DE GRUYTER OPEN

Int. J. Appl. Math. Comput. Sci., 2016, Vol. 26, No. 1, 245–258
DOI: 10.1515/amcs-2016-0017

amcs

MODELLING DNA AND RNA SECONDARY STRUCTURES USING MATRIX INSERTION–DELETION SYSTEMS

LAKSHMANAN KUPPUSAMY [a], ANAND MAHENDRAN [b,*]

[a] School of Computing Science and Engineering
VIT University, Vellore 632014, India
e-mail: klakshma@vit.ac.in

[b] Department of Computer Science, College of Computer Science and Information Systems
Jazan University, Jazan 45142, Kingdom of Saudi Arabia
e-mail: anandmahendran82@gmail.com

Insertion and deletion are operations that occur commonly in DNA processing and RNA editing. Since biological macromolecules can be viewed as symbols, gene sequences can be represented as strings and structures can be interpreted as languages. This suggests that the bio-molecular structures that occur at different levels can be theoretically studied by formal languages. In the literature, there is no unique grammar formalism that captures various bio-molecular structures. To overcome this deficiency, in this paper, we introduce a simple grammar model called the *matrix insertion–deletion system,* and using it we model several bio-molecular structures that occur at the intramolecular, intermolecular and RNA secondary levels.

Keywords: bio-molecular structures, insertion–deletion systems, intermolecular, intramolecular, secondary structures.

1. Introduction

Natural computing is a field of research that investigates various computing models and computational (algorithms) techniques that are inspired by nature. It is an interdisciplinary area that nudges natural science with computing science, and attempts to understand the world around us in terms of information processing. In the last few decades, natural computing which includes biologically inspired computing, has been pursued with a great deal of interest. This comprises *evolutionary computing* (Eiben and Smith, 2003), *membrane computing* (Calude and Paŭn, 2001), *genetic algorithms* (Goldberg, 1989), *DNA computing* (Păun *et al.*, 1998), *ant colony optimization* (Dorigo and Stutzle, 2004) and many other computing models that are inspired from biology and nature. The developments which have taken place in DNA computing inspired the definition and study of new theoretical models in formal language theory, such as *sticker systems, splicing systems, Watson–Crick automata, insertion–deletion systems* and *P systems* (Calude and Paŭn, 2001; Păun *et al.*, 1998; Păun, 2002).

Insertion–deletion systems are introduced to theoretically analyze the insertion and deletion operations that take place in gene sequences. These operations frequently occur in DNA processing and RNA editing. The insertion operation was first studied by Galiukschov (1982). A study of properties of the insertion operation was carried out by Haussler (1982; 1983). Informally, the insertion and deletion operations of an insertion–deletion system are defined as follows: If a string β is inserted between two parts w_1 and w_2 of a string w_1w_2 to get $w_1\beta w_2$, we call the operation *insertion*, whereas if a substring α is deleted from a string $w_1\alpha w_2$ to get w_1w_2, we call the operation *deletion*. Consider a production of the form $A \to cAB$ in a rewriting system. In a derivation step, if there is a presence of non-terminal A, then it will be replaced by cAB. But in the insertion–deletion system the derivations are obtained either by inserting or by deleting a string. As the system is not exactly based on the rewriting mechanism, it has attracted particular attention in the field of *formal language theory*.

The biological sequences that occur in DNA, RNA

*Corresponding author

and protein molecules can be considered words formed over well-defined chemical alphabets. The DNA molecule consists of sequences that are built of nucleotides, which are in four forms: $a(adenine), t(thymine), g(guanine), c(cytosine)$. The RNA molecule consists of sequences that are built of nucleotides, which are in four forms; a, $u(uracil)$, g, c. The complementary pair for RNA (DNA) is given as $\bar{a} = u(t)$, $\bar{u}(\bar{t}) = a$, $\bar{g} = c$ and $\bar{c} = g$. Based on the complementary pairs in chemical objects and other biological constraints, sequences form patterns and these patterns are considered structures. These structures play a vital role in governing the functionality and behavior of bio-molecules (Brendel and Busse, 1984; Searls, 1993).

Gene sequence prediction is considered one of the important and fundamental problems in computational biology. Such a sequence prediction problem is dealt with by developing suitable string (pattern) matching algorithms. The above mentioned problem is somewhat akin to analyzing the structural descriptions in computational linguistics. The following example shares a common point between formal languages and molecular strings. Consider the context-free language $L = \{ww^R \mid w \in \{a,b\}^*\}$, where w^R is the reversal of w. For example, if w is $aabb$, then w^R will be $bbaa$. Consider the gene sequence $cggcaacggc$. This gene sequence resembles the palindrome (context-free) language $\{ww^R \mid w \in \{a,u,g,c\}^*\}$. Also, there exist some relations between bio-molecular sequences and non-context-free natural language constructions such as *triple agreements:* $\{a^n b^n c^n \mid n \geq 1\}$, *crossed dependencies:* $\{a^n b^m c^n d^m \mid n,m \geq 1\}$ and *copy language:* $\{ww \mid w \in \{a,b\}^*\}$ (Searls, 1992; 1993; 2002). They are discussed in the next paragraph.

We now discuss briefly the bio-molecular structures that are frequently noticed in bio-molecules such as protein, DNA and RNA. Figure 1 shows two popular structures, *stem and loop* and *hairpin*, which can be modelled by context-free grammars (Searls, 1988). Figure 2 shows two structures, *pseudoknot* and *attenuator*, which are beyond the power of context-free grammars (Searls, 1992). In Figs. 1 and 2, the strings are obtained by reading the symbols as per directed dotted lines. The string $cuucaucagaaaaugac$ represents the stem and loop language (Fig. 1(a)) and the string $atcgcgat$ represents the hairpin language (Fig. 1(b)). The string $gcucgcga$ (Fig. 2(a)) represents the pseudoknot structure and the string $gucgacgucgac$ (refer Fig. 2(b)) represents attenuator structure. Figure 2 shows the coherence between the natural language constructs and the gene sequences. Figure 2(a) represents the pseudoknot structure, which has the crossed dependency pattern, and Figure 2(b) represents the attenuator structure, which has the copy language pattern. The formal language notations for such structures and for a few other structures are discussed in detail in the coming sections. For more details on genome structures, their corresponding languages and gene structure prediction using linguistic methods, we refer to the works of Brendel and Busse (1984), Chiang *et al.* (2006), Dong and Searls (1994), Durbin *et al.* (1998), as well as Searls (1988; 1992; 2002).

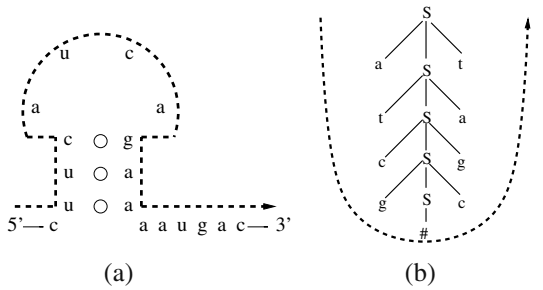

Fig. 1. Bio-molecular structures: stem and loop (where the ∘ denotes the complementary pair) (a), hairpin (where S is a non-terminal for the context-free grammar and # denotes the empty string) (b).

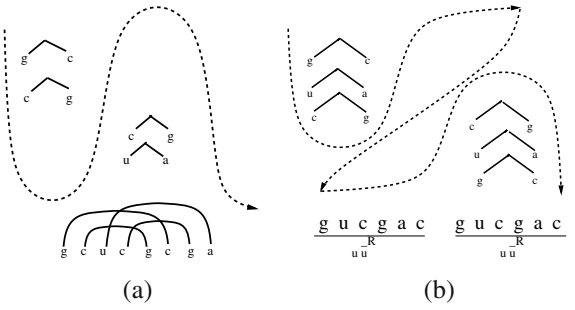

Fig. 2. Pseudoknot (a), attenuator (b).

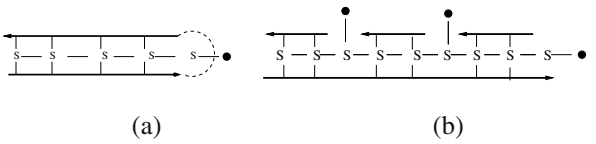

Fig. 3. Intermolecular structures: double stranded molecule (where S is a non-terminal for the context-free grammar) (a), nick language (where S is a non-terminal for the context-free grammar) (b).

The structures that are formed in RNA are mostly intermolecular. Figure 3 represents some of the intermolecular structures (Searls, 1995): (a) *double strand language* and (b) *nick language*, where the cut takes place at arbitrary positions, which is represented by a $\bullet$. The double strand language can be given as $\{w \bullet w^R \mid w \in \{a,u,g,c\}^*\}$. In a double stranded molecule, if the cut

is done at many places, then the molecule is said to be nicked.

The study of linguistic behavior of biological sequences using formal grammars was initiated in the work of Brendel and Busse (1984) as well as Head (1987). Soon after, it was carried out by Searls (1988; 1992; 1993). In this regard, in the last two decades, there have been many attempts made to establish the linguistic behavior of biological sequences by defining new grammar formalisms like *cut grammars*, *ligation grammars* (Searls, 1988; 1992; 1993) *crossed-interaction grammar* (Rivas and Eddy, 2000), *simple linear tree adjoining grammars* and *extended simple linear tree adjoining grammars* (Uemura *et al.*, 1999). These are capable of generating some of the biological structures mentioned above.

Like DNA and protein, RNA is also considered one of the important and essential macromolecules that occur in all forms of life. RNA structures are mainly classified as primary, secondary and tertiary structures. The primary structures of a nucleic acid molecule represent the exact sequence of nucleotides that forms the complete molecule. The secondary structures are a two dimensional representation formed by folding back onto itself the base pairing of complementary nucleotides (Watson–Crick pairs). The tertiary structures are 3D structures formed by a single molecule. 3D structures formed by more than one molecule are called quaternary structures. Study of such structures tends to be more complex as it is very difficult to predict the interactions between the molecules. In an RNA secondary structure, the basic

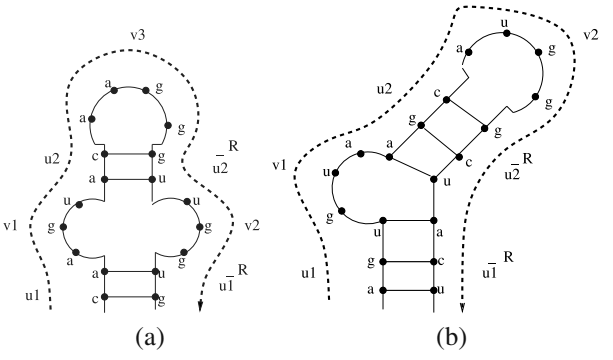

Fig. 4. RNA secondary structure: internal loop (a), bulge loop (b).

structural motifs can be classified as *stem and loop (H-loop)* (Fig. 1(a)), *internal loop (I-loop)* (Fig. 4(a)), *bulge loop (B-loop)* (Fig. 4(b)) and *multi branch loop (M-loop)* (Fig. 5(a)). Pseudoknots are also considered to be a structural motif and are formed later in the folding process. *Extended pseudoknots* (Fig. 5(b)) and *kissing hairpin* (Fig. 6) are considered to be a common folding motif belonging to the class of pseudoknots.

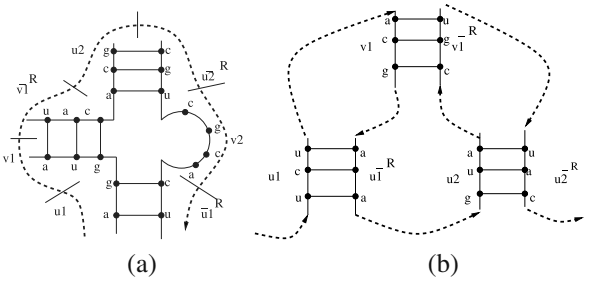

Fig. 5. RNA secondary structures: multi branch loop (a), extended pseudoknot (b).

To model and predict such structures, many attempts have been made by defining new grammar formalisms like *stochastic context-free grammars* (Sakakibara *et al.*, 1996), *pair hidden Markov models* (Sakakibara, 2003) and *stochastic multiple context-free grammars* (Yuki and Kasami, 2006). In particular, more research work is carried out on RNA pseudoknotted secondary structure prediction. In the work of Theis *et al.* (2010), prediction of RNA secondary structure is carried out including kissing hairpins. Cai *et al.* (2003) propose a grammatical approach for stochastic modelling of RNA pesudoknotted structures. In the work of Brown and Wilson (1995), RNA pseudoknot interactions are modelled using the intersection of stochastic context-free grammars. For more details on RNA secondary structures, we refer to the works of Lyngso *et al.* (1999), Lyngso and Pedersen (2000) or Rivas and Eddy (2000). Figure 7 shows

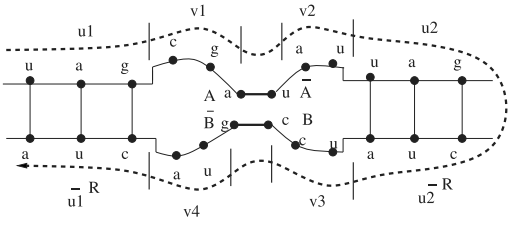

Fig. 6. RNA secondary structure: kissing hairpin.

the simple H-type and recursive pseudoknot structure. Figure 8 presents the three-knot structure. Elements $u_1, u_2, u_3, u_4, u_5, \bar{u_1}^R, \bar{u_2}^R, \bar{u_3}^R, \bar{u_4}^R, \bar{u_5}^R, v, v_1, v_2, v_3, v_4, A\bar{A}, B\bar{B}$ used in Figs. 4(a), 4(b), 5(a), 5(b), 6, 7(a), 7(b) and 8 are explained in Section 5. Table 1 shows various bio-molecular structures that are commonly noticed in DNA, protein, RNA secondary structures and their corresponding formal grammars which generate the structure.

However, there is no unique grammar model that encapsulates all the above-discussed bio-molecular structures. For example, the double copy language cannot be modelled by a simple linear tree adjoining

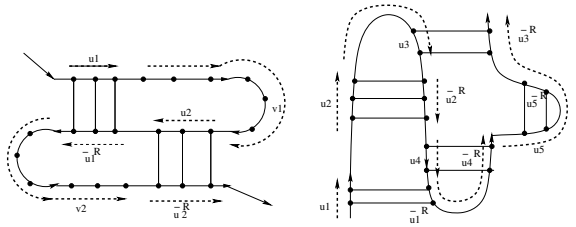

Fig. 7. RNA secondary structures: simple H-type (a), recursive pseudoknot (b).

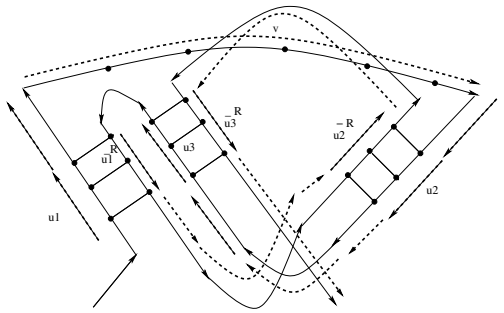

Fig. 8. RNA secondary structure: three-knot structure.

Table 1. Bio-molecular structure and the corresponding formal grammar.

Bio-molecular structure Figure number(s)	Formal grammar
Hairpin Fig. 1(b)	Context free grammar
Stem and loop Fig. 1(a)	Context free grammar
Attenuator Fig. 2(b)	Tree adjoining grammar
Pseudoknot Fig. 2(a)	Tree adjoining grammar
Cloverleaf Fig. 9	Context free grammar
Nick Fig. 3(b)	Cut grammar
Double strand Fig. 3(a)	Ligation grammar
Holliday Fig. 10	Ligation grammar
RNA structures Figs. 4(a), 4(b), 5(a), 5(b) 6, 7(a), 7(b), 8	Stochastic CFG Multiple stochastic CFG Pair HMM Stochastic multiple CFG

grammar (Uemura *et al.*, 1999). To overcome this failure, we introduced (Kuppusamy *et al.*, 2011a) a simple and powerful grammar model called *matrix insertion–deletion systems* that captures all the popular and important bio-molecular structures noticed often in bio-molecules. We also modelled the various bio-molecular structures that occur at the intramolecular level such as *pseudo knot*, *hairpin*, *stem and loop*, *attenuator* (Kuppusamy *et al.*, 2011a). We have modelled the various bio-molecular structures that occur at intermolecular level such as *double strand language*, *nick language*, *holliday structure*, *replication fork* (Kuppusamy *et al.*, 2011b).

In this paper, we substantially extend our work by introducing many RNA bio-molecular structures (see Section 5) and give a formal language representation for each such structure. Further, we model such structures using the matrix insertion–deletion system. Thus, this paper is an extended journal version of past conference papers (Lakshmanan *et al.*, 2011a; 2011b). Incidentally, in the work of Petre and Verlan (2012), the same matrix insertion–deletion system was discussed and analyzed the computational completeness result for the system. However, the motivation was not from a biological inspiration; it was rather an extension of matrix grammars. In the work of Petre and Verlan (2012), matrix insertion–deletion systems were introduced with the following measures: (i) the maximum number of rules in a matrix is denoted by k, (ii) the maximal length of the left and right context used in the insertion rules is denoted with m and m', respectively, (iii) the maximal length of the left and right context used in deletion rules is denoted with q and q', respectively, (iv) the maximal length of the inserted string is denoted with n, (v) the maximal length of the deleted string is denoted with p.

Based on these measures, the family of languages generated by matrix insertion–deletion systems is denoted by $Mat_k INS_n^{m,m'} DEL_p^{q,q'}$. With these measures, in the work of Petre and Verlan (2012), the computational completeness result (i.e., showing equivalence to recursively enumerable languages) for the matrix insertion–deletion system was proved for the combinations $Mat_3 INS_1^{1,0} DEL_2^{0,0}$, $Mat_3 INS_1^{1,0} DEL_1^{1,0}$, $Mat_3 INS_1^{1,0} DEL_1^{0,1}$ and $Mat_3 INS_2^{0,0} DEL_1^{1,0}$. In the same paper, with binary matrices (matrices having two rules) the computational completeness result was proved for the combinations $Mat_2 INS_2^{0,0} DEL_1^{1,0}$ and $Mat_2 INS_1^{1,0} DEL_2^{0,0}$. Note that, in these results when no context is considered in deletion rules, the maximal length of the deleted string is 2. Also, insertion and deletion rules are together used in a matrix. In this paper, we have modelled the bio-molecular structures using the matrix insertion–deletion systems where the length of the deletion string is 1 only. Also, insertion and deletion rules are not used together in a matrix.

This paper is organized as follows. In Section 2,

we deal with the preliminaries. In Section 3, we briefly introduce matrix insertion–deletion systems. In Section 4, we show that these systems can encompass several essential bio-molecular structures that occur at intramolecular and intermolecular levels in DNA and RNA. In Section 5, we give the language representation for RNA secondary structures and we model them using our grammar model. In Section 6, we conclude the paper with further research direction.

2. Preliminaries

We start with recalling some basic notation used in formal language theory. A finite non-empty set V or Σ is called an alphabet. Σ_{DNA} is a finite non-empty set over the symbols $\{a, t, g, c\}$. Σ_{RNA} is a finite non-empty set over the symbols $\{a, u, g, c\}$. We denote by V^* or Σ^* the free monoid generated by V or Σ, by λ its identity or the empty string, and by V^+ or Σ^+ the set $V^* - \{\lambda\}$ or $\Sigma^* - \{\lambda\}$. The elements of V^* or Σ^* are called *words* or *strings*. A language L is defined as $L \subseteq \Sigma^*$. Let w be a string, and $|w|_a$ denote the number of a in w. For more details on formal language theory, we refer to Rozenberg and Salomaa (1997).

Next, we recall the basic definition of insertion–deletion systems. Given an insertion–deletion system $\gamma = (V, T, A, R)$, where V is an alphabet (set of non-terminal and terminal symbols), $T \subseteq V$ (set of terminal symbols), A is a finite language over V, R is a set of finite triples of the form $(u, \alpha/\beta, v)$, where $(u, v) \in V^* \times V^*$, $(\alpha, \beta) \in (V^+ \times \{\lambda\}) \cup (\{\lambda\} \times V^+)$. The pair (u, v) is called contexts, which will be used in deletion/insertion rules. The insertion rule is of the form $(u, \lambda/\beta, v)$, which means that β is inserted between u and v. The deletion rule is of the form $(u, \alpha/\lambda, v)$, which means that α is deleted between u and v. In other words, $(u, \lambda/\beta, v)$ corresponds to the rewriting rule $uv \to u\beta v$, and $(u, \alpha/\lambda, v)$ corresponds to the rewriting rule $u\alpha v \to uv$.

Consequently, for $x, y \in V^*$ we can write $x \Longrightarrow^* y$, if y can be obtained from x by using either an insertion rule or a deletion rule which is given as follows (the down arrow $\downarrow$ indicates the position where the string is inserted, the down arrow $\Downarrow$ indicates the position where the string is deleted and the underlined string indicates the string inserted):

(i) $x = x_1 u^{\downarrow} v x_2,\ y = x_1 u \underline{\beta} v x_2$, for some $x_1, x_2 \in V^*$ and $(u, \lambda/\beta, v) \in R$.

(ii) $x = x_1 u \alpha v x_2,\ y = x_1 u^{\Downarrow} v x_2$, for some $x_1, x_2 \in V^*$ and $(u, \alpha/\lambda, v) \in R$.

The language generated by γ is defined by

$$L(\gamma) = \{w \in T^* \mid x \Longrightarrow^* w, \text{ for some } x \in A\},$$

where $\Longrightarrow^*$ is the reflexive and transitive closure of the relation $\Longrightarrow$.

Next, we discuss *matrix grammar*. A matrix grammar is an ordered quadruple $G = (N, T, S, M)$ where N is a set of non-terminals, T is a set of terminals, $S \in N$ is the start symbol and M is a finite set of nonempty sequences whose elements are ordered pairs (P, Q). The pairs are referred to as productions and written in the form $P \to Q$. The sequences are referred to as matrices and written as $m = [P_1 \to Q_1, \ldots, P_r \to Q_r], r \geq 1$. For a matrix grammar G, the relation $\Longrightarrow$ on the set V^* is defined as follows. For any $P, Q \in V^*$, $P \Longrightarrow Q$ holds if and only if there exist an integer $r \geq 1$ and words $\beta_1, \ldots, \beta_{r+1}$, $P_1, \ldots, P_r$, $Q_1, \ldots, Q_r$, $R_1, \ldots, R_r$, $R^1, \ldots, R^r$ over V such that (i) $\alpha_i = P$ and $\alpha_{r+1} = Q$, (ii) m is one of the matrices of G, (iii) $\alpha_i = R_i P_i R^i$ and $\alpha_{i+1} = R_i Q_i R^i$. If the above conditions are satisfied, it is also said that $P \Longrightarrow Q$ holds with specifications (m, R_1). The reflexive and transitive closure of $\Longrightarrow$ is denoted by $\Longrightarrow^*$. The above matrix grammar is without *appearance checking*. The language generated by the matrix grammar is defined by $L(G) = \{w \in T^* \mid S \Longrightarrow^* w\}$. A matrix grammar with *appearance checking* is defined as $G = (N, T, S, M, F)$, where F is a set of occurrences of rules in the matrices of M. While deriving, a rule may be exempted to apply if the rule is in F. The language generated by the matrix grammar with appearance checking is defined as $L_{ac}(G, F) = \{w \in T^* \mid S \Longrightarrow^* w\}$. For more details on matrix grammars, we refer to the work of Rozenberg and Salomaa (1997).

Next, we discuss *cut grammars* (Searls, 1995) designed specifically for modelling intermolecular structures. A *cut grammar* $G = (N, T, S, P)$ where N is a finite set of non-terminals, T is a finite set of terminals, S is a start symbol and P is a finite set of productions in $(N \cup T)^* N (N \cup T)^* \times (N \cup T \cup \{\bullet\})^*$ where $\bullet$ is a new symbol called *cut symbol* not in N or T. The language generated by the cut grammar is defined as $L(G) = \{w \in (T \cup \bullet)^* \mid S \Longrightarrow^* w\}$.

Given any string $w = w_1 \bullet w_2 \bullet \ldots \bullet w_n$ where $w_i \in T^*$ for $1 \leq i \leq n$, the *cut function* is given as $\widehat{w} = \{w_1, w_2, \ldots, w_n\}$ and the *uncut function* is given as $\widetilde{w} = w_1 w_2 \ldots w_n$. For a given cut grammar G and start symbol S, the *cut language* is defined as $\widehat{L}(G) = \{\widehat{w} \in 2^{T^*} \mid S \Longrightarrow^* w\}$ and the *uncut language* is defined as $\widetilde{L}(G) = \{\widetilde{w} \in T^* \mid S \Longrightarrow^* w\}$. With cut grammars, the structures *double strand language*, *nick language*, *holliday structure* are represented. For more details on cut grammars, we refer to the work of Searls (1995).

3. Matrix insertion–deletion systems

In this section, we explain the grammar model *matrix insertion–deletion systems*. A matrix insertion–deletion

system is a construct $\Upsilon = (V, T, A, R)$, where V is an alphabet, $T \subseteq V$, A is a finite language over V, R is a set of finite triples in the matrix format $[(u_1, \alpha_1/\beta_1, v_1), \ldots, (u_n, \alpha_n/\beta_n, v_n)]$, where $(u_k, v_k) \in V^* \times V^*$, and $(\alpha_k, \beta_k) \in (V^+ \times \{\lambda\}) \cup (\{\lambda\} \times V^+)$, with $(u_k, \alpha_k/\beta_k, v_k) \in R_{I_i} \cup R_{D_j} \cup R_{I_i/D_j}$, for $1 \leq i \leq m$, $1 \leq j \leq m$, $1 \leq k \leq n$, where m is the number of rules in the matrix format in R. Here R_{I_i} denotes the matrix which consists of only insertion rules, R_{D_j} denotes the matrix which consists of only deletion rules and R_{I_i/D_j} denotes the matrix which consists of both insertion and deletion rules.

Consequently, for $x, y \in V^*$ we can write $x \Longrightarrow x' \Longrightarrow x'' \Longrightarrow \ldots \Longrightarrow y$, if y can be obtained from x by using a matrix consisting of insertion rules (R_{I_i}), or deletion rules (R_{D_j}) or insertion and deletion rules (R_{I_i/D_j}). In a derivation step the rules in a matrix are applied sequentially one after the other in order, and no rule is in appearance checking (note that the rules in a matrix are not applied in parallel). The language generated by Υ is defined by

$$L(\Upsilon) = \{w \in T^* \mid x \Longrightarrow^*_{R_\chi} w, \text{ for some } x \in A,\ \chi \in \{I_i, D_j, I_i/D_j\}\},$$

where R_χ denotes the matrix rules from an insertion matrix or a deletion matrix or a combination of both the rules. $\Longrightarrow^*$ is the reflexive and transitive closure of the relation $\Longrightarrow$. Note that the string w is collected after applying all the rules in a matrix and also $w \in T^*$.

4. Modelling bio-molecular structures

In this section, we show that matrix insertion–deletion systems can capture the commonly noticed biological structures that are discussed earlier in the paper. In most of the following derivations, in each derivation step, we directly write the resultant string obtained by applying all the rules in a matrix. In all the lemmas, we adopt the method of proof by construction in modelling the bio-molecular structures using matrix insertion–deletion systems. In the derivation step, the rule at the suffix of $\Longrightarrow$ denotes the corresponding matrix rule applied. From the formal language theory perspective, since structures can be viewed as languages, in many places we refer to be structure as language.

4.1. Representation of intramolecular structures.

In this section, we model some of the bio-molecular structures that occur at the intramolecular level.

Lemma 1. *The pseudoknot structure language (see Fig. 2(a)) $L_{ps} = \{uv\bar{u}^R\bar{v}^R \mid u, v \in \Sigma^*_{DNA}\}$ can be generated by a matrix insertion–deletion system.*

Proof. The language L_{ps} can be generated by the matrix insertion–deletion system $\Upsilon_{ps} = (\{b, \bar{b}, \dagger_1, \dagger_2, \dagger_3, \dagger_4\}, \{b, \bar{b}\}, \{\lambda, \dagger_1\ \dagger_2\ \dagger_3\dagger_4\}, R)$, where $b \in \{a, t, g, c\}$, $\bar{b}$ is complement of b and R is given as follows:

$$\begin{aligned} R_{I_1} &= [(\lambda, \lambda/b, \dagger_1), (\lambda, \lambda/\bar{b}, \dagger_3)], \\ R_{I_2} &= [(\lambda, \lambda/b, \dagger_2), (\lambda, \lambda/\bar{b}, \dagger_4)], \\ R_{D_1} &= [(\lambda, \dagger_1/\lambda, \lambda), (\lambda, \dagger_3/\lambda, \lambda)], \\ R_{D_2} &= [(\lambda, \dagger_2/\lambda, \lambda), (\lambda, \dagger_4/\lambda, \lambda)]. \end{aligned}$$

The Υ_{ps} generates only the language L_{ps}. The idea behind the construction of the system is given as follows. $\dagger_1$, $\dagger_2$, $\dagger_3$, $\dagger_4$ are used as markers. Whenever a b is adjoined to the left of $\dagger_1$, its corresponding complementary $\bar{b}$ should be adjoined to the left of $\dagger_3$ using the rule R_{I_1}. So, $\dagger_1$ and $\dagger_3$ are used to control the $u\bar{u}^R$ part of the language. Similarly, whenever a b is adjoined to the left of $\dagger_2$, its corresponding complementary $\bar{b}$ should be adjoined to the left of $\dagger_4$ using the rule R_{I_2}. So, $\dagger_2$ and $\dagger_4$ are used to control the $v\bar{v}^R$ part of the language. When the rule R_{D_1} is used first, then system Υ_{ps} generates only $v\bar{v}^R$ part of the language. When the rule R_{D_2} is used first, then the system Υ_{ps} generates only the $u\bar{u}^R$ part of the language. We present a sample derivation for a better understanding (the rule at the suffix of the derivation symbol $\Longrightarrow$ denotes whether an insertion rule or a deletion rule is applied),

$$\begin{aligned} &{}^{\downarrow}\dagger_1\, \dagger_2^{\downarrow}\, \dagger_3\, \dagger_4 \Longrightarrow_{R_{I_1}} \underline{a}\, \dagger_1^{\downarrow}\, \dagger_2 \underline{t}\, \dagger_3^{\downarrow}\, \dagger_4 \Longrightarrow_{R_{I_2}} \\ &a\, \dagger_1\, \underline{g}^{\downarrow}\, \dagger_2\, t\, \dagger_3\, \underline{c}^{\downarrow} \dagger_4 \Longrightarrow_{R_{I_2}} a\, \dagger_1\, g\underline{a}\, \dagger_2\, t\, \dagger_3\, c\underline{t}\, \dagger_4 \\ &\Longrightarrow_{R_{D_1}} a^{\Downarrow} ga\, \dagger_2\, t^{\Downarrow} ct \dagger_4 \Longrightarrow_{R_{D_2}} aga^{\Downarrow} tct^{\Downarrow}. \end{aligned}$$

■

From Fig. 2(b), the attenuator language can be given as $L_{an} = \{u\bar{u}^R u\bar{u}^R \mid u \in \Sigma^*_{DNA}\}$.

Lemma 2. *The attenuator language L_{an} (see Fig. 2(b)) can be generated by a matrix insertion–deletion system.*

Proof. The language L_{an} can be generated by the matrix insertion–deletion system

$$\Upsilon_{an} = (\{a, t, g, c, \dagger_1, \dagger_2\}, \{a, t, g, c\}, \{\lambda, \dagger_1\dagger_2\}, R),$$

where R is given as follows:

$$\begin{aligned} R_{I_1} &= [(\lambda, \lambda/a, \dagger_1), (\dagger_1, \lambda/t, \lambda), (\lambda, \lambda/a, \dagger_2), \\ &\quad (\dagger_2, \lambda/t, \lambda)], \\ R_{I_2} &= [(\lambda, \lambda/t, \dagger_1), (\dagger_1, \lambda/a, \lambda), (\lambda, \lambda/t, \dagger_2), \\ &\quad (\dagger_2, \lambda/a, \lambda)], \\ R_{I_3} &= [(\lambda, \lambda/c, \dagger_1), (\dagger_1, \lambda/g, \lambda), (\lambda, \lambda/c, \dagger_2), \\ &\quad (\dagger_2, \lambda/g, \lambda)], \\ R_{I_4} &= [(\lambda, \lambda/g, \dagger_1), (\dagger_1, \lambda/c, \lambda), (\lambda, \lambda/g, \dagger_2), \\ &\quad (\dagger_2, \lambda/c, \lambda)], \\ R_{D_1} &= [(\lambda, \dagger_1/\lambda, \lambda), (\lambda, \dagger_2/\lambda, \lambda)]. \end{aligned}$$

Here Υ_{an} generates only the language L_{an}. Marker $\dagger_1$ is used to control the first part of the language ($u\bar{u}^R$) and marker $\dagger_2$ is used to control the second part of the language ($u\bar{u}^R$). Whenever a b and its corresponding complementary $\bar{b}$ are adjoined by using the $\dagger_1$, simultaneously by using the $\dagger_2$, the same b and its complementary $\bar{b}$ are adjoined. As the rule R_{I_1} uses both the markers $\dagger_1$ and $\dagger_2$, synchronization is easily maintained. A similar procedure holds for the remaining rules R_{I_2}, R_{I_3} and R_{I_4}, e.g.,

$$
\begin{aligned}
&{}^{\downarrow}\dagger_1^{\downarrow}\,{}^{\downarrow}\dagger_2^{\downarrow} \Longrightarrow_{R_{I_1}} \underline{a}^{\downarrow}\dagger_1^{\downarrow}\underline{t}\,\underline{a}^{\downarrow}\dagger_2^{\downarrow}\underline{t} \Longrightarrow_{R_{I_2}}\\
&at\underline{t}^{\downarrow}\dagger_1^{\downarrow}\underline{a}ta\underline{t}^{\downarrow}\dagger_2^{\downarrow}\underline{a}t \Longrightarrow_{R_{I_3}} atc\underline{c}^{\downarrow}\dagger_1^{\downarrow}\underline{g}atatc\underline{c}^{\downarrow}\dagger_2^{\downarrow}\underline{g}at\\
&\Longrightarrow_{R_{I_4}} atc\underline{g}\,\dagger_1\,\underline{c}gatatc\underline{g}\,\dagger_2\,\underline{c}gat \Longrightarrow_{R_{D_1}}\\
&atcg^{\Downarrow}cgatatcg^{\Downarrow}cgat.
\end{aligned}
$$

■

Lemma 3. *The hairpin language (see Fig. 1(b)) $L_{hp} = \{w\bar{w}^R \mid w \in \Sigma^*_{DNA}\}$ can be generated by a matrix insertion–deletion system.*

Proof. The hairpin language L_{hp} can be generated by the matrix insertion–deletion system

$$\Upsilon_{hp} = (\{b, \bar{b}, \dagger\}, \{b\bar{b}\}, \{\lambda, \dagger\}, R),$$

where $b \in \{a, t, g, c\}$, $\bar{b}$ is complement of b and R is given as follows:

$$
\begin{aligned}
R_{I_1} &= [(\lambda, \lambda/b, \dagger), (\dagger, \lambda/\bar{b}, \lambda),\\
R_{D_1} &= [(\lambda, \dagger/\lambda, \lambda)].
\end{aligned}
$$

We present a sample derivation which itself suffices to see that $L(\Upsilon_{hp}) = L_{hp}$,

$$
\begin{aligned}
&{}^{\downarrow}\dagger^{\downarrow} \Longrightarrow_{R_{I_1}} \underline{t}^{\downarrow}\dagger^{\downarrow}\underline{a} \Longrightarrow_{R_{I_1}} t\underline{g}^{\downarrow}\dagger^{\downarrow}\underline{c}a \Longrightarrow_{R_{I_1}}\\
&tg\underline{c}^{\downarrow}\dagger^{\downarrow}\underline{g}ca \Longrightarrow_{R_{I_1}} tgc\underline{g}^{\downarrow}\dagger^{\downarrow}\underline{cg}ca \Longrightarrow_{R_{D_1}}\\
&tgc\underline{g}^{\Downarrow}\underline{c}gca.
\end{aligned}
$$

■

Lemma 4. *The stem and loop language (see Fig. 1(a)) $L_{sl} = \{uv\bar{u}^R \mid u, v \in \Sigma^*_{DNA}\}$ can be generated by a matrix insertion–deletion system.*

Proof. The stem and loop language L_{sl} can be generated by the matrix insertion–deletion system $\Upsilon_{sl} = (\{b, \bar{b}, \dagger_1, \dagger_2, \dagger_3\}, \{b, \bar{b}\}, \{\lambda, \dagger_1\ \dagger_3\ \dagger_2\}, R)$, where $b \in \{a, t, g, c\}$, $\bar{b}$ is the complement of b and R is given as follows:

$$
\begin{aligned}
R_{I_1} &= [(\lambda, \lambda/b, \dagger_1), (\dagger_2, \lambda/\bar{b}, \lambda)],\\
R_{I_2} &= [(\lambda, \lambda/b, \dagger_3)],\\
R_{D_1} &= [(\lambda, \dagger_1/\lambda, \lambda), (\lambda, \dagger_2/\lambda, \lambda)],\\
R_{D_2} &= [(\lambda, \dagger_3/\lambda, \lambda)].
\end{aligned}
$$

A sample derivation is given follows:

$$
\begin{aligned}
&{}^{\downarrow}\dagger_1\,\dagger_3\,\dagger_2^{\downarrow}\\
&\Longrightarrow_{R_{I_1}} \underline{c}\,\dagger_1^{\downarrow}\dagger_3\,\dagger_2\,\underline{g} \Longrightarrow_{R_{I_2}}\\
&c\,\dagger_1\,\underline{t}^{\downarrow}\,\dagger_3\,\dagger_2 g \Longrightarrow_{R_{I_2}} c\,\dagger_1\,t\underline{c}\,\dagger_3\,\dagger_2 g\\
&\Longrightarrow_{R_{D_1}} c^{\Downarrow}tc\,\dagger_3^{\Downarrow}\,g \Longrightarrow_{R_{D_2}} ctc^{\Downarrow}g.
\end{aligned}
$$

■

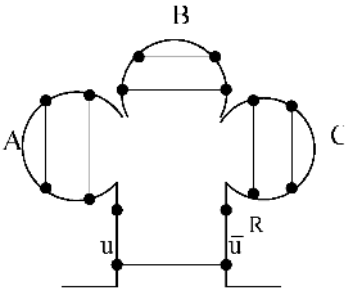

Fig. 9. Cloverleaf representation (where $A = v_1\bar{v}_1^R$, $B = v_2\bar{v}_2^R$, $C = v_3\bar{v}_3^R$).

Lemma 5. *The cloverleaf language (see Fig. 9, for $n = 3$) (Searls, 1988; 1992)*

$$
\begin{aligned}
L_{cl} = \{uv_1\bar{v}_1^R v_2\bar{v}_2^R \ldots v_n\bar{v}_n^R\bar{u}^R \mid u, v_1, v_2, \ldots,\\
v_n \in \Sigma^*_{DNA}, n \geq 0\}
\end{aligned}
$$

can be generated by a matrix insertion–deletion system.

Proof. The cloverleaf language L_{cl} (for $n = 3$) can be generated by the matrix insertion–deletion system $\Upsilon_{cl} = (\{b, \bar{b}, \dagger_1, \dagger_2, \dagger_3, \dagger_4, \dagger_5\}, \{b, \bar{b}\}, \{\lambda, \dagger_1\dagger_2, \dagger_3\dagger_4\dagger_5, \dagger_1\dagger_3\dagger_4\dagger_5\ \dagger_2\}, R)$, where $b \in \{a, t, g, c\}$, $\bar{b}$ is a complement of b and R is given as follows:

$$
\begin{aligned}
R_{I_1} &= [(\lambda, \lambda/b, \dagger_1), (\dagger_2, \lambda/\bar{b}, \lambda)],\\
R_{I_2} &= [(\lambda, \lambda/b, \dagger_3), (\dagger_3, \lambda/\bar{b}, \lambda)],\\
R_{I_3} &= [(\lambda, \lambda/b, \dagger_4), (\dagger_4, \lambda/\bar{b}, \lambda)],\\
R_{I_4} &= [(\lambda, \lambda/b, \dagger_5), (\dagger_5, \lambda/\bar{b}, \lambda)],\\
R_{D_1} &= [(\lambda, \dagger_1/\lambda, \lambda), (\lambda, \dagger_2/\lambda, \lambda)],\\
R_{D_2} &= [(\lambda, \dagger_3/\lambda, \lambda)],\\
R_{D_3} &= [(\lambda, \dagger_4/\lambda, \lambda)],\\
R_{D_4} &= [(\lambda, \dagger_5/\lambda, \lambda)].
\end{aligned}
$$

A sample derivation is given as follows:

$$
\begin{aligned}
&{}^{\downarrow}\dagger_1\,\dagger_3\,\dagger_4\,\dagger_5\dagger_2^{\downarrow} \Longrightarrow_{R_{I_1}} \underline{c}^{\downarrow}\,\dagger_1\,\dagger_3\,\dagger_4\,\dagger_5\,\dagger_2^{\downarrow}\,\underline{g}\\
&\Longrightarrow_{R_{I_1}} cg\,\dagger_1^{\downarrow}\,\dagger_3^{\downarrow}\,\dagger_4\,\dagger_5\,\dagger_2\,\underline{c}gt\\
&\Longrightarrow_{R_{I_2}} acg\,\dagger_1\,\underline{t}\,\dagger_3\,\underline{a}^{\downarrow}\,\dagger_4^{\downarrow}\,\dagger_5\,\dagger_2\,cg\\
&\Longrightarrow_{R_{I_3}} cg\,\dagger_1\,t\,\dagger_3\,a\,\underline{c}\,\dagger_4\,\underline{g}^{\downarrow}\,\dagger_5^{\downarrow}\,\dagger_2 cg\\
&\Longrightarrow_{R_{I_4}} cg\,\dagger_1\,t\,\dagger_3\,ac\,\dagger_4\,g\underline{a}\,\dagger_5\,\underline{t}\,\dagger_2\,cg\\
&\Longrightarrow_{R_{D_1}} cg^{\Downarrow}t\,\dagger_3\,ac\,\dagger_4\,ga\,\dagger_5\,t^{\Downarrow}cg\\
&\Longrightarrow_{R_{D_2}} cgt^{\Downarrow}ac\,\dagger_4\,ga\,\dagger_5\,tcg\\
&\Longrightarrow_{R_{D_3}} cgtac^{\Downarrow}ga\,\dagger_5\,tcg\\
&\Longrightarrow_{R_{D_4}} cgtacga^{\Downarrow}tcg.
\end{aligned}
$$

■

Using five markers, $\dagger_1, \dagger_2, \dagger_3, \dagger_4, \dagger_5$ the system Υ_{cl} generates the cloverleaf language L_{cl} for $n = 3$. By introducing more markers, the system Υ_{cl} can generate a cloverleaf language for an arbitrary value of n.

4.2. Representation of intermolecular structures. In this section, we model some of the bio-molecular structures that occur at intermolecular level.

Lemma 6. *The double strand language*

$$L_{ds} = \{u \bullet \bar{u}^R \mid u \in \Sigma^*_{DNA}\}$$

can be modelled by a matrix insertion–deletion system.

Proof. The double strand language (see Fig. 3(a)) L_{ds} can be be modelled by a matrix insertion–deletion system $\Upsilon_{ds} = (\{b, \bar{b}, \bullet\}, \{b, \bar{b}, \bullet\}, \{\bullet\}, R)$ where $b \in \{a, t, g, c\}$, $\bar{b}$ is complement of b and R is given as $R_{I_1} = [(\lambda, \lambda/b, \bullet), (\bullet, \lambda/\bar{b}, \lambda)]$. We present a sample derivation which itself is sufficient to see that $L(\Upsilon_{ds}) = L_{ds}$,

$$
\begin{aligned}
&{}^{\downarrow}\bullet^{\downarrow} \Longrightarrow_{R_{I_1}} \underline{a}^{\downarrow}\bullet^{\downarrow}\underline{t} \Longrightarrow_{R_{I_1}} a\underline{g}^{\downarrow}\bullet^{\downarrow}\underline{c}t \Longrightarrow_{R_{I_1}}\\
&ag\underline{a}^{\downarrow}\bullet^{\downarrow}\underline{t}ct \Longrightarrow_{R_{I_1}} agac\underline{c}^{\downarrow}\bullet^{\downarrow}\underline{g}tct.
\end{aligned}
$$

■

From Fig. 3(b) the nick language can be informally described as $L_{nl} = \{w_1 \bullet w_2 \mid \widetilde{w}_2 = \bar{w_1}^R\}$, where $w_1 \in \Sigma^*$ and $w_2 \in (\Sigma \cup \{\bullet\})^*$ (i.e., w_2 is a string which contains a number of $\bullet$).

Lemma 7. *The nick language L_{nl} can be generated by matrix insertion–deletion system.*

Proof. The nick language (see Fig. 3(b)) L_{nl} can be generated by the cut grammar $G_{nl} = S \rightarrow bS\bar{b} \mid S\bullet \mid \bullet$ for each $b \in \Sigma_{DNA}$. The grammar G_{nl} can be modelled by the matrix insertion–deletion system $\Upsilon_{nl} = (\{b, \bar{b}, \dagger, \bullet\}, \{b, \bar{b}, \bullet\}, \{b\,\dagger\,\bar{b}, \dagger\bullet, \bullet\}, R)$ where $b \in \{a, t, g, c\}$, $\bar{b}$ is a complement of b and R is given as

$$
\begin{aligned}
R_{I_1} &= [(\lambda, \lambda/b, \dagger), (\dagger, \lambda/\bar{b}, \lambda)],\\
R_{I_2} &= [(\dagger, \lambda/\bullet, \lambda)],\\
R_{D_1} &= [(\lambda, \dagger/\lambda, \lambda)].
\end{aligned}
$$

A sample derivation is given as follows:

$$
\begin{aligned}
&a^{\downarrow}\dagger^{\downarrow}\,t \Longrightarrow_{R_{I_1}} a\underline{t}^{\downarrow}\dagger^{\downarrow}\,\underline{a}t \Longrightarrow_{R_{I_1}} atg\,\dagger^{\downarrow}\,\underline{c}at \Longrightarrow_{R_{I_2}}\\
&atg^{\downarrow}\dagger^{\downarrow}\,\bullet cat \Longrightarrow_{R_{I_1}} atg\underline{a}\,\dagger^{\downarrow}\,\underline{t}\bullet cat \Longrightarrow_{R_{I_2}}\\
&atga\,\dagger\,\bullet\underline{t}\bullet cat \Longrightarrow_{R_{D_1}} atga^{\Downarrow}\bullet t\bullet cat.
\end{aligned}
$$

■

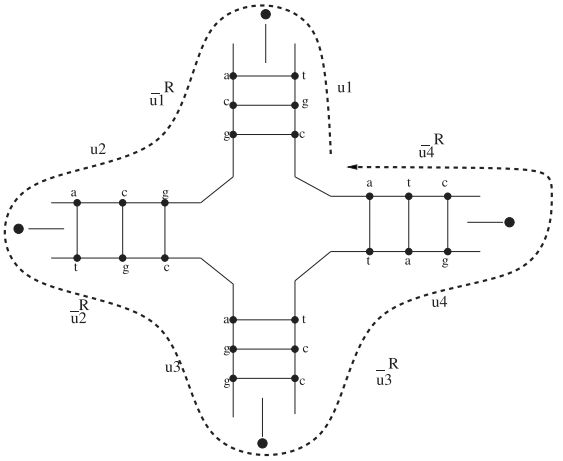

Fig. 10. Holliday structure.

Lemma 8. *The holliday structure (see Fig. 10)*

$$L_{hs} = \{u_1 \bullet \bar{u_1}^R u_2 \bullet \bar{u_2}^R u_3 \bullet \bar{u_3}^R u_4 \bullet \bar{u_4}^R \mid u_1, u_2, u_3, u_4 \in \Sigma^*_{DNA}\}$$

can be generated by a matrix insertion–deletion system.

Proof. The language L_{hs} can be generated by matrix insertion–deletion system

$$\Upsilon_{hs} = (\{b, \bar{b}, \dagger_1, \dagger_2, \dagger_3, \dagger_4, \dagger_5, \bullet\}, \{b, \bar{b}, \bullet\}, \{\dagger_1 \bullet \dagger_2 \bullet \dagger_3 \bullet \dagger_4 \bullet \dagger_5, \bullet\bullet\bullet\bullet\}, R)$$

where $b \in \{a, t, g, c\}$, $\bar{b}$ is a complement of b and R is

given as

$$\begin{aligned}
R_{I_1} &= [(\dagger_1, \lambda/b, \lambda), (\lambda, \lambda/\bar{b}, \dagger_2)],\\
R_{I_2} &= [(\dagger_2, \lambda/b, \lambda), (\lambda, \lambda/\bar{b}, \dagger_3)],\\
R_{I_3} &= [(\dagger_3, \lambda/b, \lambda), (\lambda, \lambda/\bar{b}, \dagger_4)],\\
R_{I_4} &= [(\dagger_4, \lambda/b, \lambda), (\lambda, \lambda/\bar{b}, \dagger_5)],\\
R_{D_1} &= [(\lambda, \dagger_1/\lambda, \lambda)],\\
R_{D_2} &= [(\lambda, \dagger_2/\lambda, \lambda)],\\
R_{D_3} &= [(\lambda, \dagger_3/\lambda, \lambda)],\\
R_{D_4} &= [(\lambda, \dagger_4/\lambda, \lambda)],\\
R_{D_5} &= [(\lambda, \dagger_5/\lambda, \lambda)].
\end{aligned}$$

A sample derivation is given as follows:

$$\begin{aligned}
&\dagger_1^{\downarrow} \bullet^{\downarrow} \dagger_2 \bullet \dagger_3 \bullet \dagger_4 \bullet \dagger_5\\
&\Longrightarrow_{R_{I_1}} \dagger_1^{\downarrow} \underline{a} \bullet^{\downarrow} \underline{t}\, \dagger_2 \bullet \dagger_3 \bullet \dagger_4 \bullet \dagger_5\\
&\Longrightarrow_{R_{I_1}} \dagger_1 \underline{c}a \bullet t\underline{g}\, \dagger_2^{\downarrow} \bullet^{\downarrow} \dagger_3 \bullet \dagger_4 \bullet \dagger_5\\
&\Longrightarrow_{R_{I_2}} \dagger_1 ca \bullet tg\, \dagger_2^{\downarrow} \underline{a} \bullet^{\downarrow} \underline{t}\, \dagger_3 \bullet \dagger_4 \bullet \dagger_5\\
&\Longrightarrow_{R_{I_2}} \dagger_1 ca \bullet tg\, \dagger_2 \underline{c}a \bullet t\underline{g}\, \dagger_3^{\downarrow} \bullet^{\downarrow} \dagger_4 \bullet \dagger_5\\
&\Longrightarrow_{R_{I_3}} \dagger_1 ca \bullet tg\, \dagger_2 ca \bullet tg\, \dagger_3^{\downarrow} \underline{g} \bullet^{\downarrow} \underline{c}\, \dagger_4 \bullet \dagger_5\\
&\Longrightarrow_{R_{I_3}} \dagger_1 ca \bullet tg\, \dagger_2 ca \bullet tg\, \dagger_3 \underline{a}g \bullet c\underline{t}\, \dagger_4^{\downarrow} \bullet^{\downarrow} \dagger_5\\
&\Longrightarrow_{R_{I_4}} \dagger_1 ca \bullet tg\, \dagger_2 ca \bullet tg\, \dagger_3 ag \bullet ct\, \dagger_4^{\downarrow} \underline{a} \bullet \underline{t}^{\downarrow}\, \dagger_5\\
&\Longrightarrow_{R_{I_4}} \dagger_1 ca \bullet tg\, \dagger_2 ca \bullet tg\, \dagger_3 ag \bullet ct\, \dagger_4 \underline{c}a \bullet t\underline{g}\, \dagger_5\\
&\Longrightarrow_{R_{D_1}} {}^{\Downarrow}ca \bullet tg\, \dagger_2 ca \bullet tg\, \dagger_3 ag \bullet ct\, \dagger_4 ca \bullet tg\, \dagger_5\\
&\Longrightarrow_{R_{D_2}} ca \bullet tg^{\Downarrow}ca \bullet tg\, \dagger_3 ag \bullet ct\, \dagger_4 ca \bullet tg\, \dagger_5\\
&\Longrightarrow_{R_{D_3}} ca \bullet tgca \bullet tg^{\Downarrow}ag \bullet ct\, \dagger_4 ca \bullet tg\, \dagger_5\\
&\Longrightarrow_{R_{D_4}} ca \bullet tgca \bullet tgag \bullet ct^{\Downarrow}ca \bullet tg\, \dagger_5\\
&\Longrightarrow_{R_{D_5}} ca \bullet tgca \bullet tgag \bullet ctca \bullet tg^{\Downarrow}.
\end{aligned}$$

■

5. RNA secondary structures

In this section, first we show the interpretation of various RNA secondary structures in terms of formal language representations (as shown in Table 2), and we model such structures using matrix insertion–deletion systems. If the strings are collected as per the dotted directed lines, the RNA secondary structures represented in Figs. 4(a), 4(b), 5(a), 5(b), 6, 7(a), 7(b) and 8 can be given in terms of languages as shown in Table 2.

Thus, given a DNA/RNA sequence, we can (try to) first identify the corresponding the formal language and then one can think of what matrix insertion–deletion system can generate the language.

Lemma 9. *The internal loop structure (see Fig. 4(a))*

$$L_{il} = \{u_1v_1u_2v_3\bar{u_2}^{R}v_2\bar{u_1}^{R} \mid u_1, u_2, v_1, v_2, v_3 \in \Sigma^*_{RNA}\}$$

can be generated by a matrix insertion–deletion system.

Table 2. Formal language representation: $\#_1$ represents the $A\bar{A}$ and $\#_2$ represents the $B\bar{B}$ in Fig. 8.

Fig. no.	Bio-molecular structure Formal language representation
4(a)	Internal loop $L_{il} = \{u_1v_1u_2v_3\bar{u_2}^{R}v_2\bar{u_1}^{R}\}$
4(b)	Bulge loop $L_{bl} = \{u_1v_1u_2v_2\bar{u_2}^{R}\bar{u_1}^{R}\}$
5(a)	Multi branch loop $L_{mbl} = \{u_1v_1\bar{v_1}^{R}u_2\bar{u_2}^{R}v_2\bar{u_1}^{R}\}$
5(b)	Extended pseudoknot $L_{epk} = \{u_1v_1\bar{u_1}^{R}u_2\bar{v_1}^{R}\bar{u_2}^{R}\}$
6	Kissing hairpin $L_{khp} = \{u_1v_1\#_1v_2u_2\bar{u_2}^{R}v_3\#_2v_4\bar{u_1}^{R}\}$
7(a)	Simple H-type $L_{sht} = \{u_1v_1u_2\bar{u_1}^{R}v_2\bar{u_2}^{R}\}$
7(b)	Recursive pseudoknot $L_{rps} = \{u_1u_2u_3\bar{u_2}^{R}u_4\bar{u_1}^{R}\bar{u_4}^{R}u_5\bar{u_5}^{R}\bar{u_3}^{R}\}$
8	Three-knot structure $L_{tks} = \{u_1vu_2u_3\bar{u_1}^{R}\bar{u_2}^{R}\bar{u_3}^{R}\}$

Proof. The language L_{il} can be generated by the matrix insertion–deletion system

$$\Upsilon_{il} = (\{b, \bar{b}, \dagger_1, \dagger_2, \dagger_3, \dagger_4\}, \{b, \bar{b}\}, \{\dagger_1\, \dagger_3\, \dagger_4\dagger_2\}, R),$$

where $b \in \{a, u, g, c\}$, $\bar{b}$ is the complement of b and R is given as follows:

$$\begin{aligned}
R_{I_1} &= [(\lambda, \lambda/b, \dagger_1), (\dagger_2, \lambda/\bar{b}, \lambda)],\\
R_{I_2} &= [(\lambda, \lambda/b, \dagger_3), (\dagger_4, \lambda/\bar{b}, \lambda)],\\
R_{I_3} &= [(\dagger_1, \lambda/b, \lambda)],\\
R_{I_4} &= [(\dagger_3, \lambda/b, \lambda)],\\
R_{I_5} &= [(\lambda, \lambda/b, \dagger_2)],\\
R_{D_1} &= [(\lambda, \dagger_1/\lambda, \lambda)],\\
R_{D_2} &= [(\lambda, \dagger_2/\lambda, \lambda)],\\
R_{D_3} &= [(\lambda, \dagger_3/\lambda, \lambda)],\\
R_{D_4} &= [(\lambda, \dagger_4/\lambda, \lambda)].
\end{aligned}$$

A sample derivation is given as follows:

$$\begin{aligned}
&{}^{\downarrow}\dagger_1\, \dagger_3\, \dagger_4\, \dagger_2^{\downarrow} \Longrightarrow_{R_{I_1}} \underline{a}^{\downarrow}\, \dagger_1\, \dagger_3\, \dagger_4\, \dagger_2^{\downarrow}\underline{u}\\
&\Longrightarrow_{R_{I_1}} a\underline{u}\, \dagger_1^{\downarrow}\, \dagger_3\, \dagger_4^{\downarrow}\, \dagger_2\underline{a}u\\
&\Longrightarrow_{R_{I_2}} au\, \dagger_1\, \underline{u}^{\downarrow}\, \dagger_3\, \dagger_4^{\downarrow}\underline{a}\, \dagger_2\, au\\
&\Longrightarrow_{R_{I_2}} au\, \dagger_1^{\downarrow}\, u\underline{a}\, \dagger_3\, \dagger_4\underline{u}a\, \dagger_2\, au\\
&\Longrightarrow_{R_{I_3}} au\, \dagger_1^{\downarrow}\, \underline{g}ua\, \dagger_3\, \dagger_4 ua\, \dagger_2\, au\\
&\Longrightarrow_{R_{I_3}} au\, \dagger_1\, \underline{c}gua\, \dagger_3^{\downarrow}\, \dagger_4 ua\, \dagger_2\, au
\end{aligned}$$

$$\Longrightarrow_{R_{I_4}} au\dagger_1 cgua\dagger_3^{\downarrow}\underline{c}\dagger_4 ua\dagger_2 au$$
$$\Longrightarrow_{R_{I_4}} au\dagger_1 cgua\dagger_3 \underline{g}c\dagger_4 ua\dagger_2 au$$
$$\Longrightarrow_{R_{D_1}} au^{\Downarrow}cgua\dagger_3 gc\dagger_4 ua\dagger_2 au$$
$$\Longrightarrow_{R_{D_2}} aucgua\dagger_3 gc\dagger_4 ua^{\Downarrow}au$$
$$\Longrightarrow_{R_{D_3}} aucgua^{\Downarrow}gc\dagger_4 uaau$$
$$\Longrightarrow_{R_{D_4}} aucguagc^{\Downarrow}uaau.$$

The idea is that $\dagger_1$, $\dagger_2$, $\dagger_3$ and $\dagger_4$ are used as markers. $\dagger_1$ and $\dagger_2$ are used to control the $u_1\bar{u_1}^R$ part of the language. Whenever a b is adjoined to the left of $\dagger_1$, its corresponding complementary $\bar{b}$ is adjoined to the right of $\dagger_2$ and the synchronization is maintained. Similarly, $\dagger_3$ and $\dagger_4$ are used to control the $u_2\bar{u_2}^R$ part of the language, $\dagger_1$,$\dagger_2$ and $\dagger_3$ are used to control the v_1, v_2 and v_3 part of the language, respectively. ■

Lemma 10. *The bulge loop structure (see Fig. 4(b))*

$$L_{bl} = \{u_1v_1u_2v_2\bar{u_2}^R\bar{u_1}^R \mid u_1, u_2, v_1, v_2 \in \Sigma^*_{RNA}\}$$

can be generated by a matrix insertion–deletion system.

Proof. The language L_{bl} can be generated by the matrix insertion–deletion system

$$\Upsilon_{bl} = (\{b, \bar{b}, \dagger_1, \dagger_2, \dagger_3, \dagger_4\}, \{b, \bar{b}\}, \{\dagger_1\,\dagger_4\,\dagger_3\dagger_2\}, R),$$

where $b \in \{a, u, g, c\}$, $\bar{b}$ is the complement of b and R is given as follows:

$$\begin{aligned}
R_{I_1} &= [(\lambda, \lambda/b, \dagger_1), (\dagger_2, \lambda/\bar{b}, \lambda)],\\
R_{I_2} &= [(\lambda, \lambda/b, \dagger_4), (\dagger_3, \lambda/\bar{b}, \lambda)],\\
R_{I_3} &= [(\dagger_1, \lambda/b, \lambda)],\\
R_{I_4} &= [(\dagger_4, \lambda/b, \lambda)],\\
R_{D_1} &= [(\lambda, \dagger_1/\lambda, \lambda),\\
R_{D_2} &= [(\lambda, \dagger_2/\lambda, \lambda)],\\
R_{D_3} &= [(\lambda, \dagger_3/\lambda, \lambda)],\\
R_{D_4} &= [(\lambda, \dagger_4/\lambda, \lambda)].
\end{aligned}$$

As the derivation and the language are similar to $internal\ loop\ structure$, we omit the sample derivation. ■

Lemma 11. *The multi-branch loop structure (see Fig. 5(a))*

$$L_{mbl} = \{u_1v_1\bar{v_1}^Ru_2\bar{u_2}^Rv_2\bar{u_1}^R \mid u_1, u_2, v_1, v_2 \in \Sigma^*_{RNA}\}$$

can be generated by a matrix insertion–deletion system.

Proof. The language L_{mbl} can be generated by the matrix insertion–deletion system

$$\Upsilon_{mbl} = (\{b, \bar{b}, \dagger_1, \dagger_2, \dagger_3, \dagger_4\}, \{b, \bar{b}\}, \{\dagger_1\,\dagger_3\,\dagger_4\dagger_2\}, R),$$

where $b \in \{a, u, g, c\}$, $\bar{b}$ is complement of b and R is given as follows:

$$\begin{aligned}
R_{I_1} &= [(\lambda, \lambda/b, \dagger_1), (\dagger_2, \lambda/\bar{b}, \lambda)],\\
R_{I_2} &= [(\lambda, \lambda/b, \dagger_4), (\dagger_4, \lambda/\bar{b}, \lambda)],\\
R_{I_3} &= [(\lambda, \lambda/b, \dagger_3), (\dagger_3, \lambda/\bar{b}, \lambda)],\\
R_{I_4} &= [(\lambda, \lambda/b, \dagger_2)],\\
R_{D_1} &= [(\lambda, \dagger_1/\lambda, \lambda)],\\
R_{D_2} &= [(\lambda, \dagger_2/\lambda, \lambda)],\\
R_{D_3} &= [(\lambda, \dagger_3/\lambda, \lambda)],\\
R_{D_4} &= [(\lambda, \dagger_4/\lambda, \lambda)].
\end{aligned}$$

As the derivation and the language are similar to $internal\ loop\ structure$, we omit the sample derivation. ■

Lemma 12. *The extended pseudoknot structure (see Fig. 5(b))*

$$L_{epk} = \{u_1v_1\bar{u_1}^Ru_2\bar{v_1}^R\bar{u_2}^R \mid u_1, u_2, v_1 \in \Sigma^*_{RNA}\}$$

can be generated by a matrix insertion–deletion system.

Proof. The language L_{epk} can be generated by the matrix insertion–deletion system

$$\Upsilon_{epk} = (\{b, \bar{b}, \dagger_1, \dagger_2, \dagger_3, \dagger_4\}, \{b, \bar{b}\}, \{\dagger_1\,\dagger_2\,\dagger_3\dagger_4\}, R),$$

where $b \in \{a, u, g, c\}$, $\bar{b}$ is a complement of b and R is given as follows:

$$\begin{aligned}
R_{I_1} &= [(\lambda, \lambda/b, \dagger_1), (\dagger_2, \lambda/\bar{b}, \lambda)],\\
R_{I_2} &= [(\dagger_1, \lambda/b, \lambda), (\dagger_3, \lambda/\bar{b}, \lambda)],\\
R_{I_3} &= [(\lambda, \lambda/b, \dagger_3), (\dagger_4, \lambda/\bar{b}, \lambda)],\\
R_{D_1} &= [(\lambda, \dagger_1/\lambda, \lambda)],\\
R_{D_2} &= [(\lambda, \dagger_2/\lambda, \lambda)],\\
R_{D_3} &= [(\lambda, \dagger_3/\lambda, \lambda)],\\
R_{D_4} &= [(\lambda, \dagger_4/\lambda, \lambda)].
\end{aligned}$$

A sample derivation is given as follows:

$$^{\downarrow}\dagger_1\dagger_2^{\downarrow}\dagger_3\dagger_4 \Longrightarrow_{R_{I_1}} \underline{a}^{\downarrow}\dagger_1\dagger_2^{\downarrow}\underline{u}\dagger_3\dagger_4$$
$$\Longrightarrow_{R_{I_1}} a\underline{g}^{\downarrow}\dagger_1\dagger_2^{\downarrow}\underline{c}u\dagger_3\dagger_4$$
$$\Longrightarrow_{R_{I_1}} agc\dagger_1^{\downarrow}\dagger_2\underline{g}cu\dagger_3^{\downarrow}\dagger_4$$
$$\Longrightarrow_{R_{I_2}} agc\dagger_1^{\downarrow}\underline{a}\dagger_2 gcu\dagger_3^{\downarrow}\underline{u}\dagger_4$$
$$\Longrightarrow_{R_{I_2}} agc\dagger_1\underline{g}a\dagger_2 gcu^{\downarrow}\dagger_3\underline{c}u\dagger_4^{\downarrow}$$
$$\Longrightarrow_{R_{I_3}} agc\dagger_1 ga\dagger_2 gcu\underline{a}^{\downarrow}\dagger_3 cu\dagger_4^{\downarrow}\underline{u}$$
$$\Longrightarrow_{R_{I_3}} agc\dagger_1 ga\dagger_2 gcua\underline{c}\dagger_3 cu\dagger_4\underline{g}u$$
$$\Longrightarrow_{R_{D_1}} agc^{\Downarrow}ga\dagger_2 gcuac\dagger_3 cu\dagger_4 gu$$
$$\Longrightarrow_{R_{D_2}} agcga^{\Downarrow}gcuac\dagger_3 cu\dagger_4 gu$$
$$\Longrightarrow_{R_{D_3}} agcgagcuac^{\Downarrow}cu\dagger_4 gu$$
$$\Longrightarrow_{R_{D_4}} agcgagcuaccu^{\Downarrow}gu.$$

■

Lemma 13. *The kissing hairpin structure (see Fig. 6)*

$$L_{khp} = \{u_1v_1A\bar{A}v_2u_2\bar{u_2}^R v_3B\bar{B}v_4\bar{u_1}^R \mid u_1, u_2, v_1, v_2, v_3, v_4 \in \Sigma^*_{RNA} \text{ and } A, B \in \Sigma_{RNA}\}$$

can be generated by a matrix insertion–deletion system.

Proof. The language L_{khp} can be generated by the matrix insertion–deletion system

$$\Upsilon_{khp} = (\{b, \bar{b}, \dagger_1, \dagger_2, \dagger_3, \dagger_4\}, \{b, \bar{b}\}, \{\dagger_1 A\bar{A} \dagger_3 \dagger_4 B\bar{B}\dagger_2\}, R),$$

where $A, B, b \in \{a, u, g, c\}$, $\bar{A}, \bar{B}, \bar{b}$ is the complement of A, B, b and R is given as follows:

$$\begin{aligned}
R_{I_1} &= [(\lambda, \lambda/b, \dagger_1), (\dagger_2, \lambda/\bar{b}, \lambda)],\\
R_{I_2} &= [(\dagger_3, \lambda/b, \lambda), (\lambda, \lambda/\bar{b}, \dagger_4)],\\
R_{I_3} &= [(\dagger_1, \lambda/b, \lambda)],\\
R_{I_4} &= [(\lambda, \lambda/b, \dagger_3)],\\
R_{I_5} &= [(\lambda, \lambda/b, \dagger_2)],\\
R_{I_6} &= [(\dagger_4, \lambda/b, \lambda)],\\
R_{D_1} &= [(\lambda, \dagger_1/\lambda, \lambda)],\\
R_{D_2} &= [(\lambda, \dagger_2/\lambda, \lambda)],\\
R_{D_3} &= [(\lambda, \dagger_3/\lambda, \lambda)],\\
R_{D_4} &= [(\lambda, \dagger_4/\lambda, \lambda)].
\end{aligned}$$

A sample derivation is given as follows:

$$\begin{aligned}
&{}^{\downarrow}\dagger_1 au \dagger_3 \dagger_4 ua \dagger_2^{\downarrow} \Longrightarrow_{R_{I_1}} \underline{c}^{\downarrow} \dagger_1 au \dagger_3 \dagger_4 ua \dagger_2^{\downarrow} \underline{g}\\
&\Longrightarrow_{R_{I_1}} c\underline{u} \dagger_1 au \dagger_3 {}^{\downarrow\downarrow}\dagger_4 ua \dagger_2 \underline{ag}\\
&\Longrightarrow_{R_{I_2}} cu \dagger_1^{\downarrow} au \dagger_3 \underline{c}\,\underline{g} \dagger_4 ua \dagger_2 ag\\
&\Longrightarrow_{R_{I_3}} cu \dagger_1 \underline{g}au^{\downarrow} \dagger_3 cg \dagger_4 ua \dagger_2 ag\\
&\Longrightarrow_{R_{I_4}} cu \dagger_1 gau\underline{u} \dagger_3 cg \dagger_4 ua^{\downarrow} \dagger_2 ag\\
&\Longrightarrow_{R_{I_5}} cu \dagger_1 gauu \dagger_3 cg \dagger_4^{\downarrow} ua\underline{c} \dagger_2 ag\\
&\Longrightarrow_{R_{I_6}} cu \dagger_1 gauu \dagger_3 cg \dagger_4 \underline{g}uac \dagger_2 ag\\
&\Longrightarrow_{R_{D_1}} cu^{\Downarrow}gauu \dagger_3 cg \dagger_4 guac \dagger_2 ag\\
&\Longrightarrow_{R_{D_2}} cugauu \dagger_3 cg \dagger_4 guac^{\Downarrow}ag\\
&\Longrightarrow_{R_{D_3}} cugauu^{\Downarrow}cg \dagger_4 guacag\\
&\Longrightarrow_{R_{D_4}} cugauucg^{\Downarrow}guacag.
\end{aligned}$$

The idea for generating the language L_{khp} is given as follows. As $A\bar{A}$ and $B\bar{B}$ are already present in the axiom, markers are not required to generate it. To generate the remaining part of the language, $\dagger_1$, $\dagger_2$, $\dagger_3$ and $\dagger_4$ are used as markers. $\dagger_1$ and $\dagger_2$ are used to control the $u_1\bar{u_1}^R$ part of the language. Whenever a b is adjoined to the left of $\dagger_1$, its corresponding complementary $\bar{b}$ is adjoined to the right of $\dagger_2$ and the synchronization is maintained. Similarly, $\dagger_3$ and $\dagger_4$ are used to control the $u_2\bar{u_2}^R$ part of the language, $\dagger_1$,$\dagger_3$, $\dagger_4$ and $\dagger_2$ are used to control the v_1, v_2, v_3 and v_4 part of the language, respectively. ■

Lemma 14. *The simple H-type structure (see Fig. 7(a))*

$$L_{sht} = \{u_1v_1u_2\bar{u_1}^R v_2\bar{u_2}^R \mid u_1, u_2, v_1, v_2 \in \Sigma^*_{RNA}\}$$

can be generated by a matrix insertion–deletion system.

Proof. The language L_{sht} can be generated by the matrix insertion–deletion system

$$\Upsilon_{sht} = (\{b, \bar{b}, \dagger_1, \dagger_2, \dagger_3\}, \{b, \bar{b}\}, \{\dagger_1 \dagger_2 \dagger_3\}, R),$$

where $b \in \{a, u, g, c\}$, $\bar{b}$ is a complement of b and R is given as follows:

$$\begin{aligned}
R_{I_1} &= [(\lambda, \lambda/b, \dagger_1), (\dagger_2, \lambda/\bar{b}, \lambda)],\\
R_{I_2} &= [(\lambda, \lambda/b, \dagger_2), (\dagger_3, \lambda/\bar{b}, \lambda)],\\
R_{I_3} &= [(\dagger_1, \lambda/b, \lambda)],\\
R_{I_4} &= [(\lambda, \lambda/b, \dagger_3)],\\
R_{D_1} &= [(\lambda, \dagger_1/\lambda, \lambda)],\\
R_{D_2} &= [(\lambda, \dagger_2/\lambda, \lambda)],\\
R_{D_3} &= [(\lambda, \dagger_3/\lambda, \lambda)].
\end{aligned}$$

As the derivation and the language are similar to *extended pseudoknot structure*, we omit the sample derivation. ■

Lemma 15. *The recursive pseudoknot structure (see Fig. 7(b))*

$$L_{rps} = \{u_1u_2u_3\bar{u_2}^R u_4\bar{u_1}^R\bar{u_4}^R u_5\bar{u_5}^R\bar{u_3}^R \mid u_1, u_2, u_3, u_4, u_5 \in \Sigma^*_{RNA}\}$$

can be modelled using a matrix insertion–deletion system.

Proof. The language L_{rps} can be generated by the matrix insertion–deletion system

$$\Upsilon_{rps} = (\{b, \bar{b}, \dagger_1, \dagger_2, \dagger_3, \dagger_4 \dagger_5\}, \{b, \bar{b}\}, \{\dagger_1\dagger_2\dagger_3\dagger_4 \dagger_5\}, R),$$

where $b \in \{a, u, g, c\}$, $\bar{b}$ is a complement of b and R is given as follows:

$$\begin{aligned}
R_{I_1} &= [(\lambda, \lambda/b, \dagger_1), (\dagger_3, \lambda/\bar{b}, \lambda)],\\
R_{I_2} &= [(\dagger_1, \lambda/b, \lambda), (\dagger_2, \lambda/\bar{b}, \lambda)],\\
R_{I_3} &= [(\lambda, \lambda/b, \dagger_2), (\dagger_5, \lambda/\bar{b}, \lambda)],\\
R_{I_4} &= [(\lambda, \lambda/b, \dagger_3), (\lambda, \lambda/\bar{b}, \dagger_4)],\\
R_{I_5} &= [(\dagger_4, \lambda/b, \lambda), (\lambda, \lambda/\bar{b}, \dagger_5)],\\
R_{D_1} &= [(\lambda, \dagger_1/\lambda, \lambda)],\\
R_{D_2} &= [(\lambda, \dagger_2/\lambda, \lambda)],\\
R_{D_3} &= [(\lambda, \dagger_3/\lambda, \lambda)],\\
R_{D_4} &= [(\lambda, \dagger_4/\lambda, \lambda)],\\
R_{D_5} &= [(\lambda, \dagger_5/\lambda, \lambda)].
\end{aligned}$$

A sample derivation is given as follows:

$$
\begin{aligned}
&{}^{\downarrow}\dagger_1\,\dagger_2\,\dagger_3^{\downarrow}\,\dagger_4\dagger_5 \Longrightarrow_{R_{I_1}} \underline{a}^{\downarrow}\,\dagger_1\,\dagger_2\,\dagger_3^{\downarrow}\,\underline{u}\,\dagger_4\,\dagger_5\\
&\Longrightarrow_{R_{I_1}} \underline{a}u\,\dagger_1^{\downarrow}\,\dagger_2^{\downarrow}\,\dagger_3\,\underline{a}u\,\dagger_4\,\dagger_5\\
&\Longrightarrow_{R_{I_2}} au\,\dagger_1^{\downarrow}\,\underline{g}\,\dagger_2^{\downarrow}\,\underline{c}\,\dagger_3\,au\,\dagger_4\,\dagger_5\\
&\Longrightarrow_{R_{I_2}} au\,\dagger_1\,\underline{u}g^{\downarrow}\,\dagger_2\,\underline{a}c\,\dagger_3\,au\,\dagger_4\,\dagger_5^{\downarrow}\\
&\Longrightarrow_{R_{I_3}} au\,\dagger_1\,ug\underline{u}^{\downarrow}\,\dagger_2\,ac\,\dagger_3\,au\,\dagger_4\,\dagger_5^{\downarrow}\underline{a}\\
&\Longrightarrow_{R_{I_3}} au\,\dagger_1\,ugu\underline{c}\,\dagger_2\,ac^{\downarrow}\,\dagger_3\,au^{\downarrow}\,\dagger_4\,\dagger_5\underline{g}a\\
&\Longrightarrow_{R_{I_4}} au\,\dagger_1\,uguc\,\dagger_2\,ac\underline{a}^{\downarrow}\,\dagger_3\,au\underline{u}^{\downarrow}\,\dagger_4\,\dagger_5ga\\
&\Longrightarrow_{R_{I_4}} au\,\dagger_1\,uguc\,\dagger_2\,aca\underline{u}\,\dagger_3\,auu\underline{a}\,\dagger_4\,{}^{\downarrow\downarrow}\,\dagger_5\,ga\\
&\Longrightarrow_{R_{I_5}} au\,\dagger_1\,uguc\,\dagger_2\,acau\,\dagger_3\,auua\,\dagger_4\,{}^{\downarrow}\underline{g}\underline{c}^{\downarrow}\,\dagger_5\,ga\\
&\Longrightarrow_{R_{I_5}} au\,\dagger_1\,uguc\,\dagger_2\,acau\,\dagger_3\,auua\,\dagger_4\,\underline{u}gc\underline{a}\,\dagger_5\,ga\\
&\Longrightarrow_{R_{D_1}} au^{\Downarrow}uguc\,\dagger_2\,acau\,\dagger_3\,auua\,\dagger_4\,ugca\,\dagger_5\,ga\\
&\Longrightarrow_{R_{D_2}} auuguc^{\Downarrow}acau\,\dagger_3\,auua\,\dagger_4\,ugca\,\dagger_5\,ga\\
&\Longrightarrow_{R_{D_3}} auugucacau^{\Downarrow}auua\,\dagger_4\,ugca\,\dagger_5\,ga\\
&\Longrightarrow_{R_{D_4}} auugucacauauua^{\Downarrow}ugca\,\dagger_5\,ga\\
&\Longrightarrow_{R_{D_5}} auugucacauauuaugca^{\Downarrow}ga.
\end{aligned}
$$

■

Lemma 16. *The three-knot structure (see Fig. 8)*

$$L_{tks} = \{u_1vu_2u_3\bar{u_1}^R\bar{u_2}^R\bar{u_3}^R \mid u_1,u_2,u_3,v \in \Sigma^*_{RNA}\}$$

can be generated by a matrix insertion–deletion system.

Proof. The language L_{tks} can be generated by the matrix insertion–deletion system

$$\Upsilon_{tks} = (\{b,\bar{b},\dagger_1,\dagger_2,\dagger_3,\dagger_4\},\{b,\bar{b}\},\{\dagger_1\,\dagger_2\,\dagger_3\dagger_4\},R),$$

where $b \in \{a,u,g,c\}$, $\bar{b}$ is the complement of b and R is given as follows:

$$
\begin{aligned}
R_{I_1} &= [(\lambda,\lambda/b,\dagger_1),(\lambda,\lambda/\bar{b},\dagger_3)],\\
R_{I_2} &= [(\lambda,\lambda/b,\dagger_2),(\dagger_3,\lambda/\bar{b},\lambda)],\\
R_{I_3} &= [(\dagger_2,\lambda/b,\lambda),(\lambda,\lambda/\bar{b},\dagger_4)],\\
R_{I_4} &= [(\dagger_1,\lambda/b,\lambda)],\\
R_{D_1} &= [(\lambda,\dagger_1/\lambda,\lambda)],\\
R_{D_2} &= [(\lambda,\dagger_2/\lambda,\lambda)],\\
R_{D_3} &= [(\lambda,\dagger_3/\lambda,\lambda)],\\
R_{D_4} &= [(\lambda,\dagger_4/\lambda,\lambda)].
\end{aligned}
$$

A sample derivation is given as follows:

$$
\begin{aligned}
&{}^{\downarrow}\dagger_1\,\dagger_2^{\downarrow}\,\dagger_3\,\dagger_4 \Longrightarrow_{R_{I_1}} \underline{a}^{\downarrow}\,\dagger_1\,\dagger_2\underline{u}^{\downarrow}\,\dagger_3\,\dagger_4\\
&\Longrightarrow_{R_{I_1}} a\underline{u}\,\dagger_1^{\downarrow}\,\dagger_2 u\underline{a}^{\downarrow}\,\dagger_3^{\downarrow}\,\dagger_4\\
&\Longrightarrow_{R_{I_2}} au\,\dagger_1\,\underline{g}^{\downarrow}\,\dagger_2\,ua^{\downarrow}\,\dagger_3^{\downarrow}\,\underline{c}\,\dagger_4\\
&\Longrightarrow_{R_{I_2}} au\,\dagger_1\,g\underline{u}\,\dagger_2^{\downarrow}\,ua\,\dagger_3\,\underline{a}c^{\downarrow}\,\dagger_4\\
&\Longrightarrow_{R_{I_3}} au\,\dagger_1\,gu\,\dagger_2^{\downarrow}\,\underline{u}ua\,\dagger_3\,ac\underline{a}^{\downarrow}\,\dagger_4\\
&\Longrightarrow_{R_{I_3}} au\,\dagger_1^{\downarrow}\,gu\,\dagger_2\,\underline{c}uua\,\dagger_3\,acag\,\dagger_4\\
&\Longrightarrow_{R_{I_4}} au\,\dagger_1^{\downarrow}\,\underline{g}gu\,\dagger_2\,cuua\,\dagger_3\,acag\,\dagger_4\\
&\Longrightarrow_{R_{I_4}} au\,\dagger_1\,\underline{c}ggu\,\dagger_2\,cuua\,\dagger_3\,acag\,\dagger_4\\
&\Longrightarrow_{R_{D_1}} au^{\Downarrow}cggu\,\dagger_2\,cuua\,\dagger_3\,acag\,\dagger_4\\
&\Longrightarrow_{R_{D_2}} aucggu^{\Downarrow}cuua\,\dagger_3\,acag\,\dagger_4\\
&\Longrightarrow_{R_{D_3}} aucggucuua^{\Downarrow}acag\,\dagger_4\\
&\Longrightarrow_{R_{D_4}} aucggucuuaacag^{\Downarrow}.
\end{aligned}
$$

■

6. Conclusion

In this paper, we introduced the matrix insertion–deletion system and, using it, we modelled several bio-molecular structures that occur at the intramolecular, intermolecular and RNA secondary structure level. These structures are often discussed in the literature, thus they can be assumed to be important bio-molecular structures.

We remark that, in this paper, to model all the bio-molecular structures we used a matrix of insertion rules and a matrix of deletion rules separately (i.e., the system has no insertion rule and deletion rule together in a matrix) thus forming a new subclass. In all the systems we considered here, the insertion rule uses context and the deletion rule uses no context. This can be viewed as follows: the insertion operation works in a context-sensitive manner whereas the deletion operation works in a context-free manner. Thus, the system uses both the nature of context-sensitiveness and context-freeness, and it seems to be a promising model for application to various domains such as molecular biology and linguistics. With matrices consisting of both insertion and deletion rules, more complicated structures such as tertiary structures, quaternary structures, protein secondary structures (Mamitsuka and Abe, 1994) can be modelled, and this is left as a future research work.

From the computational linguistic point of view, one natural question is where to place this newly introduced formalism among the other formalisms that exist in this domain. Though the parsing algorithm for tree adjoining grammars (*TAG*) and *linear indexed grammar* take $O(n^6)$ (Pardo *et al.*, 1997), they have their own deficiency. For example, *TAG* cannot generate $L_i = \{a_1^n a_2^n \ldots a_i^n \mid n \geq 1\}$ for $i \geq 5$ and the triple copy

language $L_{3cp} = \{www \mid w \in \{a,b\}^*\}$. However, multi-component TAG (*MCTAG*) can cover them. But there are languages like $L_{nsl} = \{a^{2^n}b^n \mid n \geq 1\}$ that cannot be covered by *MCTAG*, although, on the other hand, they are accepted by a Turing machine in polynomial time (Boullier and Sagot, 2011). With the introduced subclass variant, we can easily generate $L_i,\ i \geq 5$ and L_{3cp} languages; however, to generate L_{nsl}, it seems required that both insertion and deletion rules be used together in a matrix. We suppose that the introduced grammar formalism can emulate *TAG*, but subsumed by the range concatenation grammar (*RCG*). Analyzing the relationship among these grammar formalisms with the introduced formalism in detail is out of objective of this paper, but can be carried out as a future work.

Acknowledgment

The first author would like to acknowledge the project SR/S3/EECE/054/2010, Department of Science and Technology, New Delhi, India. The second author is on leave from the parent institute at VIT University, Vellore, India. The authors would also like to thank the anonymous referees for their useful suggestions and corrections.

References

Boullier, P. and Sagot, B. (2011). Multi-component tree insertion grammars, *in* P. De Groote *et al.* (Eds.), *Formal Grammar 2009*, Lecture Notes in Artificial Intelligence, Vol. 5591, Springer, Berlin/Heidelberg, pp. 31–46.

Brendel, V. and Busse, H.G. (1984). Genome structure described by formal languages, *Nucleic Acids Research* **12**(5): 2561–2568.

Brown, M. and Wilson, C. (1995). RNA pseudoknot modelling using intersections of stochastic context free grammars with applications to database search, *Proceedings of the Pacific Symposium on Biocomputing, Big Island, HI, USA*, pp. 109–125.

Cai, L., Russell, L. and Wu, Y. (2003). Stochastic modelling of RNA pseudoknotted structures: A grammatical approach, *Bioinformatics* **19**(1): 66–73.

Calude, C.S. and Paŭn, Gh. (2001). *Computing with Cells and Atoms: An Introduction to Quantum, DNA and Membrane Computing*, Taylor and Francis, London.

Chiang, D., Joshi, A.K. and Searls, D.B. (2006). Grammatical representations of macromolecular structure, *Journal of Computational Biology* **13**(5): 1077–1100.

Dong, S. and Searls, D.B. (1994). Gene structure prediction by linguistic methods, *Genomics* **23**(3): 540–551.

Dorigo, M. and Stutzle, T. (2004). *Ant Colony Optimization*, MIT Press, Cambridge, MA.

Durbin, R., Eddy, S., Krogh, A. and Mitchison, G. (1998). *Biological Sequence Analysis*, Cambridge University Press, Cambridge.

Eiben, A.E. and Smith, J.E. (2003). *Introduction to Evolutionary Computing*, Springer, Berlin/Heidelberg.

Galiukschov, B.S. (1981). Semicontextual grammars, *Matematicheskaya Logika i Matematicheskaya Lingvistika*: 38–50, (in Russian).

Goldberg, E.D. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Boston, MA.

Haussler, D. (1982). *Insertion and Iterated Insertion as Operations on Formal Languages*, Ph.D. thesis, University of Colorado, Boulder, CO.

Haussler, D. (1983). Insertion languages, *Information Science* **131**(1): 77–89.

Head, T. (1987). Formal language theory and DNA: An analysis of the generative capacity of specific recombinant behaviors, *Bulletin of Mathematical Biology* **49**(6): 737–750.

Kuppusamy, L., Mahendran, A. and Krishna, S.N. (2011a). Matrix insertion–deletion systems for bio-molecular structures, *in* R. Natarajan and A. Ojo (Eds.), *ICDCIT-2011*, Lecture Notes in Computer Science, Vol. 6536, Springer, Berlin/Heidelberg, pp. 301–311.

Kuppusamy, L., Mahendran, A. and Clergerie, E.V. (2011b). Modelling intermolecular structures and defining ambiguity in gene sequences using matrix insertion–deletion systems in biology, computation and linguistics, *in* G.B. Enguix *et al.* (Eds.), *New Interdisciplinary Paradigms*, IOS Press, Amsterdam, pp. 71–85.

Lyngso, R.B., Zuker, M. and Pedersen, C.N.S. (1999). Internal loops in RNA secondary structure prediction, *RECOMB99, Proceedings of the 3rd International Conference on Computational Molecular Biology, Lyon, France*, pp. 260–267.

Lyngso, R.B. and Pedersen, C.N.S. (2000). Pseudoknots in RNA secondary structure, *RECOMB00, Proceedings of the 4th Annual International Conference on Computational Molecular Biology, Tokyo, Japan* pp. 201–209.

Mamitsuka, H. and Abe, N. (1994). Prediction of beta-sheet structures using stochastic tree grammars, *Proceedings of the 5th Workshop on Genome Informatics, Yokohama, Japan*, pp. 19–28.

Pardo, M.A.A., Clergerie, E.V. and Ferro, M.V. (1997). Automata-based parsing in dynamic programming for LIG, *in* A.S. Narinyani (Ed.), *Proceedings of the DIALOGUE'97 Computational Linguistics and Its Applications Workshop, Moscow, Russia*, pp. 22–27.

Păun, Gh., Rozenberg, G. and Salomaa, A. (1998). *DNA Computing: New Computing Paradigms*, Springer, Berlin/Heidelberg.

Păun, Gh. (2002). *Membrane Computing: An Introduction*, Springer, Berlin/Heidelberg.

Petre, I. and Verlan, S. (2012). Matrix insertion–deletion systems, *Theoretical Computer Science* **456**: 80–88.

Rivas, E. and Eddy, S.R. (2000). The language of RNA: A formal grammar that includes pseudoknots, *Bioinformatics* **16**(4): 334–340.

Rozenberg, G. and Salomaa, A. (1997). *Handbook of Formal Languages*, Vol. 1, Springer, New York, NY.

Sakakibara, Y., Brown, R., Hughey, R., Mian, I.S., Sjolander, K., Underwood, R.C. and Haussler, D. (1996). Stochastic context-free grammars for tRNA modelling, *Nucleic Acids Research* **22**(23): 5112–5120.

Sakakibara, Y. (2003). Pair hidden Markov models on tree structures, *Bioinformatics* **19**(1): 232–240.

Searls, D.B. (1988). Representing genetic information with formal grammars, *Proceedings of the National Conference on Artificial Intelligence, Saint Paul, MN, USA*, pp. 386–391.

Searls, D.B. (1992). The linguistics of DNA, *American Scientist* **80**(6): 579–591.

Searls, D.B. (1993). The computational linguistics of biological sequences, *in* L. Hunter (Ed.), *Artificial Intelligence and Molecular Biology*, AAAI Press, Paolo Alto, CA, pp. 47–120.

Searls, D.B. (1995). Formal grammars for intermolecular structures, *1st International IEEE Symposium on Intelligence and Biological Systems, Washington, DC, USA*, pp. 30–37.

Searls, D.B. (2002). The language of genes, *Nature* **420**(6912): 211–217.

Theis, C., Janssen, S. and Giegerich, R. (2010). Prediction of RNA secondary structure including kissing hairpin motifs, *Proceedings of WABI 2010, Liverpool, UK*, pp. 52–64.

Uemura, Y, Hasegawa, A., Kobayashi, S. and Yokomori, T. (1999). Tree adjoining grammars for RNA structure prediction, *Theoretical Computer Science* **210**(2): 277–303.

Yuki, S. and Kasami, T. (2006). RNA pseudoknotted structure prediction using stochastic multiple context-free grammar, *IPSJ Transactions on Bioinformatics* **47**: 12–21.

Lakshmanan Kuppusamy received his Ph.D. from IIT Madras, India (2004), in theoretical computer science. His research interests include formal language theory and automata, bio-inspired computing models and epistemic logic. He has published more than 30 papers in international journals and refereed international conferences. He is currently working as a professor in the School of Computing Science and Engineering at VIT University, Vellore, India.

Anand Mahendran received his Ph.D. (computer science and engineering) from VIT University, India, in 2012, his M.E. (computer science and engineering) from Anna University, India, in 2005, and his B.E. (computer science and engineering) from VIT University, India, in 2003. His research interests include formal language theory and automata, and bio-inspired computing models. He has published more than 20 papers in international journals and refereed international conferences. He works as an assistant professor in the College of Computer Science and Information Systems, Jazan University, Kingdom of Saudi Arabia.

Received: 8 August 2014
Revised: 17 March 2015
Re-revised: 30 July 2015