}$, written as $[_{i_0} \lambda_{-i_0}]$ for some number $i_0 \in \mathbb{Z}_0^+$. The empty word is always a balanced word.
The length of a bond word w from $\mathcal{B}_\Sigma^*$ or $\mathcal{B}_\Sigma^{\circledast}$ is denoted by $|w|$ and is equal to the number of letters in w. In particular, the empty bond word is of length 0.

Once the derivation process has ended, the words generated by bonded systems will still contain bonds on each letter, which does not allow us to compare the language generated with languages of other families, especially those in the Chomsky hierarchy. This would make it difficult to classify the languages generated by bonded systems. Therefore, to remove the bonds while still retaining the well-formedness of the words, the bond erasing homomorphism has been introduced, as defined in Definition 2.6.

Definition 2.6 [10] The *bond erasing homomorphism* is a homomorphism

$$h_{\text{be}} : \mathcal{B}_\Sigma^{\circledast} \to \Sigma^* \text{ defined by } h_{\text{be}}([_i a_{-j}]) = a \text{ for every } [_i a_{-j}] \in \mathcal{B}_\Sigma.$$

From there, an insertion-deletion system [14] that works on the bonding alphabet has been constructed, where the language generated by the system contains bond words under the bond erasing homomorphism. Introduced in [13], the bonded sequential insertion-deletion systems (bSINSDEL-systems) combine bonded insertion systems and bonded deletion systems to increase the generative power to RE. There, the family generated by bSINSDEL-systems has indeed been shown to be equivalent to the family of recursively enumerable languages, $\mathcal{L}(\text{RE})$ also known as the family generated by Type-0 grammars, denoted by $\mathcal{L}_0$.

Generally, we seek to increase the generative power of language systems to RE to achieve computational completeness. By Church's Thesis in [15], a computationally complete system can carry out any real-world computation by way of a Turing machine.

Formally, bSINSDEL-systems are defined as follows.

Definition 2.7 [13] Let Σ be a finite alphabet, $A \subseteq \mathcal{B}_\Sigma^\circledast$ be a finite set of *axioms* that contains only neutral words, and $I, D \subseteq \mathcal{B}_\Sigma^*$ be a finite set of *insertion strings* and *deletion strings*, respectively, such that the insertion strings in I and the deletion strings in D need not be balanced. A *bonded sequential insertion-deletion system* (bSINSDEL-system) is a quadruple $\sigma = (\Sigma, A, I, D)$, where the derivation relation $\Rightarrow$ of a bSINSDEL-system $\sigma = (\Sigma, A, I, D)$ is defined as follows: for any $\alpha \in A$, $\alpha \Rightarrow_\sigma \beta, \beta \in \mathcal{B}_\Sigma^\circledast$ if and only if at least one of the following is true:

- $\alpha = uv$ for two words $u, v \in \mathcal{B}_\Sigma^*$ and there exists an insertion string $\delta \in I$ such that $\beta = u\delta v$.
- $\alpha = u\delta' v$ for two words $u, v \in \mathcal{B}_\Sigma^*$ and there exists a deletion string $\delta' \in D$ such that $\beta = uv$.

By definition, concatenation and quotient to the left and right is also possible. The reflexive and transitive closure of $\Rightarrow_\sigma$ is denoted by $\Rightarrow_\sigma^*$. If there is no risk of ambiguity, $\Rightarrow$ and $\Rightarrow^*$ are written instead of $\Rightarrow_\sigma$ and $\Rightarrow_\sigma^*$, respectively. The *language generated* by a bSINSDEL-system $\sigma = (\Sigma, A, I, D)$ is defined as

$$L(\sigma) = \{\, h_{\mathrm{be}}(\beta) \mid \text{there is an axiom } \alpha \in A \text{ such that } \alpha \Rightarrow_\sigma^* \beta \,\},$$

where h_{be} is the bond erasing homomorphism.
The family of all languages generated by bSINSDEL-systems is denoted by $\mathcal{L}(\mathrm{bSINSDEL})$.

To clarify the construction of bSINSDEL-systems, the following example is provided.

Example 2.1 Let $\sigma_1 = (\{a, b, c\}, A, I, D)$ be a bSINSDEL-system, where

$$A = \{[_0a_{-1}][_1b_{-2}][_2c_0]\},$$

$$I = \{[_1a_{-1}], [_2b_{-2}][_2c_{-2}], [_0p_{-5}]\},$$

$$D = \{[_0p_{-5}]\}.$$

Here, it is shown that the bSINSDEL-system σ_1 generates the context-sensitive language

$$L = \{a^n b^n c^n, n \geq 1\}.$$

First, the word abc is obtained directly from the axiom.
Next, the word $a^2b^2c^2$ is obtained from the derivation as follows:

$$\begin{aligned} [_0a_{-1}][_1b_{-2}][_2c_0] &\Rightarrow [_0a_{-1}][_1b_{-2}][_2c_0][_0p_{-5}] \\ &\Rightarrow [_0a_{-1}][_1a_{-1}][_1b_{-2}][_2c_0][_0p_{-5}] \\ &\Rightarrow [_0a_{-1}][_1a_{-1}][_1b_{-2}][_2b_{-2}][_2c_{-2}][_2c_0][_0p_{-5}] \\ &\Rightarrow [_0a_{-1}][_1a_{-1}][_1b_{-2}][_2b_{-2}][_2c_{-2}][_2c_0]. \end{aligned}$$

Then, the word $a^3b^3c^3$ is obtained from the derivation as follows:

$$\begin{aligned} [_0a_{-1}][_1b_{-2}][_2c_0] &\Rightarrow [_0a_{-1}][_1b_{-2}][_2c_0][_0p_{-5}] \\ &\Rightarrow [_0a_{-1}][_1a_{-1}][_1b_{-2}][_2c_0][_0p_{-5}] \\ &\Rightarrow [_0a_{-1}][_1a_{-1}][_1a_{-1}][_1b_{-2}][_2c_0][_0p_{-5}] \\ &\Rightarrow [_0a_{-1}][_1a_{-1}][_1a_{-1}][_1b_{-2}][_2b_{-2}][_2c_{-2}][_2c_0][_0p_{-5}] \\ &\Rightarrow [_0a_{-1}][_1a_{-1}][_1a_{-1}][_1b_{-2}][_2b_{-2}][_2b_{-2}][_2c_{-2}][_2c_{-2}][_2c_0][_0p_{-5}] \\ &\Rightarrow [_0a_{-1}][_1a_{-1}][_1a_{-1}][_1b_{-2}][_2b_{-2}][_2b_{-2}][_2c_{-2}][_2c_{-2}][_2c_0]. \end{aligned}$$

Subsequently, the derivations for the remaining words are done in the same way, that is, the

insertion rule $[_0p_{-5}]$ must be used in the first step. This rule, called the *unbalancing rule* ensures that the sentential forms of words not in the form of $a^nb^nc^n$ do not appear in the generated language due to the unbalanced structure. This is because only balanced words appear in the language as stated in Definition 2.7.
Next, the insertion rule $[_1a_{-1}]$ is used followed by either the insertion rule $[_1a_{-1}]$ or the insertion rule $[_2b_{-2}][_2c_{-2}]$, depending on the length of the word to be generated. The rule $[_2b_{-2}][_2c_{-2}]$ must be inserted as many times as the rule $[_1a_{-1}]$. Also, in each derivation, the rule $[_1a_{-1}]$ can be inserted wherever it fits but the rule $[_2b_{-2}][_2c_{-2}]$ can only be inserted in between the letters b and c to retain the desired form.
At the end of the derivation sequence, the rule $[_0p_{-5}]$ is then deleted to form a balanced word, now in the form of $a^nb^nc^n$.
By repeating these steps ad infinitum, the desired language

$$L(\sigma_1) = \{a^nb^nc^n, n \geq 1\}$$

is obtained.

By using the concepts presented in this section, we now determine the closure properties of bSINSDEL-systems.

3. Results

The closure properties of bSINSDEL-systems are determined by whether there exists a system that generates the language obtained under a specific language operation. The language operations are union, concatenation, concatenation closure, λ-free concatenation closure, substitution, and intersection with regular languages. In this section, it is shown that $\mathcal{L}$(bSINSDEL) is closed under all the aforementioned operations, making it a full abstract family of languages [2].

First, closure under union is shown.

Theorem 3.1 $\mathcal{L}$(bSINSDEL) is closed under union.
Proof Let $\sigma_1 = (\Sigma_1, A_1, I_1, D_1)$ and $\sigma_2 = (\Sigma_2, A_2, I_2, D_2)$ be two bSINSDEL-systems, where L_1 and L_2 are the languages generated by σ_1 and σ_2, respectively. To generate the language $L_1 \cup L_2$, construct the bSINSDEL-system $\sigma = (\Sigma, A, I, D)$, where

$$\Sigma = \Sigma_1 \cup \Sigma_2,$$

$$A = A_1 \cup A_2,$$

$$I = I_1 \cup I_2,$$

$$D = D_1 \cup D_2.$$

Clearly, any word in $L_1 \cup L_2$ can be generated by σ since all of the insertion rules and deletion rules of σ_1 and σ_2 are contained in the insertion set and deletion set of σ, respectively. If there are positions where insertion or deletion rules from σ_1 can be applied to the axiom or words from σ_2, and vice versa, the sequential derivation of σ provides the option to not do so.
Therefore,

$$L(\sigma) = L_1 \cup L_2.$$

Next, it is shown that the family of bonded sequential insertion-deletion systems is closed under concatenation.

Theorem 3.2 $\mathcal{L}$(bSINSDEL) is closed under concatenation.
Proof Let $\sigma_1 = (\Sigma_1, A_1, I_1, D_1)$ and $\sigma_2 = (\Sigma_2, A_2, I_2, D_2)$ be two bSINSDEL-systems, where L_1 and L_2 are the languages generated by σ_1 and σ_2, respectively. To generate the language L_1L_2, where all the words are the result of appending a word from L_2 onto the right-end of a word from L_1, construct the bSINSDEL-system $\sigma = (\Sigma, A, I, D)$, where

$$\Sigma = \Sigma_1 \cup \Sigma_2,$$
$$A = A_1A_2,$$
$$I = I_1 \cup I_2 \cup \{[_0p_{-k}] \mid k \geq 1\},$$
$$D = D_1 \cup D_2 \cup \{[_0p_{-k}] \mid k \geq 1\}.$$

Here, the system σ generates all words in the language L_1L_2 in a sequential way by using the respective insertion or deletion rules according to positions from their respective systems. Recall that all words in A_1 and A_2 are neutral, thus have 0 as the outer bonds, which makes it possible for the concatenation of the axioms. To preserve the desired form, the unbalancing rule $[_0p_{-k}]$ is used.
Therefore,

$$L(\sigma) = L_1L_2.$$

Furthermore, the closure of $\mathcal{L}$(bSINSDEL) under concatenation closure and consequently λ-free concatenation closure is shown.

Theorem 3.3 $\mathcal{L}$(bSINSDEL) is closed under concatenation closure.
Proof Let $\sigma_1 = (\Sigma_1, A_1, I_1, D_1)$ be a bSINSDEL-system, where L_1 is the language generated by σ_1. To generate the language L_1^*, where all the words are the result of repeating the same word from L_1 any number of times, construct the bSINSDEL-system $\sigma = (\Sigma, A, I, D)$, where

$$\Sigma = \Sigma_1,$$
$$A = A_1,$$
$$I = I_1 \cup A_1 \cup \{[_0p_{-k}] \mid k \geq 1\},$$
$$D = D_1 \cup \{[_0p_{-k}] \mid k \geq 1\}.$$

Here, the system σ generates all words in the language L_1^* by either first following the sequential derivation of the original system σ_1 or inserting any number of axioms to satisfy the catenation of L_1 onto itself before applying the rules. Again, to preserve the desired form, the unbalancing rule $[_0p_{-k}]$ is used.
Therefore,

$$L(\sigma) = L_1^*.$$

Theorem 3.4 $\mathcal{L}$(bSINSDEL) is closed under λ-free concatenation closure.
Proof The proof follows from Theorem 3.3 and Theorem 3.3 [2], which states that for any L in a family of languages of type i, denoted by $\mathcal{L}_i, 0 \leq i \leq 3$, the language L^+ is also a member of the same family of languages. We know from [13] that $\mathcal{L}$(bSINSDEL) $= \mathcal{L}_0$. Therefore, for all $L \in \mathcal{L}$(bSINSDEL), $L^+ \in \mathcal{L}$(bSINSDEL).

Next, closure under substitution is shown, which implies that $\mathcal{L}$(bSINSDEL) is closed under arbitrary homomorphism, including λ-free homomorphism and inverse homomorphism [2].

Theorem 3.5 $\mathcal{L}$(bSINSDEL) is closed under substitution.
Proof Let $\sigma = (\Sigma, A, I, D)$ be a bSINSDEL-system, where

$$\Sigma = \{u_1, u_2, u_3, ..., u_r\}.$$

Let $s : \Sigma \to \Sigma_i^*$ be a substitution defined by $s([_k u_{i-l}]) = [_k \omega_{i-l}]$, where Σ_i is the alphabet of some bSINSDEL-system $\sigma_i = (\Sigma_i, A_i, I_i, D_i)$. The language generated by bSINSDEL-system σ, $L(\sigma)$ under the substitution s is denoted by $s(L(\sigma))$.
Now, construct a bSINSDEL-system $\sigma' = (\Sigma', A', I', D')$, where

$$\Sigma' = \{s(u_i) \mid 1 \leq i \leq r\},$$
$$A = \{s(\alpha) \mid \alpha \in A\},$$
$$I = \{s(\iota) \mid \iota \in I\},$$
$$D = \{s(\delta) \mid \delta \in D\}.$$

Here, the language generated by bSINSDEL-system σ', $L(\sigma') = s(L(\sigma))$ and obviously $L(\sigma') \in \mathcal{L}$(bSINSDEL).
Therefore, $\mathcal{L}$(bSINSDEL) is closed under substitution. Again, since $\mathcal{L}$(bSINSDEL) $= \mathcal{L}_0$, Theorem 3.5 in [2] applies, which means that $\mathcal{L}$(bSINSDEL) is closed under arbitrary homomorphism, subsequently closed under both λ-free homomorphism and inverse homomorphism.

Lastly, it is shown that the family of bSINSDEL-systems is closed under intersection with regular languages.

Theorem 3.6 $\mathcal{L}$(bSINSDEL) is closed under intersection with regular languages.
Proof Let $\sigma_1 = (\Sigma_1, A_1, I_1, D_1)$ and $\sigma_2 = (\Sigma_2, A_2, I_2, D_2)$ be two bSINSDEL-systems, where L_1 and L_2 are the languages generated by σ_1 and σ_2, respectively.
If $L_1 \cap L_2 = \varnothing$, then there is no need to construct a bSINSDEL-system to generate the resulting language.
Otherwise, if $L_1 \cap L_2 \neq \varnothing$, then there must be a bSINSDEL-system $\sigma = (\Sigma, A, I, D)$ such that the language generated by σ, $L(\sigma)$ is equal to the interesecting language, such that

$$L(\sigma) = L_1 \cap L_2.$$

To this end, firstly, let

$$L_1 \cap L_2 = \{\omega_{i_1}, \omega_{i_2}, \omega_{i_3}, ..., \omega_{i_n} \mid i = 1, 2, 3, ...\},$$

where $\omega_{i_k},\ k = 1, 2, 3, ..., n$ are words in $L_1 \cap L_2$ arranged by ascending of length and $\omega_i,\ i = 1, 2, 3, ...$ are all possible forms of the words in $L_1 \cap L_2$.
Next, construct a bSINSDEL-system $\sigma = (\Sigma, A, I, D)$, where

$$\Sigma = \Sigma_1 \cup \Sigma_2,$$
$$A = \{\omega_i \mid i = 1, 2, 3, ...\},$$
$$I = I_1 \cup I_2 \cup \{[_0 p_{-k}] \mid k \geq 1\},$$
$$D = D_1 \cup D_2 \cup \{[_0 p_{-k}] \mid k \geq 1\}.$$

Thus, all possible forms of the words in $L_1 \cap L_2$ are generated by the bSINSDEL-system σ by using the shortest words of each possible form as the axioms and applying the necessary insertion and deletion rules. The unbalancing rule $[_0 p_{-k}]$ is included in case the rules in $I_1 \cup I_2$ and $D_1 \cup D_2$ are insufficient to generate the desired words.
Therefore, for any two languages L_1 and L_2 in $\mathcal{L}$(bSINSDEL), there exists another language generated by a bSINSDEL-system $L(\sigma)$ such that

$$L(\sigma) = L_1 \cap L_2.$$

Now, since $\mathcal{L}(\text{REG}) \subset \mathcal{L}(\text{RE})$ [9] and $\mathcal{L}(\text{bSINSDEL}) = \mathcal{L}(\text{RE})$ [13], we have

$$\mathcal{L}(\text{REG}) \subset \mathcal{L}(\text{bSINSDEL}).$$

Therefore, let $L_2 \in \mathcal{L}(\text{REG})$ and the proof is complete.

Finally, it has been shown that $\mathcal{L}$(bSINSDEL) is closed under all the necessary operations that qualify it as a full AFL, as summarized in Theorem 3.7.

Theorem 3.7 $\mathcal{L}$(bSINSDEL) is a full AFL.
Proof This theorem is a direct result of the closure results presented in Theorems 3.1-3.6.

4. Conclusion
In this research, the closure properties of the family of language generated by bonded sequential insertion-deletion systems $\mathcal{L}$(bSINSDEL) have been determined, where it is closed under the operations of union, concatenation, concatenation closure, λ-free concatenation closure, substitution, and intersection with regular languages. Hence, we conclude that $\mathcal{L}$(bSINSDEL) is a full AFL. This result is consistent with the established theorem that $\mathcal{L}$(RE) is a full AFL [2]. Since $\mathcal{L}$(bSINSDEL) is equal to $\mathcal{L}$(RE) as proven in [13], then $\mathcal{L}$(bSINSDEL) must also be a full AFL.

The next step in this research would be to study the computational complexity of bSINSDEL-systems. This would act as another way of classifying the bonded systems in a more theoretical computer science-centric sense. The results could prove to be handy for those looking to build abstract computers using these systems. Other than that, we also suggest that the closure properties of bonded parallel insertion-deletion systems (bPINSDEL-systems) be determined. Some extra steps may be required to account for the parallelism of the bPINSDEL-systems.

This research into bonded insertion-deletion systems provides an accurate model for the atomic behavior of DNA molecules during DNA recombination. Here, the integers appended to each side of each letter in a word act as the chemical bonds between molecules in DNA, the well-formedness of the words reflects the conformity to the stable electron arrangement between two DNA molecules, and finally, the operations of insertion and deletion simulate the recombination of the DNA sequence. Till today, the field of DNA computing continues to grow, and with every new finding, we are that much closer to innovative, biotechnological breakthroughs.

Acknowledgement
The first author would like to acknowledge The Ministry of Higher Education (MOHE) for his MyBrainSc scholarship. The authors would also like to thank The Ministry of Higher Education (MOHE) and Research Management Centre (RMC) of Universiti Teknologi Malaysia (UTM) for the financial funding through Fundamental Research Grant Scheme (FRGS) Vote No. FRGS/1/2018/STG06/UTM/02/10 and Research University Grant (RUG) 16J09.

References

[1] Linz P 2011 *An Introduction to Formal Languages and Automata, 5th Ed.* (Jones and Bartlett Publishers)
[2] Salomaa A 1973 *Formal Languages* ACM Monograph Series (Academic Press, Inc.)
[3] Herman G T 1974 *Information and Control* **24** 101–121 ISSN 0019-9958
[4] Rozenberg G and Salomaa A 1980 *Mathematical Theory of L Systems* (USA: Academic Press, Inc.) ISBN 0125971400
[5] Holzer M, Truthe B and Yosman A F 2018 *RAIRO-Theor. Inf. Appl.* **52** 127–151
[6] Yosman A F, Fong W H and Mat Hassim H I 2020 *Journal of Critical Reviews* **7** 902–909 ISSN 2394-5125
[7] Rahman A A, Fong W H, Sarmin N H and Turaev S 2020 *AIP Conference Proceedings* **2266** 060005
[8] Rozenberg G and Salomaa A 1997 *Handbook of Formal Languages* (Berlin: Springer-Verlag)
[9] Paun G, Rozenberg G and Salomaa A 1998 *DNA Computing: New Computing Paradigms* (Springer-Verlag Berlin Heidelberg) ISBN 978-3-642-08388-4
[10] Fong W H, Holzer M, Truthe B, Turaev S and Yosman A F 2016 *8th Workshop on Non-Classical Models of Automata and Applications (NCMA), Debrecen, Hungary, August 29 – 30, 2016, Proceedings* books@ocg.at (Österreichische Computer Gesellschaft, Austria) pp 163–178
[11] Yosman A F, Holzer M, Truthe B, Fong W H and Turaev S 2017 *Mal. J. Fund. Appl. Sci* **13** 769–773
[12] Fong W H, Sarmin N H, Turaev S and Yosman A F 2017 *AIP Conference Proceedings* vol 1830 (AIP Publishing LLC) p 070005
[13] Yosman A F, Fong W H and Mat Hassim H I 2020 *Proceedings of the 8th International Graduate Conference on Engineering, Science and Humanities (IGCESH 2020)* pp 228–231
[14] Kari L and Thierrin G 1996 *Information and Computation* **131** 47–61
[15] Church A 1936 *American Journal of Mathematics* **58** 345–363 ISSN 00029327, 10806377