International Journal of Engineering & Technology, 7 (1) (2018) 19-24

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i1.8627
Research paper

Evolving state grammar for modeling DNA and RNA structures

Ajay Kumar *, Nidhi Kalra, Sunita Garhwal

CSED, Thapar University, Patiala, India,147004
**Corresponding author E-mail: ajayloura@gmail.com*

Abstract

In this paper, we represent bio-molecular structures (Attenuator, Extended Pseudoknot Structure, Kissing Hairpin, Simple H-type structure, Recursive Pseudoknot and Three-knot Structure) using state grammar. These representations will be measured using descriptional complexity point of views. Results indicate that the proposed approach is more succinct in terms of production rules and variables over the existing approaches. Another major advantage of the proposed approach is state grammar can be represented by deep pushdown automata, whereas no such automaton exists for matrix ins-del system.

Keywords: *Descriptional Complexity; Kissing Hairpin; Pseudoknot; Ribonucleic Acid; Simple H-Type; State Grammar; Three-Knot.*

1. Introduction

Natural computing combines the formal models and algorithmic techniques to solve the problem inspired by nature with the inclusion of natural materials (i.e. molecules). Deoxyribonucleic Acids (DNA) computing also comes under the umbrella of natural computing. Nowadays, a major concern of the bioinformatics is the analysis of DNA, Ribonucleic Acids (RNA) and protein sequences. DNA and RNA molecules form a complimentary pair which results in a pattern formation in the sequence. The grammatical formalism of these biological sequences is used in solving many bioinformatics problems such as multiple alignment calculations, classification, and prediction of primary and secondary structures. DNA and RNA are responsible for the development, growth and functioning of all known living organisms and viruses.

Chomsky grammar systems [1] are found to be ideal for representing the interactions of nucleotides as there is a similarity between formal languages and bio-molecules language. Some sequences like a hairpin (the language of palindrome) can be represented by a context-free grammar, whereas other sequences like attenuator $\{u\bar{u}^R u\bar{u}^R \mid u \in \Sigma^*_{DNA}\}$ cannot be represented by a context-free grammar. Similarly, other biological structures such as extended pseudoknot, recursive pseudoknot, simple-H type, kissing hairpin, Three-knot includes cross-dependency, and they require a higher class of formal language than context-free. In this paper, we will represent these bio-molecular structures using state grammar (a type of regulated grammar). The regulated grammar consists of production rules similar as context-free grammar, but certain restrictions are imposed on these grammars to represent the cross dependencies. State grammar is a rule-based regulated grammar in which restrictions are imposed in terms of states.

Prior Work: The concept of state grammar was introduced by Kasai [2]. A state grammar is a rule-based regulated grammar in which restrictions are imposed in terms of states. Various representations of DNA and RNA sequences using formal grammar and automata have been found in the literature. Sung [3] represented RNA secondary structure loops such as a hairpin loop, an internal loop, bulge loop and double helix using context-sensitive grammar. Sakakibara et al. [4] modeled RNA structure loops using stochastic context-free grammar. Further, Sakakibara [5] modeled RNA structure loops using pair hidden Markov models. Yuki and Kasami [6] modeled RNA structure loops using stochastic multiple context-free grammars. Brown and Wilson [7] modeled RNA pseudoknot structure using the intersection of stochastic context-free grammars.

Rivas and Eddy [8] used cross-interaction grammar for representing RNA secondary structure. Searls [9] used indexed grammar to represent DNA and RNA sequences such as tandem repeat, inverted repeat and pseudoknot. Searls [10] also represented DNA sequences using string variable grammar. Mizoguchi et al. [11] used stochastic multiple context-free grammars to represent various classes of pseudoknots. Cai et al. [1] represented RNA pseudoknot structure using parallel communicating grammar systems. Kuppusamy et al. [12] represented DNA and RNA secondary structures using matrix insertion-deletion system. Kalra and Kumar [3] represented tandem repeat, inverted repeat and pseudoknot using state grammar.

In this paper, we analyze the representation of bio-molecular structures with the basic descriptional complexity in terms of a number of production rules and number of variables. Results are compared with the matrix insertion-deletion system for the similar representations. After introducing some preliminary concepts in Section 2, we represent attenuator, extended pseudoknot, H-type, three-knot structure, recursive pseudoknot structure and kissing hairpin using state grammar. Section 4 consists of results and discussion.

2. Preliminaries

In this section, some basic notations and definition are discussed. $\Sigma_D = \{g, c, a, t\}$ and $\Sigma_R = \{g, c, a, u\}$ denote DNA and RNA alphabet respectively. Σ^*_D denotes the free monoid generated by Σ_D. λ denotes empty string or null string. In DNA and RNA, pairing occurs between complement pair in purines and pyrimidine. The complement of a symbol d is denoted by $\bar{d}$. Purines are classified into adenine (a) and guanine (g), while pyrimidine is classified into cytosine (c) thymine (t) and uracil (u).

In DNA:

$\bar{a}=t$

$\bar{t}=a$

$\bar{g}=c$

$\bar{c}=g$

In DNA:

$\bar{a}=u$

$\bar{u}=a$

$\bar{g}=c$

$\bar{c}=g$

Watson-Crick pairing occurs in RNA. DNA and RNA are important macromolecules that exist in every form of life. They are made from monomers known as nucleotides. Each nucleotide consists of a pentose carbon sugar, a phosphate group, and a nitrogenous base. If the sugar is ribose, then the polymer is RNA. If the sugar is deoxyribose, then the polymer is DNA.

Def. 2.1: Regulated grammar [14-16] is quintuple (N,Σ,S,P,RG) where N is a set of non-terminals, Σ is an alphabet, S is the start symbol, P is the set of production rules, and RG is the restriction applied on the derivations of strings, and it depends on the type of regulated grammar.

Regulated grammar is classified into rule-based and context-based grammatical regulation [14].

Def. 2.2 [13]: A state grammar is a quintuple $G(V, Q, \Sigma, P, S)$, where V is a finite set of symbols, Q is a finite set of states such that $V \cap Q=\phi$, $\Sigma \subseteq V$ is an alphabet of terminals, $P \subseteq (Q \times (V-\Sigma) \times (Q \times V^{+}))$ is a finite relation over the productions, and $S \in V-\Sigma$ is the start symbol.

Example 1: Consider the state grammar $G_S=(\{S,A,B,0,1,2\},\{p_0,p_1,p_2\},\{0,1,2\},P,S)$ where the production rules are

$$(p_0,S)\rightarrow(p_0,AB) \quad (p_0,A)\rightarrow(p_1,0A1) \quad (p_1,B)\rightarrow(p_0,2B)$$
$$(p_0,A)\rightarrow(p_2,01) \quad (p_2,B)\rightarrow(p_2,2)$$

Consider the string $s=000111222$

$$\boldsymbol{S}_{p0}\rightarrow\boldsymbol{A}B_{p0}\rightarrow 0A1\boldsymbol{B}_{p1}\rightarrow 0\boldsymbol{A}12B_{p0}\rightarrow 00A112\boldsymbol{B}_{p1}\rightarrow$$
$$00\boldsymbol{A}1122B_{p0}\rightarrow 00011122\boldsymbol{B}_{p2}\rightarrow 000111222_{p2}$$

The non-context-free language generated by G_s is $L(G_S)=\{0^n1^n2^n \mid n\geq 1\}$.

Def. 2.3 [17]: Matrix insertion-deletion system $I(V,\Sigma,\mathrm{A},R)$ where V is a finite set of symbols, $\Sigma\subseteq V$ is an alphabet of terminals, A is a finite language over V, R is a finite set of triple in a matrix format $[(u_1,\alpha_1 \mid \beta_1,v_1)\dots(u_n,\alpha_n \mid \beta_n,v_n)]$, $(u_i,v_i)\in V^{*}\times V^{*}$ and $(\alpha_i,\beta_i)\in (V^{+}\times\{\lambda\})\cup(\{\lambda\}\times V^{+})$.

Def. 2.4 [17]: Given a matrix insertion-deletion system $I(V,\Sigma,\mathrm{A},R)$. Descriptional complexity measure of I in terms of variables and production is defined by

$$prod(I)=\left(\sum_{m\in R} m+|R|\right).|\mathrm{A}|,$$

$$\mathrm{var}(I)=|V|-|T|$$

Here $|R|$ denote a total number of rules in a matrix insertion-deletion system.

Def. 2.5: Descriptional complexity of a state grammar $G(V, Q, \Sigma, P, S)$ is defined by

$$prod(G)=|P|+|Q|$$

$$\mathrm{var}(I)=|V|-|T|$$

The descriptional complexity of example 1 is

$$prod(G)=|P|+|Q|=5+3=8$$

And

$$\mathrm{var}(I)=|V|-|T| \;=|\{S,A,B,0,1,2\}|-|\{0,1,2\}|=5-3=2.$$

Recently, various researchers had researched different direction of automata theory, especially focusing on deep pushdown automata and state grammar (See [13, 18-22] for more details)

3. State grammar for bio-molecular structures

This section describes the state grammar for bio-molecular sequences found in DNA and RNA.

Proposition 1. The attenuator language $L_A=\{u\bar{u}^R u\bar{u}^R \mid u\in\Sigma_{DNA}^{*}\}$ can be generated by state grammar. Fig 1. represents attenuator structure of the sequence $gtcgacagcgct$.

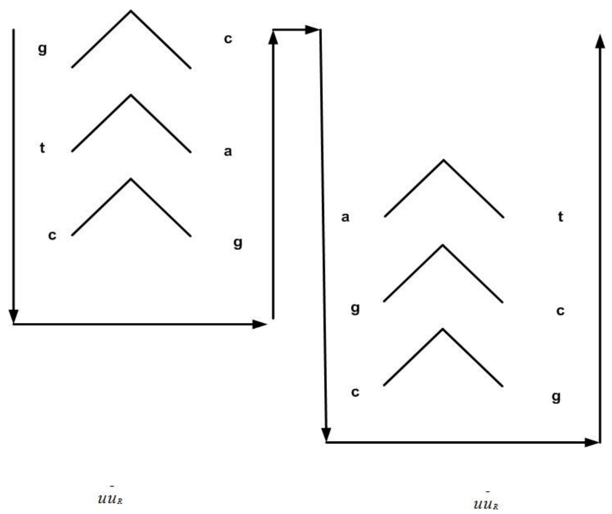

Fig. 1: Attenuator Structure.

Proof: Grammar $G_1=\{(S,A,B,u,\bar{u}),(q_0,q_1,q_2),(u,\bar{u}),P,S\}$, where $u\in\{a,g,c,u\}$ and $\bar{u}$ is the complement of u. State grammar productions are defined as follows:

$$(q_0,S)\rightarrow(q_0,AB)$$

$$(q_0,A)\rightarrow(q_1,uA\bar{u})$$

$$(q_1,B)\rightarrow(q_0,uB\bar{u})$$

$$(q_0,A)\rightarrow(q_2,\lambda)$$

$(q_2, B) \to (q_2, \lambda)$

Derivation for input string $w = gtatacgtatac$

$$\mathbf{S}_{q0} \to \mathbf{A}B_{q0} \to gAc\mathbf{B}_{q1} \to g\mathbf{A}cgBc_{q0} \to gtAacg\mathbf{B}c_{q1} \to gt\mathbf{A}acgtBac_{q0}$$
$$\to gtaAtacgt\mathbf{B}ac_{q1} \to gta\mathbf{A}tacgtaBtac_{q0} \to gtatacgta\mathbf{B}tac_{q2}$$
$$\to gtatacgtatac_{q2}$$

Proposition 2. The extended pseudoknot language $L_{ExP} = \{u_1 v_1 \bar{u}_1^R u_2 \bar{v}_1^R \bar{u}_2^R \mid u_1, u_2, v_1 \in \Sigma^*_{RNA}\}$ can be generated by state grammar. Fig. 2 represents the structure of extended pseudoknot sequence $cugcuacagcguuagacg$.

Proof: Grammar $G_2 = \{(S, A, B, u_1, u_2, v_1, \bar{u}_1, \bar{u}_2, \bar{v}_1), (q_0, q_1, q_2, q_3), (u_1, u_2, v_1, \bar{u}_1, \bar{u}_2, \bar{v}_1), P, S\}$ be the state grammar where production rules P are as follows:

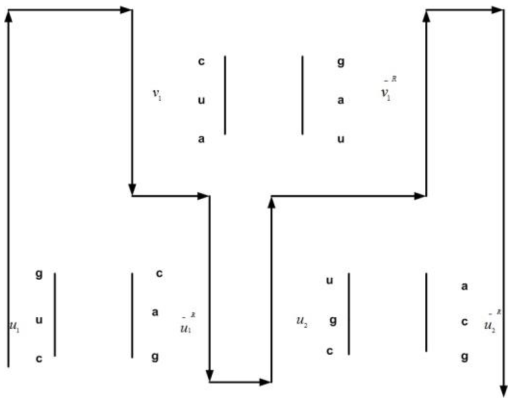

Fig. 2: Extended Pseudo knot Structure.

$(q_0, S) \to (q_0, AB)$

$(q_0, A) \to (q_0, u_1 A \bar{u}_1)$

$(q_0, B) \to (q_0, u_2 B \bar{u}_2)$

$(q_0, A) \to (q_1, v_1 A)$

$(q_1, B) \to (q_2, B\bar{v}_1)$

$(q_2, A) \to (q_1, v_1 A)$

$(q_2, A) \to (q_3, \lambda)$

$(q_3, B) \to (q_3, \lambda)$

Derivation for input string $w = cugcuacagcguuagacg$

$$\mathbf{S}_{q0} \to \mathbf{A}B_{q0} \to c\mathbf{A}gB_{q0} \to cu\mathbf{A}agB_{q0} \to cugAcag\mathbf{B}_{q0} \to$$
$$cugAcagc\mathbf{B}g_{q0} \to cugAcagcg\mathbf{B}cg_{q0} \to cug\mathbf{A}cagcguBacg_{q0}$$
$$\to cugcAcagcgu\mathbf{B}acg_{q1} \to cugc\mathbf{A}cagcguBgacg_{q2} \to$$
$$cugcuAcagcgu\mathbf{B}gacg_{q1} \to cugcu\mathbf{A}cagcguBagacg_{q2} \to$$
$$cugcuaAcagcgu\mathbf{B}agacg_{q1} \to cugcua\mathbf{A}cagcguBuagacg_{q2} \to$$
$$cugcuacagcgu\mathbf{B}uagacg_{q3} \to cugcuacagcguuagacg_{q3}$$

Bold non-terminal indicates the non-terminal to be expanded next.

Proposition 3. The simple H-type language $L_{SH} = \{u_1 v_1 u_2 \bar{u}_1^R v_2 \bar{u}_2^R \mid u_1, u_2, v_1, v_2 \in \Sigma^*_{RNA}\}$ can be generated by state grammar. Fig. 3 represents the structure of the simple H-type sequence.

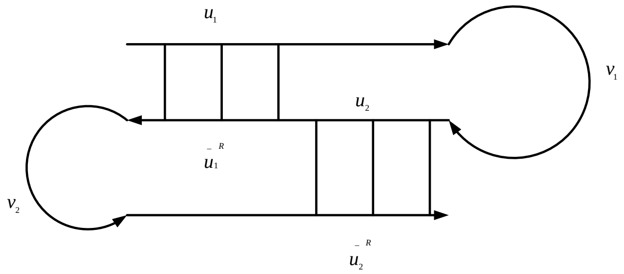

Fig. 3: Simple H-Type Structure.

Proof: Grammar $G_3 = (\{S, A, B, u_1, u_2, v_1, v_2, \bar{u}_1, \bar{u}_2\}, \{q_0, q_1, q_2, q_3, q_4\}, \{u_1, u_2, v_1, v_2, \bar{u}_1, \bar{u}_2\}, P, S)$ be the state grammar where production rules P are as follows:

$(q_0, S) \to (q_0, AB)$

$(q_0, A) \to (q_0, u_1 A \bar{u}_1)$

$(q_0, A) \to (q_1, v_1 A)$

$(q_1, A) \to (q_1, v_1 A)$

$(q_1, B) \to (q_1, v_2 B)$

$(q_1, A) \to (q_2, u_2 A)$

$(q_2, B) \to (q_3, B\bar{u}_2)$

$(q_3, A) \to (q_2, u_2 A)$

$(q_3, A) \to (q_4, \lambda)$

$(q_3, B) \to (q_4, \lambda)$

Derivation for input string $w = cugucugcagacag$

$$\mathbf{S}_{q0} \to \mathbf{A}B_{q0} \to c\mathbf{A}gB_{q0} \to cu\mathbf{A}agB_{q0} \to cug\mathbf{A}cagB_{q0} \to cuguAcag\mathbf{B}_{q1}$$
$$\to cugu\mathbf{A}cagaB_{q1} \to cugucAcaga\mathbf{B}_{q2} \to cuguc\mathbf{A}cagaBg_{q3} \to$$
$$cugucuAcaga\mathbf{B}g_{q2} \to cugucu\mathbf{A}cagaBag_{q3} \to cugucugAcaga\mathbf{B}ag_{q2} \to$$
$$cugucug\mathbf{A}cagaBcag_{q3} \to cugucugcaga\mathbf{B}cag_{q4} \to cugucugcagacag_{q4}$$

Proposition 4. The simple three-knot language $L_{TK} = \{u_1 v u_2 u_3 \bar{u}_1^R \bar{u}_2^R \bar{u}_3^R \mid u_1, u_2, u_3, v \in \Sigma^*_{RNA}\}$ can be generated by state grammar. Fig 4. represents the structure of the three-knot sequence.

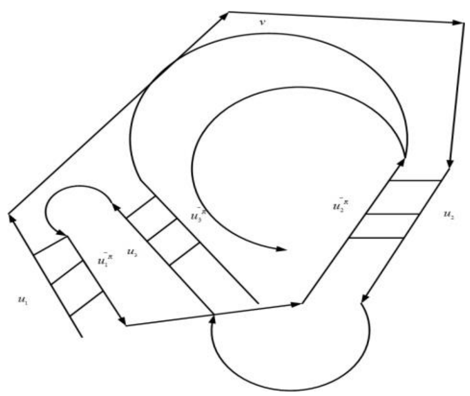

Fig. 4: Three-Knot Structure.

Proof: Grammar $G_4=(\{S,A,B,C,u_1,u_2,u_3,v,\bar{u}_1,\bar{u}_2,\bar{u}_3\},\{q_0,q_1,q_2,q_3,q_4,q_5\},\{u_1,u_2,u_3,v,\bar{u}_1,\bar{u}_2,\bar{u}_3\},P,S)$ be the state grammar where production rules P are as follows:

$$(q_0,S)\to(q_0,ABC)$$

$$(q_0,A)\to(q_0,u_1A\bar{u}_1)$$

$$(q_0,A)\to(q_1,vA)$$

$$(q_1,A)\to(q_1,vA)$$

$$(q_1,B)\to(q_2,u_3B)$$

$$(q_2,C)\to(q_1,C\bar{u}_3)$$

$$(q_1,A)\to(q_3,u_2A)$$

$$(q_3,B)\to(q_3,\lambda)$$

$$(q_3,C)\to(q_4,C\bar{u}_2)$$

$$(q_4,A)\to(q_3,u_2A)$$

$$(q_5,A)\to(q_5,\lambda)$$

$$(q_5,C)\to(q_5,\lambda)$$

Derivation for input string $w=cugcugacacagucug$

$$\mathbf{S}_{q0}\to\mathbf{A}BC_{q0}\to c\mathbf{A}gBC_{q0}\to cu\mathbf{A}agBC_{q0}\to cug\mathbf{A}cagBC_{q0}\to cugc\mathbf{A}cagBC_{q1}\to cugcuAcag\mathbf{B}C_{q1}\to cugcuAcagcB\mathbf{C}_{q2}\to cugcuAcagc\mathbf{B}Cg_{q1}\to cugcuAcagcaB\mathbf{C}g_{q2}\to cugcu\mathbf{A}cagcaBCug_{q1}\to cugcugAcagcaBCug_{q3}\to cugcugAcagca\mathbf{C}ug_{q3}\to cugcug\mathbf{A}cagcaCcug_{q4}\to cugcugaAcagca\mathbf{C}cug_{q3}\to cugcuga\mathbf{A}cagcaCucug_{q4}\to cugcugacagca\mathbf{C}ucug_{q5}\to cugcugacagcaucug_{q5}$$

Proposition 5. The recursive pseudoknot language $L_{RC}=\{u_1u_2u_3\bar{u}_2^Ru_4\bar{u}_1^R\bar{u}_4^Ru_5\bar{u}_5^R\bar{u}_3^R\mid u_1,u_2,u_3,u_4,u_5\in\Sigma_{RNA}^*\}$ can be generated by state grammar. Fig 5. represents the structure of recursive pseudoknot sequence.

Proof: Grammar $G_5=(\{S,A,B,C,u_1,u_2,u_3,u_4,u_5,\bar{u}_1,\bar{u}_2,\bar{u}_3,\bar{u}_4,\bar{u}_5\},\{q_0,q_1,q_2,q_3,q_4,q_5,q_6,q_7,q_8\},\{u_1,u_2,u_3,u_4,u_5,\bar{u}_1,\bar{u}_2,\bar{u}_3,\bar{u}_4,\bar{u}_5\},P,S)$ be the state grammar where production rules P are as follows:

$$(q_0,S)\to(q_0,ABC)$$

$$(q_0,B)\to(q_0,u_4B\bar{u}_4)$$

$$(q_0,A)\to(q_1,u_1A)$$

$$(q_1,B)\to(q_2,B\bar{u}_1)$$

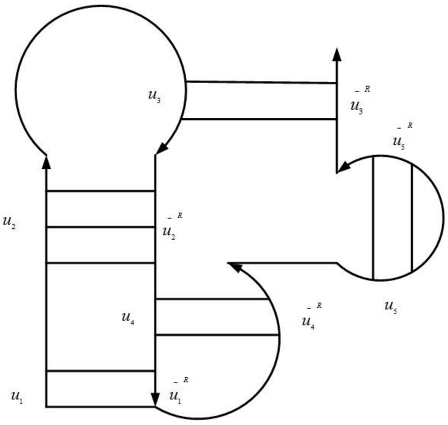

Fig. 5: Recursive Pseudoknot Structure.

$$(q_2,A)\to(q_1,u_1A)$$

$$(q_2,A)\to(q_3,u_2A\bar{u}_2)$$

$$(q_3,A)\to(q_3,u_2A\bar{u}_2)$$

$$(q_3,B)\to(q_3,\lambda)$$

$$(q_3,C)\to(q_3,u_5C\bar{u}_5)$$

$$(q_3,A)\to(q_4,u_3A)$$

$$(q_4,C)\to(q_5,Cu_5)$$

$$(q_5,A)\to(q_4,u_3A)$$

$$(q_5,A)\to(q_6,\lambda)$$

$$(q_6,C)\to(q_6,\lambda)$$

Derivation for input string $w=cagcucugagucuaag$

$$\mathbf{S}_{q0}\to A\mathbf{B}C_{q0}\to Ag\mathbf{B}aC_{q0}\to AgaBuaC_{q0}\to \mathbf{A}gaBuaC_{q0}\to cAga\mathbf{B}uaC_{q1}\to c\mathbf{A}gaBguaC_{q2}\to ca\mathbf{A}ugaBguaC_{q3}\to cagAcuga\mathbf{B}guaC_{q3}\to cagAcugagua\mathbf{C}_{q3}\to cag\mathbf{A}cugaguauCa_{q3}\to cagAcugaguau\mathbf{C}a_{q4}\to cagc\mathbf{A}cugaguauCga_{q5}\to cagcuAcugaguau\mathbf{C}ga_{q4}\to cagcu\mathbf{A}cugaguauCaga_{q5}\to cagcucugaguau\mathbf{C}aga_{q6}\to cagcucugaguauaga_{q6}$$

Proposition 6. The kissing hairpin language $L_{KH}=\{u_1v_1A\bar{A}v_2u_2\bar{u}_2^Rv_3B\bar{B}v_4\bar{u}_1^R\mid u_1,u_2,v_1,\ v_2,v_3,v_4\in\Sigma_{RNA}^*,A,B\in\Sigma_{RNA}\}$ can be generated by state grammar. Fig 6. represents the structure of kissing hairpin sequence.

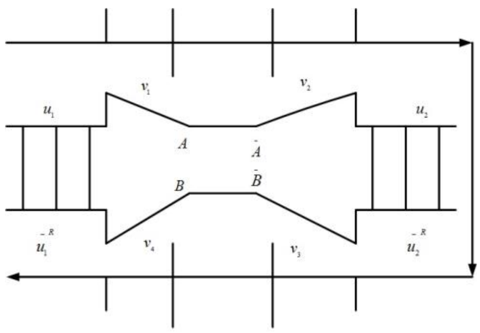

Fig. 6: Kissing Hairpin Structure.

Proof: Grammar $G_6 = \{\{S, A, B, C, D, u_1, u_2, v_1, v_2, v_3, v_4\}, \{q_0, q_1, q_2, q_3, q_4, q_5\}, \{u_1, u_2, v_1, v_2, v_3, v_4\}, P, S\}$ be the state grammar where production rules P are as follows:

$(q_0, S) \to (q_0, u_1 S \bar{u}_1)$

$(q_0, S) \to (q_1, v_1 C)$

$(q_1, C) \to (q_1, v_1 C)$

$(q_1, C) \to (q_1, C v_4)$

$(q_1, C) \to (q_2, CD)$

$(q_2, C) \to (q_1, A C \bar{A})$

$(q_2, D) \to (q_2, B D \bar{B})$

$(q_2, C) \to (q_3, v_2 C)$

$(q_3, C) \to (q_3, v_2 C)$

$(q_3, C) \to (q_3, C v_3)$

$(q_3, C) \to (q_4, u_2 C \bar{u}_2)$

$(q_4, C) \to (q_4, u_2 C \bar{u}_2)$

$(q_4, D) \to (q_5, \lambda)$

$(q_4, C) \to (q_5, \lambda)$

Derivation for input string $w = gaugaucuguaccagcgcuc$

$\boldsymbol{S}_{q0} \to g\boldsymbol{S}c_{q0} \to ga\boldsymbol{S}uc_{q0} \to gau\boldsymbol{C}uc_{q1} \to gauu\boldsymbol{C}uc_{q1} \to gauu\boldsymbol{C}auc_{q1} \to gauu\boldsymbol{C}Dauc_{q2} \to gauucCg\boldsymbol{D}auc_{q2} \to gauucCgc\boldsymbol{D}gauc_{q2} \to gauuc\boldsymbol{C}gcuDagauc_{q2} \to gauucu\boldsymbol{C}gcuDagauc_{q3} \to gauucua\boldsymbol{C}gcuDagauc_{q3} \to gauucua\boldsymbol{C}agcuDagauc_{q3} \to gauucua\boldsymbol{C}aagcuDagauc_{q3} \to gauucua\boldsymbol{C}aagcuDagauc_{q3} \to gauucua\boldsymbol{C}uaagcuDagauc_{q4} \to gauucuag\boldsymbol{C}caagcuDagauc_{q4} \to gauucuaga\boldsymbol{C}ccaagcuDagauc_{q4} \to gauucuagaccaagcu\boldsymbol{D}agauc_{q5} \to gauucuagaccaagcuagauc_{q5}$

4. Results and discussion

In this section, we compared the above-designed state grammar with the matrix insertion-deletion system based on descriptional complexity. Table 1, depicts the comparison between the matrix insertion-deletion system and state grammar in terms of production and variable used for attenuator, extended pseudoknot structure, kissing hairpin, simple H-type structure, recursive pseudoknot and three-knot structure. Clearly, state grammar is more succinct in terms of the number of productions and variables except in attenuator (one more variable used than the matrix insertion-deletion system) based on the type-1 descriptional complexity. Design of state grammar will be helpful in the prediction of secondary structure. Design of the parser for the above-designed state grammar is kept as a future work.

Table 1: Summary of Results

System	Matrix Insertion-deletion system		State grammar	
Language	Prod	Var	Prod	Var
Attenuator Language	18+5=23	2	5+3=8	3
Extended Pseudoknot Language	10+7=17	4	8+4=12	3
Simple H-type	9+7=16	3	10+5=15	3
Three knot structure	11+8=19	4	12+6=18	4
Recursive Pseudoknot	15+10=25	5	14+7=21	4
Kissing hairpin	12+10=22	4	14+6=20	3

Acknowledgements

Nidhi Kalra was supported under Visvesvaraya PhD Scheme fellowship by Department of Electronics and Information Technology, Ministry of Communication & IT, Government of India. We would like to Prof. Meduna, Prof. Kuppusamy as their research work help us in understanding state grammar and Bio-molecular structures.

References

[1] L. Cai, L.R. Malmberg and Y. Wu, “Stochastic modeling of RNA pseudoknotted structures: a grammatical approach”, Bioinformatics, Vol. 19, No. 1, i66-i73, (2003). https://doi.org/10.1093/bioinformatics/btg1007.

[2] T. Kasai, “An hierarchy between context-free and context-sensitive languages, Journal of Computer and System Sciences”, Vol. 4, No. 5, (1970), pp: 492-508. https://doi.org/10.1016/S0022-0000(70)80045-9.

[3] K.Y. Sung, “The Use of Context-Sensitive Grammar For Modeling RNA Pseudoknots”, In- Proceedings of the International Conference on Bioinformatics & Computational Biology (BIOCOMP’06); Las Vegas, Nevada, USA, 338-344, 26-29 June (2006).

[4] Y. Sakakibara, M. Brown, R. Hughey, I.S. Mian, K. Sjölander, R.C. Underwood, D. Haussler, “Stochastic context-free grammars for tRNA modelling”, Nucleic Acids Research, Vol. 22, No. 23, (1996), pp: 5112-5120. https://doi.org/10.1093/nar/22.23.5112.

[5] Y. Sakakibara, “Pair hidden Markov models on tree structures”, *Bioinformatics*, Vol. 19, No. 1, (2003), pp: 232-240. https://doi.org/10.1093/bioinformatics/btg1032.

[6] S. Yuki and T. Kasami, “RNA pseudoknotted structure prediction using stochastic multiple context-free grammar”, *IPSJ Transactions on Bioinformatics*, Vol. 47, (2006), pp: 12-21.

[7] M. Brown and C. Wilson, “RNA pseudoknot modeling using intersections of stochastic context free grammars with applications to database search”, Proceedings of the Pacific Symposium on Biocomputing, Big Island, HI, USA, (1995), pp: 109-125.

[8] E. Rivas and S. R. Eddy, “The language of RNA: a formal grammar that includes pseudoknots”, Bioinformatics, Vol. 16, No. 4, (2000), pp: 334-340. https://doi.org/10.1093/bioinformatics/16.4.334.

[9] D. B. Searls, “The computational linguistics of biological sequences, Artificial intelligence and molecular biology”, American Association for Artificial Intelligence Menlo Park, CA, USA, Vol. 2, (1993), pp: 47-120.

[10] D. B. Searls, “String variable grammar: A logic grammar formalism for the biological language of DNA”, The Journal of Logic Programming, Vol. 24, No. 1, (1995), pp: 73-102. https://doi.org/10.1016/0743-1066(95)00034-H.

[11] N. Mizoguchi, Y. Kato, and H. Seki, “A grammar-based approach to RNA pseudoknotted structure prediction for aligned sequences”, In - Proceedings of 1st IEEE International Conference on Computational Advances in Bio and Medical Sciences (ICCABS), Orlando, FL, (2011), pp: 135-140. https://doi.org/10.1109/ICCABS.2011.5729868.

[12] L. Kuppusamy and A. Mahendran, “Modelling DNA and RNA secondary structures using matrix insertion-deletion systems”, International Journal of Applied Mathematics and Computer Science, Vol. 26, No. 1, (2016), pp: 245-258. https://doi.org/10.1515/amcs-2016-0017.

[13] N. Kalra and A. Kumar, “State Grammar and Deep Pushdown Automata for Biological Sequences of Nucleic Acids”, Current Bioinformatics, Vol. 11, No. 4, (2016), pp: 470-479. https://doi.org/10.2174/1574893611666151231185112.

[14] A. Meduna and P. Zemek, "Regulated Grammars and Automata", Springer, New York, 2014. https://doi.org/10.1007/978-1-4939-0369-6.

[15] J. Dassow and G. Pawn, "Regulated rewriting in formal language theory", 1st ed Springer-Verlag Berlin Heidelberg, (2012).

[16] J. Dassow, "Grammars with regulated rewriting", In: Formal Languages and Applications: Studies in Fuzziness and Soft Computing Ed. Springer Berlin Heidelberg, 148, (2004), pp: 249-273. https://doi.org/10.1007/978-3-540-39886-8_13.

[17] L. Kuppusamy, I. Raman and K. Krithivasan, "On Succinct Description of Certain Context-Free Languages by Ins-Del and Matrix Ins-Del Systems", International Journal of Foundations of Computer Science, Vol. 27, No.7, (2016), pp: 775-786. https://doi.org/10.1142/S0129054116500295.

[18] N. Kalra and A. Kumar, "Fuzzy state grammar and fuzzy deep pushdown automaton", Journal of Intelligent and Fuzzy Systems, Vol. 31, No.1, (2016), pp: 249-258. https://doi.org/10.3233/IFS-162138.

[19] K. Singh, "Conversion of deterministic finite automata to regular expression using bridge state", M.E. Thesis, Thapar University, (2011).

[20] N. Kalra and A. Kumar, "Deterministic Deep Pushdown Transducer and its Parallel Version", The Computer Journal, doi.org/10.1093/comjnl/bxx036. https://doi.org/10.1093/comjnl/bxx036.

[21] T. Singla and A. Kumar, "Reducing Mutation Testing Endeavor using the similar conditions for the same Mutation Operators occurs at Different Locations", Applied mathematics & Information Science, Vol. 8, No. 5, (2014), pp: 2389-2393. https://doi.org/10.12785/amis/080534.

[22] R. Kumar and A. Kumar, "Metamorphosis of fuzzy regular expressions to fuzzy automata using the follow automata", arXiv preprint arXiv: 1411.2865.