Analysis and Visualization of Gene Expressions and Protein Structures

Ashraf S. Hussein
Faculty of Computer and Information Sciences, Ain Shams University, Cairo, 11566, Egypt.
Email: ashrafh@acm.org

Abstract—**This paper describes a web-based interactive framework for the analysis and visualization of gene expressions and protein structures. The formulation of the proposed framework was encountered by many challenges due to the wide range of relevant analysis and visualization techniques, in addition to the existence of a diversity of biological data types, on which these techniques operate. The main challenges that guided the formulation of the present framework are: (a) the integration of data from heterogeneous resources, such as expert-driven data from text, public domain databases and diverse large scale experimental data sets, and (b) difficulty in integrating the most recent analysis and visualization tools due to the lack of standard I/O. Therefore, the fundamental innovation in the proposed framework is the integration of the state-of-the-art techniques of both analysis and visualization for gene expressions and protein structures through a unified workflow. In addition, it supports a wide range of input data types and exports three dimensional interactive outputs using Virtual Reality Modeling Language (VRML) to be ready for exploration via off-the-shelf monitors as well as immersive, 3D, stereo display environments.**

Index Terms—**computer visualization, bioinformatics, gene expressions, protein structures**

I. Introduction

Gigantic amounts of biological data are being generated from researches and experiments everyday. This, along with the evolution of computer science, has resulted in the emerging of the world of Bioinformatics that is growing at an undeniably fast rate. Bioinformatics is officially known as the science of storing, organizing, retrieving and analyzing biological data in order to make use of them in many vital applications [1]. Since the different types of biological data come in enormous amounts, analysis and visualization techniques became essential for the data to be meaningful and interpretable. Consequently, these techniques support biologists in revealing conclusions from their data, leading to more understanding about the principles of biology.

Developing efficient and easy to use tools for the analysis and visualization of gene expressions and protein structures is a goal which has been pursued by a diversity of research groups with various objectives and approaches [32]. The available analysis tools can be categorized into two main categories. The first category handles gene and protein sequences, compares them to the online sequence-databases, and returns the best matching sequences. This proves extremely useful providing that the returned sequences are accompanied with thorough descriptions of them, which assists biologists to assume (or determine with certainty) the properties of the query gene/protein sequence sent [2]. Two effective tools professionally perform this type of analysis which are BLAST [3] and ClustalW [4]. The second category of analysis tools involves performing various analysis techniques on data resulting from the micro-array experiments. One of these analysis techniques is 'clustering' that may be performed on the data of the experiments conducted on genes. Similarities in the genes' behaviors lead to the corresponding genes to be grouped together. Consequently, biologists will be able to know the behavior of these groups and why they behave in such ways [5]. The main famous tools that carry out such analysis techniques are GEPAS [6] and Cluster 3.0 [7].

Visualization tools provide the biologists with graphical representations of the data to be interpreted in more easily way than by merely looking at the raw data. Desktop applications like RasMol [7], PyMOL [9] and IBM Protein Viewer [10] render several representations of the protein structures [10]. Also, ADN-Viewer is specialized in visualizing the DNA sequences using various methods [12].

The data sets, on which these analysis and visualization tools operate, are stored in large online databases such as the National Center of Biotechnology Information (NCBI) [13]. To access these databases, tools that provide web services such as BioMoby [14] and myGrid [15] have to be emerged .

Two main challenges in Bioinformatics have evolved due to this wide variety of tools in addition to the existence of a diversity of biological data types, on which these tools operate. The first challenge is the integration of data from heterogeneous sources, such as expert-driven data from text, public domain databases and diverse large scale experimental data sets. The second is the difficulty in integrating the available analysis and visualization tools due to the lack of standard I/O. To overcome these problems, the proposed framework introduces a unified workflow, which integrates various

types of the state of the art techniques of both analysis and visualization specifically for gene expressions and protein structures. On the other hand, it handles wide range of the commonly used data formats in addition to the support of the virtual reality modeling language (VRML) output to be ready for exploration via off-the-shelf monitors as well as immersive, 3D, stereo display environments. The design of the formulated framework allows users to analyze and visualize their data through a web-portal. Also, users can access, retrieve and acquire data from online biological databases through this portal.

The next sections will be organized as follows: section 2 outlines the basic architecture of the proposed framework. Section 3 presents the implemented analysis functionalities while the adopted visualization functionalities are described in section 4. The analysis and visualization functionalities of the proposed framework are studied and discussed in section 5. Finally, section 6 gives the conclusions of this work and an overview of the future work.

II. ARCHITECTURE

Analysis and visualization are important to interpret scientific data and reveal conclusions in many areas of bioinformatics. It is particularly significant in the structural bioinformatics since the foundation of this field is the 3D structure of biological macromolecules, which can only be interpreted using a molecular graphics program. Historically, to run a visualization program, a user needed to install and configure the program locally. With the advent of the World Wide Web, users can run increasingly complex applications without having to explicitly download them, relying instead on the Web page's delivery with a single mouse click. In addition, web applications have the edge of directly accessing the online databases.

The proposed framework is composed from several modules: the analysis and visualization manager, the user interface and the web services as depicted in figure 1.

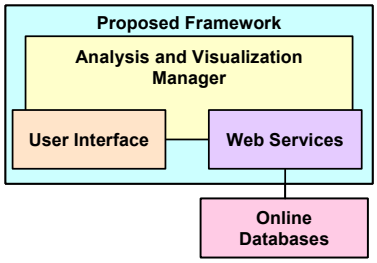


Figure 1. The Proposed Framework Components.

Based on the user request, the present framework follows the corresponding scenario. Each scenario starts from the client site by uploading the user's data to be processed. This data is entered in the client site using one of three methods: (1) entering the data directly in its corresponding fields, (2) uploading the data file or (3) retrieving the data from one of the available online databases employing the search engine. The considered types of the input data files are: (a) the files containing DNA/Protein sequences like Fasta [28], Swiss-Prot [29], GenBank [30] and NBRF files, (b) BLAST[2] and ClustalW[4] files which contain the results of query sequences comparisons with other sequences in the online, biological databases and (c) PDB files [31] which contain 3D protein structures.

The search engine has been included in the formulated framework using the web services technology to permit users to retrieve the desired biological data for analysis and/or visualization. The user can directly enter a protein name, select the database name or the URL of that protein. Biomedical articles can also be accessed via the search engine by entering the article name in the search parameter.

As the user action reaches the server site, the server starts searching in the public domain databases or passes the input directly for further processing based on the user request. Once the desired data has been acquired, the user selects either analysis or visualization to handle the data using the analysis and visualization manager. Finally, the framework returns the results to the client site in a concise, informative, and visual manner. The basic architecture of the proposed framework is shown in figure 2. It is based on three-tier architecture: client, server and online databases as a third-party component.

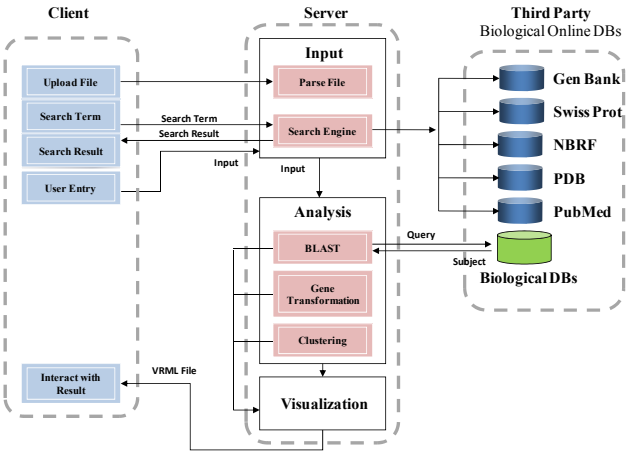


Figure 2. Proposed Framework Architecture

The considered analysis techniques are BLAST [2], gene transformations and clustering. If the user input is a sequence (DNA, RNA or a protein), it can be sent by the BLAST functionality, as a query sequence, to one of the online biological databases. Then, the subject sequence is returned back (the best matching sequence). The gene transformations, including translation and transcription processes (and their reverses), can also be performed on the gene/protein sequences, and the resulting sequences are presented in the user interface. Finally, the clustering module handles the results of the performed experiments on different genes. This technique provides three different schemes: partitional, hierarchical and grid-based clustering.

Visualization operations can be performed directly on the input data or on the output(s) of the analysis techniques. The visualization techniques, which can be performed directly on the input data, include the PDB visualization and the gene/protein sequences

visualization. The PDB visualization is carried out via wire-frame, sticks, ball-and-stick, non-bonded, backbone, ribbons, and cartoons. On the other hand, the sequences can be visualized using gene scale, atomic scale, DB-Curves and 3D representation.

III. Analysis Functionalities

The analysis functionalities described in the following subsections are the first part of the analysis and visualization manager, which is the main module within the proposed framework. These functionalities facilitate the identification of genes/proteins and the detection of their behavior and characteristics. This is useful because the more similar two sequences are, the more their corresponding genes/proteins are alike (properties-wise and behavior-wise). The considered analysis techniques are shown in figure 3.

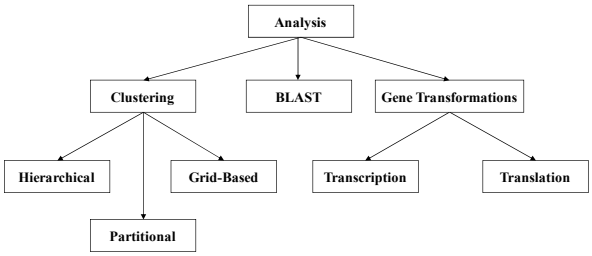


Figure 3. The considered analysis techniques

A. Gene Transformations

Protein production is carried out via cascading transformations. RNA is first generated from DNA, and then proteins are produced from RNA [16]. This is known as the "Central Dogma" in molecular biology as shown in figure 4. The data of DNA, RNA and proteins may be represented in the form of sequences. The gene transformations were considered and implemented in the present framework by connecting to the ExPASy's online translation tool [17], using web services. The resultant sequence can then be presented and/or visualized to the user.

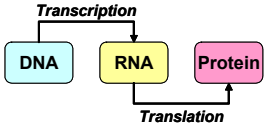


Figure 4. Central Dogma Theory

B. BLAST

Gene/protein sequences carry properties and functionalities of their corresponding genes or proteins. Thus, similarity of sequences can be used as an approach for identifying unknowns and implicit relationships in the sequence world [2]. This is why the web services are provided for connecting to both the BLAST, and the sequence-comparison tool at the NCBI website [16]. Figure 5 shows the events flow in the user interface when performing the BLAST operation. The resulting BLAST report contains the nearest matching sequences that can be visualized, afterwards, by the visualization functionalities provided within the proposed framework.

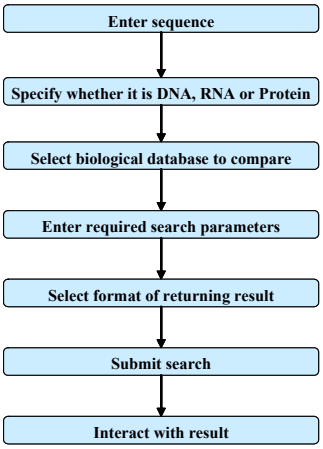


Figure 5. Event Flow during the BLAST Operation

C. Clustering

Micro-array technologies facilitate measuring the gene expression levels for thousands of genes simultaneously. To extract knowledge from the generated datasets, these datasets have to be presented to the user in a meaningful way. Gene clustering methods fulfill this requirement in a professional way. Gene clustering is the process of grouping each set of related genes in a cluster based on their behavior [17]. The considered clustering algorithms are: K-Algorithm [19], Hierarchical [20] and Self-Organized Maps (SOM) [21]. The input for the K-Algorithm is a set **V** consists of **N** points and a parameter **K**. The output is a set **X** consists of **K** points (cluster centers) that minimizes the squared error distortion **d(V,X)** over all possible choices of **X**. The algorithm is described in figure 6.

Arbitrarily assign the **K** cluster centers
While the cluster centers keep changing:
Assign each data point to cluster C_i corresponding to the closest cluster representative (center) having the least **d (V, X)**
where $1 \le i \le K$
After the assignment of all data points, compute new cluster representatives according to the center of gravity of each cluster
(for every cluster C_i the new cluster representative is $\sum v / |C_i|$ for all v in C_i)

Figure 6. K Algorithm

The input to the hierarchical clustering technique [20] is a **N*N** matrix **D** of pair-wise distances between points. The output is a tree **T** with a single all-inclusive cluster at the top and single point clusters at the bottom. The user selects one of four ways to compute distances (called distance measures) to be used in the clustering operation. These distance measures are: single, complete, average, and centroid linkage. Figure 7 illustrates the algorithm of the hierarchical clustering technique.

Form **N** clusters each with one element
Construct a graph **T** by assigning one vertex to each cluster
while there is more than one cluster:
  Find the two closest centers C_1 and C_2
  Merge C_1 and C_2 into a new cluster C with $|C_1|+|C_2|$ elements
  Compute the distance from C to all the other clusters
  Add a new vertex C to **T** and connect it to C_1 and C_2 as their parent
  Remove rows and columns of **D** corresponding to C_1 and C_2
  Add a row and column to **D** corresponding to the cluster C
Return **T**

Figure 7. Hierarchical Clustering Algorithm

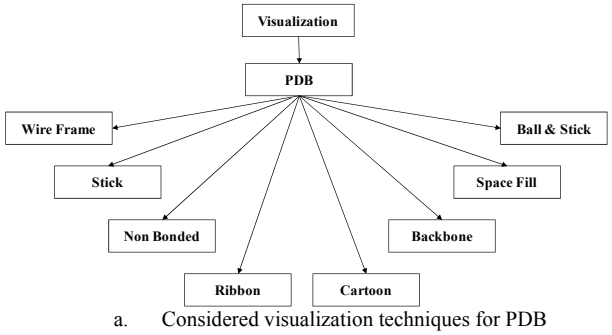
Finally, for the SOM technique [21], as a grid based technique, the input is a set **X** consists of **N** points, and the output is a 2D grid of clusters. Figure 8 illustrates the algorithm of the SOM clustering technique.

Randomly initialize all weights
while there are input vectors in **X**:
  Select next input vector $x = [x_1, x_2, \ldots, x_N]$
  for each neuron j:
    Compute the distance between x and w_j
    Find neuron j with minimum distance (winner)
    Update winner so that it becomes more like x, together with the winner's neighbors for units within the neighborhood according to: $w_{ij}(n+1) = w_{ij}(n) + \eta(n)[x_i - w_{ij}(n)]$
  Adjust parameters; learning rate η and neighborhood function (each neuron represents a cluster that contains similar data).

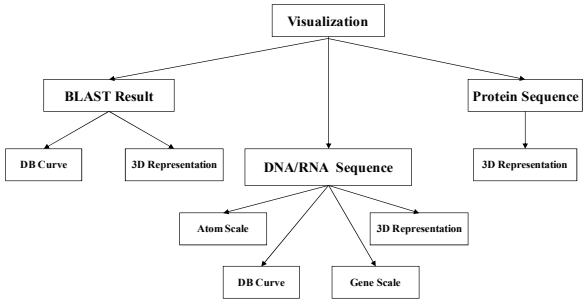
Figure 8. SOM Algorithm

IV. VISUALIZATION FUNCTIONALITIES

The second part of the analysis and visualization manager is concerned with the visualization functionalities, which described in the following subsections. These functionalities assist biologists to reveal conclusions from the massive amounts of data generated from genetics and proteomics experiments. The considered visualization techniques are shown in figure 9.



a. Considered visualization techniques for PDB



b. Considered Visualization Techniques for Blast results, DNA/RNA Sequences and Protein Sequences.

Figure 9. The considered visualization techniques

A. Sequence Visualization

The 3D structure of genes are very important in many essential biological mechanisms, such as the "Central Dogma" [16]. 3D visualization of the gene sequences can be performed using one of the following different methods.

1) Gene Scale

The entire gene sequence can be visualized in a double-helix form representing the connections between the nitrogenous base pairs (A-T and C-G). This representation can be performed along with the connections backbones [12]. Each of the nucleotides, A, C, G and T, is distinguished by a distinct color. Also, each nucleotide in the input sequence is visualized with a cylinder, connecting it to its complementary nucleotide (the nucleotide base pairs are A-T and C-G). Each base pair is visualized above its preceding base pair by a small fixed translation (in the y-direction) and a small fixed rotation (around the y-axis). Figure 10 shows the algorithm of the DNA sequence visualization.

Rotation Angle = 0
Translation Distance = 0
for each nucleotide in the sequence:
  Rotation Angle += fixed rotation value about y-axis
  Translation Distance += fixed translation value in +ve y-direction
  Draw nucleotide and its complementary nucleotide connected by a cylinder
  Apply rotation and translation to drawn base pair

Figure 10. DNA Sequence Visualization at the Gene Scale

2) Atomic Scale

The atomic scale visualizes gene sequences (or part of them) to show the atomic structure of the nitrogenous bases and their connections [12]. There are two standard representations for the atomic scale: the 'wire-frame' (the connections are visualized as cylinders) and the 'ball-and-stick' (the nucleotides are visualized as spheres and the connections between them as cylinders). The atomic scale visualization is similar to that of the gene scale, but with some slight modifications, (atomic scale focuses on a part of the DNA while the gene scale visualizes the entire DNA).

3) DB-Curves

DB-Curves are the 2D representation of the nucleotides in the DNA sequences. This visualization technique facilitates studying the gene characteristics in terms of the behavior of two nucleotides [22]. The DNA sequence is entered as an input and one of six curve types can then be selected. The types of the curves are AC, TC, CG, AT, TG and AG. For the AC DB-Curve, as an example, the vector (1, 0) is defined corresponding to the base A and the vector (0, 1) is defined corresponding to the base C. The rest of the bases, T and G, are given by the vector (1, 1). If the starting point is defined as (0, 0), the DNA sequence can be mapped to the 2D-coordinate system by a cumulative plot of the bases in the sequence using the above notation. The AC DB-Curve emphasizes the A and C bases and makes their visualization more obvious than the rest of the bases. Colors are assigned to the four different bases of the DB-Curves according to

the type of the nucleotide. Figure 11 describes the DB-Curves visualization algorithm.

```
Curve Type = XY
Start at point (0,0)
for each nucleotide in the sequence:
    if (nucleotide type == X)
        Assign (1,0) vector to X
    else if (nucleotide type == Y)
        Assign (0,1) vector to Y
    else if (nucleotide type is otherwise)
        Assign (1,1) vector
    Assign color to nucleotide according to its type
    Draw vector
```

Figure 11. Visualization of DNA sequences via "DB-Curves"

4) 3D Representation

The 3D representation of the DNA sequences is performed by assigning a vector to each nucleotide in the DNA sequence [23]. It is similar to the DB-Curves except that the vectors are assigned in the 3D. Figure 12 describes the algorithm of creating the "3D Representation" to visualize DNA sequences.

In this work, the "3D representation" was modified to visualize linear sequences of proteins. Consequently, the letters in the protein sequence which represent the different amino acids can be divided into several categories based on their side chains.

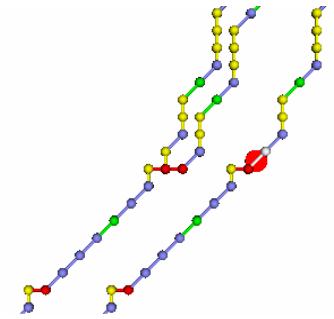
The algorithm of creating "3D Representation" visualization for the protein sequences is quite similar to that of the DNA sequences. The only difference is that there are six vectors for the six categories of amino acids instead of the four vectors used for the four types of nucleotides.

```
for each nucleotide in the sequence:
    if (nucleotide type == A)
        Assign vector (1,0,0)
    if (nucleotide type == C)
        Assign vector (0,1,0)
    if (nucleotide type == G)
        Assign vector (0,0,1)
    if (nucleotide type == T)
        Assign vector (0,1,1)
    Assign color to nucleotide according to its type
    Draw vector
```

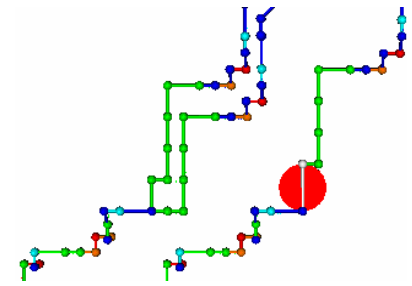
Figure 12. Visualizing DNA Sequences using "3D Representation

5) Visualization of BLAST/ClustalW Results

For the visualization of the BLAST operation results, the query sequence and the subject sequence are both visualized, in a superimposed representation, using the technique of the DB-Curves. Red dots are placed at the locations of mismatches between the two sequences as shown in figure 13. ClustalW uses almost the same idea in its visualization but more than two sequences being compared simultaneously. Figure 14 describes this algorithm.



a. BLAST result on a DNA sequence (DB-Curves)



b. BLAST result on a protein sequence (3D Representation)

Figure 13. The visualization of BLAST result

```
Curve Type = XY
fore ach sequence (from BLAST/ClustalW):
    for each nucleotide:
        if (nucleotide type == X)
            Assign (1,0) vector to X
        if (nucleotide type == Y)
            Assign (0,1) vector to Y
        else
            Assign (1,1) vector
        Assign color according to its type
        if ( gap(ClustalW)/mismatch(BLAST) ):
            Draw a red dot at the vector
```

Figure 14. Visualizing BLAST/ClustalW results using "DB-Curves"

The "3D Representation" algorithm was modified again to visualize the results of both the BLAST and the ClustalW as well. This algorithm is described in figure 15.

```
for each sequence (from BLAST/ClustalW)
    for each sequence symbol:
        if (nucleotide):
            Assign to the nucleotide its corresponding vector
            Assign corresponding color
        else
            Assign new vector
            Assign white color
        Draw vector
```

Figure 15. Visualizing BLAST/ClustalW results using "3D Representation"

B. Protein Structure Visualization

The functionality of any protein can be determined from its 3D structure. There are several techniques to represent the protein 3D structure. The availability of these techniques confirms that the biologist's interpretation of the protein's properties will be quite high. In this work, the considered visualization techniques for the protein structure are: wire-frame, sticks, ball-and-stick, non-bonded, backbone, ribbons, and cartoons representation. The protein structures are saved in standard 'PDB files'. The CPK color scheme

[27] was used within the algorithms described in this sub-section. A distinct standard color is assigned to each atom type (Oxygen atoms are red colored, carbon atoms are green colored, and so on).

The number of 'standard' amino acids, that form proteins, is known to be 20 so far. These amino acids are called 'standard' because the connections between the atoms are already known and can be found in the references concerning biochemistry. In addition, these amino acids are connected by means of peptide bonds as shown in figure 16. The primary structure of the protein is the amino-acid sequence of the formed polypeptide chains [24].

All amino acids have in common a central carbon atom, a hydrogen atom, an amino group (NH_2) and a carboxyl group (COOH), as shown in Figure 17. The central carbon atom is called the C_{alpha} (or CA) atom. The side chain of each amino acid is what makes it unique and different from the other amino acids. There are 20 side chains found in the proteins encoded by the genetic machinery of the cell (one for each amino acid) [24].

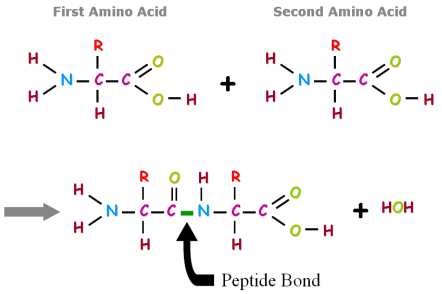


Figure 16. Peptide Chain Formation

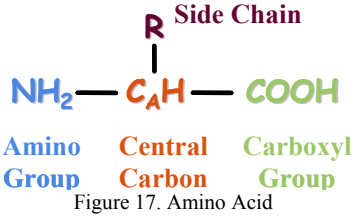


Figure 17. Amino Acid

'Wire-frame', 'sticks' and 'ball-and-stick' are the standard representations which show the atoms along with the connections between them [25]. In order to create these representations, a text specifying the connections within each amino acid (i.e. residue) is needed.

As implied by the primary structure of the protein, the amino acids in the PDB file should be connected unless there is a TER record in the middle (which indicates the end of a peptide chain and, possibly, the beginning of another one). Finally, the CONECT records in the PDB file contain the non-standard connections (which are not peptide bond connections) between the atoms. Having determined all the required connections, these representations can hence be created. The algorithm for the 'wire-frame' representation is shown in figure 18.

```
Initialize CONN as an empty array
for each residue:
    Store the atoms of the residue (or copy) in some temporary
array
    Determine the connections and store them in the form:
(atomSerial1, atomSerial2)
    Add these connections to CONN
    if the next record is not of type TER
        Add a connection between its C-atom and the next
residue's
            N-atom to CONN
for each CONECT record:
    Add its connections to CONN
for each connection (atomSerial1, atomSerial2) in CONN:
    Retrieve atom1 and atom2
    Color1 = color of atom1
    Color2 = color of atom2
    Vector = atom2 - atom1
    Midpoint = (atom1 + atom2)/2
    Use Vector and Midpoint to compute required rotation and
translation transformations respectively
    Create a cylinder (half of it in Color1, and the other half in
Color2)
    Apply transformations to the created cylinder
    Draw the cylinder
```

Figure 18. Wire-frame Representation

The 'sticks' and 'ball-and-stick' visualization techniques are quite similar to the 'wire-frame'. The cylinders in the 'sticks' representation are little thicker. In the 'ball-and-stick' representation, spheres are drawn at the atoms' locations to emphasize them. All of the atoms in the protein (not only oxygen atoms as in other viewers) are simply drawn as spheres with no connections between them. The 'backbone' and 'ribbons' representations focus on the backbone of the protein [26]. The 'backbone' representation is created by connecting the central carbon atom (CA) of each amino acid to that of the next one. The idea is to create peptide planes along the amino acids for the ribbons to be twisted around [26]. Figure 19 shows the algorithm for the 'backbone' representation.

```
tempCA = CA-atom of first amino acid
for each atom (starting from the second amino acid):
    if atom is a CA-atom:
        Draw cylinder between it and tempCA
        tempCA = atom
```

Figure 19. Backbone Representation

Finally, the 'cartoons' visualization technique concerns with the secondary structures of the protein. In this framework, the "cartoon" representation is developed by a new methodology. Helices and sheets are represented as red cylinders and yellow arrows respectively. The red cylinders are drawn between the first and the last CA atoms in each helix. The yellow arrows are drawn using Bezier Curves with CA atoms as their control points (also between the first and the last CA atoms in each sheet). For the rest of the protein, a Hermite Spline connects each consecutive pair of CA atoms (i.e. CA atoms of consecutive amino acids in the PDB file). The tangent of the spline at each CA atom is double the vector from the CA atom to the C atom (i.e. the carbon atom of the carboxyl group) of the same amino acid. This algorithm is illustrated in figure 20.

```
for each helix (HELIX record):
    seqR1 = the sequence number of the starting residue
    seqR2 = the sequence number of the terminating residue
    serialCA1 = serial of the CA atom of the residue of sequence
number 'seqR1'
    serialCA2 = serial of the CA atom of the residue of sequence
number 'seqR2'
    Retrieve atomCA1 and atomCA2
    Vector = atomCA2 – atomCA1
    Midpoint = (atom1 + atom2)/2
    Use Vector to compute required rotation transformation
    Use Midpoint to compute required translation transformation
    Create a red thick cylinder
    Apply transformations to created cylinder
    DRAW the cylinder
for each sheet (SHEET record):
    seqR1 = the sequence number of the starting residue
    seqR2 = the sequence number of the terminating residue
    CAserials = serials of the CA atoms from seqR1 to seqR2
    Retrieve the CA atoms and use them as control points to
create Bezier curve
    Draw yellow arrows along the generated Bezier curve to
represent the sheet
for each CA-atom not in a helix or a sheet:
    Compute tangent: T1 = CA - C
    Similarly compute tangent T2 at the next CA-atom
    Generate Hermite Spline using T1 and T2
    Draw curve using the generated Hermite Spline
```

Figure 20. Cartoons Representation

C. Clustering Visualization

The proposed framework also provides visualization service for the data resulting from most of the clustering techniques previously mentioned in section 3.

For the K-Algorithm, each cluster of genes is represented as a group of spheres having a distinct color. In the hierarchical clustering, evolutionary trees are drawn with genes as their leaves. Every node (except the leaf nodes) has two children. The node can be a child of another node, which may be also a child of another node and so on. If each node is assigned the correct children, this will lead to the formation of a tree data structure. The depth-first algorithm can then be used to visualize the results of the hierarchical clustering, also in the form of a tree. The use of the depth-first technique prevents any visual intersections between the stems of the visualized trees.

Finally, For the SOM clustering technique, each set of genes, which are similar in behavior are grouped up in a cluster. When the user selects one of the clusters, a Cartesian graph look-alike is drawn (where the x-axis represents the experiments, the y-axis represents the gene's behavior). Each of the genes is represented by a number of connected line segments. These segments show the selected gene cluster behavior in different experiments.

V. DISCUSSION

In this work, the analysis and visualization functionalities were implemented using python, and C++ programming languages while the portal interface was developed using ASP.NET and PHP.

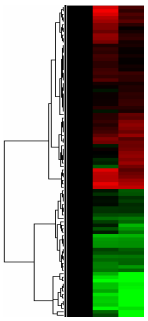
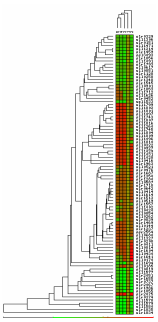
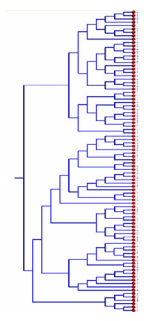
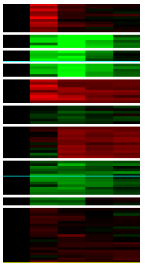
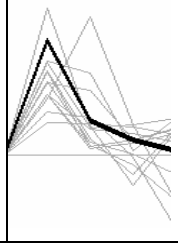
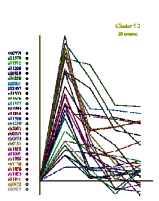
The results of the different functionalities of the proposed framework were validated against other analysis and visualization tools. The considered tools for validation purposes are Cluster 3.0 [7] and GEPAS[6] for the clustering results, ADN-Viewer [12] for the DNA/RNA/Protein Sequence visualization, and RasMol [7] and PyMol [9] for the protein structure visualization.

The features of the present framework are compared with that of the Cluster 3.0 [7] and GEPAS[6]. The metrics of comparison include the web-enablement, the range of the algorithms supported, the distance functions used and the I/O types. As shown in table 1, the proposed framework covers most of the other two applications features. Also, the visualization functionalities of the proposed framework for the different types of clustering results are compared with that of the Cluster 3.0 and GEPAS as depicted in table 2. As shown, the proposed framework bested others since it supports hierarchical, partitional and grid-based clustering results.

TABLE I. COMPARISON BETWEEN THE PROPOSED FRAMEWORK AND CLUSTER 3.0 AND GEPAS

	Cluster 3.0	**GEPAS**	**Proposed Framework**
Web-based/ Console Application	Console	Web-based	Web-based
Algorithms	• K-Mean • K-Median • SOM • Hierarchical • PCA	• K-Mean • SOM • Hierarchical	• K-Mean • K-Median • SOM • Hierarchical
Distance Functions Used	• Correlation "Centered" • Correlation " Not Centered" • Euclidean Distance • Absolute Correlation "Centered" • Absolute Correlation "Not Centered" • Spearman Rank Correlation • Kendall's Tau • City block	• Correlation "Centered" • Correlation "Not Centered" • Euclidean Distance • Absolute Correlation "Centered" • Absolute Correlation " Not Centered" • Spearman Rank Correlation • Kendall's Tau • City Block • Harmonically Summed Euclidean Distance	• Correlation "Centered" • Correlation "Not Centered" • Euclidean Distance • Absolute Correlation "Centered" • Absolute Correlation "Not Centered" • Spearman Rank Correlation • Kendall's Tau • City block
Input Format	A text file contains the results of certain experiments on genes		
Output	• Files that contain the clustering results. • Visualization can be obtained for results of hierarchical clustering only (using TreeView).	• Files that contain the clustering results (can be downloaded). • Visualizes the result in *jpg* format.	• Files that contain the clustering results. • Visualization of all clustering techniques in 3D *VRML* format.

TABLE II. THE PROPOSED FRAMEWORK OUTPUT FOR VISUALIZING CLUSTERING RESULTS VERSUS THAT OF THE OTHER PUBLIC DOMAIN TOOLS

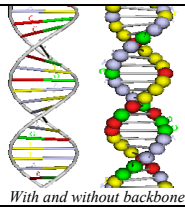
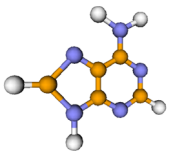
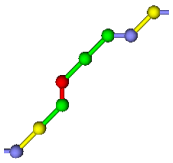
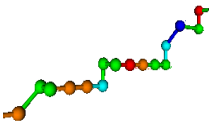
	Cluster 3.0	GEPAS	Proposed Framework
Hierarchical Clustering Result			
Partitional Clustering Result		*No Visualization for Partitional Clustering Technique*	
Grid-Based Clustering Result	*No visualization for Grid-Based Clustering Technique*		

The features of the present framework are compared with that of the ADN-Viewer [12] for visualizing DNA sequences. The metrics of comparison are web-enablement, supported techniques and output format. The comparison is summarized in table 3. For DNA/RNA/Protein sequences visualization, the proposed framework includes three new visualization techniques which are not included in the ADN-Viewer [12]. These techniques are the Atomic Scale, DB-Curves representations, and the 3D Representation. In addition, it has some extra features like the web-enablement and the interactive 3D output provided in VRML format. Also, it provides some enhancements on the gene scale representation by representing the gene scale with back-bone. Sample of the visualization outputs are given in table 4.

TABLE III. COMPARISON BETWEEN THE PROPOSED FRAMEWORK AND THE ADN-VIEWER FOR VISUALIZING DNA SEQUENCES.

	ADN-Viewer	Proposed Framework
Web-based/ Text-based	Text-based	Web-based
Supported Techniques	• Gene Scale Only	• Gene Scale • Atomic Scale • DB-Curves • 3D Representation
Output Format	OpenGL	VRML

TABLE IV. AMPLE OUTPUTS FROM THE PROPOSED FRAMEWORK AND THE ADN-VIEWER FOR VISUALIZING DNA SEQUENCES.

	ADN-Viewer	Proposed Framework
Gene Scale	 *Only Without Backbone*	 *With and without backbone*
Atomic Scale	*Not supported*	
DB-Curves	*Not supported*	
3D Representation	*Not supported*	

Finally, the protein structure visualization functionalities of the proposed framework are compared to that of the RasMol [7] and PyMol [9]. The metrics of comparison include interactivity, labeling atoms, supported techniques, web-enablement and coloring schemes. As shown in table 5, the proposed framework covers most of the other two applications features.

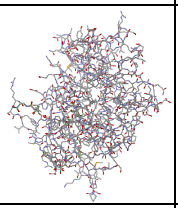
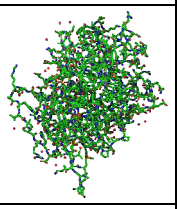
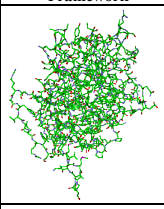
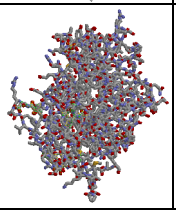
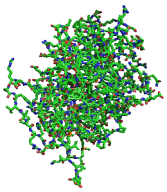
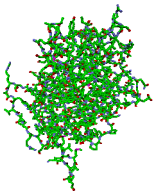
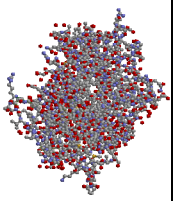
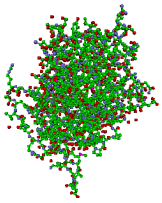
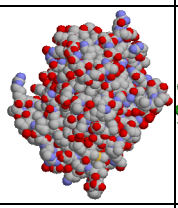
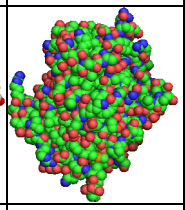
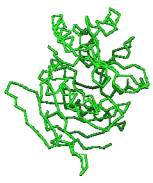
TABLE V. COMPARISON BETWEEN THE PROPOSED FRAMEWORK AND THE OTHER PROTEIN VIEWERS

	RasMol	PyMol	Proposed Framework
Interactivity	Rotation, Zooming	Rotation, Zooming	VRML facilitates : navigation, immersive navigation and fly-through
Labeling Atoms	Yes	Yes	Yes
Supported Techniques	• Wire-frame • Sticks • Space-fill • Ball & Stick • Backbone • Ribbons • Strands • Cartoon	• Wire-frame • Sticks • Space-fill • Backbone • Cartoon	• Wire-frame • Sticks • Non bonded • Ball & Stick • Backbone • Ribbons • Cartoon
Web-based/ Text-based Application	Text-based	Text-based	Web-based
Coloring Schemes	• CPK • Monochrome • Chains • Temperature • Structure	• CPK • Monochrome • Chains • Temperature • Structure	• CPK

The output generated by the proposed framework compares favorably with that generated from RasMol [8] and PyMol [9] for most of the techniques as shown in table 6. Due to the enhancements in the implemented cartoon and non-bounded representations, some differences appear. Unlike RasMol and PyMol, the

helices in the cartoons representation in our framework are represented by cylinders instead of twirls, which give better representation as shown. For the non-bonded representation, the proposed framework visualizes all of the atoms in the protein as small spheres (not just the oxygen atoms as in some of the other protein viewers).

TABLE VI. COMPARISON BETWEEN THE VISUALIZATION OUTPUT OF THE PROPOSED FRAMEWORK AND THE OTHER PROTEIN VIEWERS

	RasMol	PyMol	Proposed Framework
Wire-frame			
Stick			
Ball-and-Stick		*Not Supported*	
Space-fill			*Not Supported*
Backbone			

For all the considered comparisons, the Virtual Reality Modeling Language (VRML) provides high interactivity for exploration employing either traditional VRML navigators or immersive, 3D, stereo display environments.

VI. CONCLUSION

In this work, a unified web-based framework is proposed to interactively analyze and visualize gene expressions and protein structures. The proposed framework integrates the state-of-the-art analysis and visualization techniques through a complete coherent web-based interactive environment. The considered analysis techniques include gene transformations, BLAST and clustering, while the considered visualization techniques include wire-frame, sticks, ball-and-stick, non-bonded, backbone, ribbons, cartoons representation, gene scale, atomic scale, DB-Curves and 3D representations. On the other hand, the proposed framework handles a wide range of the commonly used input data types and employs the Virtual Reality Modeling Language (VRML) for the interactive output. The proposed framework was compared favorably with other applications. In addition, the proposed framework covers most of the available features within these applications and has more features especially in the visualization functionalities. As a future work, the distributed memory parallelization will be considered for the analysis functionalities to reduce their total execution time.

ACKNOWLEDGMENT

The author would like to thank Prof. M. F. Tolba, Dr. O. H. Karam and Dr. T. Hassan for valuable discussions. This work was supported in part by a grant from Information Technology Academia Collaboration Programs (ITAC), Information Technology Industry Development Agency (ITIDA), Ministry of Communication and Information Technology, Egypt.

REFERENCES

[1] J. Barker and J. Thornton, "Software Engineering Challenges in Bioinformatics", *Proc. of the 26th Int. Conf. on Software Engineering (ICSE'04)*, pp. 12- 15, 2004.

[2] J. Bedell, I. Korf and M. Yandell, *BLAST*. O'Reilly, July 2003.

[3] "Basic Local Alignment Search Tool," *National Center for Biotechnology Information, U.S. National Library of Medicine*. http://www.ncbi.nlm.nih.gov/BLAST. 2007.

[4] "ClustalW2: A General Purpose Multiple Sequence Alignment Program for DNA or Proteins," *European Bioinformatics Institute*, http://www.ebi.ac.uk/clustalw. 2007.

[5] M. Dettling and P. Bühlmann, "Supervised Clustering of Genes," *Genome Biology*, vol. 3, No. 12, http://genomebiology.com/2002/3/12/research/0069.3. 2002.

[6] J.M. Vaquerizas, L. Conde, P. Yankilevich, A. Cabezon, P. Minguez, R. Diaz-Uriarte, F. Al-Shahrour, J. Herrero and J. Dopazo, "GEPAS: An Experiment-Oriented Pipeline for the Analysis of Microarray Gene Expression Data," *Nucleic Acids Research*, vol. 33, (Web Server issue): W616-W620, 2005.

[7] M. J. L. de Hoon, S. Imoto, J. Nolan, and S. Miyano, "Open Source Clustering Software," *Bioinformatics*, vol. 20, No. 9, pp. 1453—1454, 2004.

[8] H. J. Bernstein, "Home Page for RasMol and OpenRasMol," Bernstein and Sons, Information Systems Consultants, http://www.openrasmol.org. 2007.

[9] "PyMOL: A User Sponsored Molecular Visualization System," http://pymol.sourceforge.net. 2007.

[10] F. Suits and D. Gresh, "Prototype Protein Viewer," *IBM's T. J. Watson Research Center*, http://www.alphaworks.ibm.com/tech/ppv/. 2001.

[11] P. Lai, W. Kaplan, W. B. Church and R. K. Wong, "Informative 3D Visualization of Multiple Protein Structures", *Proc. of the 2nd Asia-Pacific Bioinformatics Conf. (APBC2004)*, vol. 29, pp. 201 – 208, 2004.

[12] J. Hérisson, P. E. Gros, N. Férey, O. Magneau and R. Gherbi, "DNA in Virtuo: Visualization and Exploration of 3D Genomic Structures", *Proc. of the 3rd Int. Conf. on Computer Graphics, Virtual reality, Visualisation and interaction in Africa*, pp. 35-40, 2004.
[13] "National Center for Biotechnology Information," *U.S. National Library of Medicine*, http://www.ncbi.nlm.nih.gov/. 2007.
[14] M. D. Wilkinson and M. Links, "BioMoby: An Open Source Biological Web Services Proposal," *Briefings in Bioinformatics*, vol. 3, No. 4, pp. 331-341, 2002.
[15] Y. L. Simmhan, B. Plale, D. Gannon, "A Survey of Data Provenance in e-Science," *ACM SIGMOD Record*, vol. 34, No. 3, pp. 31-36, 2005.
[16] Jacques Cohen, "Bioinformatics—An Introduction for Computer Scientists," *ACM Computing Surveys (CSUR)*, vol. 36, No. 2, pp. 122–158, 2004.
[17] E. Gasteiger, A. Gattiker, C. Hoogland, I. Ivanyi, R.D. Appel and A. Bairoch, "ExPASy: the proteomics server for in-depth protein knowledge and analysis," *Nucleic Acids Research*, vol. 31, pp. 3784-3788, 2003.
[18] X. Xiao, E. R. Dow, R. Eberhart, Z. B. Miled, R. J. Oppelt, "Gene Clustering Using Self-Organizing Maps and Particle Swarm Optimization," *Proc. of the Int. Parallel and Distributed Processing Symposium*, 2003.
[19] A. M. Fahim, A. M. Salem, F. A. Torkey, F. A. Ramadan, "An Efficient Enhanced K-means Clustering Algorithm," *Journal of Zhejiang University SCIENCE A*, vol. 7, No. 10, pp. 1626-1633, 2006.
[20] A. K. Jain, M.N. Murty and P.J. Flynn, "Data Clustering: A Review," *ACM Computing Surveys*, vol. 31, No. 3, 1999.
[21] T. Honkela, T. Leinonen, K. Lonka, A. Raike, "Self-Organizing Maps and Constructive Learning," Proc. of Int. Conf. on Educational Uses of Communication and Information Technologies (ICEUT'2000), pp. 339-343, 2000.
[22] Y. Wu, A. W. Liew, H. Yan and M. Yang, "DB-Curve: A Novel 2D Method of DNA Sequence Visualization and Representation," *Chemical Physics Letters*, vol. 367, No. 1, pp. 170-176, 2003.
[23] C. Li and J. Wang, "On a 3-D Representation of DNA Primary Sequences," *Combinatorial Chemistry & High Throughput Screening*, vol. 7, No. 1, pp. 23-27, 2004.
[24] A. Halm, L. Offen and D. Fellner, "BioBrowser: A Framework for Fast Protein Visualization," EUROGRAPHICS - IEEE VGTC Symposium on Visualization, pp. 287–294, 2005.
[25] "PROTEIN DATA BANK: Atomic Coordinate and Bibliographic Entry Format," *Research Collaboratory for Structural Bioinformatics (RCSB)*, www.rcsb.org/. 2007.
[26] M. Carson and C. E. Bugg, "Algorithm for Ribbon Models of Proteins," *Journal of Molecular Graphics*, vol. 4, No.2, pp.121-122, 1986.
[27] CPK Color Schema, http://pdb.bu.edu/oca-docs/cpk.html and http://en.wikipedia.org/wiki/CPK_coloring/. 2007.
[28] FASTA Sequence Comparison at the U. of Virginia, http://fasta.bioch.virginia.edu/fasta_www2/fasta_list2.shtml/. 2007.
[29] Protein knowledgebase, http://www.expasy.ch/sprot/. 2007.
[30] GenBank Databases, http://www.psc.edu/general/software/packages/genbank/genbank.php /. 2007.
[31] Protein Data Bank, www.PDB.org/. 2007.
[32] Q. Zhang, S. Veretnik and P. E. Bourne, "Overview of Structural Bioinformatics," Bioinformatics Technologies, Springer Berlin Heidelberg, pp. 15-44, 2005.

Ashraf S. Hussein was born in Giza, Egypt, in 1970. He received his B.Sc., M.Sc. and Ph.D. degrees in Aerospace Engineering from Cairo University, Egypt in 1992, 1996 and 1999, respectively. The major fields of his studies are Modeling and Simulation, and Computer Visualization.

During 1992-1996, he worked as a research engineer in the Department of Aerospace Engineering, Cairo University. He joined IBM Cairo Technology Development Centre in 1996 and IBM Cairo Center for Advanced Studies in 2005. In addition, he joined Faculty of Computer and Information Sciences, Ain Shams University, Egypt in 2001. Currently, he is an Associate Professor in the Department of Scientific Computing, Faculty of Computer and Information Sciences, Ain Shams University. Also, he is a Visiting Scientist, IBM Center for Advanced Studies in Cairo.

His research areas cover Modeling and Simulation, Computational Techniques in Science and Engineering, Computer Graphics and Visualization, and High Performance Computing Applications. He has more than 35 published scientific papers in international conferences and journals. He has participated in more than 15 R&D and incubation projects.

Dr. Hussein served as a member in numerous technical committees of international scientific conferences. He is a member of the editorial board of one technical journal and a professional member of the ACM.