Analyzing the Trade-Off Between Complexity Measures, Ambiguity in Insertion System and Its Applications

ANAND MAHENDRAN1, KUMAR KANNAN1, MOHAMED HAMADA2, (Senior Member, IEEE), AND MANUEL MAZZARA3

1School of Computer Science and Engineering, Vellore Institute of Technology, Vellore 632014, India
2Software Engineering Laboratory, The University of Aizu, Aizuwakamatsu 965-8580, Japan
3Institute of Software Development and Engineering, Innopolis University, 420500 Innopolis, Russia

Corresponding author: Anand Mahendran (manand@vit.ac.in)

The work of Anand Mahendran was supported by the Council of Scientific and Industrial Research (CSIR) under Project 25(0291)/18/EMR-II.

ABSTRACT Insertion is one of the basic operations in DNA computing. Based on this operation, an evolutionary computation model, the insertion system, was defined. For the above defined evolutionary computation model, varying levels of ambiguity and basic descriptional complexity measures were defined. In this paper, we define twelve new (descriptional) complexity measures based on the integral parts of the derivation, such as axioms, strings to be inserted, and contexts used in the insertion rules. Later, we analyze the trade-off among the (newly defined) complexity measures and the existing ambiguity levels. Finally, we examine the application of the analyzed trade-off in natural languages and modelling of bio-molecular structures.

INDEX TERMS Insertion systems, complexity measures, ambiguity levels, trade-off, natural languages.

I. INTRODUCTION

In the recent decades, the usage of computer has been increased enormously starting from storing and retrieving of data, manipulating scientific computations and performing other complex operations. To capture the needs of the fast growing world, there is a constant research happening in the domain of computer science. Due to the need of increase in computation speed and storage of data, the computing models used for computation and the technologies used for storage medium needs to be changed rapidly. As nature is always more faster than human brains and the computing devices, researchers felt that the nature would play a critical role, in specific, if biology is introduced in the domain of computer science. This initiated the notion of natural computing or bio-inspired computing models which bridged the gap between nature and computer science. As a result, lot of bio-inspired computing models have been defined namely membrane computing, sticker systems, splicing systems, Watson-Crick automata, insertion-deletion systems, DNA Computing, H-systems [6], [35], [36]. In formal language theory, the language generation depends on the rewriting operations, which paved a new dimension for insertion systems. If a string $\beta$ is lodged between two substrings $\alpha_1, \alpha_2$ of a string $\alpha_1\alpha_2$ to get a new string $\alpha_1\beta\alpha_2$, then the performed operation on the strings is called insertion. Insertion operation was first theoretically studied in [16]. In DNA computing, the insertion operations have (some) biological relevance, which in turn has (some) biological relevant properties in human genetics. In [34], the application of the insertion operation in the domain of genetics has been investigated.

In 1969, Solomon Marcus introduced Contextual grammars which are mainly based on the descriptive linguistics [30]. In contextual grammars based on the selector, the context is inserted to the left and right of selectors. Using the adjoining operation, iteratively, the strings are generated in the language, where as in insertion system based on the left and right context, the string is inserted. In [33], different ambiguity levels were defined and studied for external, internal contextual grammars depending on the parts that are used in the derivation. For more details, on the ambiguity issues related to contextual grammars, we cite [18], [21], [22], [32], [34]. As insertion system can be viewed as the counterpart of contextual grammars, in the similar line of
direction, in [23], various ambiguity levels were interpreted for insertion systems. As there will be more than one grammars (systems) G₁, G₂, . . . , Gₙ which generates a language L, a situation arises to choose an economical grammar (system) G which generates L. This idea of economical grammar (system) leads to the introduction of the notion called descriptional complexity measures. In [15], the complexity measures were investigated for context-free grammars. Several complexity measures were defined for contextual grammars such as \( A_{x}, M_{x}, T_{x}, C_{x}, P_{x}, M_{x} \), \( T_{Sel} \) [20], [37], [38]. In [24], depending on the insertion and deletion rules, different complexity measures were introduced and analyzed for insertion-deletion systems.

In programming and natural languages ambiguity is one of the interesting problems that needs to be investigated. First, we will discuss about the ambiguity issues in programming languages. Given a grammar G and an input string \( w \in L(G) \), if it has more than one derivation or derivation trees (for the same string \( w \)), then the grammar G is said to be ambiguous. On the other hand the grammar G is unambiguous, if there exists only one derivation (derivation tree) for all the words in \( L(G) \). The following example shows the importance and necessity of studying about the trade-off in programming languages. Consider the following Context Free Grammar (CFG) \( G \) which generates \( L \) (the set of arithmetic expressions): Let \( G = ((X, Y), \{a, +, *, (, )\}, X, \{X \rightarrow Y, X \rightarrow X \ast X, X \rightarrow (X), Y \rightarrow a\}) \). The grammar G is ambiguous. The string \( a + a \ast a \in L(G) \) can be derived by two (distinct) left most derivations (LMD): LMD 1: \( X \rightarrow X + X \ast X \rightarrow a + X \ast X \rightarrow a + a \ast X \rightarrow a + a \ast a \). LMD 2: \( X \rightarrow X + X \rightarrow a + X \rightarrow a + X \ast X \rightarrow a + a \ast X \rightarrow a + a \ast a \). The grammar G is ambiguous in addition to that the grammar G is minimal in terms of the measures non-terminals and productions. The minimal values of the measures based on the grammar G is 2 and 5 respectively. For the same language L, an unambiguous grammar \( G' \) can be derived. \( G' = ((X, Y, X', Y'), \{a, +, *, (, )\}, X, \{X \rightarrow X', X' \rightarrow Y', Y' \rightarrow Y, X \rightarrow X + X', X' \rightarrow X \ast Y', Y' \rightarrow (X), Y \rightarrow a\}) \). The interesting fact about the above grammar \( G' \) is unambiguous but is not minimal with respect to the measures non-terminals and number of productions. The Table 1 shows the comparison between the measures of the grammars G and \( G' \) respectively.

Based on the (minimal) measures number of non-terminals and productions a minimal CFG can be given for the expression language, but the given CFG is ambiguous. Whereas if the grammar G is unambiguous, it is not minimal in the measures non-terminals and productions. As insertion system is mainly based on the insertion operation, it has a potential application in generating natural languages and modelling of bio-molecular structures [30], [35]. In general, if we want to store natural languages, we will prefer economical and an unambiguous system. Under these circumstances, a trade-off needs to be performed based on the descriptional complexity measures and the ambiguity levels. As far as considering the research work on insertion systems it is mainly focused on the introduction of variants, reducing the weights towards the computational completeness, analyzing the relationship with Chomskian hierarchy of grammars, closure properties, ambiguity issues and decidability issues. For more details, we refer to [9], [10], [11], [12], [13], [23], [24], [25], [35]. This motivated to define new descriptional complexity measures for insertion systems, perform the trade-off and to investigate the application of the analyzed trade-off.

The organization of the paper is given as, the preliminaries are dealt in Section II. The newly introduced descriptional complexity measures of insertion systems were discussed in Section III. The trade-off results between the newly defined complexity measures and various ambiguity levels of insertion systems were investigated in Section IV. The application of the trade-off between ambiguity and measures in natural languages and modelling of bio-molecular structures has been probed in Section V. The comparative study is dealt in Section VI. The conclusion and the future work is dealt in Section VII.

### Table 1. Comparison of complexity measures.

| Measures | Ambiguous Grammar (\( G \)) | Unambiguous Grammar (\( G' \)) |
|----------|----------------------------|-----------------------------|
| Number of Non-terminals | 2 | 4 |
| Number of Terminals | 5 | 5 |
| Number of productions | 3 | 7 |
TABLE 2. Different ambiguity levels of insertion system.

| Ambiguity Level | Description |
|----------------|-------------|
| 0 – ambiguous   | From two different axioms \((A_1, A_2 \in A, A_1 \neq A_2)\), a same word \(w\) can be derived |
| 1 – ambiguous   | If the same word \(w\) can be obtained by two distinct unordered CS |
| 2 – ambiguous   | If the same word \(w\) can be obtained by two distinct ordered CS |
| 3 – ambiguous   | If the same word \(w\) can be obtained by two distinct ordered CCS |
| 4 – ambiguous   | If the same word \(w\) can be obtained by two distinct ordered CCS |
| 5 – ambiguous   | If the derivations are different based on the descriptions |

Now, we recall the various complexity measures introduced for insertion-deletion (ins-del) systems. Given an ins-del system \(\gamma = (V, T, A, R)\), the existing descriptional complexity measures of ins-del systems are defined as follows in Table 3. For more details, we refer [20], [24], [37], [38]. Given the measure \(M\) and language \(L\), the minimal system \(\gamma\) for the language \(L\) is defined as: \(M(L) = \min \{M(\gamma) | L = L(\gamma)\}\).

For a given measure \(M\) and a language \(L\), we define \(M^{-1}(L) = \{\gamma | L(\gamma) = L \text{ and } M(\gamma) = M(L)\}\). In the above definition, \(M^{-1}(L)\) denotes the set of all minimal systems that generate \(L\) which are minimal in the measures \(M\). For a language \(L\), two measures \(M_1, M_2\) are said to be incompatible if the following relation \(M_1^{-1}(L) \cap M_2^{-1}(L) = \emptyset\) holds true. If \(M_1^{-1}(L) \cap M_2^{-1}(L) \neq \emptyset\), then the measures \(M_1\) and \(M_2\) are called compatible. Based on the above definition, in [34], any two of the measures \(Ax, \text{MAx, TAx}\) are proved to be compatible. From the above measures (Table 3), if the deletion rules were not used by the system \(\gamma\), the measures \(\text{TDEL} - \text{StrCon}, \text{TDEL} - \text{Str}\) are not applicable to insertion systems.

TABLE 3. Existing descriptive complexity measures of ins-del system.

| S.No | Measure | Notation |
|------|---------|----------|
| 1    | \(Ax\)  | card(A)  |
| 2    | \(\text{MAx}\) | \(\text{MAx}_{\text{cc}}(A)\) |
| 3    | \(\text{TAX}\) | \(\sum_{w \in A^{+}}|w|\) |
| 4    | \(\text{Prod}\) | card(R) |
| 5    | \(TLength - Con\) | \(\sum_{(u, v) \in E} |uv|\) |
| 6    | \(TLength - Con\) | \(\sum_{(u, v) \in E} |u| + |v|\) |
| 7    | \(\text{TINS} - \text{StrCon}\) | \(\sum_{(u, v) \in E} |u| + |v|\) |
| 8    | \(\text{TDEL} - \text{StrCon}\) | \(\sum_{(u, v) \in E} |u| + |v|\) |
| 9    | \(\text{TINS} - \text{Str}\) | \(\sum_{(u, v) \in E} |u| + |v|\) |
| 10   | \(\text{TDEL} - \text{Str}\) | \(\sum_{(u, v) \in E} |u| + |v|\) |

LCon, MLen – LCon, TLen – LCon + InsStr, MLen – LCon + InsStr, and \(M_2 \in \{Ax\}\).

Proof: Let the language \(L_1 = \{b^a c^m | m \geq 0\}\). The following 5-ambiguous insertion system \(\gamma_1\) can be used to generate \(L_1, \gamma_1 = \{(a, b), (b^3, b^3 a), [(a, a, \lambda)]\}\). The system \(\gamma_1\) is minimal in \(TLen – LCon, MLen – LCon, TLen – LCon + InsStr, MLen – LCon + InsStr\). Now, we will proceed on the minimal measures of the system \(\gamma_1\). In this regard, first we will prove for the measures \(TLen – LCon, MLen – LCon\). From the system \(\gamma_1\), we can see that \(TLen – LCon(\gamma_1) = 1\). As the system \(\gamma_1\), uses only one insertion rule and since \(TLen – LCon(\gamma_1) = 1\), which implies \(MLen – LCon(\gamma_1) = 1\). Now, we will discuss on the other measures \(TLen – LCon + InsStr, MLen – LCon + InsStr\). Any system which generates \(L_1\), should have a string to be inserted of minimum length one. Therefore, \(TINS – Str(\gamma_1) = 1\). Earlier, we have proved that the \(TLen – LCon(\gamma_1) = 1 = MLen – LCon(\gamma_1)\). Therefore, \(TLen – LCon + InsStr(\gamma_1) = 2 = MLen – LCon + InsStr(\gamma_1)\). From the above arguments, we can conclude that the system \(\gamma_1\) is minimal in \(TLen – LCon, MLen – LCon, TLen – LCon + InsStr, MLen – LCon + InsStr\).

Consider any \(\gamma_1'\) which generates \(L_1\) which has \(TLen – LCon(\gamma_1') = 1 = MLen – LCon(\gamma_1')\) and \(TLen – LCon + InsStr(\gamma_1') = 2 = MLen – LCon + InsStr(\gamma_1')\). Consider a word \(b^{3} a^{k} \in L_1\), for a large value of \(k\). In the derivation of the above words, different \(a\) can be chosen, thus producing two different descriptions. Therefore, the system \(\gamma_1'\) is 5-ambiguous.

However, the language \(L_1\) is unambiguous as there exists an unambiguous insertion system \(\gamma''_1\) which generates \(L_1\). Consider the system \(\gamma''_1 = \{(a, b), (b^3, a, \lambda)\}\). The system \(\gamma''_1\) is unambiguous. From the system \(\gamma''_1\), it is clear that \(\gamma''_1\) generates \(L_1\). While deriving \(b^{3} a^{r}, r \geq 1 \in L_1\), the position of the string \(a\) to be inserted is unique in the derivation. As the system \(\gamma''_1\) has only one axiom \(b^3\), it is minimal with respect to \(Ax\). Note that the system \(\gamma''_1\) is not minimal in \(TLen – LCon, TLen – LCon, TLen – LCon + InsStr, MLen – LCon + InsStr\).
### TABLE 4. Newly introduced descriptive complexity measures of insertion system.

| S.No | Measure                  | Notation              | Description                                                                 |
|------|--------------------------|-----------------------|-----------------------------------------------------------------------------|
| 1    | MLen – InsStr            | \( \max_{(u, v) \in R} | \beta \) | Maximum length of the IS                                                   |
| 2    | mLen – InsStr            | \( \min_{(u, v) \in R} | \beta \) | Minimum length of the IS                                                   |
| 3    | MLen – LCon              | \( \max_{(u, v) \in R} | | \) | Maximum length of the LC used in the IR                                    |
| 4    | MLen – RCon              | \( \max_{(u, v) \in R} | | \) | Maximum length of the RC used in the IR                                    |
| 5    | mLen – LCon              | \( \min_{(u, v) \in R} | | \) | Minimum length of the LC used in the IR                                    |
| 6    | mLen – RCon              | \( \min_{(u, v) \in R} | | \) | Minimum length of the RC used in the IR                                    |
| 7    | TLen – LCon              | \( \sum_{(u, v) \in R} | | u \) | Total length of all LC used in the IR                                      |
| 8    | TLen – RCon              | \( \sum_{(u, v) \in R} | | v \) | Total length of all RC used in the IR                                      |
| 9    | TLen – LCon + InsStr     | \( \sum_{(u, v) \in R} | | u + | \beta \) | Total length of LC + the length of the IS                                 |
| 10   | TLen – RCon + InsStr     | \( \sum_{(u, v) \in R} | | v + | \beta \) | Total length of LC + the length of the IS                                 |
| 11   | MLen – LCon + InsStr     | \( \max_{(u, v) \in R} | | u + | \beta \) | Maximum length of LC + the length of the IS                               |
| 12   | MLen – RCon + InsStr     | \( \max_{(u, v) \in R} | | v + | \beta \) | Maximum length of RC + the length of the IS                               |

**Corollary 1:** There are pseudo inherently 5-ambiguous insertion languages with \( M_1 \in \{ TLen – LCon, MLen – LCon, TLen – LCon + InsStr, MLen – LCon + InsStr \} \) and \( M_2 \in \{ M\!\!\text{Ax}, T\!\!\text{Ax} \} \).

**Theorem 2:** There are pseudo inherently 5-ambiguous insertion languages which are minimal in \( M_1 \in \{ TINS – Str \} \) and \( M_2 \in \{ TLen – RCon, MLen – RCon, mLen – RCon \} \).

**Proof:** Let the language \( L_2 = \{ d(a^3b)^k \mid k \geq 0 \} \cup \{ ((a^2b)^k c \mid k \geq 0 \} \). The following 5-ambiguous insertion system \( \gamma_2 \) can be used to generate \( L_2 \): \( \gamma_2 = ((a, b, c, d), (d, da^2b, c, a^2bc), ((\lambda, a^3b, a^3b))) \). The system \( \gamma_2 \) is minimal in TINS – Str. Any insertion system \( \gamma'_2 \) which generates \( L_2 \) should have an insertion string of length \( \leq 4 \). Therefore, \( \gamma_2 \) is minimal in TINS – Str and TINS – Str(\( L_2 \)) = 4.

Consider any \( \gamma'_2 \) which generates \( L_2 \) which has TINS – Str = 4. Consider the words \( d(a^3b)^{i+j} \) or \( (a^3b)^{i+j} c \in L_2 \), for a large values of \( i, j \). In the derivation of the above word, different \( a^3b \) can be chosen, thus producing two different descriptions. Therefore, the system \( \gamma'_2 \) is 5-ambiguous.

However, the language \( L_2 \) is unambiguous as there exists an unambiguous insertion system \( \gamma''_2 \) which generates \( L_2 \). Consider the system \( \gamma''_2 = ((a, b, c, d), (c, d), (d, a^3b, a^3b), ((\lambda, a^3b, c))) \). The system \( \gamma''_2 \) is unambiguous. With the help of the insertion rule \( (d, a^3b, a^3b), (a^3b)^{i+j} c, k \geq 0 \) can be generated. Likewise, by using the insertion rule \( \lambda, (a^3b)^{i+j}, (a^3b)^{i+j} c, k \geq 0 \) can be generated. While deriving \( d(a^3b)^{i+j} \) or \( (a^3b)^{i+j} c, r, s \geq 1 \in \gamma''_2 \), the position of the string to be inserted \( a^3b \) is unique in the derivation. From the system \( \gamma''_2 \), it is easy to see that the \( \gamma''_2 \) is minimal with respect to \( \{ TLen – RCon, MLen – RCon, mLen – RCon \} \).

**Corollary 2:** In the above result, if the insertion rule is changed as \( (a^3b, a^3b, \lambda) \), the language \( L_2 \) is represented as \( L'_2 \). For the language \( L'_2 \), there exists a result for the following measures. There are pseudo inherently 5-ambiguous insertion languages with respect to \( M_1 \in \{ TINS – Str \} \) and \( M_2 \in \{ TLen – LCon, MLen – LCon, mLen – LCon \} \).

**Theorem 3:** There are pseudo inherently 5-ambiguous insertion languages which are minimal in \( M_1 \in \{ TINS – Str, TLen – RCon, MLen – RCon, mLen – RCon, TLen – RCon + InsStr, MLen – RCon + InsStr \} \) and \( M_2 \in \{ Ax, TLen – LCon, MLen – LCon, TLen – LCon + InsStr \} \).

**Proof:** Let the language \( L_3 = \{ ca^3cda^n \mid n \geq 1 \} \cup \{ dba^3da^n \mid n \geq 1 \} \). The following 5-ambiguous insertion system \( \gamma_3 \) can be used to generate \( L_3 \): \( \gamma_3 = ((a, b, c, d, \{ ca^3c, ca^2ca^2, ca^2ca^3, dba^3da, db3da^2, db3da^3 \}, (a^3, a, \lambda))) \). The system \( \gamma_3 \) is minimal in TINS – Str, TLen – RCon, MLen – RCon, mLen – RCon, TLen – RCon + InsStr, MLen – RCon + InsStr. Any insertion system \( \gamma'_3 \) which generates \( L_3 \) should have an insertion string of length \( \leq 1 \). Therefore, \( \gamma_3 \) is minimal in TINS – Str and TINS – Str(\( L_3 \)) = 1. Now, we will prove for other minimal measures. As the insertion system has only one insertion rule and it uses \( \lambda \) as the right context, the system \( \gamma_3 \) is minimal in the measures TLen – RCon, MLen – RCon, mLen – RCon. As the TINS – Str(\( L_3 \)) = 1 and TLen – RCon(\( L_3 \)) = 1 = MLen – RCon(\( L_3 \)), we can conclude the system \( \gamma_3 \) is minimal in the measures TLen – RCon + InsStr, MLen – RCon + InsStr.
Consider any $\gamma'_3$ which generates $L_3$ which has $TINS - Str = 4$, $\{TLen - RCon, MLen - RCon, mLen - RCon\} = \lambda$, $\{TLen - RCon + InsStr, MLen - RCon + InsStr\} = 1$. Consider the words $cb a^2 c^d$, or $d b a^2 d a \in L_3$, for a large values of $s, t$. In the derivation of the abovewords, different $a$ can be chosen, thus producing two different descriptions. Therefore, the system $\gamma'_3$ is 5-ambiguous.

However, the language $L_3$ is unambiguous as there exists an unambiguous insertion system $\gamma'_3$ which generates $L_3$. Consider the system $\gamma'_4 = \{(a, b, c, d), \{c b^2 a c, d b^2 a d\}, \{(c, a, a), (d, a, a)\}\}$. The system $\gamma'_4$ is unambiguous and minimal in the measures $Ax, TLen - LCon, MLen - LCon, TLlen - LCon + InsStr$. With the help of the insertion rule $(c, a, a)$, $cba^2 c^k, k \geq 1$ can be generated. Likewise, by the using the insertion rule $(d, a, a)$, $d b a^2 d^k, k \geq 1$ can be generated. While deriving $cba^2 c^r$ or $d b a^2 d^r$, $r, s \geq 1$ in $L_3$, the position of the string to be inserted $a$ is unique in the derivation. Any system which generates $L_3$ should have minimum two axioms. As the system uses the insertion rules of the form $\{(c, a, a), (d, a, a)\}$, it is easy to see that the $\gamma'_4$ is minimal with respect to $\{Ax, TLen - LCon, MLen - LCon, TLlen - LCon + InsStr\}$. 

**Corollary 3:** There are pseudo inherently 5-ambiguous insertion languages with respect to $M_1 \in \{TINS - Str, TLlen - RCon, MLen - RCon, mLen - RCon, TLen - RCon + InsStr, MLen - RCon + InsStr\}$ and $M_2 \in \{Ax, Max, Tax, TLen - LCon, MLen - LCon, TLen - LCon + InsStr\}$.

**Theorem 4:** There are pseudo inherently 4-ambiguous insertion languages with $M_1 \in \{Ax\}$ and $M_2 \in \{TLen - LCon, MLen - LCon, TLen - LCon + InsStr\}$.

**Proof:** Let the language $L_4 = \{c^2 a^m | n \geq 0\} \cup \{d^2 a^m | n \geq 0\} \cup \{c^2 a d^2 a^m | n, m \geq 0\}$. The following 4-ambiguous insertion system $\gamma_4$ can be used to generate $L_4$. $\gamma_4 = \{(a, b, c, d), \{c^2 a^2 d^2, d^2 a^2, (c^2, a, a), (d^2, a, a)\}\}$. To generate $L_4$, any insertion system $\gamma'_4$ should have the axioms of the form (which should be minimum three) $c^2 a^2 d^2, c^2 a^2 d^2$. Therefore, $Ax(L_4) = 3$.

Consider any $\gamma'_4$ which generates $L_4$, where $Ax(L_4) = 3$. To generate $c^2 a^2 d^2$, $l \geq 0$ of $L_4$, definitely, the insertion system must have an insertion rule of the following form $(c^2, a^l, \lambda)$, $i \geq 1$. To generate $d^2 a^2 d^k, k \geq 0$ of $L_4$, definitely, the insertion system should have a rule of the form $(d^2, a^l, \lambda), j \geq 1$. In order to prove $\gamma'_4$ is 4-ambiguous, let us examine a string $c^2 a^l d^2 a^l d^2 a^{l+1} \in L_4$. The above string can be acquired by two different ordered CCS. In one CCS, first the following insertion rule $(c^2, a, \lambda)$ can be used, followed by the other insertion rule $(d^2, a, \lambda)$. In another CCS, first the following insertion rule $(d^2, a, \lambda)$ can be used, followed by the other insertion rule $(c^2, a, \lambda)$. As the string to be inserted $a^l$ is same for any arbitrary system, the insertion system $\gamma'_4$ is 1 and 3-ambiguous.

However, $L_4$ is unambiguous as there exists an unambiguous system $\gamma''_4$ which generates $L_4$. Consider the system $\gamma''_4 = \{(a, b, c, d), \{c^2 a^2 d^2, d^2 c^2 a^2, (c^2, a^2 d, \lambda)\}, \{(a, a, a)\}\}$. The system $\gamma''_4$ is minimal while considering the following measures: $\{TLen - LCon, MLen - LCon, TINS - Str\}$. As the system $\gamma''_4$ uses only one insertion rule $(a, a, \lambda)$, it is clear that the system is 4-ambiguous.

**Corollary 4:** There are pseudo inherently 4-ambiguous insertion languages with $M_1 \in \{Max, Tax, mLen - LCon, MLen - LCon + InsStr\}$ and $M_2 \in \{TLen - LCon\}$.

**Theorem 5:** There are pseudo inherently 4-ambiguous insertion languages with respect to $M_1 \in \{Ax, mLen - LCon, MLen - LCon + InsStr\}$ and $M_2 \in \{TLen - LCon\}$.

**Proof:** Let the language $L_5 = \{c^2 a(b^3)^m | n \geq 1\} \cup \{d^2 (a b^3)^m | n \geq 1\} \cup \{c^2 a(b^3)^m d^2 a(b^3)^m | n, m \geq 0\}$. The following 4-ambiguous insertion system $\gamma_5$ can be used to generate $L_5$. $\gamma_5 = \{(a, b, c, d), \{c^2 a b^3, d^2 a b^3, c^2 d^2, (c^2, a b^3, \lambda), \{d^2, ab^3, \lambda\}\}\}$. First, we will discuss on the measure $Ax$. The axioms $c^2 a b^3, d^2 a b^3, c^2 d^2$ can be used to derive the first, second and third part of $L_5$. It is easy to see that the minimum three axioms should be there to generate $L_5$. Therefore, $Ax(L_5) \leq 3$, which implies $Ax(L_5) = 3$. Next, we will discuss on the measure $mLen - LCon$. The insertion rule $(c^2, ab^3, \lambda)$ is used to generate the first part of the language $L_5$. The $d^2 (a b^3)^m, n \geq 1$ part of the language $L_5$ can be derived using the following insertion rule $(d^2, ab^3, \lambda)$. By using the insertion rules alternatively, the third part of the language $L_5$ can be generated. Any system which generates $L_5$, should have insertion rules whose $mLen - LCon(L_5) \leq 2$, which implies $mLen - LCon(L_5) = 2$. Next, we will discuss on the measure $MLen - LCon + InsStr$. Any system which generates $L_5$ should have the left contexts $c^2, d^2$ in the insertion rules, which implies $MLen - LCon(L_5) = 2$. Likewise, the inserted string should be $ab^3$, which implies $MLen - InsStr(L_5) = 4$. As the measure $MLen - LCon + InsStr$ is the combination of the above two measures, we can conclude $MLen - LCon + InsStr(L_5) = 6$.

Consider any $\gamma'_5$ which generates $L_5$, where $Ax(L_5) = 3$, $mLen - LCon(L_5) = 2$ and $MLen - LCon + InsStr(L_5) = 6$. Since $c^2 a(b^3)^m \in L_5$, the insertion system $\gamma'_5$ should have an insertion rule which should be of the following form $(c^2, a(b^3)^r, \lambda), r \geq 1$. Likewise, $d^2 (a b^3)^m \in L_5$, the system should have an insertion rule of the form $(d^2, a(b^3)^r, \lambda), s \geq 1$. Consider the word $c^2 a(b^3)^r d^2 a(b^3)^r \in L_5$. This word can be generated from the axiom $c^2 d^2$ by means of two different ordered CCS. In one CCS, the insertion rule $(c^2, ab^3, \lambda)$ is used followed by $(d^2, ab^3, \lambda)$. In another CCS, first the following insertion rule $(d^2, ab^3, \lambda)$ is used in the derivation followed by the another insertion rule $(c^2, ab^3, \lambda)$. Therefore, it implies the insertion system $\gamma'_5$ is 4-ambiguous. The system $\gamma'_5$ is 1 and 3-unambiguous, because the same string $ab^3$ is inserted in both the derivations.

However, the language $L_5$ is unambiguous since there exists an 4-ambiguous system $\gamma'''_5 = \{(a, b, c, d), \{c^2 a b^3, d^2 a b^3, c^2 d^2 a b^3, c^2 d^2 a b^3, (b^3, ab^3, \lambda)\}\}$. The system $\gamma'''_5$ is 4-ambiguous as it uses only one insertion rule. The system $\gamma'''_5$ is minimal with respect to $TLlen - LCon$. Note that, the insertion system $\gamma'''_5$ is not minimal in the measures $\{Ax, mLen - LCon, MLen - InsStr\}$. 

**Corollary 5:** There exists pseudo inherently 4-ambiguous insertion languages with $M_1 \in \{Max, Tax, mLen - LCon, MLen - LCon + InsStr\}$ and $M_2 \in \{TLlen - LCon\}$. 

100517
Theorem 6: There are pseudo inherently 2-ambiguous insertion languages with $M_1 \in \{Ax\} \text{ and } M_2 \in \{TLen - LCon, TLen - RCon, MLen - LCon, MLen - RCon, TLen - LCon + InsStr, TLen - RCon + InsStr\}$.

Proof: Let the language $L_6 = \{a^3b^m | m \geq 0\} \cup \{b^m | m \geq 0\}$ and $L_7 = \{a^3b^m c^2 | m \geq 0\}$. The following 2-ambiguous insertion system $\gamma_0$ can be used to generate $L_6$: 

$$\gamma_0 = \{(a, b, c), (a^2, b, c^2), (a^2, c), (a, b, \lambda), (\lambda, b, c^2)\}.$$

The insertion system $\gamma_0$ is minimal in $Ax$. Any system which generates $L_6$ should have minimum three axioms $(a^3, c, a^2c)$. Therefore, $Ax(L_6) \leq 3$, in turn it infers $Ax(L_6) = 3$.

Consider any insertion system $\gamma_0'$ which is used to generate $L_6$, where $Ax(L_6) = 3$. Since the strings of the form $a^3b^i, i \geq 0 \in L_6$, definitely in the insertion rule there should be a context of the form $(a, b(c^3), t \geq 0)$. Likewise, since the strings of the form $(bc)^y, j \geq 1 \in L_7$, definitely in the insertion rule there should be a context of the form $(bc)^y, d, s \geq 0$. In both cases, the inserted string will be $(bc)^y, k \geq 1$. In order to prove the insertion system is $\gamma_0'$ is 2-ambiguous, let us take a string $a(b(c^3))^y$ if $d \in L_7$. From two (different) unordered CCS, the above word can be obtained from the (same) axiom. In one sequence, using the context $(a, b(c^3))$ completely, the string $a(b(c^3))^y$ can be obtained. In another sequence, using the context $LCon$ completely, the string $a(b(c^3))^y$ can be derived. Thus, the same word $a(b(c^3))^y$ is obtained from two distinct unordered CCS. Therefore, the language $L_6$ is 2-ambiguous.

The language $L_6$ is unambiguous as $L(\gamma_0') = L_6$. The 2-ambiguous insertion system is given as: $\gamma_0'' = \{(a, b, c), (a^2, b, c^2), (a^2, b, c), (a, b, \lambda)\}$. Since the system $\gamma_0''$ has only one context in the insertion rule $(b, \lambda)$, it implies $\gamma_0''$ is 2-ambiguous. The insertion system $\gamma_0''$ is minimal in the measures $\{TLen - LCon, TLen - RCon, MLen - LCon, MLen - RCon, TLen - LCon + InsStr, TLen - RCon + InsStr\}$. From the system $\gamma_0''$, it is not minimal in $Ax$.

Corollary 6: There are pseudo inherently 0-ambiguous insertion languages in the measures $M_1 \in \{Max, TAx\}$ and $M_2 \in \{TLen - LCon, TLen - RCon, MLen - LCon, MLen - RCon, TLen - LCon + InsStr, TLen - RCon + InsStr\}$.

Theorem 7: There are pseudo inherently 2-ambiguous insertion languages which are minimal in $M_1 \in \{TLen - LCon, TLen - LCon + InsStr\}$ and $M_2 \in \{TLen - LCon, TLen - RCon + InsStr\}$.

Proof: Let the language $L_7 = \{a(b(c^3))^y | n \geq 1\}$ and $(bc)^yd, n \geq 1 \cup \{a(b(c^3))^y | n \geq 0\}$. The following 2-ambiguous insertion system $\gamma_1$ can be used to generate $L_7$: 

$$\gamma_1 = \{(a, b, c), (b, c^3, d, ad), ((a, c^3), \lambda), (\lambda, b, c^3, d)\}.$$

The system $\gamma_1$ is minimal with respect to $\{TLen - LCon, TLen - LCon + InsStr\}$. Any system which generates $L_7$ should have the following contexts $(\{(a, \lambda), (\lambda, d)\})$ in the insertion rules and the inserted string of the form $bc^3$. From the system $\gamma_1$, it is clear that the system $\gamma_1$ is minimal in the measures $\{TLen - LCon, TLen - LCon + InsStr\}$. It is easy to see that $TLen - LCon \leq 1$ and $TLen - LCon + InsStr \leq 5$, which implies that $TLen - LCon = 1$ and $TLen - LCon + InsStr = 5$.

Consider any system $\gamma_0''$, which generates $L_7$, where $TLen - LCon(L_7) = 1$ and $TLen - LCon + InsStr(L_7) = 5$. Since the strings of the form $a(b(c^3))^y, i \geq 1 \in L_7$, definitely in the insertion rule there should be a context of the form $(a, (bc)^y), t \geq 0$. Likewise, since the strings of the form $(bc)^yd, j \geq 1 \in L_7$, definitely in the insertion rule there should be a context of the form $(bc)^y, d, s \geq 0$. In both cases, the inserted string will be $(bc)^y, k \geq 1$. In order to prove the insertion system is $\gamma_0''$ is 2-ambiguous, let us take a string $a(b(c^3))^y, d \in L_7$. From two (different) unordered CCS, the above word can be obtained from the (same) axiom. In one sequence, using the context $(a, b(c^3))$ completely, the string $a(b(c^3))^y$ can be obtained. In another sequence, using the context $(bc)^y, d$ completely, the string $a(b(c^3))^y$ can be derived. Thus, the same word $a(b(c^3))^y$ is obtained from two different unordered CCS. Therefore, the language $L_7$ is 2-ambiguous.

However, the language $L_7$ is 2-ambiguous since there exists an 2-ambiguous insertion system which is minimal in $\{TLen - LCon, TLen - LCon + InsStr\}$ $\gamma_0'' = \{(a, b, c, d), (abc, b(c^3), ad, abc(c^3), ((c^3, bc^3), \lambda))\}$. As the system $\gamma_0''$ uses only one insertion rule, obviously, there will be only one context in the insertion rule $(c^3, \lambda)$. Therefore, the system $\gamma_0''$ is 2-ambiguous. The system $\gamma_0''$ is minimal in the measures $\{TLen - LCon, TLen - LCon + InsStr\}$. Note that, the insertion system $\gamma_0'''$ is not minimal in the measure $\{TLen - LCon, TLen - LCon + InsStr\}$.

Theorem 8: There are pseudo inherently 0-ambiguous insertion languages with respect to $M_1 \in \{Ax, MLen - InsStr, TINS - Str, MLen - InsStr\}$ and $M_2 \in \{MLen - LCon, MLen - RCon, TLen - LCon, TLen - RCon\}$.

Proof: Let the language $L_8 = \{a^2b^{2n} | n \geq 1\} \cup \{a^2b^m c^2 | n \geq 1\} \cup \{a^2b^m c^2 | n \geq 0\}$. The following 0-ambiguous insertion system $\gamma_8$ can be used to generate $L_8$: $\gamma_8 = \{(a, b, c), (a^2b^m c^2, a^2b^m c^2, a^2b^m c^2), ((a^2, b, c), (a^2, b^2, \lambda), (\lambda, b, c^3))\}$. The system $\gamma_8$ is minimal with respect to $\{Ax, MLen - InsStr, TINS - Str\}$. First, we will prove for the measures $\{MLen - InsStr, TINS - Str\}$. From the system $\gamma_8$, it is easy to see that $MLen - InsStr(\gamma_8) \leq 3$ and $TINS - Str(\gamma_8) \leq 5$. To generate the strings of the form $a^2b^{2k+2}, k \geq 0 \in L_8$, the insertion rule should have the string $b$. However, if such an insertion string is present in any of the insertion rules, then the system $\gamma_8$ may generate some strings $a^2b^{2p}, p \geq 1$ and $b^{2q+2}, q \geq 1$ which doesn’t $\notin L_8$. From the above claim, all the parts of $L_8$ cannot be produced by the insertion string $b$, which implies $MLen - InsStr(\gamma_8) = 2$. Next, we will discuss on the following measure $TINS - Str$. Since the strings of the structure $a^2b^{2p}, p \geq 1 \in L_8$, insertion rule will certainly have the string $b^3$. Likewise, since the strings of the structure $b^{2q+2}, q \geq 1 \in L_8$, insertion rule will certainly have the string $b^3$. Therefore, we conclude $MLen - InsStr(\gamma_8) \geq 3$ and $TINS - Str(\gamma_8) \geq 5$.

Next, we will discuss on the measure Axiom. Any system which generates $L_8$ will have three axioms $a^2b^2, b^2c^2, a^2c^2$. Next, we will discuss why the system should have an axiom $a^2bc^2$. If the system is not having the axiom $a^2bc^2$, probably it can be generated by using the axiom $a^2c^2$ by inserting
the string $b$. But previously, we have proved that the system cannot have $b$ as the string to be inserted. Therefore, it implies $a^2bc^2$ should be present in the axiom. Therefore, the system $\gamma_8$ is minimal in the measure $Ax$.

Consider any system $\gamma'_8$ which generates $L_8$. The system $\gamma'_8$ is minimal in the measure $Ax$, $MLen - InsStr$, $TINS - Str$, $MLen - InsStr$. In order to claim $\gamma_8$ is 0-ambiguous, let us take the strings $a^2b^m\gamma$ and $b^2c^m\gamma \in L_8$, for a large values of $r$ and $s$. To produce the words of the form $a^2b^p$, $p \geq 1$ and $b^q\gamma$, $q \geq 1$, the insertion system $\gamma_8$ should have strings of the form $b^{3i}$, $t \geq 1$ and $b^{2u}$, $u \geq 1$ respectively. Consider a word $a^2b^{3m+2au}c^2 \in L_8$, for $m \geq 1$, $n \geq 0$. The above word can be achieved from two unique axioms $a^2c^2$ and $a^2bc^2$. Starting from the axiom $a^2c^2$, the word $a^2b^{3m+2am}c^2$ can be obtained by inserting the strings $b^{3i}$, $m$-times and $b^{2u}$, $u$-times. On the other hand, the word $a^2b^{3m+2am}c^2$ can be derived from the axiom $a^2bc^2$. In one derivation, the string $b^{3i}$ can be inserted for $m - i_1$-times, $i_1 \geq 1$. In another derivation, the string $b^{2u}$ can be inserted for $n + i_2$-times, $i_2 \geq 1$. Thus, the word $a^2b^{3m+2am}c^2$ is obtained from two different axioms $a^2c^2$, $a^2bc^2$. Therefore, the system $\gamma'_8$ is 0-ambiguous. For the measures $MLen - InsStr$, $TINS - Str$, $MLen - InsStr$, an akin reasoning can be given.

Next, we have to prove the $L_8$ is 0-ambiguous, by showing there exists an 0-ambiguous system $\gamma''_8 = \{(a, b, c), \{a^2b^2, a^2b^3, b^3c^3, b^2c^2, b^2c^3, a^2c^2, a^2bc^2, a^2b^2c^2, a^2b^3c^2, (b, b^6, \lambda)\}$ which generates $L_8$. The system will produce a unique derivation step for any word in $L_8$, starting from an axiom by inserting the string $b^6$, which shows $\gamma''_8$ is 0-ambiguous. As the system uses the following insertion rule $(b, b^6, \lambda)$, the system $\gamma''_8$ is minimal in the measures $MLen - LCon, MLen - RCon, TLen - LCon, MLen - RCon$.

Corollary 7: There are pseudo inherently 0-ambiguous insertion languages in the measures $M_1 \in \{M\Ax, TAx, MLen - InsStr, TINS - Str, mLen - InsStr\}$ and $M_2 \in \{MLen - LCon, MLen - RCon, TLen - LCon, TLen - RCon\}$.

Theorem 9: There are pseudo inherently 0-ambiguous insertion languages with respect to $M_1 \in \{Ax, mLen - InsStr\}$ and $M_2 \in \{MLen - RCon, mLen - RCon, TLen - LCon, TLen - RCon + InsStr, MLen - RCon + InsStr\}$.

Proof: Let the language $L_9 = \{ba^{2n} | n \geq 1\} \cup \{a^{4n}c | n \geq 1\} \cup \{ba^nc | n \geq 0\}$. The following 0-ambiguous insertion system $\gamma_9$ can be used to generate $L_9$. $\gamma_9 = \{(a, b, c), \{ba^2, a^4c, bc, bac\}, \{(b, a^2, \lambda), (\lambda, a^4, c)\}\}$. The system $\gamma_9$ is minimal with respect to $\{Ax, mLen - InsStr\}$. First, we will prove for the measure $mLen - InsStr$. From the system $\gamma_9$, it is easy to see that $mLen - InsStr(\gamma_9) \leq 2$. To generate the strings of the form $ab^k, k \geq 0 \in L_9$, the insertion rule should have the string $a$. However, if such an insertion string is present in any of the insertion rules, then the system $\gamma_9$ may generate some strings $ba^{2n}, p \geq 1$ and $a^4qc, q \geq 1$ which doesn’t $\notin L_9$. From the above claim, all the parts of $L_9$ cannot be produced by the insertion string $a$. So, obviously the minimum length of the insertion string should be $a^2$. Therefore, we conclude $mLen - InsStr(\gamma_9) \geq 2$.

Next, we will discuss on the measure $Ax$. Any system which generates $L_9$ will have three axioms $ba^2, a^4c, bc$. Next, we will discuss why the system $\gamma_9$ should have an axiom $bac$. If the system is not having the axiom $bac$, probably it can be generated by using the axiom $bc$ by inserting the string $a$. But previously we have proved that the system cannot have $a$ as the string to be inserted. Therefore, it implies $bac$ should be present in the axiom. Therefore, the system $\gamma_9$ is minimal in the measure $Ax$.

Consider any system $\gamma'_9$ which generates $L_9$. The system $\gamma'_9$ is minimal in the measure $Ax$, $mLen - InsStr$. In order to claim $\gamma'_9$ is 0-ambiguous, let us take the strings $ba^{2n}$ and $a^4vc \in L_9$, for a large values of $r$ and $s$. To produce the words of the form $ba^{2p}$, $p \geq 1$ and $a^{4q}c$, $q \geq 1$, the insertion system $\gamma'_9$ should have strings of the form $a^{2t}$, $t \geq 1$ and $a^{4u}$, $u \geq 1$ respectively. Consider a word $ba^{4m+2au}c \in L_9$, for $m \geq 1$, $n \geq 0$. The above word can be achieved from two unique axioms $bc$ and $bac$. Starting from the axiom $bc$, the word $ba^{4m+2au}c$ can be obtained by inserting the strings $a^{2t}$, $m$-times and $a^{4u}$, $u$-times. On the other hand, the word $ba^{4m+2au}c$ can be derived from the axiom $bac$. In one derivation, the string $a^{2t}$ can be inserted for $m - i_1$-times, $i_1 \geq 1$. In another derivation, the string $a^{4u}$ can be inserted for $n + i_2$-times, $i_2 \geq 1$. Thus, the word $ba^{4m+2au}c$ is obtained from two different axioms $bc$, $bac$. Therefore, the system $\gamma'_9$ is 0-ambiguous. Next, we have to prove the $L_9$ is 0-ambiguous, by showing there exists an 0-ambiguous system $\gamma''_9 = \{(a, b, c), \{ba^2, ba^4, a^4c, bc, bac, ba^2c, ba^4c\}, \{(a, a^4, \lambda)\}\}$ which generates $L_9$. The system will produce a unique derivation step for any word in $L_9$, starting from an axiom by inserting the string $a^4$, which shows $\gamma''_9$ is 0-ambiguous. As the system uses the following insertion rule $(a, a^4, \lambda)$, the system $\gamma''_9$ is minimal in the measures $\{MLen - RCon, mLen - RCon, TLen - RCon + InsStr, MLen - RCon + InsStr\}$.

Corollary 8: There are pseudo inherently 0-ambiguous insertion languages in the measures $M_1 \in \{M\Ax, TAx, mLen - InsStr\}$ and $M_2 \in \{MLen - RCon, mLen - RCon, TLen - LCon, TLen - RCon + InsStr, MLen - RCon + InsStr\}$.

Table 5 shows the different trade-off results acquired for various blends of the ambiguity levels and descriptional complexity measures. The intersecting entry at $M_1$ and $M_2$ shows there exists a pseudo inherently ambiguous insertion languages with respect to the measure $M_1$ and $M_2$. For example, the intersection of row 7 ($mLen - LCon$) and column 9 ($TLen - LCon$) indicates the language $L_5$ which is pseudo inherently 4-ambiguous with respect to the measures $M_1 \in mLen - LCon$ and $M_2 \in TLen - LCon$. In Table 4, the intersection of ($M_1$, $M_2$) entries that are empty are left as open problems. The entries which are in diagonal and those marked by • are not suitable for the trade-off study.

V. APPLICATION OF THE TRADE-OFF RESULTS

In this section, we analyze the application and significance of the trade-off in natural languages, modelling of bio-molecular structures. Before moving on to the application, first, we will discuss about the controlling parameters and limitations of
the proposed trade-off study. Given an insertion system, the weight of the system is the sum of $n$, $i$, $j$. The $n$ denotes the maximal length of insertion string and $i$, $j$ denotes the maximal length of the left and right context used in insertion rules. The weight of a insertion system is given as $(n, i, j)$. Several attempts were made on the insertion systems to characterize recursively enumerable languages with a lesser weights. In addition to that many variants has been introduced such as universal matrix insertion grammars, graph controlled insertion-deletion systems, path controlled insertion systems, insertion-deletion P Systems, Context-free insertion-deletion systems [9], [10], [11], [19], [31]. Natural languages such as English, Dutch has some grammatical structures that are beyond the power of context-free languages. As insertion system can characterize recursively enumerable languages the system can be considered as one of the prominent grammar models in generating natural language constructs and modelling the bio-molecular structures. The computational completeness of the insertion systems mainly depends on the weights used in the insertion rules. In practical, such weights will play a limiting factor while generating the natural language constructs and modelling the bio-molecular structures. Despite of such practical difficulty the application we had investigated in this paper will throw a new light on theoretical study on the trade-off.

A. APPLICATION OF THE ANALYZED TRADE-OFF IN NATURAL LANGUAGES

Syntactic and semantic ambiguity deserves a special attention in natural, programming and artificial languages. As the programming language constructs are mainly based on syntax and semantic rules handling these ambiguities is not a great deal of interest, whereas in natural languages handling syntactic ambiguity is easier when compared to semantic ambiguity. The main reason is, while dealing with the natural languages, one sentence (or a word) can convey different meaning. Even in Google translator, if the translation is carried out word by word the meaning may be different from the source to the target language. Under these circumstances, natural languages should be translated (stored) in an unambiguous manner. As we know, for every natural/programming/artificial language, there is a grammar $G$, such that $L(G) = L$. To generate the natural languages such as English, Dutch, we need grammars that are beyond the (generative) capability of Context-free grammar [5], [39]. In addition to that, many natural languages has the existence of sentences beyond context free [7], [28]. In this regard, to generate (store) such natural languages the grammar $G$ which generates $L$ should be unambiguous and at the same time it should be minimal in terms of measures. In practical, such a minimal unambiguous system will not be there for all languages. Under these, circumstances a necessary trade-off needs to be studied between the (descriptive) complexity measures and ambiguity.

To prove why such a trade-off is very important in natural languages, lets consider the following sentence, They are hunting dogs. The sentence is syntactically correct, where the sentence is having semantic ambiguity, as it can be elucidated in a different manner. The different interpretation of the above mentioned sentence can be: Whether any group is hunting for dogs? or Whether the category of dogs belongs to the hunting type or Whether the phrase hunting dogs refers to a music band or a basketball team or a secret code. In fact, the right phrases of the sentence are They are, They are hunting. They are dogs, They are hunting dogs. Assume that, we want to construct an insertion system which generates the above sentence. As there is no concept of non-terminals(variables) in insertion system, it can be called as pure grammars. Since the insertion system is a pure grammar, every derivation step should represent a correct phrase, the correct phrases are They are, They are hunting, They are dogs, They are hunting dogs. Consider, ‘They are’ is an axiom and the insertion rules are of the form: (They are, dogs, $\lambda$) and (They are, hunting, $\lambda$).

By using the above axiom and the insertion rules, the derivations can be of the forms (the underlined words indicates the inserted string): (1) They are $\Rightarrow$ They are dogs $\Rightarrow$ They are hunting dogs, which gives all the three correct phrases. (2) They are $\Rightarrow$ They are hunting $\Rightarrow$ They are dogs hunting, which is not a correct phrase. So, with the above insertion rules all the correct phrases cannot be generated. However, if we consider three insertion rules (They are, dogs, $\lambda$), (They are, hunting, $\lambda$), (They are, hunting, dogs) all the three correct phrases can be derived from the axiom or else using different insertion rules we may get all correct phrases of the sentence, but the number of insertion rules will be more. So, to derive the above sentence, we need three insertion rules.

Such sentences can be stored compactly if there exists an unambiguous system which generates it, but may happen to be not minimal with respect to measure(s). As insertion systems is found to be one of the prominent (rewriting) grammar mechanisms, the system can be recognized to be one of the fit (rewriting) mechanisms to generate natural languages [30]. The above example clearly shows that the sentence can be generated by an unambiguous system but not minimal in terms of components used to iterate the sentence. The above case study explicit the importance of studying the trade-off in natural languages.

B. APPLICATION OF THE ANALYZED TRADE-OFF IN MODELLING OF BIO-MOLECULAR STRUCTURES

In computational biology there are lot of research problems needs to be addressed based on the gene sequence such as gene structure prediction, gene sequence alignment, bio-molecular modelling, construction of phylogenetic trees. Such gene structure prediction, bio-molecular modelling problems are effectuated by progressing with relevant pattern matching algorithms. The above discussed computational biological problems are somewhat akin to investigating the structural frameworks in computational linguistics. The gene structure prediction, bio-molecular modelling problems can be handled in an effective and succinct manner, if there exists a unique grammar model/system which generates/models it.
To model or predict the structures, first, it should be expressed as a gene sequence. Such sequences can be visualized as strings formed over the four basic chemical symbols $a, t, g$ and $c$ ($\Sigma_{DNA}$). The complementary of the above four chemical symbols is given as $\bar{a} = t$, $\bar{g} = c$, $\bar{t} = a$, $\bar{c} = g$. As the bio-molecular structures can be expressed in terms of (gene) sequences it has kindled the researchers to study the connection and application of formal language theory and computational biology [42]. In addition to that, the genetic structural descriptions that are found in the bio-molecular structures has some coherence in natural language constructs such as triple agreements: $L_{ot} = \{a^i b^j c^k d^n | n \geq 1\}$, quadruple agreements: $L_{qua} = \{a^i b^j c^k d^m | n, m \geq 1\}$, crossed dependencies: $L_{cd} = \{a^i b^j c^k d^m | n, m \geq 1\}$, copy language: $L_{cp} = \{ww | w \in \{a, b\}^*\}$. More precisely, $L_{ot}$ and $L_{qua}$ resembles triple and quadruple helix structure. Likewise, $L_{cd}$ and $L_{cp}$ has some pertinence with pseudoknot and attenuator structures respectively [41], [43]. For modelling of the bio-molecular structures that occurs at intramolecular, intermolecular and RNA secondary structures, we refer to [17], [25], [26], [27], [29], [44].

Before we discuss about the application of the trade-off in modelling of the bio-molecular structures, first, we will show that insertion system is capable of modelling some of the bio-molecular structures. Consider the following bio-molecular structures like hairpin, stem and loop, orthodox. The language description and modelling of the above structures by using insertion system are given in the following lemmas. In the forthcoming lemmas $y \in \Sigma_{DNA}$, the counterpart of $y$ is $y'$, $\bar{w}^{R}$, $\bar{y}^{R}$ is the complementary reversal of the string $w, u$ respectively.

Lemma 1: The hairpin language $L_{hp} = \{w = \bar{y}^{R} | w \in \Sigma_{DNA}^*\}$ can be spawned by the insertion system $\gamma_{hp} = (\{y, y'\}, \{\lambda, y'\}, \{y, y', y'\})$.

Lemma 2: The stem and loop language $L_{sl} = \{uv\bar{u}^{R} | u, v \in \Sigma_{DNA}^*\}$ can be achieved by insertion system $\gamma_{sl} = (\{y, y'\}, \{\lambda, y\}, \{(y, y'), \{y, y', y\}, (y, y, y')\})$.

A string $w$ is said to be orthodox over $\Sigma_{DNA}$(complementary alphabet) iff it fulfills the following conditions (i) it should be an empty string $\lambda$, or (ii) the string obtained by the insertion of $y'y'$ anywhere in an orthodox string. A language which contains only orthodox strings is called orthodox language $L_{od}$.

Lemma 3: The orthodox language $L_{od}$ can be spawned by the insertion system $\gamma_{od} = (\{y, b'\}, \{\lambda\}, \{(\lambda, y'), (\lambda, y)\})$.

1) AMBIGUITY ISSUES IN ORTHODOX LANGUAGE $L_{od}$

In this subsection, we will discuss about the ambiguity issues in the orthodox language $L_{od}$. In the derivations/descriptions/sequence the underlined string denotes the inserted gene sequence and $^\dagger$ denotes the position where the gene sequence is to be inserted.

**Case 1:** Consider the string $cgata\bar{g}c\bar{c}g \in L_{od}$, which can be obtained from two different axioms.

| Derivation 1: | $cg \dagger \rightarrow cg\bar{a}\dagger \rightarrow cgat\dagger \rightarrow cgatg\dagger \rightarrow cgatgc\dagger \rightarrow cgatgccccg. $ |

In both the derivations, the same sequence $cgatgccccg$ is derived from two different axioms $cg$ and $at$. Therefore, the system $\gamma_{od}$ evinces 0-ambiguous.

**Case 2:** Consider an orthodox string $cgata\bar{g}c\bar{c}g \in L_{od}$, which can be obtained by two different ordered CS:

Ordered CS 1: $\dagger \lambda \rightarrow ta \dagger \rightarrow tagc \dagger \rightarrow tagccg \dagger \rightarrow tcagc \dagger \rightarrow cgtagccg.$

Ordered CS 2: $\dagger \lambda \rightarrow ta \dagger \rightarrow cgta \dagger \rightarrow cgtaat \rightarrow tcagc \dagger \rightarrow cgtagccg.$

In CS 1, the order of gene sequence used by the insertion rules are $ta, gc, at, cg$, whereas in CS 2, the order of gene sequence used by the insertion rule are $ta, cg, at, gc, cg$. Thus, the gene sequence $cgtagccg$ can be derived by two different ordered CS. Therefore, the system $\gamma_{od}$ evinces 1-ambiguous also.

**Case 3:** Consider the string $ateg\bar{c}gta \in L_{od}$, which can be derived in two different descriptions by $\gamma_{od}$ which are given below:

| Description 1: | $cg \dagger \rightarrow cgcg \dagger \rightarrow cgc\bar{g}a \rightarrow ateg\bar{c}gta.$ |
| Description 2: | $cg \dagger \rightarrow atcg \dagger \rightarrow atcg \dagger \rightarrow ta \rightarrow ateg\bar{c}gta.$ |

In both the descriptions the axioms are same $cg$ and the contexts used in the insertion rules ($\lambda, \lambda$) are also same, but the position where the inserted gene sequence $y'y'$ are different. Therefore, the system $\gamma_{od}$ is 5-ambiguous also.

The above example shows a clear evidence on the existence of different levels of ambiguity for the same language $L_{od}$ on different gene sequences. In addition to that, the above (ambiguity) example reveals that analysis of the ambiguity issues in gene sequences has to be carried out with utmost care because ambiguity issues plays a pivot role in some of the computational biology problems such as protein sequence analysis, parallel gene recognition, prediction of gene locations. For more practical applications on the importance of ambiguity in gene sequences, we refer to [1], [2], [3], [4]. The evolutionary relationship among the various biological species can be depicted in terms of trees. The trees can be derived based on the differences and similarities among the species. Such trees are called as phylogenetic trees [45]. The phylogenetic trees plays an important role in DNA/Protein sequence divergence problem [8].

The axiom, intermediates sequence and the final sequence to be generated can be represented as a tree. If the intermediate gene sequences are different then we will have more than one phylogenetic trees for the same gene sequence. Such a study on the different intermediate sequences will help us to study more on the inheritance properties. The following example of a phylogenetic tree will give a better understanding on the ambiguity. One such phylogenetic tree is shown.
TABLE 5. Obtained trade-off results.

| Unambiguous & M2 → | Az MAX TAx | mL en - LCon | mL en - RCOn | mL en - LCon | mL en - RCOn | TLen - LCon | TLen - RCOn | TLen - LCon+ InsStr | TLen - RCOn+ InsStr | MLen - LCon | MLen - RCOn+ InsStr | MLen - InsStr | MLen - RCon+ InsStr | TINS - Str |
|---------------------|-----------|---------------|--------------|--------------|--------------|-------------|-------------|---------------------|---------------------|--------------|----------------------|--------------|----------------------|-------------|
| 100522              | VOLUME 10, 2022 | L4 - 4A       | L6 - 2A      | L6 - 2A      | L9 - 0A      | L4 - 4A     | L5 - 4A     | L6 - 2A             | L6 - 2A             | L9 - 0A      | L9 - 0A               | L5 - 4A      | L9 - 0A               | L9 - 0A     |
| 100522              | VOLUME 10, 2022 | L8 - 0A       | L8 - 0A      | L9 - 0A      | L9 - 0A      | L8 - 0A     | L8 - 0A     | L9 - 0A             | L9 - 0A             | L9 - 0A      | L9 - 0A               | L9 - 0A      | L9 - 0A               | L9 - 0A     |
| 100522              | VOLUME 10, 2022 | L3 - 5A       | L3 - 5A      | L3 - 5A      | L3 - 5A      | L3 - 5A     | L3 - 5A     | L3 - 5A             | L3 - 5A             | L3 - 5A      | L3 - 5A               | L3 - 5A      | L3 - 5A               | L3 - 5A     |
| 100522              | VOLUME 10, 2022 | L7 - 2A       | L7 - 2A      | L7 - 2A      | L7 - 2A      | L7 - 2A     | L7 - 2A     | L7 - 2A             | L7 - 2A             | L7 - 2A      | L7 - 2A               | L7 - 2A      | L7 - 2A               | L7 - 2A     |
| 100522              | VOLUME 10, 2022 | L5 - 4A       | L5 - 4A      | L5 - 4A      | L5 - 4A      | L5 - 4A     | L5 - 4A     | L5 - 4A             | L5 - 4A             | L5 - 4A      | L5 - 4A               | L5 - 4A      | L5 - 4A               | L5 - 4A     |
| 100522              | VOLUME 10, 2022 | L3 - 5A       | L3 - 5A      | L3 - 5A      | L3 - 5A      | L3 - 5A     | L3 - 5A     | L3 - 5A             | L3 - 5A             | L3 - 5A      | L3 - 5A               | L3 - 5A      | L3 - 5A               | L3 - 5A     |
| 100522              | VOLUME 10, 2022 | L3 - 5A       | L3 - 5A      | L3 - 5A      | L3 - 5A      | L3 - 5A     | L3 - 5A     | L3 - 5A             | L3 - 5A             | L3 - 5A      | L3 - 5A               | L3 - 5A      | L3 - 5A               | L3 - 5A     |
| 100522              | VOLUME 10, 2022 | L3 - 5A       | L3 - 5A      | L3 - 5A      | L3 - 5A      | L3 - 5A     | L3 - 5A     | L3 - 5A             | L3 - 5A             | L3 - 5A      | L3 - 5A               | L3 - 5A      | L3 - 5A               | L3 - 5A     |
| 100522              | VOLUME 10, 2022 | L3 - 5A       | L3 - 5A      | L3 - 5A      | L3 - 5A      | L3 - 5A     | L3 - 5A     | L3 - 5A             | L3 - 5A             | L3 - 5A      | L3 - 5A               | L3 - 5A      | L3 - 5A               | L3 - 5A     |
To reach the Human Virus leaf node, the path can be explored as: \( \text{RootNode} \rightarrow \text{IN2} \rightarrow \text{IN3} \rightarrow \text{IN4} \rightarrow \text{IN5} \). Starting from the root node there is a unique path up to the intermediate node IN5, whereas, after reaching the purple color intermediate node IN5, there exists two paths. One path will be Violet line from \( \text{IN5} \rightarrow \text{HV} \) and another path will be Yellow line from \( \text{IN5} \rightarrow \text{HV} \). So, the Human Virus node can be reached by two different paths from the root node. The above scenario clearly shows that a different perspective can be given in the visualization of ambiguity in phylogenetic trees. On the other hand, while predicting the gene structure, we need an optimal system and at the same time the system which generates/models the bio-molecular structure should be unambiguous. Consider the system \( y_{od} = ([y, b'], \lambda, ((\lambda, yy', \lambda)) \) which generates the \( L_{od} \). One insertion rule is enough to generate all the strings in \( L_{od} \). The system \( y_{od} \) is minimal \( \{Ax, MLen − LCon, MLen − RCon, mLen − LCon, mLen − RCon, TLen − LCon, TLen − TCon\} \). The language \( L_{od} \) can be generated by an unambiguous system but definitely the unambiguous system will not be minimal in the above mentioned measures. This example shows the importance and application of the trade-off study between complexity measures and ambiguity levels in modelling of the bio-molecular structures.

VI. COMPARATIVE STUDY

In this section, we discuss about the comparative study of trade-off results obtained for the insertion systems and its applications in natural languages, modelling of bio-molecular structures. Table 6 shows the comparative study of the proposed results and applications with other relevant grammar models. From the comparative study, it has a clear evidence, that the insertion systems, insertion-deletion systems, variants of insertion deletion systems are mainly motivated towards reducing the weights in simulating the recursively enumerable languages by means of suitable normal forms where as, in this paper, we have defined some new descriptive complexity measures, analyzed the trade-off between ambiguity levels and descriptional complexity measures. In addition to that, we have discussed about the application of the analyzed trade-off which was missing in the various research work carried out on insertion systems.

VII. CONCLUSION

In this paper, we defined twelve new descriptional complexity measures namely \( MLen − \text{InsStr}, mLen − \text{InsStr}, MLen − LCon, MLen − RCon, mLen − LCon, mLen − RCon, TLen − LCon, TLen − RCon, TLen + \text{InsStr}, TLen − RCon + \text{InsStr}, MLen − LCon + \text{InsStr}, MLen − RCon + \text{InsStr} \) based on the components used in the derivations. Later, we discussed the trade-off between the newly defined ambiguity levels and measures in insertion systems by showing that there exists pseudo inherently ambiguous insertion languages which can be generated by an ambiguous system that are minimal in \( M_1 \) and unambiguous if they are minimal in \( M_2 \). Finally, we have studied the application of the investigated trade-off in natural languages and modelling of bio-molecular structures. Analyzing the trade-off between measures and
ambiguity levels which are not considered in this paper is left as a future work. More, precisely it would be interesting such a trade-off results can be obtained for the ambiguity levels 1 and 3. As insertion systems can be recognized as a good model to generate some of the programming language constructs, analyzing the trade-off in programming languages would be another line of future work.

ACKNOWLEDGMENT

An earlier version of this paper was presented at the International Conference (ICRTC-2022) held at SRMIST, New Delhi.

REFERENCES

[1] M. J. Bishop and C. J. Rawlings, Nucleic Acid and Protein Sequence Analysis. London, U.K.: Oxford Univ. Press, 1987.
[2] M. Borodovsky and J. McIninch, “Recognition of genes in DNA sequence with ambiguities,” Biosystems, vol. 30, nos. 1–3, pp. 161–171, 1993.
[3] M. Borodovsky and J. McIninch, “Prediction of gene locations using DNA Markov chain models,” in Bioinformatics: Supercomputing and Complex Genome Analysis, 1993, pp. 231–248.
[4] M. Borodovsky and J. McIninch, “GENMARK: Parallel gene recognition for both DNA strands,” Comput. Chem., vol. 17, no. 2, pp. 123–133, Jun. 1993.
[5] J. Bresnan, R. M. Kaplan, S. Peters, and S. Zaenen, “Cross-serial dependencies in Dutch,” Linguistic Inquiry, vol. 13, no. 4, pp. 613–663, 1982.
[6] C. S. Calude and G. Paun, Computing With Cells and Atoms An Introduction to Quantum, DNA and Membrane Computing. London, U.K.: Taylor & Francis, 2001.
[7] N. Chomsky, “Formal properties of grammars,” in Handbook of Mathematical Psychology, R. D. Luce, Ed. New York, NY, USA: Wiley, 1963, pp. 323–418.
[8] S. Choudhuri, Bioinformatics for Beginners: Genes, Genomes, Molecular Evolution, Databases and Analytical Tools. Amsterdam, The Netherlands: Elsevier, 2014.
[9] H. Fernau, L. Kuppusamy, and S. Verlan, “Universal matrix insertion grammars with small size,” in Proc. Int. Conf. Unconventional Computation. Natural Comput. Springer, Cham, Jun. 2017, pp. 182–193.
[10] H. Fernau, L. Kuppusamy, and I. Raman, “Graph-controlled insertion–deletion systems generating language classes beyond linearity,” in Proc. Int. Conf. Descriptive Complexity. Formal Syst. Cham, Switzerland: Springer, Jun. 2017, pp. 128–139.
[11] H. Fernau, L. Kuppusamy, and I. Raman, “On path-controlled insertion–deletion systems,” in Acta Inf., vol. 56, no. 1, pp. 35–59, Feb. 2019.
[12] H. Fernau, L. Kuppusamy, and I. Raman, “Computational completeness of simple semi-conditional insertion–deletion systems,” in Proc. Int. Conf. Unconventional Computation. Natural Comput. Cham, Switzerland: Springer, Jun. 2018, pp. 86–100.
[13] H. Fernau, L. Kuppusamy, and I. Raman, “Computational completeness of path-structured graph-controlled insertion–deletion systems,” in Proc. Int. Conf. Implement. Appl. Automata. Cham, Switzerland: Springer, Jun. 2017, pp. 89–100.
[14] D. Haussler, “Insertion Languages,” in Inf. Sci., vol. 31, no. 1, pp. 77–89, 1983.
[15] J. Gruska, “Descriptive complexity of context-free languages,” in Mathematical Foundations of Computer Science. Bratislava, Slovakia: High Tatras, 1973, pp. 71–84.
[16] D. Haussler, “Insertion languages,” in Inf. Sci., vol. 31, no. 1, pp. 77–89, Oct. 1983.
[17] T. Head, “Formal language theory and DNA: An analysis of the generative capacity of specific recombinant behaviors,” Bull. Math. Biol., vol. 49, pp. 737–750, Nov. 1987.
[18] L. Hie, “On ambiguity in internal contextual languages,” in Proc. 2nd Int. Conf. Math. Lang., C. Martin-Vide, Eds. Amsterdam, The Netherlands: John Benjamins, 1997, pp. 29–45.
[19] N. S. Krishna and R. Rama, “Insertion-deletion P systems,” in Proc. Int. Workshop DNA-Based Comput. Berlin, Germany: Springer, 2001, pp. 362–370.
[20] K. Lakshmanan, “Incompatible measures of internal contextual grammars,” in Proc. DCFS, C. Mereghetti, Eds. Como, Italy, 2005, pp. 253–260.
[21] K. Lakshmanan, “A note on ambiguity of internal contextual grammars,” Theor. Comput. Sci., vol. 369, pp. 436–441, Dec. 2006.
[22] K. Lakshmanan, M. Anand, and K. Krithivasan, “On the trade-off between ambiguity and measures in internal contextual grammars,” in Proc. DCFS, C. Câmpineanu and G. Pighizzini, Eds., 2008, pp. 216–223.
[23] L. Kuppusamy, A. Mahendran, and K. Krithivasan, “On the ambiguity of insertion systems,” Int. J. Found. Comput. Sci., vol. 22, no. 7, pp. 1747–1758, Nov. 2011.
[24] K. Lakshmanan, M. Anand, and K. Krithivasan, and K. Mohammed, “On the study of ambiguity and the trade-off between measures and ambiguity in insertion–deletion languages,” Nano Commun. Netw., vol. 2, nos. 2–3, pp. 106–118, 2011.
[25] K. Lakshmanan, M. Anand, and S. N. Krishna, “Matrix insertion-deletion systems for bio-molecular structures,” in Proc. ICDCCIT in Lecture Notes in Computer Science, vol. 6536, R. Natarajan and A. Ojo, Eds., 2011, pp. 301–311.
[26] K. Lakshmanan, M. Anand, and E. V. Clergerie, “Modelling intermolecular structures and defining ambiguity in gene sequences using matrix insertion-deletion systems,” in Biology, Computation and Linguistics, New Interdisciplinary Paradigms, vol. 228. Amsterdam, The Netherlands IOS Press, 2011, pp. 71–85.
[27] L. Kuppusamy and A. Mahendran, “Modelling DNA and RNA secondary structures using matrix insertion–deletion systems,” Int. J. Appl. Math. Comput. Sci., vol. 26, no. 1, pp. 245–258, Mar. 2016.
[28] D. T. Langendoen and P. M. Postal, The Vastness of Natural Language. Oxford, U.K.: Blackwell, 1984.
[29] A. Mahendran and L. Kuppusamy, “Formal language representation and modelling structures underlying RNA folding process,” in Theoretical Computer Science and Discrete Mathematics (Lecture Notes in Computer Science), vol. 10398, S. Arunugam, J. Bagga, L. Beinke, and B. Panda, Eds. Cham, Switzerland: Springer, 2017, pp. 20–29.
[30] S. Marcus, “Contextual grammar,” Rev. Roum. Pures Appl., vol. 14, pp. 1525–1534, 1969.
[31] M. Margenstern, G. Paun, Y. Rogozhin, and S. Verlan, “Context-free insertion–deletion systems,” Theor. Comput. Sci., vol. 330, no. 2, pp. 339–348, (2005).
[32] C. Martin-Vide, J. Miguel-Verges, A. G. Paun, and A. Salomaa, “Attempting to define the ambiguity in internal contextual languages,” in Proc. 2nd Intl. Conf. Math. Ling., C. Martin-Vide, Ed. Amsterdam, The Netherlands: John Benjamins, 1997, pp. 59–81.
[33] G. Paun, Contextual Grammars. Bucuresti, Romania: Publishing House of the Romanian Academy of Sciences, 1982.
[34] G. Paun, Marcus Contextual Grammars. Norwell, MA, USA: Kluwer, 1997.
[35] G. Paun, G. Rozenberg, and A. Salomaa, DNA Computing, New Computing Paradigms. Cham, Switzerland: Springer, 1998.
[36] G. Paun, Membrane Computing An Introduction. Cham, Switzerland: Springer, 2002.
[37] G. Paun, “On the camp of contextual grammars with choice,” Stud. Cerc. Matem., vol. 27, pp. 559–569, 1975.
[38] G. Paun, “Further remarks on the syntactical complexity of Marcus contextual languages,” Ann. Univ. Buc., Ser. Matem.-Inform., pp. 72–82, 1991.
[39] G. H. Pullum, “On two recent attempts to show that english is not a context-free language,” Comput. Linguistics, vol. 10, nos. 3–4, pp. 182–186, 1980.
[40] G. Rozenberg and A. Salomaa, Handbook of Formal Languages, Word, Language, Grammar, vol. 1. Cham, Switzerland: Springer, (1997).
[41] E. Rivas and S. R. Eddy, “The language of DNA: A formal grammar that includes pseudoknots,” Bioinformatics, vol. 16, no. 4, pp. 334–340, Apr. 2000.
[42] D. B. Searls, “The linguistics of DNA,” Amer. Scientist, vol. 80, no. 6, pp. 579–591, 1992.
[43] D. B. Searls, “The computational linguistics of biological sequences,” in Artificial Intelligence and Molecular Biology, L. Hunter. Monto Park, CA, USA: AAAI Press, 1993, pp. 47–120.
[44] D. B. Searls, “Formal grammars for intermolecular structure,” in Proc. 1st Int. Symp. Intell. Neural Biol. Syst. (INBS), May 1995, pp. 30–37.
[45] J. C. Setubal and J. Meidanis, Introduction to Computational Molecular Biology, Brooks, vol. 5, no. 9. Pacific Grove, CA, USA: Cole Publishing Company, 1997, p. 18.
ANAND MAHENDRAN received the B.E. degree in computer science and engineering from Madras University, India, in 1998, the M.E. degree in computer science and engineering from Anna University, India, in 2005, and the Ph.D. degree in computer science and engineering from the Vellore Institute of Technology (VIT), Vellore, India, in 2012. He worked as a Postdoctoral Research Fellow with the Laboratory of Theoretical Computer Science, National Research University, Higher School of Economics (HSE), Moscow, Russia. He is currently an Associate Professor (Senior) with the School of Computer Science and Engineering, VIT. He has published more than 50 papers in international journals and refereed international conferences. His research interests include formal language and automata theory, and bio-inspired computing models.

KUMAR KANNAN received the bachelor’s degree in computer science and engineering (CSE) from Madras University, in 1998, the master’s degree in CSE from Pondicherry University, in 2004, and the Ph.D. degree in CSE from the Vellore Institute of Technology, Vellore, in 2016. From 1998 to 2001, he worked as a Programmer/Software Engineer at Computer Access Ltd., and Bhari Information Technology, and later he worked as a Lecturer by passion. Since 2005, he has been an Associate Professor with the VIT. He is currently a Researcher and an Engineer who has been working in software engineering, recommender patterns, context-aware patterns, and machine learning applications. He has presented several works in the area of software patterns and machine learning, and published various articles, and book chapters in international and national level.

MOHAMED HAMADA (Senior Member, IEEE) received the Ph.D. degree from the University of Tsukuba, Japan, under the scholarship from the Japanese Government (MEXT). He got a Japan International Cooperation Agency (JICA) Fellowship for six months. He is currently a Senior Associate Professor at The University of Aizu, Aizuwakamatsu, Fukushima, Japan. He is a Regular Visiting Professor at Fatih University, Istanbul, Turkey, and the African University of Science and Technology, Abuja, Nigeria. He leaded several funded research projects and supervised several graduate (M.Sc. and Ph.D.) and undergraduate students. He edited three books and has more than 100 papers in major international journals and conferences published by major publishers, such as IEEE, ACM, Elsevier, and Springer. His research interests include artificial intelligence and learning technologies. He is also interested in smart devices (such as smartphones and tablets) applications development and innovation. He is a member of the editorial board of several international journals and a program committee member of several international conferences. He is a member of the IEEE Technical Committee on Multimedia, and the IEEE Technical Committee on Learning Technologies. He is a Senior Member of IEEE and ACM Computer Societies.

MANUEL MAZZARA received the Ph.D. degree in computing science from the University of Bologna, Italy. He is currently a Professor of computer science at Innopolis University, Russia, with a research background in software engineering, service-oriented architecture, concurrency theory, formal methods, and software verification. He is also the Director of the Institute of Software Development and Engineering and the Head of the International Cooperation Office at Innopolis University. He published many relevant and highly-cited papers, in particular in the field of service engineering and software architectures. He has collaborated with European and U.S. industries, plus governmental and inter-governmental organizations, such as the United Nations, always at the edge between science and software production. The work conducted by Dr. Manuel Mazzara and his team in recent years focuses on the development of theories, methods, tools, and programs covering the two major aspects of software engineering: the process side, related to how we develop software, and the product side, concerning the results of this process.

***