IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 19, Issue 2, Ver. I (Mar.-Apr. 2017), PP 15-25
www.iosrjournals.org

A New Approach to DNA, RNA, and Protein Motifs Templates Visualization and Analysis via Compilation Technique

Hussein K. Al-Khafaji[1], Zainab M.Jameel[2]
[1]*Al-Rafidain University College*
[2]*University of Information Technology and Communication*

Abstract *:Motifs template is the input for many bioinformatics systems such codons finding, transcription, transaction, sequential pattern miner, and bioinformatics databases analysis. The size of motifs arranged from one base up to several Mega bases, therefore, the typing errors increase according to the size of motifs. In addition, when the structures motifs are submitted to bioinformatics systems, the specifications of motifs components are required, i.e. the simple motifs, gaps, and the lower bound and upper bound of each gap. The motifs can be of DNA, RNA, or Protein. In this research, a motif parser and visualization module is designed depending on a proposed a context free grammar, CFG, and colors human recognition system. GFC describes the motif structure to parse the motifs, detect, debug the errors, and analyze the motifs template to its components. Many experiments are accomplished using motifs templates of various sizes arranged from 10 Kbase to 10 Mbase, various numbers of gaps arranged from 15 gaps to 15000 gaps, and different numbers of errors arranged from 100 errors to 1820 errors. The proposed systems, in all these experiments, exhibited linear behavior in parsing phase and visualization phase that indicates its scalability to motifs template sizes.*

I. Introduction

This section contains a review for some terms and principles related to bioinformatics, which are necessary to preceed in this research.

1.1 Bioinformatics

It is difficult to give exactly the source of bioinformatics, both as a term and as propriety. The expression was employed as early as 1977 by Dutch theoretical biologist PaulienHogeweg when she depicted her key of research as bioinformatics [1].PaulienHogeweginnovated the expression bioinformatics in 1979 for the study of informatics operations in bioticsystems. It was mainly used since late 1980s in genetics and genomics especially in those fields ofgenomics cover a wide area of DNA series. One can define Bioinformatics as the use of computer technology for the control of information in biology. Bioinformatics is concerned with storing, extracting, organizing, analyzing, interpreting and usinginformation from biological sequences and molecules. It has been primarily motivated by progress in DNA sequencingand mapping methods. Recently the rapid progress in genomic and other molecular researchmethods and developments in information technologies have combined to generate a great amount ofinformation on molecular biology [2]. The main goal of bioinformatics is to enhance the understanding of biological activities. The differentiating factor that distinguishes bioinformatics from other techniques, however, is its concentration on developing and using computationally intensive methods to carry out this goal. Examples cover pattern recognition, data mining, machine learning algorithms, and visualization. Major research attempts in the area cover sequence alignment, gene discovery, genome assembly, drug design, drug discovery, protein structure alignment, protein structure prediction, prediction of gene expression and protein–protein interactions, genome-wide association studies and the evolution modeling [3].

1.2 Deoxyribonucleic Acid (DNA)

A double helix molecule is composed of a linear nucleotides structure (See Figure 1). DNA carries the genetic code for an organism in the order of the bases. The double helix of DNA that comes from the hydrogen bonds is formative between bases when two polynucleotide chains are identical, but employment in other directions. The information in DNA is stored as a code consist of chemical bases contain of about three billion bases. The sequence of these bases determines the information available for constructing and maintaining an organism. DNA bases contained four nucleotides (A) adenine, (T) thymine , (C) cytosine , (G) guanine that joined with each other, (A) with (T) and (C) with (G), to make base pairs. Each base is also attached to a sugar molecule and a phosphate molecule. Nucleotide contains a base, sugar, and phosphate. Nucleotides are placed in two long strands that make a double helix. An essential Characteristics of DNA is that it can replicate itself. In the double helix of DNA each strand can be used as a pattern for duplicating the sequence of bases. This is very important because when cells divided each new cell need a delicate copy of DNA that found in the old cell [4].

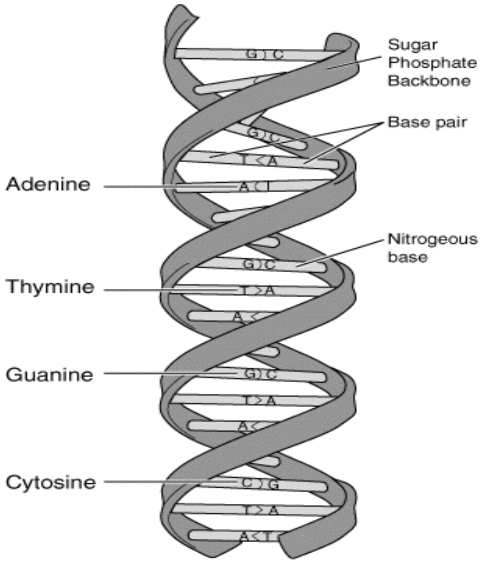

Figure (1) DNA is a double helix formed by base pairs [4]

1.3Ribonucleic Acid (RNA)

Is a sequence of nucleotidessimilar to DNA, RNA nucleotides containing ribose rings in state of deoxyribose .RNA bases contained four nucleotides (A) adenine, (U) *uracil* , (C) cytosine , (G) guanine that joined with each other, (A) with (U) and (C) with (G), to make base pairs [4]. Figure (2) show the difference between RNA and DNA.The major function of RNA is to carry the genetic code which is important in establishing of proteins from the nucleus to the ribosome. RNA contains multiple types the main types are:

1) (mRNA) messenger RNA: which is mounted from a DNA gene. Messenger *ribonucleic acid* contains information of the main sequence of amino acids in protein to be mounted. Messenger RNA responsible to move the code into cytoplasm which protein mounted.
2) (tRNA) transfer RNA :which is contain twenty different transfer ribonucleic acid ,one for each of amino acid. The transfer RNA reads the messenger RNA "codon" by utilization its own "anticodon". The actual reading is done during the match pairs through hydrogen bonds.Every "codon" is reading by multiple Transfer RNA until suitable matching of the "anticodon" with the "codon" happen.
3) (r RNA)ribosomal RNA: through the cytoplasm protein and ribosomal RNA joined to each other from a nucleoprotein dub a ribosome .the main job of ribosomal RNA that move the enzymes for to create protein[5].

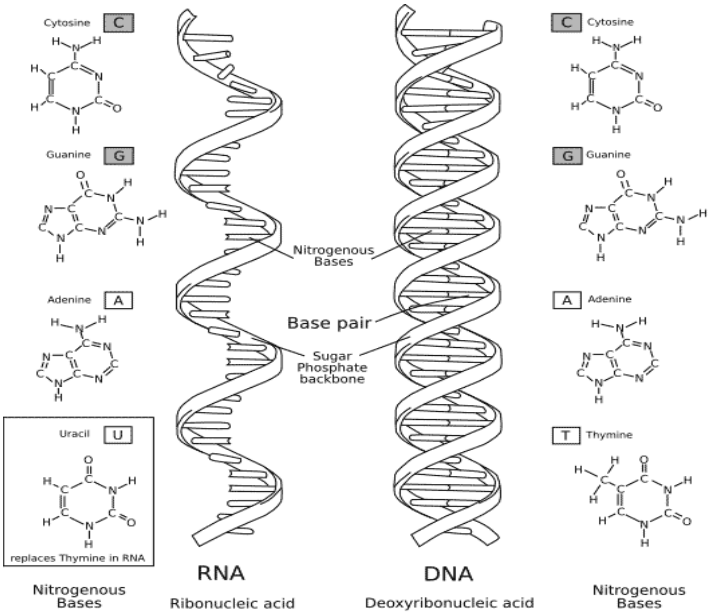

Figure (2) depicted differences between DNA & RNA [4]

1.4 Protein

A Protein is composed of small unit called amino acids [6]. There are more twenty type of amino acid. Proteins are containing different type of amino acids. Proteins are large molecules contain multiple hundred of amino acids in order of sequence. The main job of protein is build and repairs a body [7]. Proteins contain twenty bases are (A) adenine, (T) thymine, (C) cytosine, (G) guanine, (I) isoleucine, (D) aspartic acid, (E) glutamic acid, (F) phenylalanine, (K) lysine, (R) arginine, (H) histidine, (N) asparagines, (Q) glutamine, (S) serine, (L) leucine, (V) valine, (W) tryptophan, (Y) tyrosine, (M) methionine and (P) proline that order in sequence[8].

1.5 Motif &Motif Template

Motif is used in two ways in the structure. First way indicates sequence of amino acid that is referring to properties of a special biochemical task. Second way, motif represents as a collection of contiguous structure elements that either select a part of task significance or select a part of folded domain independently [9].

Motifs composed of two types simple or structure motif; a simple motif also named single pattern. Simple motif does not contain gaps, that separate simple motif because easy to matching relatively short bases, and the second type is structure motif also named compound pattern, structure motif contains gaps for this reason named structure motif, that approach of it is more difficult than simple motif [3].

DNA motif is a serial of a nucleic acid pattern.DNA motif contacts with structural motifs in proteins. Motifs might happen in each of the strands of DNA [10]. DNA motif contains four bases A, T, C and G. the shape of DNA motif pattern is ATACCGATTTCGCGCGCGTT and sometimes contain gaps such as AATATA [3,5]CTG[2,4] AATGCCG.

RNA motif is defined as a sequence of levels that occurs a short sequences in functional RNAs, for examples (tRNA) transfer RNA or (rRNA) ribosomal RNA [11]. RNA motif contains four bases A, U, C and G. the final form of RNA motifs is AUGUUCGAGGGCAUU and sometimes RNA motifs contain gaps in the form such as AUU[5,8]CCCG[7,9]AAAUGGC.

A protein motif is asequence of amino acid found in proteins. Change the motif according to biological function [12]. Protein motifs contain twenty bases in form that A, T, C, G, I, E, D, F, M, Y, Q, N, H, S, L, V, W, R, K and P. the final form of protein motif is RQWPPAATTTVDE and sometimes contain gaps such as RQWPP[3,6]TAQWVF.

1.6 Motif Template

Definition of Motif template contains a sequence of bases and gaps. These contain lower number and upper number. Simple motif consists of bases for any type of motif that used. The general structure of motif represented by the form:

M_1 {[l_1, u_1] M_2 {[l_2, u_2] M_3} {….. M_n [l_n,u_n]} M_{n+1}}……. (1)

So, for example consider the following protein motif:

KMKKVVVVVQ [8, 10] AVCCEW [2, 3] CW

It is a structure motif template which consists of three simple motifs in protein alphabets, M1 [8, 10] M2 [2, 3] M3 such that:

- M1 is KMKKVVVVVQ with length of 10 bases.
- M2 is AVCCEW with length of 6 bases.
- M3 is CW with length of 2 bases.
- [8, 10] is the gap between two simple motifs as lower bound of 8 bases and upper bound of 10 bases.
- [2, 3] is the gap between two simple motifs as lower bound of 2 bases and upper bound of 3 bases.

The lower bound must be less than or equal upper bound and integer number.

According to the previous introduction, the reader can conclude that a motif is a sequence of alphabets contains bases and gaps. The motif contains millions of bases in accordance with its uses. This paper produces a new parser to detect and correct the errors in the motif templates before further processing such as sequential pattern mining in bioinformatics system, cordon finding, etc. In addition, the paper introduces a design and implementation of a motifs visualization facility. The next subsections describe the design and implementation of this motif template parser and visualization facility in details.

II. The Proposed System of Motifs Template Parsing And Visualization

The following subsection describes the design of the motif template parser and visualization system.

2.1 The proposed system of Motifs Template parsing

The general structure of the motif parser is show in figure (3). Motif source selector enables user to input the motif template using the keyboard, (for small size motif), or to load a motif from data set.

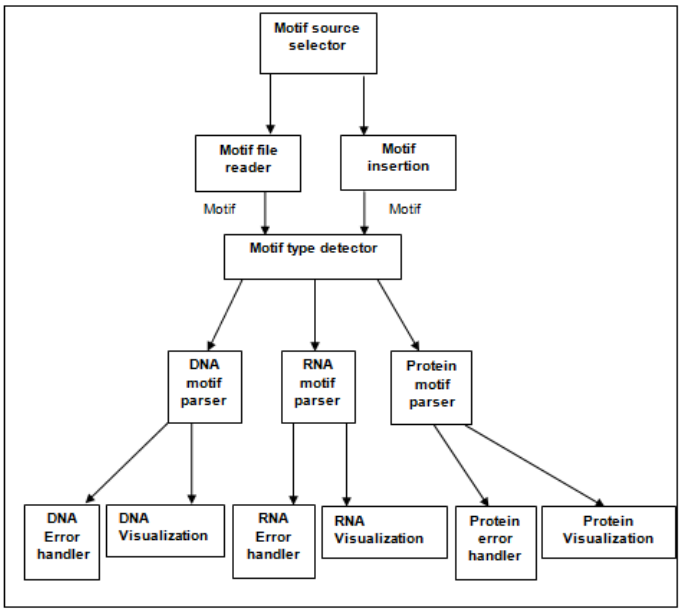

Figure (3) architecture of motif parsing and visualization system

Motif files reader loads and segments huge files according to memory size. Motif type detector is responsible for determining the type of motif template. According to the type of the Motif, suitable parser will treat it. If the Motif template passed the parsing process, specialized visualization module will visualize it.

As Motif has fixed structure and it as well as its components cannot be changed it must always be correct and without errors (both oral and grammatical) because it is used to give accurate results.

In this research, a parser is proposed which is similar to a standard parser to detect the syntax and semantic errors in addition to intended and unintended noise. As shown in Figure (4), It consists of the following phases:-

1) Lexical analyzer
2) Syntax analyzer
3) Semantic analyzer
4) Motifs and gaps generator

Introducing Motif Pattern into the program either directly or through a file already stored in the computer will correct the error by passing it through several stages in the compiler. The result will appear in the form of a report showing whether there is an error or not. The Motif contains bases and gaps and it should be error-free to be ready for mining process. Three types of Motif were processed in this research; they are DNA, RNA and Protein.

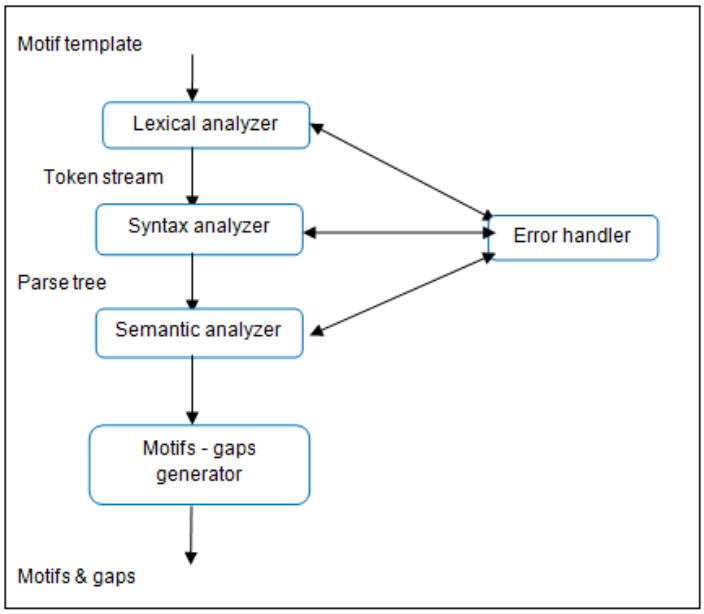

Figure (4) phases of the Motif template parser [3]

1) Lexical analyzer

Lexical analysis is the first phase of a parser. It takes the source motif (motif template) from one of the types of resources; keyboard or file. The lexical analyzer decomposes the structure of the Motif template into a number of tokens, by eliminating any whitespace or comments in the source motif. If a token is found invalid by the lexical analyzer, the latter produces an error. The lexical analyzer operates closely with the syntax analyzer. Motif streams are read from the Motif template, legal tokens are checked and the data are passed to the syntax analyzer when it demands. The alphabets of DNA, RNA, and protein are shown in table (1).

Table (1) Alphabets of DNA, RNA, and protein

Number	{0,1,2,3,4,5,6,7,8,9} digit for gap.
Strings	DNA={A,T,C,G} RNA={A,U,C,G} Protein={I,T,P,G,K,R,C,N,Q,S,L,V,F,W,Y,H,M,A,E,D}
Special Symbols	Comma(,) , Parentheses ([,])

To elucidate the duty of the lexical analyzer, consider the following Motifs, which represent DNA, RNA, and Protein Motifs respectively.

1. ATCT[1,10]CC.
2. AUGC[3,34]GC.
3. ADEFH[23,67]DE.

The result of lexical analyzer phase related to the Motif templates are shown in table (2), table (3), and table (4) respectively.

Table (2) DNA tokens

DNA token values	Description
ATCT	Simple DNA motif
[	Left side bracket gap
1	Low integer number
,	Comma
10	High integer number
]	right side bracket gap
CC	Simple DNA motif

Table (3) RNA tokens

RNA token values	Description
AUGC	Simple RNA motif
[	Left side bracket gap
3	Low integer number
,	Comma
34	High integer number
]	right side bracket gap
GC	Simple RNA motif

Table (4) protein tokens

Protein token values	description
ADEFH	Simple protein motif
[	Left side bracket gap
23	Low integer number
,	Comma
67	High integer number
]	right side bracket gap
DE	Simple protein motif

Figure (5), figure (6), and figure (7) depict the skeleton of the lexical analyzer. Indeed, they are written in self-documentation style, therefore, there is no extended explanation for the steps of the algorithms.

```
Lexical_Analyzer Algorithm
Input: motif_template;
Output: Motif_value, Motive_type;

Step 1:  {   Call Simple_motif_recognition (motif_template, motif_value, motif_type);
Step 2:      If (motif_type<>"simple_motif")
Step 3:              Call Gap_recognition (motif_template,  motif_value, motif_type);
Step 4:      Else     exit;
Step 5:      If (motif_template) not in ("LB","RB","Comma","Number")
Step 6:                          Call Error_handler_and_visualizer(Motif_template);
Step 7:  }
……..
```

Figure (5) skeleton of lexical analyzer algorithm

```
Simple_motif_recognition algorithm

Input: motif_ template, motif_template_type;
Output: simple_motif

Step 100:  {    base=left_trim (motif_template, motif_template);
                  token_string="";   simple_motif=" ";
Step 101:       case moif_template _type of
Step 102        {
Step 103:      "DNA": {
Step 104:                  while (base in ['A', 'T', 'C', 'G']) do
Step 105:                       {   token_string+=base;
Step 106:                           base=left_trim (motif_template, motif_template); }
Step 107:              }
Step 108:    "RNA": {
Step 109:                  while (base in ['A', 'U', 'C', 'G']) do
Step 110:                       {   token_string+=base;
Step 111:                           base=left_trim (motif_template, motif_template); }
Step 112:              }
Step 113:    "Protein": {
Step 114:                  while (base in ['A', 'T', 'C', 'G', 'I', 'D', 'E', 'F', 'K', 'R', 'H',
                                          'N', 'Q', 'S', 'L', 'V', 'W', 'Y', 'M', 'P']) do
Step 115:                       {   token_string+=base;
Step 116:                           base=left_trim (motif_template, motif_template);}
Step 117:              }
Step 118:                   }//end case
Step 119:         concat (base, motif_template, motif_template);
Step 120:         simple_motif=motif_string; motif_type="simple_motif";
Step 121:      } // of simple motif recognition.
.............
```

Figure (6) simple motif recognition algorithm

```
Gap_recognition algorithm

Input: motif_ template;
Output: token_string; token_value;

Step 200:   {   num="";
Step 201:      base=left_trim (motif_template, motif_template);
Step 202:       case base of   {
Step 203:              '[':{ token_string='['; token_value = "LB";}
Step 204:              '] ':{ token_string=']'; token_value = "RB";}
Step 205:              ', ':{ token_string=','; token_value = "comma";}
Step 206:              '0'|'1'|'2'|'3'|'4'|'5'|'6'|'7'|'8'|'9':
Step 207:                   {
Step 208:                    while(base in['0', '1', '2', '3', '4', '5', '6', '7', '8', '9'])do
Step 209:                   {
Step 210:                     concat(num, base, num);
Step 211:                     left_trim (motif_template, motif_template);
Step 212:                   }
Step 213:                        string_to_number (num, num2);
Step 214:            token_string=num; token_value="number";
Step 215:                 }// end digit
Step 216:            }//case
Step 217:  Error_handler("illegal character");
Step 218:  Call Motif_editor and_visiualizer( Motif_template);
Step 219:            Exit;
.............
```

Figure (7) Gap recognition algorithm

2) Syntax Analyzer

A syntax analyzer, also known as parser, receives the input from a lexical analyzer in the form of token flow. The source motif, (token stream), is analyzed by the parser against the production rules to discover any mistake in the motif template.

The Parser is supposed to parse the entire Motif even though some errors are found in the Motif template. The Parser uses strategies for error recovering called context free grammar (CFG), which is illustrated in figure (8). A parse tree is the output of this phase. The leaves of this tree are the tokens of lexical analyzer phase, and these leaves read from left to right. The parse tree of motif is illustrated in figure (9). In this way the parser performs two functions, i.e., parsing the Motif template, searching for mistakes and produces a parse tree as the output of the phase.

Fragment of Context Free Grammar of the Motif Parser

Rule 1: MT ⟶ MT_DNA
Rule 2: MT ⟶ MT_RNA
Rule 3: MT ⟶ MT_PROTEIN
Rule 4: MT_DNA ⟶ DNA_ Base
Rule 5: MT_DNA ⟶ DNA_ Base MT_DNA
Rule 6: MT_DNA ⟶ DNA_ Base GAP MT_DNA
Rule 7: DNA_ Base ⟶ a|c|g|t
Rule 8: MT_RNA ⟶ RNA_ Base
Rule 9: MT_RNA ⟶ RNA_ Base MT_RNA
Rule10: MT_RNA ⟶ RNA_ Base GAP MT_RNA
Rule11: RNA_ Base ⟶ a|c|g|u
Rule12: MT_PROTEIN ⟶ PROTEIN_ Base
Rule13: MT_PROTEIN ⟶ PROTEIN_ Base MT_PROTEIN
Rule14: MT_PROTEIN ⟶ PROTEIN_ Base GAP MT_PROTEIN
Rule15: PROTEIN_ Base ⟶ a|c|g|t|i|d|e|f|k|r|h|n|q|s|l|v|w|y|m|p
Rule16: GAP ⟶ [Number, Number]
Rule17: Number ⟶ Digit Number
Rule18: Number ⟶ Digit
Rule19: Digit ⟶ 0|1|2|3|4|5|6|7|8|9
.....

Figure (8) context free grammar rules of motif parser

Accordingly, if the protein motif template, ID [1,3]N[2,4]M is correct ,then the validation process will be as follows:

MT → MT_PROTEIN……..……...Rule (3)
MT → PROTEIN_BASE MT_PROTEIN……Rule (13)
MT → i MT_PROTEIN…….....Rule (15)
MT →i PROTEIN_BASE GAP MT PROTEIN.……Rule (14)
MT→ i d GAP MT_PROTEIN……...…..……...Rule (15)
MT→ i d [Number, Number] MT_PROTEIN……..Rule (16)
MT → i d [Digit, Number] MT_PROTEIN……....Rule (18)
MT →i d [1, Number] MT_PROTEIN……...….Rule (19)
MT→ i d [1, Digit] MT_PROTEIN…….……...Rule (18)
MT→i d [1, 3] MT_PROTEIN……...Rule (19)
MT→ i d [1, 3] PROTEIN_BASE GAP MT_PROTEIN ……...Rule (14)
MT →i d [1, 3] n GAP MT_PROTEIN…….……...…….Rule (15)
MT→ i d [1, 3] n [Number, Number] MT_PROTEIN……...Rule (16)
MT → i d [1, 3] n [Digit, Number] MT_PROTEIN…….……..Rule (18)
MT →i d [1, 3] n [2, Number] MT_PROTEIN…….....Rule (19)
MT→i d [1, 3] n [2, Digit] MT_PROTEIN……...Rule (18)
MT→ i d [1, 3] n [2, 4] MT_PROTEIN……Rule (19)
MT → i d [1, 3] n [2, 4] PROTEIN_BASE……..……Rule (12)
MT→ i d [1, 3] n [2, 4] m……...…….Rule (15)

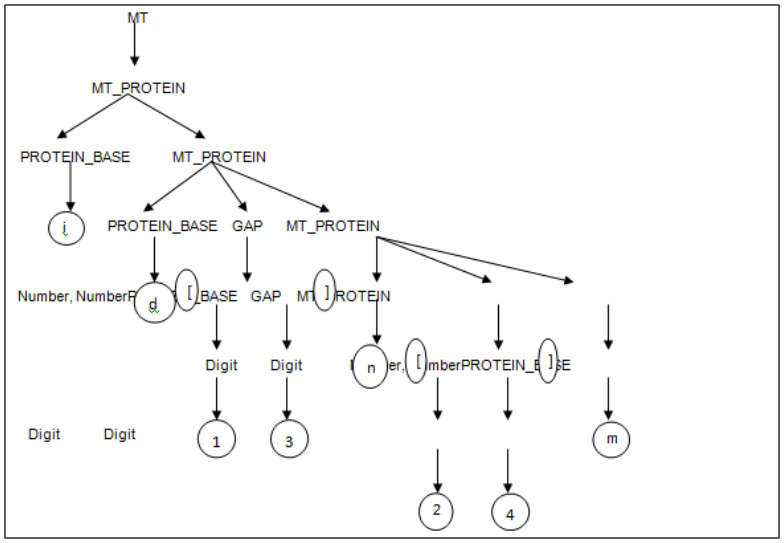

Figure (9) protein motif template parse tree

3) Semantic Analyzers

Meaning is provided to Motif constructs by its semantics, like tokens and syntax structure. Semantics help interpret symbols, their kinds, and their relationships with each other. Semantic analysis decides whether the syntactical structure made in the source Motif derives any meaning or not .it should not issue an error in lexical and syndical analysis phase, as it is lexically and structurally correct, but it should produce a semantic mistake as the kind of the assignment varies. The grammar of the Motif sets these rules which are evaluated in semantic analysis. The following is some of semantic errors that can be manipulated by the semantic analyzer, figure (10) shows the semantic phase deals with motif in specific rules:

1) The right number in gap in the motif must be valued smaller than left hand side value.

EX: DNA motif = AT [5,7]CCGA

RNA motif = UUA [3,6] UACA.

2) Low and high number in gap in the motif must be integer.

EX: Protein motif = KQVV [2,9]RRTC

EX: DNA motif = ATC [2,9]CCGA

3) The gap interval must not exceed the size of data set to be mined.

```
Semantic analyzer algorithm

Input:  Parsing tree
Output:  semantic tree

Step 1:  From parse tree construct GapList;
Step 2:  While (more Gap in GapList) DO
Step 3:  {
                Gap=next Gap in GapList;
Step 6:         If (Gap.LowBound>Gap.UpperBound)
Step 7:              Error_handler_and_visualizer ("Lower bound should be
                              Less Upper Bound of the Gap");
Step 8:
Step 9:         Elseif (Gap.LowerBound and Gap.UpperBound are not integer values)
Step 10:             Error_handler_and_visualizer ("Lower bound  and UpperBound of the should be Integer");
.....

        };
    };
```

Figure (10) fragment of semantic analyzer algorithm

4) Motifs & Gaps generator

Motifs and Gaps generator can be considered as the final phase of the proposed parser. The output of these phase considered in two tables, the first one named simple motifs table contain bases of simple motif, real position of motif contain (start real position, end real position and length of simple motif), and position of motif contain (start position, end position, and length of motif). The second table consists of gaps, lower bound in gap, upper bound in gap, start position of gap and end position of gap; you can see these tables with protein examples in figure (11).

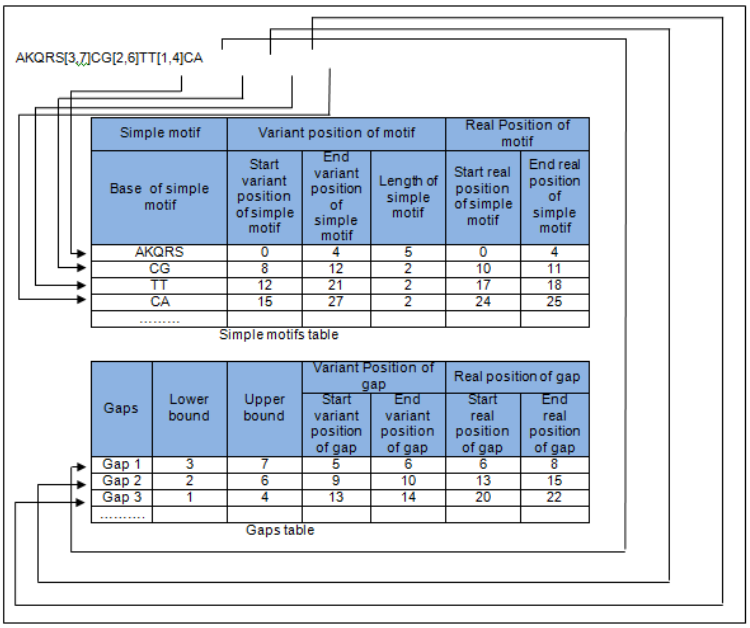

Figure (11) output of Motif and Gaps generator phase

2.2 The proposed system of Motifs Template visualization

A colorized system for the motif can be made by giving a color to every element of the Motif whether it is a symbol, number or characters. Figure (12) illustrates the colors used for this purpose. Tables (5) shows the colors specified for the alphabet of DNA, RNA, protein, and other symbols (legal and illegal). The black color is assigned for illegal symbols.

Colors will activate human colors recognition system and as a result, the time of error detection and correction degenerated in addition to minimization of error percentage.

The visualization module helps out in:

1- Activation of user concentration on errors.
2- Reducing the time consumed in finding the error.
3- Improving the way to reach an error through its pointer.
4- Accuracy in correction of errors.
5- Improving the overall performance of the system.

Table (5) DNA & RNA & protein & other symbols (legal and illegal) colors

	Characters & symbols	Description	Color
1	A	Adenine	Green
2	T	Thymine	Red
3	C	Cytosine	Blue
4	G	Guanine	Orange
5	U	Uracil	Deep pink
6	I	Isoleucine	Tan
7	D	Aspartic Acid	Dark Goldenrod
8	E	Glutamic Acid	Gold
9	F	Phenylalanine	Olive
10	K	Lysine	Dark violet
11	R	Arginine	Maroon
12	H	Histidine	Light Coral
13	N	Asparagine	Rosy Brown
14	Q	Glutamine	Lime green
15	S	Serine	Royal blue
16	L	Leucine	Mid night blue
17	V	Valine	Dodger blue
18	W	Tryptophan	Light sea green
19	Y	Tyrosine	Tomato
20	M	Methionine	Crimson
21	P	Proline	Dark sea green
22	] and [	right side bracket gaps & left side bracket gaps	Saddle brown
23	,	Comma	Lamp
24	0|1|2|3|4|5|6|7|8|9	Digit	Deep sky blue
25	illegal symbols	Any symbols is not defined within the alphabet CFG of motif	Black

```
Colorized algorithm
Input: motif_ template
Output: (colorized_char)
Step1: {
Step2: char_position =0; color=" ";
Step3: while (not end of motif) do
Step4:     {
Step5:   if (motif_template [char_position] is legal char)
Step6:   color=color_table (motif_template [char_position])
Step 7:  set_color (motif_template [char_position], color);
Step8:     }
Step9:     else  {   set_color (motif_template [char_position], black);
Step10:              char_position+=1  }
Step11:    }
.....
```

Figure (12**)** colorizing algorithm

III. Experment Results

This section presents the results of the experiment that focus on the efficiency of the proposed parser to detect and debug the error included in a motif template .The experiments used various motif template sized up to 10000000 bases. The time of processes depends on the size of motif.The proposed parser that proved efficiency to load motif, colorization motif, and handle the errors to huge number of motifs within acceptable time.
Table (6) and figure (13) illustrate the relation between motifs size, number of characters, number of errors, processing time, and time consumed to load and visualizing the entered motif.

Table (6) time of parsing and colorization for various size + error motif

Size of motif	No. of character in motif	No. of simple motif	No. of gap	No. of error	Time of loading and colorized file	Time of parser H:M:S:MS
10 k base	10249	10172	15	100	0:0:0:742	0:0:3:262
50 k base	51245	50860	75	508	0:0:2:738	0:0:3:356
100 k base	102490	101720	150	650	0:0:5:012	0:0:5:998
500 k base	512450	508600	750	992	0:0:17:578	0:0:6:332
1000 k base	1024900	1017200	1500	1002	0:0:38:581	0:0:8:223
5000 k base	5260234	5086000	7500	1350	0:3:12:373	0:0:25:221
10000 k base	10520468	10222573	15000	1820	0:6:2:123	0:1:54:832

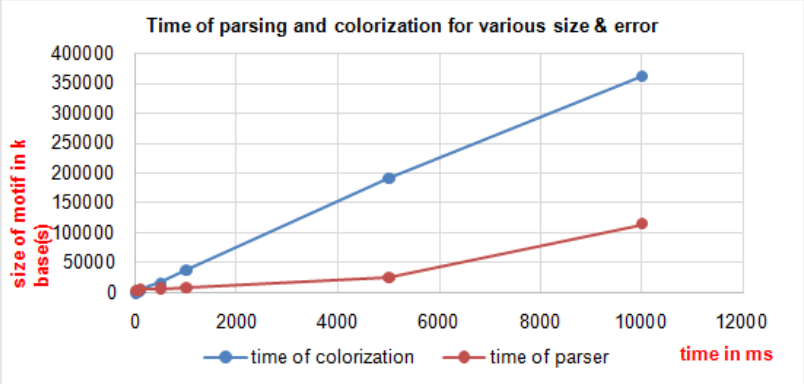

Figure (13) time of parsing and colorization for various size & error motif

Table (7) and figure (14) illustrate the relation between motifs size, number of characters, processing time, and time consumed to load and visualizing the entered error-free motifs.

Table (7) time of parsing and colorization for various size motif

Size of motif	No. of character in motif	No. of simple motif	No. of gap	Time of loading and colorized file	Time of parser H:M:S:MS
10 k base	10249	10172	15	0:0:0:515	0:0:0:113
50 k base	51245	50860	75	0:0:1:876	0:0:0:082
100 k base	102490	101720	150	0:0:3:646	0:0:0:125
500 k base	512450	508600	750	0:0:17:537	0:0:0:541
1000 k base	1024900	1017200	1500	0:0:34:718	0:0:1:116
5000 k base	5260234	5086000	7500	0:2:55:590	0:0:5:675
10000 k base	10520468	10222573	15000	0:5:14:273	0:0:12:075

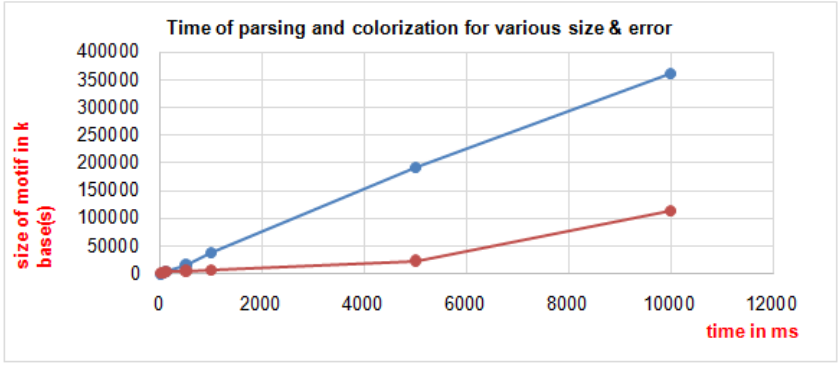

Figure (14) time of parsing and colorization for various size

Table (8) and figure (15) illustrate the relation between number of error and time of parser for (1000 k bases or 1M bases), (5000 K bases or 5 M bases) and (10000 K bases or 10 M bases).

Table (8) time of parsing and number of error for size file 1m, 5m and 10m

No. of error	Time of parser H:M:S:MS		
	1 M bases	5 M bases	10 M bases
100	0:0:1:051	0:0:5:147	0:0:12:320
250	0:0:1:092	0:0:5:356	0:0:12:550
500	0:0:1:110	0:0:5:398	0:0:12:904
750	0:0:1:147	0:0:5:642	0:0:13:183
1000	0:0:1:721	0:0:5:685	0:0:13:545
1250		0:0:6:001	0:0:13:822
1500		0:0:6:626	0:0:14:330
1750		0:0:6:904	0:0:15:206
2000		0:0:7:034	0:0:15:899

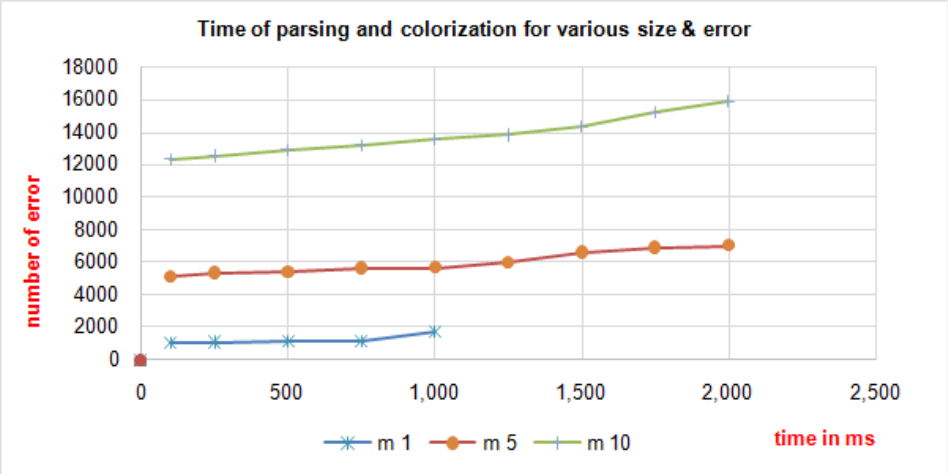

Figure (15) time of parsing and number of error for size file 1m, 5m and 10m

IV. Conclusion

Many conclusions can be extracted from this research, such as:

1) The compilation technique is very suitable for detection and debugging of the errors and intended and unintended noise in a motif template or bioinformatics databases.
2) The experiments accomplished on the presented system showed its ability to deal with the three types of motif DNA, RNA, and protein efficiently.
3) The proposed system manipulates huge size motifs or bioinformatics databases with various number of errors

Bibliography

[1]. Attwood T. K., Gisel A., Eriksson N-E., and BongcamRudloff E. "Concepts, Historical Milestones and the Central Place of Bioinformatics in Modern Biology: A European Perspective", Bioinformatics-Trends and Methodologies, 2012.
[2]. PhD. Alla l lapidus, bioinformatics and its applications, SPBAU, SPBSU, ST. Petersburg.
[3]. Dr. Hussein K. Al-Khafaji, a New Approach to Motif Templates Analysis via Compilation Technique, Al-Rafidain University Collage, 2014.
[4]. SeppHochreiter, bioinformatics I sequence analysis and phylogenetics, institute of bioinformatics, Johannes kepler university linz , winter semester 2013/2014.
[5]. J. Mol. Biol., "Crystal Structure of Yeast Phenylalanine Transfer RNA I. Crystallographic Refinement", 123 607, 1978
[6]. Assistant Professor Sabu M. Thampi, Bioinformatics, LBS College of Engineering.
[7]. Georgia C. Lauritzen, Food & Nutrition Specialist, What Is Protein, Utah state university, cooperative extension, extension.usu.edu, 1992.
[8]. RashidaHasan&JainalUddin,Data Mining Techniques for Informative Motif Discovery,International Journal of Computer Applications,February 2014.
[9]. Gregory A Petsko& Dagmar Ringe, "Protein structure and function", New science press ltd, 2004.
[10]. Modan K Das and Ho-Kwok Dai, "A Survey of DNA Motif Finding Algorithms", Fourth Annual MCBIOS Conference, Computational Frontiers in Biomedicine, 2007.
[11]. Donna K. Hendrix1 , Steven E. Brenner1,2 and Stephen R. Holbrook, RNA structural motifs: building blocks of a modular biomolecule, Cambridge University Press, July 2006.
[12]. Vicki Doronina, How to identify protein motifs from protein sequences, October 2013