Matrix Insertion-Deletion Systems for Bio-Molecular Structures

Lakshmanan Kuppusamy[1], Anand Mahendran[1], and Shankara Narayanan Krishna[2]

[1] School of Computing Science and Engineering, VIT University, Vellore-632 014, India
klakshma@vit.ac.in, manand@vit.ac.in
[2] Department of Computer Science and Engineering, IIT Bombay, Powai - 400 076, India
krishnas@cse.iitb.ac.in

Abstract. Insertion and deletion are considered to be the basic operations in Biology, more specifically in DNA processing and RNA editing. Based on these evolutionary transformations, a computing model has been formulated in formal language theory known as *insertion-deletion* systems. Since the biological macromolecules can be viewed as symbols, the gene sequences can be represented as strings. This suggests that the molecular representations can be theoretically analyzed if a biologically inspired computing model recognizes various bio-molecular structures like *pseudoknot, hairpin, stem and loop, cloverleaf* and *dumbbell*. In this paper, we introduce a simple grammar system that encompasses many bio-molecular structures including the above mentioned structures. This new grammar system is based on insertion-deletion and *matrix* grammar systems and is called *Matrix insertion-deletion grammars*. Finally, we discuss how the ambiguity levels defined for insertion-deletion grammar systems can be realized in bio-molecular structures, thus the ambiguity issues in gene sequences can be studied in terms of grammar systems.

Keywords: bio-molecules, structure representation, insertion-deletion systems, matrix grammars, ambiguity.

1 Introduction

In the last three decades, biology played a great role in the field of formal languages by being the root for the development of various biologically inspired computing models such as *sticker systems, splicing systems, Watson-Crick automata, insertion-deletion systems, p systems* [3], [8], [9]. Since, most of the language generating devices are based on the operation of rewriting systems, the insertion-deletion systems opened a particular attention in the field of formal languages. Informally, the insertion and deletion operations of an insertion-deletion system is defined as follows: If a string α is inserted between two parts w_1 and w_2 of a string w_1w_2 to get $w_1\alpha w_2$, we call the operation as insertion, whereas if

R. Natarajan and A. Ojo (Eds.): ICDCIT 2011, LNCS 6536, pp. 301–312, 2011.

a substring β is deleted from a string $w_1\beta w_2$ to get w_1w_2, we call the operation as deletion.

DNA molecules may be considered as strings over alphabet consisting of four symbols namely a, t, g and c. Similarly, RNA molecules may be considered as strings over alphabet consisting of four symbols namely a, u, g and c. Since the bio-molecular structures can be defined in terms of sequence of symbols (i.e., strings) there exists a correlation between formal grammars and bio-molecular structures. The following example shares a common point between formal grammar and molecular strings. Consider the following gene sequence $S = ctatcgcgatag$. As $\bar{a} = t$, $\bar{t} = a$, $\bar{g} = c$ and $\bar{c} = g$, the above gene sequence resembles the context-free (palindrome) language $\{w\bar{w}^R \mid w \in \{a, b\}^*\}$.

The gene sequences in the bio-molecular structures can be viewed as strings which has some common patterns in it. In [7], [6] it has been shown that there exists a relevance between the gene sequences and natural language constructs such as *triple agreements* : $L_1 = \{a^nb^nc^n \mid n \geq 1\}$, *crossed dependencies:* $L_2 = \{a^nb^mc^nd^m \mid n, m \geq 1\}$. We discuss in brief some of the important structures seen in bio-molecules. Fig.1. shows the two structures which are predominantly available in bio-molecules like proteins, DNA and RNA molecular structures. Fig.2. shows the coherence between natural language constructs and gene sequences. Fig.2.(a) and (b) represents the grammar oriented derivations of idealized RNA structures (# denotes empty string) whereas Fig.2.(c) represents a biological sequence which has a crossed dependency pattern. The corresponding languages for the above structures can be given as *hairpin language* (for Fig.2(a)), orthodox language (for Fig.2(b)) and pseudoknot structure (for Fig.2(c)). 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 we refer to [2].

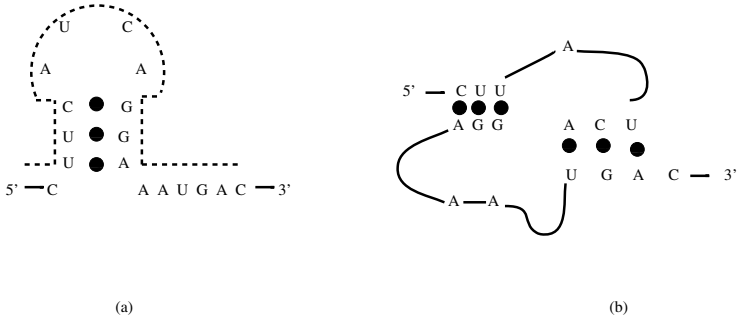

Fig. 1. Bio-molecular structures: (a)stem and loop (b)pseudo-knot

In the last two decades, many attempts have been made to establish the linguistic behaviour of biological sequences by defining new grammar formalisms like *cut grammars* [4], [5], [6], *crossed-interaction grammar* [7], *simple linear tree adjoining grammars* and *extended simple linear tree adjoining grammars* [15]

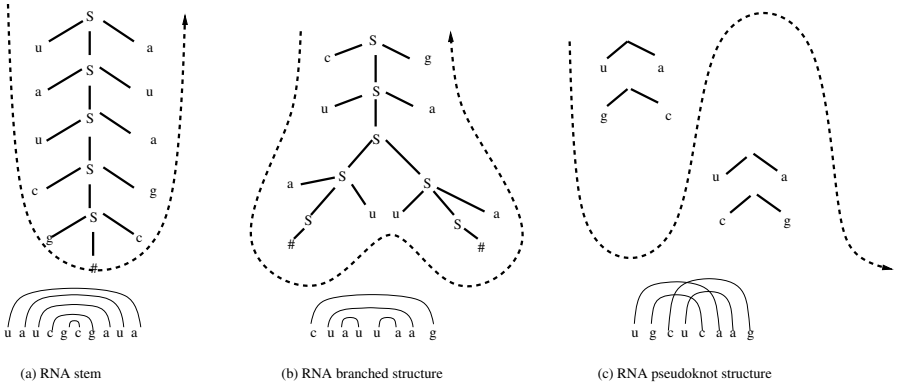

Fig. 2. RNA structures

which are capable of generating some of the biological structures mentioned above. However, there is no unique grammar system that encapsulate all discussed bio-molecular structures. This motivates us to introduce a simple and powerful grammar systems that captures all the essential and important bio-molecular structures.

Ambiguity is considered as one of the fundamental problems in formal language theory. A grammar is said to be ambiguous, if there exists more than one distinct derivation of the words in the generated language. As the insertion-deletion system can be applied in DNA processing [8], the ambiguity in DNA processing (which uses the insertion system) can happen in the following manner. Let W_1W_2 be a DNA strand and suppose we want to insert $W_3W_4W_5$ between W_1 and W_2 to obtain another DNA strand $W_1W_3W_4W_5W_2$. This can be done first by inserting W_3 between W_1 and W_2, followed by inserting W_4 between W_3 and W_2, followed by inserting W_5 between W_4 and W_2. The other sequence would be first by inserting W_5 between W_1 and W_2, followed by inserting W_4 between W_1 and W_5, followed by inserting W_3 between W_1 and W_4. This shows that ambiguity in gene sequences is also possible (i.e., starting from one sequence we are able to get another sequence in more than one way such that the intermediate sequences are different). This motivates us to study about the ambiguity in gene sequences by using Matrix insertion-deletion systems. Study of this concept of ambiguity may be useful in considering inheritance properties and phylogenetic trees [14]. More specifically, when these intermediate sequences are represented as phylogenetic trees, we can see that the trees are different and thus it might help us to identify the inheritance properties.

With the above mentioned details, in this paper we first introduce a simple and powerful bio-inspired distributed computing model named Matrix insertion-deletion systems. This model is obtained by combining insertion-deletion and matrix grammars. Next, in Section 4 we show that the newly introduced variant can capture the various important and essential bio-molecular structures described in DNA, RNA, protein like *hairpin*, *stem and loop*, *ideal*, *orthodox*,

dumbbell, *pseudoknot*, *clover-leaf*, *attenuator* and *non-ideal attenuator*. In Section 5, we analyze the application of ambiguity in gene sequences by using the Matrix insertion-deletion system and discuss about the universality result for the new system.

2 Preliminaries

We assume that the readers are familiar with the notions of formal language theory. However, we recall the basic notions which are used in the paper. A finite non-empty set V or Σ is called an alphabet. We denote by V^* or Σ^*, the free monoid generated by V or Σ, by λ it 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*. For any word $w \in V^*$ or Σ^*, we denote the length of w by $|w|$. For more details on formal language theory, we refer to [10]. RE represents the family of recursive enumerable languages.

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

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/deleted)

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

The language generated by γ is defined by

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

where $\Longrightarrow^*$ is the reflexive and transitive closure of the relation $\Longrightarrow$. The family of languages generated by the insertion-deletion systems is given as $INS_n^m DEL_p^q$ for $n, m, p, q \geq 0$, where n denotes the maximal length of the inserted string, m denotes the maximal length of the context in insertion rules, p denotes the maximal length of the deleted string and q denotes the maximal length of the context in deletion rules.

Next, we will look into the definition of matrix grammars. 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 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 $m = [P_1 \to Q_1, ..., P_r \to Q_r], r \geq 1$. Some rules in a matrix are exempted in applying the derivation if those rules are present in the set *appearance checking*. The language generated by the matrix grammar is defined by $L(G) = \{w \in T^* \mid S \Longrightarrow^* w\}$. The family of languages generated by matrix grammars is denoted by MAT^{λ} (the λ on the upper index indicates that $P \to \lambda$ is allowed). For more details on matrix grammars, we refer to [1], [12].

The language of DNA can be considered over $\Sigma_{DNA} = \{a, t, g, c\}$, where the complementary can be given as: $\bar{a} = t$, $\bar{t} = a$, $\bar{g} = c$ and $\bar{c} = g$. Similarly, the language of RNA can be considered over $\Sigma_{RNA} = \{a, u, g, c\}$, where the complementary can be given as: $\bar{a} = u$, $\bar{u} = a$, $\bar{g} = c$ and $\bar{c} = g$.

3 Matrix Insertion-Deletion Systems

In this section, we introduce our new grammar system *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 finite triples of the form in matrix format $[(u_1, \beta_1/\alpha_1, v_1), \ldots, (u_n, \beta_n/\alpha_n, v_n)]$, where $(u_k, v_k) \in V^*$, and $(\alpha_k, \beta_k) \in (V^+ \times \{\lambda\}) \cup (\{\lambda\} \times V^+)$, with $(u_k, \beta_k/\alpha_k, v_k) \in R_{I_i} \cup R_{D_j} \cup R_{I_i/D_j}$, for $1 \leq k \leq n$. 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 the matrix consisting of insertion or deletion or insertion and deletion rules 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/deleted)

1. $x = x_1u_1 \stackrel{\downarrow}{} v_1u_2v_2...u_nv_nx_2 \Longrightarrow x' = x_1u_1\underline{\alpha_1}v_1u_2 \stackrel{\downarrow}{} ...u_nv_nx_2 \Longrightarrow x'' = x_1u_1\alpha_1v_1u_2\underline{\alpha_2}v_2...u_n \stackrel{\downarrow}{} v_nx_2 \Longrightarrow^* y = x_1u_1\alpha_1v_1u_2\alpha_2v_2...u_n\underline{\alpha_n}v_nx_2$, for some $x_1, x_2 \in V^*$ and $[(u_1, \lambda/\alpha_1, v_1), (u_2, \lambda/\alpha_2, v_2), \ldots, (u_n, \lambda/\alpha_n, v_n)] \in R_{I_i}$.
2. $x = x_1u_1\underline{\beta_1}v_1u_2\beta_2v_2...u_n\beta_nv_nx_2 \Longrightarrow x' = x_1u_1 \stackrel{\Downarrow}{} v_1u_2\underline{\beta_2}v_2...u_n\beta_nv_nx_2 \Longrightarrow x'' = x_1u_1v_1u_2 \stackrel{\Downarrow}{} v_2...u_n\underline{\beta_n}v_nx_2 \Longrightarrow^* y = x_1u_1v_1u_2v_2...u_n \stackrel{\Downarrow}{} v_nx_2$ for some $x_1, x_2 \in V^*$ and $[(u_1, \beta_1/\lambda, v_1), (u_2, \beta_2/\lambda, v_2), ..., (u_n, \beta_n/\lambda, v_n)] \in R_{D_j}$.
3. $x = x_1u_1 \stackrel{\downarrow}{} v_1u_2\beta_2v_2...u_n\beta_nv_nx_2 \Longrightarrow x' = x_1u_1\underline{\alpha_1}v_1u_2\underline{\beta_2}v_2...u_n\beta_nv_nx_2 \Longrightarrow x'' = x_1u_1\alpha_1v_1u_2 \stackrel{\Downarrow}{} v_2...u_n\underline{\beta_n}v_nx_2 \Longrightarrow^* y = x_1u_1\alpha_1v_1u_2v_2...u_n \stackrel{\Downarrow}{} v_nx_2$, for some $x_1, x_2 \in V^*$ and $[(u_1, \lambda/\alpha_1, v_1), (u_2, \beta_2/\lambda, v_2), ..., (u_n, \beta_n/\lambda, v_n)] \in R_{I_i/D_j}$.

In a derivation step the rules in a matrix are applied sequentially one after 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

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

where $\Longrightarrow^*$ 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 $w \in T^*$ only. The family of languages generated by Matrix insertion-deletion systems is

given as $MATINS_n^m DEL_p^q$. The following example gives the clear understanding of the Matrix insertion-deletion systems. Consider the triple agreement language $L_1 = \{a^n b^n c^n \mid n \geq 1\}$. The language L_1 can be generated by the following Matrix insertion-deletion system Υ_1.

Example 1. $L_1 = \{a^n b^n c^n \mid n \geq 1\} \in \Upsilon_1$. The language L_1 can be generated by the Matrix insertion-deletion system $\Upsilon_1 = (\{a, b, c\}, \{a, b, c\}, \{abc\}, \{R_{I_1} = [(a, \lambda/ab, b), (b, \lambda/c, c)]\})$. A sample derivation can be given as follows:
$a^{\downarrow}bc \Longrightarrow_{R_{I_1}} a\underline{ab}b^{\downarrow}c \Longrightarrow_{R_{I_1}} aabb\underline{c}c \Longrightarrow^*_{R_{I_1}} a^n b^n c^n$. The triple agreement language has relevances to triple-stranded DNA [6]. If we include d in V, T, replace the axiom as $abcd$ and R_{I_1} as $[(a, \lambda/ab, b), (c, \lambda/cd, d)]$ in Υ_1 we can see that Υ_1 generates quadruple agreement language $\{a^n b^n c^n d^n \mid n \geq 1\}$. The quadruple agreement language has relevances to quadruple-stranded DNA [6].

4 Representing Bio-Molecular Structures

In this section, we show that the Matrix insertion-deletion systems can capture the important and essential biological structures discussed earlier in the paper. In most of the following derivations, at each derivation step, we directly write the resultant string obtained by applying all the rules in a matrix.

Lemma 1. *The pseudoknot structure language* $L_{ps} = \{uv\bar{u}^R\bar{v}^R \mid u, v \in \Sigma^*_{DNA}\}$ *can be generated by 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:
$R_{I_1} = [(\lambda, \lambda/b, \dagger_1), (\lambda, \lambda/\bar{b}, \dagger_3)]$, $R_{I_2} = [(\lambda, \lambda/b, \dagger_2), (\lambda, \lambda/\bar{b}, \dagger_4)]$
$R_{D_1} = [(\lambda, \dagger_1/\lambda, \lambda), (\lambda, \dagger_3/\lambda, \lambda)]$, $R_{D_2} = [(\lambda, \dagger_2/\lambda, \lambda), (\lambda, \dagger_4/\lambda, \lambda)]$
A sample derivation is given as follows:

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

The attenuator language in a molecular structure can be represented as $L_{an} = \{u\bar{u}^R u\bar{u}^R \mid u \in \Sigma^*_{DNA}\}$. The Fig.3. shows the attenuator structure.

Lemma 2. *The attenuator language* $L_{an} = \{u\bar{u}^R u\bar{u}^R \mid u \in \Sigma^*_{DNA}\}$ *can be generated by 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:
$R_{I_1} = [(\lambda, \lambda/a, \dagger_1), (\dagger_1, \lambda/t, \lambda), (\lambda, \lambda/a, \dagger_2), (\dagger_2, \lambda/t, \lambda)]$
$R_{I_2} = [(\lambda, \lambda/t, \dagger_1), (\dagger_1, \lambda/a, \lambda), (\lambda, \lambda/t, \dagger_2), (\dagger_2, \lambda/a, \lambda)]$
$R_{I_3} = [(\lambda, \lambda/c, \dagger_1), (\dagger_1, \lambda/g, \lambda), (\lambda, \lambda/c, \dagger_2), (\dagger_2, \lambda/g, \lambda)]$
$R_{I_4} = [(\lambda, \lambda/g, \dagger_1), (\dagger_1, \lambda/c, \lambda), (\lambda, \lambda/g, \dagger_2), (\dagger_2, \lambda/c, \lambda)]$
$R_{D_1} = [(\lambda, \dagger_1/\lambda, \lambda), (\lambda, \dagger_2/\lambda, \lambda)]$

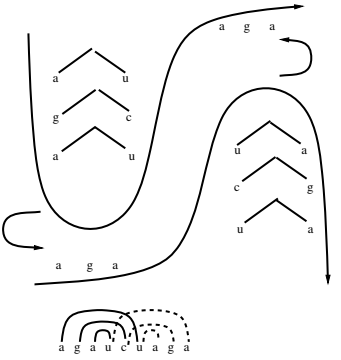

Fig. 3. Attenuator representation

A sample derivation is given as follows:

$^{\downarrow}\dagger_1^{\downarrow}\,{}^{\downarrow}\dagger_2^{\downarrow} \Longrightarrow_{R_{I_1}} \underline{a}^{\downarrow}\,\dagger_1^{\downarrow}\,\underline{t}\;\underline{a}^{\downarrow}\,\dagger_2^{\downarrow}\,\underline{t} \Longrightarrow_{R_{I_2}} a\underline{t}^{\downarrow}\,\dagger_1^{\downarrow}\,\underline{a}ta\underline{t}^{\downarrow}\,\dagger_2^{\downarrow}\,\underline{a}t \Longrightarrow_{R_{I_3}} at\underline{c}^{\downarrow}\,\dagger_1^{\downarrow}\,\underline{g}atat\underline{c}^{\downarrow}$
$\dagger_2^{\downarrow}\underline{g}at \Longrightarrow_{R_{I_4}} atc\underline{g}\,\dagger_1\,\underline{c}gatatc\underline{g}\,\dagger_2\,\underline{c}gat \Longrightarrow_{R_{D_1}} atcg^{\Downarrow}cgatatcg^{\Downarrow}cgat$ □

Lemma 3. *The non ideal-attenuator language* $L_{nian} = \{u\bar{u}^R u \mid u \in \Sigma^*_{DNA}\}$ *can be generated by Matrix insertion-deletion system.*

Proof. The language L_{nian} can be generated by the Matrix insertion-deletion system $\Upsilon_{nian} = (\{a,t,g,c,\dagger_1,\dagger_2\},\{a,t,g,c\},\{\lambda,\dagger_1\dagger_2\},R)$, where R is given as

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

A sample derivation is given as follows:

$^{\downarrow}\dagger_1^{\downarrow}\,{}^{\downarrow}\dagger_2 \Longrightarrow_{R_{I_1}} \underline{a}^{\downarrow}\,\dagger_1\,\underline{t}\;\underline{a}^{\downarrow}\dagger_2 \Longrightarrow_{R_{I_2}} a\underline{t}^{\downarrow}\,\dagger_1\,\underline{a}ta\underline{t}^{\downarrow}\dagger_2 \Longrightarrow_{R_{I_3}} at\underline{c}^{\downarrow}\,\dagger_1\,\underline{g}atat\underline{c}^{\downarrow}\dagger_2 \Longrightarrow_{R_{I_4}}$
$atc\underline{g}^{\downarrow}\,\dagger_1\,\underline{c}gatatc\underline{g}^{\downarrow}\dagger_2 \Longrightarrow_{R_{I_3}} atcg\underline{c}\,\dagger_1\,\underline{g}cgatatcg\underline{c}\dagger_2 \Longrightarrow_{R_{D_1}} atcgc^{\Downarrow}gcgatatcgc^{\Downarrow}$ □

Lemma 4. *The hairpin language* $L_{hp} = \{w = \bar{w}^R \mid w \in \Sigma^*_{DNA}\}$ *can be generated by Matrix insertion-deletion system.*

Proof. The hairpin language $L_{hp} = \{w = \bar{w}^R \mid w \in \Sigma^*_{DNA}\}$ can be generated by the Matrix insertion-deletion system $\Upsilon_{hp} = (\{b,\bar{b},\dagger\},\{b\bar{b}\},\{\lambda,b\,\dagger\,\bar{b}\},R)$, where $b \in \{a,t,g,c\}$, $\bar{b}$ is complement of b and R is given as follows:

$R_{I_1} = [(\lambda,\lambda/b,\dagger),(\dagger,\lambda/\bar{b},\lambda)], R_{D_1} = [(\lambda,\dagger/\lambda,\lambda)]$

A sample derivation is given as follows:

$a^{\downarrow}\,\dagger^{\downarrow}\,t \Longrightarrow_{R_{I_1}} a\underline{t}^{\downarrow}\,\dagger^{\downarrow}\,\underline{a}t \Longrightarrow_{R_{I_1}} atg\,\dagger\,\underline{c}at \Longrightarrow_{R_{D_1}} atgcg^{\Downarrow}cgcat$ □

Lemma 5. *The stem and loop language* $L_{sl} = \{uv\bar{u}^R \mid u, v \in \Sigma^*_{DNA}\}$ *can be generated by Matrix insertion-deletion system.*

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

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

A sample derivation is given follows:

$a^{\downarrow}\ \dagger_1\ \dagger_3\ \dagger_2^{\downarrow}\ t \Longrightarrow_{R_{I_1}} a\underline{c}\ \dagger_1^{\downarrow}\ \dagger_3\ \dagger_2\ \underline{gt} \Longrightarrow_{R_{I_2}} ac\ \dagger_1\ \underline{t}\ \dagger_3\ \dagger_2 gt \Longrightarrow_{R_{I_2}}$
$ac\ \dagger_1\ t\underline{c}\ \dagger_3\ \dagger_2 gt \Longrightarrow_{R_{D_1}} aca^{\Downarrow} tc\ \dagger_3\ gt \Longrightarrow_{R_{D_2}} acatc^{\Downarrow} gt$ □

The dumbbell language in a molecular structure can be represented as $L_{db} = \{u\bar{u}^R v\bar{v}^R \mid u, v \in \Sigma^*_{DNA}\}$. The Fig.4. shows the pictorial representation of dumbbell language.

Fig. 4. Dumbbell representation in bio-molecular structures

Lemma 6. *The dumbbell language* $L_{db} = \{u\bar{u}^R v\bar{v}^R \mid u, v \in \Sigma^*_{DNA}\}$ *can be generated by Matrix insertion-deletion system.*

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

$R_{I_1} = [(\lambda, \lambda/b, \dagger_1), (\dagger_1, \lambda/\bar{b}, \lambda)],\ R_{I_2} = [(\lambda, \lambda/b, \dagger_2), (\dagger_2, \lambda/\bar{b}, \lambda)]$
$R_{D_1} = [(\lambda, \dagger_1/\lambda, \lambda)],\ R_{D_2} = [(\lambda, \dagger_2/\lambda, \lambda)]$

A sample derivation is given follows:

$^{\downarrow}\ \dagger_1^{\downarrow}\ \dagger_2 \Longrightarrow_{R_{I_1}} \underline{a}\ \dagger_1\ \underline{t}^{\downarrow}\dagger_2^{\downarrow} \Longrightarrow_{R_{I_2}} a\ \dagger_1\ t\underline{g}^{\downarrow}\ \dagger_2^{\downarrow}\ \underline{c} \Longrightarrow_{R_{I_2}} a\ \dagger_1\ tg\underline{c}\ \dagger_2\ \underline{gc} \Longrightarrow_{R_{D_1}}$
$a^{\Downarrow} tgc\ \dagger_2\ gc \Longrightarrow_{R_{D_2}} atgc^{\Downarrow} gc$ □

Definition 1. *A string* w *over a complementary alphabet* Σ *is called ideal iff* $|w|_b = |w|_{\bar{b}}$ *for all* $b \in \Sigma$. *A language is ideal iff it contains only ideal strings.*

Lemma 7. *The ideal language* L_{id} *can be generated by Matrix insertion-deletion system.*

Proof. The ideal language L_{id} can be generated by the Matrix insertion-deletion system $\Upsilon_{id} = (\{b, \bar{b}\}, \{b, \bar{b}\}, \{\lambda\}, R)$, where $b \in \{a, t, g, c\}$, $\bar{b}$ is complement of b and R is given as $R_{I_1} = [(\lambda, \lambda/b, \lambda), (\lambda, \lambda/\bar{b}, \lambda)]$.
A sample derivation is given as follows:

$^{\downarrow}\lambda^{\downarrow} \Longrightarrow_{R_{I_1}} \underline{a}^{\downarrow}\underline{t}^{\downarrow} \Longrightarrow_{R_{I_1}} a\underline{c}^{\downarrow}t\underline{g}^{\downarrow} \Longrightarrow_{R_{I_1}} act\underline{c}^{\downarrow}tg\underline{a}^{\downarrow} \Longrightarrow_{R_{I_1}} actctga\underline{g}$ □

Definition 2. *A string w over a complementary alphabet Σ is called orthodox iff it is (i) the empty string ϵ, or (2) the result of inserting two adjacent complementary element $b\bar{b}$, for some $b \in \Sigma$, anywhere in an orthodox string. A language is orthodox iff it contains only orthodox strings.*

Lemma 8. *The orthodox language L_{od} can be generated by Matrix insertion-deletion system.*

Proof. The orthodox language L_{od} can be generated by the Matrix insertion-deletion system $\Upsilon_{od} = (\{b, \bar{b}\}, \{b, \bar{b}\}, \{\lambda\}, R)$, where $b \in \{a, t, g, c\}$, $\bar{b}$ is complement of b and R is given as $R_{I_1} = [(\lambda, \lambda/b\bar{b}, \lambda)]$. A sample derivation is given as $\lambda^{\downarrow} \Longrightarrow_{R_{I_1}} \underline{at}^{\downarrow} \Longrightarrow_{R_{I_1}} a^{\downarrow}t\underline{gc} \Longrightarrow_{R_{I_1}} a\underline{ta}tgc^{\downarrow} \Longrightarrow_{R_{I_1}} atatgc\underline{cg}$ □

The cloverleaf language in bio-molecular structures is represented as $L_{cl} = \{uv_1\bar{v}_1^R v_2\bar{v}_2^R, ..., v_n\bar{v}_n^R\bar{u}^R \mid u, v_1, v_2, v_3, ..., v_n \in \Sigma_{DNA}^*, n \geq 0\}$. The following Fig.5. represents the diagramatic representation of cloverleaf language for $n = 3$.

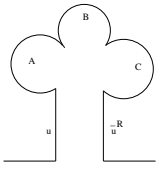

Fig. 5. Cloverleaf representation(where $A = v_1\bar{v}_1^R$, $B = v_2\bar{v}_2^R$, $C = v_3\bar{v}_3^R$)

Lemma 9. *The cloverleaf language $L_{cl} = \{uv_1\bar{v}_1^R v_2\bar{v}_2^R, ..., v_n\bar{v}_n^R\bar{u}^R \mid u, v_1, v_2, ..., v_n \in \Sigma_{DNA}^*, n \geq 0\}$ can be generated by Matrix insertion-deletion systems.*

Proof. The cloverleaf language L_{cl} (for $n = 3$) can be generated by the Matrix insertion-deletion system $\Upsilon_{cl} = (\{b, \bar{b}, \dagger_1, \dagger_2, \dagger_3, \dagger_4, \dagger_4, \dagger_5\}, \{b, \bar{b}\}, \{\lambda, b\,\dagger_1\,\dagger_2\bar{b}, \dagger_3\,\dagger_4\,\dagger_5, b\,\dagger_1\,\dagger_3\,\dagger_4\,\dagger_5\,\dagger_2\,\bar{b}\}, R)$, where $b \in \{a, t, g, c\}$, $\bar{b}$ is complement of b and R is
$R_{I_1} = [(\lambda, \lambda/b, \dagger_1), (\dagger_2, \lambda/\bar{b}, \lambda)]$, $R_{I_2} = [(\lambda, \lambda/b, \dagger_3), (\dagger_3, \lambda/\bar{b}, \lambda)]$
$R_{I_3} = [(\lambda, \lambda/b, \dagger_4), (\dagger_4, \lambda/\bar{b}, \lambda)]$, $R_{I_4} = [(\lambda, \lambda/b, \dagger_5), (\dagger_5, \lambda/\bar{b}, \lambda)]$
$R_{D_1} = [(\lambda, \dagger_1/\lambda, \lambda), (\lambda, \dagger_2/\lambda, \lambda)]$, $R_{D_2} = [(\lambda, \dagger_3/\lambda, \lambda)]$
$R_{D_3} = [(\lambda, \dagger_4/\lambda, \lambda)]$ $R_{D_4} = [(\lambda, \dagger_5/\lambda, \lambda)]$
A sample derivation is given as follows:

$a^{\downarrow}\,\dagger_1\,\dagger_3\,\dagger_4\,\dagger_5\,\dagger_2^{\downarrow}\,t \Longrightarrow_{R_{I_1}} a\underline{c}^{\downarrow}\,\dagger_1\,\dagger_3\,\dagger_4\,\dagger_5\,\dagger_2^{\downarrow}\,\underline{g}t \Longrightarrow_{R_{I_1}} ac\underline{g}\,\dagger_1^{\downarrow}\,\dagger_3^{\downarrow}\,\dagger_4\,\dagger_5\,\dagger_2\,\underline{c}gt$
$\Longrightarrow_{R_{I_2}} acg\,\dagger_1\,\underline{t}\,\dagger_3\,\underline{a}^{\downarrow}\,\dagger_4^{\downarrow}\,\dagger_5\,\dagger_2\,cgt \Longrightarrow_{R_{I_3}} acg\,\dagger_1\,\underline{t}\,\dagger_3\,\underline{a}\,\underline{c}\,\dagger_4\,\underline{g}^{\downarrow}\,\dagger_5^{\downarrow}\,\dagger_2 cgt \Longrightarrow_{R_{I_4}}$
$acg\,\dagger_1\,t\,\dagger_3\,ac\,\dagger_4\,g\underline{a}\,\dagger_5\,\underline{t}\,\dagger_2\,cgt \Longrightarrow_{R_{D_1}} acg^{\Downarrow}t\,\dagger_3\,ac\,\dagger_4\,ga\,\dagger_5\,t^{\Downarrow}cgt \Longrightarrow_{R_{D_2}}$
$acgt^{\Downarrow}ac\,\dagger_4\,ga\,\dagger_5\,tcgt \Longrightarrow_{R_{D_3}} acgtac^{\Downarrow}ga\,\dagger_5\,tcgt \Longrightarrow_{R_{D_4}} acgtacga^{\Downarrow}tcgt$ □

5 Ambiguity Issues in Gene Sequences

In this section, we study the application of various ambiguity levels of insertion-deletion systems introduced in [11]. Since the Matrix insertion-deletion systems

is an extension of insertion-deletion systems, the ambiguity levels defined for insertion-deletion system even holds for Matrix insertion-deletion systems also.

Level 0: The Matrix insertion-deletion systems is said to be 0-ambiguous if the same string can be derived from two different axioms. Consider the gene sequence *actgagct* in ideal language. This sequence can be generated by the Matrix insertion-deletion system Υ_{id} from two different axioms *at* and *ta* such that the same string is obtained at the end of the derivation. The two different derivations which differ by axioms are given as follows:

$$Derivation\ 1 : at^{\downarrow} \Longrightarrow a^{\downarrow}t^{\downarrow}\underline{gc} \Longrightarrow a\underline{c}t\underline{g}^{\downarrow}gc^{\downarrow} \Longrightarrow actg\underline{ag}c\underline{t}$$
$$Derivation\ 2 : ta^{\downarrow} \Longrightarrow^{\downarrow} ta^{\downarrow}\underline{gc} \Longrightarrow^{\downarrow} \underline{c}ta\underline{gg}c^{\downarrow} \Longrightarrow \underline{a}ctaggc\underline{t}$$

Level 1: The Matrix insertion-deletion systems is said to be 1-ambiguous if there are two different derivations for the same string which differs by the order of string inserted/deleted. Consider the gene sequence *aagtt* in stem and loop language. This sequence can be generated in two ways by the Matrix insertion-deletion system Υ_{sl}. Note that the axiom for both derivation is same. The two derivations are given as follows:

$Derivation\ 1 : a^{\downarrow} \dagger_1 \dagger_3 \dagger_2^{\downarrow} t \Longrightarrow_{R_{I_1}} a\underline{a} \dagger_1^{\downarrow} \dagger_3 \dagger_2 \underline{t}t \Longrightarrow_{R_{I_2}} aa \dagger_1 \underline{g} \dagger_3 \dagger_2 tt \Longrightarrow_{R_{D_1}}$
$aa^{\Downarrow}g \dagger_3^{\Downarrow} tt \Longrightarrow_{R_{D_2}} aag^{\Downarrow}tt$ (order of insertion is a and g)
$Derivation\ 2 : a \dagger_1^{\downarrow} \dagger_3 \dagger_2 t \Longrightarrow_{R_{I_2}} a^{\downarrow} \dagger_1 \underline{g} \dagger_3 \dagger_2^{\downarrow} t \Longrightarrow_{R_{I_1}} a\underline{a} \dagger_1 g \dagger_3 \dagger_2 \underline{t}t \Longrightarrow_{R_{D_1}}$
$aa^{\Downarrow}g \dagger_3^{\Downarrow} tt \Longrightarrow_{R_{D_2}} aag^{\Downarrow}tt$ (order of insertion is g and a)

Level 2: The Matrix insertion-deletion systems is said to be 2-ambiguous if there are two different derivations for the same string which differs by the order of contexts used for insertion/deletion. Consider the gene sequence *ctaatcgg* in cloverleaf language. This sequence can be generated in two ways by the Matrix insertion-deletion system Υ_{cl}. The two derivations are given as follows:

$Derivation\ 1 : c \dagger_1^{\downarrow} \dagger_3^{\downarrow} \dagger_4 \dagger_5 \dagger_2 g \Longrightarrow_{R_{I_2}} c \dagger_1 \underline{t} \dagger_3 \underline{a}^{\downarrow} \dagger_4^{\downarrow} \dagger_5 \dagger_2 g \Longrightarrow_{R_{I_3}} c \dagger_1 t \dagger_3 a\underline{a}$
$\dagger_4 \underline{t}^{\downarrow} \dagger_5^{\downarrow} \dagger_2 g \Longrightarrow_{R_{I_4}} c \dagger_1 t \dagger_3 aa \dagger_4 t\underline{c} \dagger_5 \underline{g} \dagger_2 g \Longrightarrow_{R_{D_1}} c^{\Downarrow}t \dagger_3 aa \dagger_4 tc \dagger_5 g^{\Downarrow}g$
$\Longrightarrow_{R_{D_3}} ct^{\Downarrow}aa \dagger_4 tc \dagger_5 gg \Longrightarrow_{R_{D_3}} ctaa^{\Downarrow}tc \dagger_5 gg \Longrightarrow_{R_{D_4}} ctaatc^{\Downarrow}gg$
$Derivation\ 2 : c \dagger_1 \dagger_3 \dagger_4^{\downarrow} \dagger_5^{\downarrow} \dagger_2 g \Longrightarrow_{R_{I_4}} c \dagger_1 \dagger_3^{\downarrow} \dagger_4^{\downarrow} \underline{c} \dagger_5 \underline{g} \dagger_2 g \Longrightarrow_{R_{I_3}} c \dagger_1^{\downarrow} \dagger_3^{\downarrow} \underline{a} \dagger_4 \underline{t}c$
$\dagger_5 g \dagger_2 g \Longrightarrow_{R_{I_2}} c \dagger_1 \underline{t} \dagger_3 \underline{a}a \dagger_4 tc \dagger_5 g \dagger_2 g \Longrightarrow_{R_{D_1}} c^{\Downarrow}t \dagger_3 aa \dagger_4 tc \dagger_5 g^{\Downarrow}g \Longrightarrow_{R_{D_2}}$
$ct^{\Downarrow}aa \dagger_4 tc \dagger_5 gg \Longrightarrow_{R_{D_3}} ctaa^{\Downarrow}tc \dagger_5 gg \Longrightarrow_{R_{D_4}} ctaatc^{\Downarrow}gg$

Note that the contexts chosen are of different order in each derivation. The Level 2 ambiguity can be pictorially represented as shown in Fig.6. Fig. 6(a) corresponds to derivation 1 and Fig.6(b) corresponds to derivation 2. This picture suggests a way of handling ambiguity issues in gene sequences and how they can be interpreted and what could be the intermediate sequences of genes in its sequence process.

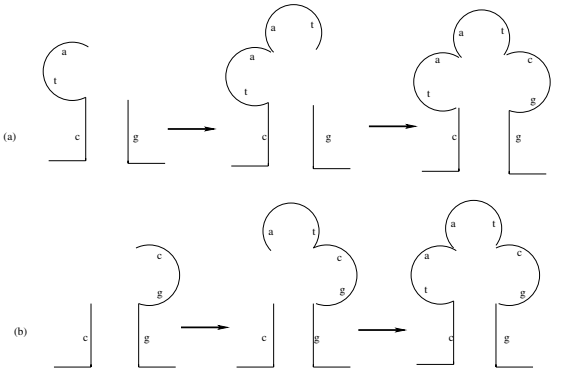

Fig. 6. Ambiguity in cloverleaf language

Level 3: The Matrix insertion-deletion systems is said to be 3-ambiguous if there are two different descriptions for the same string which differs by the position where the string is inserted/deleted. Consider the string *gctagcat* in orthodox language. This string can be derived in two different descriptions by Υ_{od} The two different descriptions are given as follows:

$$Description\ 1 :^{\downarrow} ta \Longrightarrow_{R_{I_1}} \underline{gcta}^{\downarrow} \Longrightarrow_{R_{I_1}} gctag\underline{c}^{\downarrow} \Longrightarrow_{R_{I_1}} gctagc\underline{at}$$

$$Description\ 2 : ta^{\downarrow} \Longrightarrow_{R_{I_1}} {}^{\downarrow}\underline{tagc} \Longrightarrow_{R_{I_1}} \underline{gc}tagc^{\downarrow} \Longrightarrow_{R_{I_1}} gctagc\underline{at}$$

Note that the axiom, order of insertion of strings, order of contexts (here (λ, λ)) all are same in both derivations, but the position of insertion is different in each derivation.

From the above results we can see that there may be more than one way that a gene sequence can be processed.

5.1 Universality of Matrix Insertion-Deletion Systems

Though our aim in this paper is not to analyze the introduced system with respect to Chomsky grammars, we provide the following trivial universality result in order to show that our new system is computationally complete.

Lemma 10. $RE \subseteq MATINS_1^1\ DEL_1^1$

Proof. In [13], it is proved that $RE \subseteq INS_1^1\ DEL_1^1$. Since each rule of insertion-deletion system can be considered as a set of matrices with each matrix consisting a single rule, the above result is true for Matrix insertion-deletion systems also. □

6 Conclusion

Insertion-deletion systems were defined and motivated by the way DNA strands are inserted and deleted. The bio-molecular structures like pseudoknot, attenuator, non-ideal attenuator are beyond the scope of context free grammars.

The bio-molecular structures like hairpin, stem and loop, ideal languages are within the power of context free grammars. We have defined a simple and powerful grammar system named Matrix insertion-deletion system. This is a unique grammar system which encompasses all the important biological structures that can be found in DNA, RNA, protein and other bio molecules. No other grammar system captures all the above discussed bio-molecular structures and therefore this new grammar system deserves a special attention. We have also shown the application of various ambiguity levels of insertion-deletion systems in gene sequences and how the ambiguity can be interpreted in gene sequences. As a future work, it is worth to analyze the closure properties and generative capacity of the introduced Matrix insertion-deletion systems.

References

1. Salomaa, A.: Formal languages. Academic Press, New York (1973)
2. Brendel, V., Busse, H.G.: Genome structure described by formal languages. Nucleic Acids Res., 2561–2568 (1984)
3. Calude, C.S., Paŭn, G.: Computing with cells and atoms, An intro. to Quantum, DNA and Membrane Computing. Taylor and Francis, London (2001)
4. Searls, D.B.: Representing genetic information with formal grammars. In: Proceedings of the National Conference on Artificial Intelligence, pp. 386–391 (1988)
5. Searls, D.B.: The linguistics of DNA. American Scientist, 579–591 (1992)
6. Searls, D.B.: The computational linguistics of biological sequences. In: Hunter, L. (ed.) Artificial Intelligence and Molecular Biology, pp. 47–120. AAAI Press, Menlo Park (1993)
7. Rivas, E., Reddy, S.R.: The language of RNa: A formal grammar that includes psuedoknots. Bioinformatics 16, 334–340 (2000)
8. Păun, G., Rozenberg, G., Salomaa, A.: DNA Computing, New Computing Paradigms. Springer, Heidelberg (1998)
9. Păun, G.: Membrane Computing-An introduction. Springer, Heidelberg (2002)
10. Hopcroft, J.E., Motwani, R., Ullman, J.D.: Introduction to Automata Theory, Languages and Computation. Addison-Wesley, Reading (2006)
11. Krithivasan, K., Kuppusamy, L., Mahendran, A., Khalid, M.: On the ambiguity and complexity measures in insertion-deletion systems. In: Proceedings of Bionetics 2010, USA, December 1-3. LNCS (2010)
12. Rozenberg, G., Salomaa, A.: Handbook of formal languages. Springer, Heidelberg (1997)
13. Verlan, S.: On minimal context-free insertion-deletion systems. Journal of Automata, Languages and Combinatorics 2, 317–328 (2007)
14. Setubal, J.C., Meidanis, J.: Introduction to Computational Molecular Biology. PWS Publishing Company (1997)
15. Uemura, Y., Hasegawa, A., Kobayashi, S., Yokomori, T.: Tree adjoining grammarsfor RNA structure prediction. Theoretical Computer Science 210, 277–303 (1999)