A Technique for Visualizing Dihedral Signal of Large Protein Sequences

Quang Vinh Nguyen[1], Mao Lin Huang[1], Yu Qian[2] and I-Ling Yen[2]
[1]Faculty of Information Technology, University of Technology, Sydney, Australia
[2]Department of Computer Science, University of Texas at Dallas, Texas, USA
{quvnguye, maolin}@it.uts.edu.au, { yxq012100, ilyen }@utdallas.edu

Abstract

This paper presents a clustering and visualization technique for analyzing dihedral angles of large protein sequences. The clustering is used for discovering and grouping those similar dihedrals while the visualization can display and navigate sequences of dihedral angles of several proteins as well as their clustered property. In order to visualize a very large sequence of hundred thousands of dihedral signals, we plot them on a spiral coordinate system. This spiral visualization ensures the linear distribution without distortion or interruption of a very long sequence of points. A clustering algorithm is also provided to group those dihedral signals into different clusters so that it can enhance the analysis process. Our system can also zoom to display a number of selected proteins interactively.

1. Introduction

With the growth of research in genetics, the quantity and variety of proteins increase significantly each year. This raises the necessity of developing visualization tools for contrasting and comparing the trend and patent of multiple proteins.

Protein sequence's analysis is one of the important areas in genetic research. Although proteins can be displayed as three-dimensional (3D) structure, it does not visualize well the physicochemical and kinetic factors. Therefore, it could prevent us from understanding the internal structures. A protein structure is normally characterized by α-helix, β-strand, and un-repetitive coil. These α-helix and β-strand are defined by the characteristics of a sequence of dihedral angles. The visualization of these angles is a visual aid for emphasizing the internal structure of each protein as well as any interesting trends or patents between different proteins. Each protein can contain tens to several thousands of dihedral angles. Thus, visualizing concurrently hundreds of proteins could lead to several hundred thousands of angles. This raises a question of how to visualize such a large quantity of linear points without distortion or/and interruption.

Current protein visualization is mainly in 3D. These techniques mainly provide a realistic look and feel of proteins rather than their actual internal structures. Consequently, they are not very useful for analyzing sequences of dihedral angles inside each protein or several consecutive proteins. A few typical protein visualization tools in 3D are *Interactive Protein Manipulation* [1], *ConSurf* [2], and *FPV* [3].

Visualization of protein structures is also in two-dimensional (2D) space. Most algorithms described global conformations of the proteins using a simple structural alphabet [4], [5], [6]. However, this simple type of visualization might require an extra time for the user to learn the mapping between the alphabet characters and the actual structural. In addition, the display tends to be very wide when visualizing long proteins. Some other techniques use concentric circles to visualize microbial genomes [7], [8]. Although this idea is somehow similar to our spiral visualization, the concentric circles can cause interruption between each circle which makes them unsuitable to visualize a very long single sequence.

This paper presents a new technique for visualizing the sequence of dihedrals angles of a large number of proteins as well as a few selected proteins. In short, we use spiral coordination in 2D to display the very long sequence of dihedrals angles in which each angle is draw as a point or a connected segment along a spiral line. This visualization aims to provide a smooth and highly capable way for display the long protein sequences without an interruption. Our technique includes a clustering algorithm to enhance user's spatial pattern recognition from the visualization. An interactive zooming is also provided to allow user to navigate to any specific proteins while the overall context view is retained as a small display window. In addition, coloring is also used to identify the proteins.

Section 2 describes the protein database that was used in out experiment. Section 3 presents the clustering algorithm. Section 4 shows the technical detail of the visualization system. Final section is our conclusion.

2. Protein database

A total of 2000 proteins are randomly selected in a database. For each protein, we have stored the series of dihedral angles with the value varies in (-180, 180). Each protein has from tens to several thousands of dihedral angles. Thus we have a total of more than 600,000 angles. Unfortunately, a normal screen has typically only more than 1 million pixels. To avoid the over density of the display, we only visualize a sequence of maximum around 150 proteins with around 45,000 dihedral angles at a time.

Each two consecutive dihedral angles have typically one positive value and one opposite negative. Therefore, the trend of a sequence of *n* points *{0, 2, 4, …, 2n - 2}* is often similar to a sequence of other *n* points *{1, 3, 5, …, 2n – 1}*.

3. Clustering

Given a similarity matrix, an effective approach to document classification is cluster analysis. Clustering methods can be traditionally classified into four categories including partitioning, hierarchical, density-based, and grid-based methods. Among the four kinds of methods, hierarchical methods can be directly applied to process the similarity matrix without needing the original data. To provide a comparison and demonstrate the compatibility of the layout method, two clustering methods are selected to classify the dihedral signals. The first one is the classic hierarchical agglomerate clustering (*HAC*) method and the second one is *FAÇADE* [11]. The two clustering methods merge data in different ways. While *HAC* considers the similarity value between pair of dihedral points, *FAÇADE* considers a group of documents and uses group density information to merge data hierarchically. These two techniques are next briefly described.

3.2.1 HAC

The basic idea of *HAC* method is straightforward. Given a similarity matrix, the most similar pair of data items is found and merged into one cluster. Then the similarity matrix is updated and the merging process is repeated until all items are in one cluster. According to the way of updating the similarity matrix, *HAC* methods can be single-link [12], complete-link [10], and minimum-variance [13]. Of these, the single-link and complete-link algorithms are most popular [9]. In the single-link method, the distance between two clusters is updated with the minimum of the distances between all pairs of data items drawn from the two clusters (one item from the first cluster and the other from the second). In the complete-link algorithm, the distance between two clusters is updated with the maximum of all pair-wise distances between items in the two clusters. To save sorting time of updating the similarity matrix, the single-link method is used as a representative of *HAC* method. Since single-link method suffers from a chaining effect and has a tendency to produce very big clusters, we put a constraint to limit the cluster size. The adapted single-link *HAC* method is described as follows. Given *n* points each belonging to its own group, the goal is to generate k groups/clusters by merging the n groups pair by pair. The similarity between two groups is represented by the most similar pair of papers in the two groups. The combination of groups continues along the decreasing order of similarity values with a constraint that the size of any group is smaller than *4n/k*, i.e., four times of average. In other words, if merging the pair with the current biggest similarity value breaks the constraint, the pair with the next biggest similarity will be tested until the constraint is satisfied. For each combination the number of groups decreases by 1 until the number of groups becomes k, i.e., *n-k* combinations are needed.

3.2.2 FAÇADE

As a clustering method, *HAC* has many weaknesses: it is sensitive to noise and cannot discover outliers effectively; its group merging based on individual data items is often biased when groups are big and of different shapes. *FAÇADE* [11] is a recently proposed spatial clustering algorithm with an advantage of preserving cluster shapes in a noisy environment. *FAÇADE* consists of four steps: 1) Graph construction. A k-mutual neighborhood graph is constructed based on the given similarity matrix, which will be used for noise removal and group merging. 2) Noise removal. The constructed graph will be partitioned into different parts and the parts with less significant connections will be discarded as noise. 3) Compression. This step produces initial groups for later merging step and accelerates the merging step without losing information about cluster shape. 4) Group merging. A new merging criterion based on the connections between two groups and the sizes of the two groups is proposed. With the new criterion, the relationship between two groups is better described because the information at the group level is discovered and utilized. In the original version of *FAÇADE*, the step of compression is repeated multiple times, which cannot be used in this paper because the original data is not available. This will not make a significant change on the final result because the step of compression is designed for speed purpose only. Compared with *HAC*, *FAÇADE* will produce a noise group containing documents with insignificant relationship to other documents. Also, *FAÇADE* does not require a constraint limit on the group size and therefore is independent of the step of paper collection. The speed of *FAÇADE* is supposed to be much faster than *HAC* due to the compression step. In the practical running without compression, we found that *FAÇADE* and *HAC* are of similar speed and both of them can cluster thousands of documents in seconds at a personal computer. We used *FAÇADE clustering algorithm in this prototype.* Figure 5 shows an example of the visualization using *FAÇADE* clustering algorithm. More technical detail and comparison between two clustering techniques can be found at [11].

4. Visualization

This section presents the technical detail of our technique for visualizing sequences of dihedral angles of several proteins as well as their clustered property from section 3. The visualization system includes several components including a *context view*, a *main view* and a *single protein view*.

The *context view* provides an overall view of the entire protein sequence. This view is displayed at a small panel at the top-right of the system (see Figure 1). This context panel also highlights the focus region, so that user can keep their mental map during the navigation.

The *main view* provides the detail view of one-dimensional (1D) dihedral angles of a number of selected proteins along a spiral line. This view is displayed at the left-hand side large panel. The number of focused proteins can be interactively adjusted via the "zoom panel", showing from one protein to the entire dataset. Figure 1 shows an example of the visualization for a large number of proteins while Figure 3 shows an example of the visualization for a small number of proteins.

Finally, the *single protein view* shows further detail of a selected protein using the traditional visualization. This view is displayed at the bottom-right panel which dihedral angles are drawn along a vertical coordination (see Figure 1).

The concentric spiral line was used in our visualization to visualize the 1D sequence of dihedral angles. This is because that we aim to display concurrently several proteins at the time to analyze the properties of each individual protein as well as multiple proteins. And the number of dihedral points is very large which could read to hundred thousands to millions. Therefore, a straight-line on a screen is not sufficient to show such large amount of points without breaking or a scroll-bar. This problem might be solved by using multiple-lines and/or multiple-lines with connected lines. However, in these approaches, the discontinuity also occurs in the visualization between the turning points at the connection between straight-lines and/or curves or two broken lines. The spiral drawing approach was chosen in our visualization system because it was highly capable of providing a very long smooth and continuing line at a limited display space (see Figure 1).

The following sections describe further detail of the *main view* component.

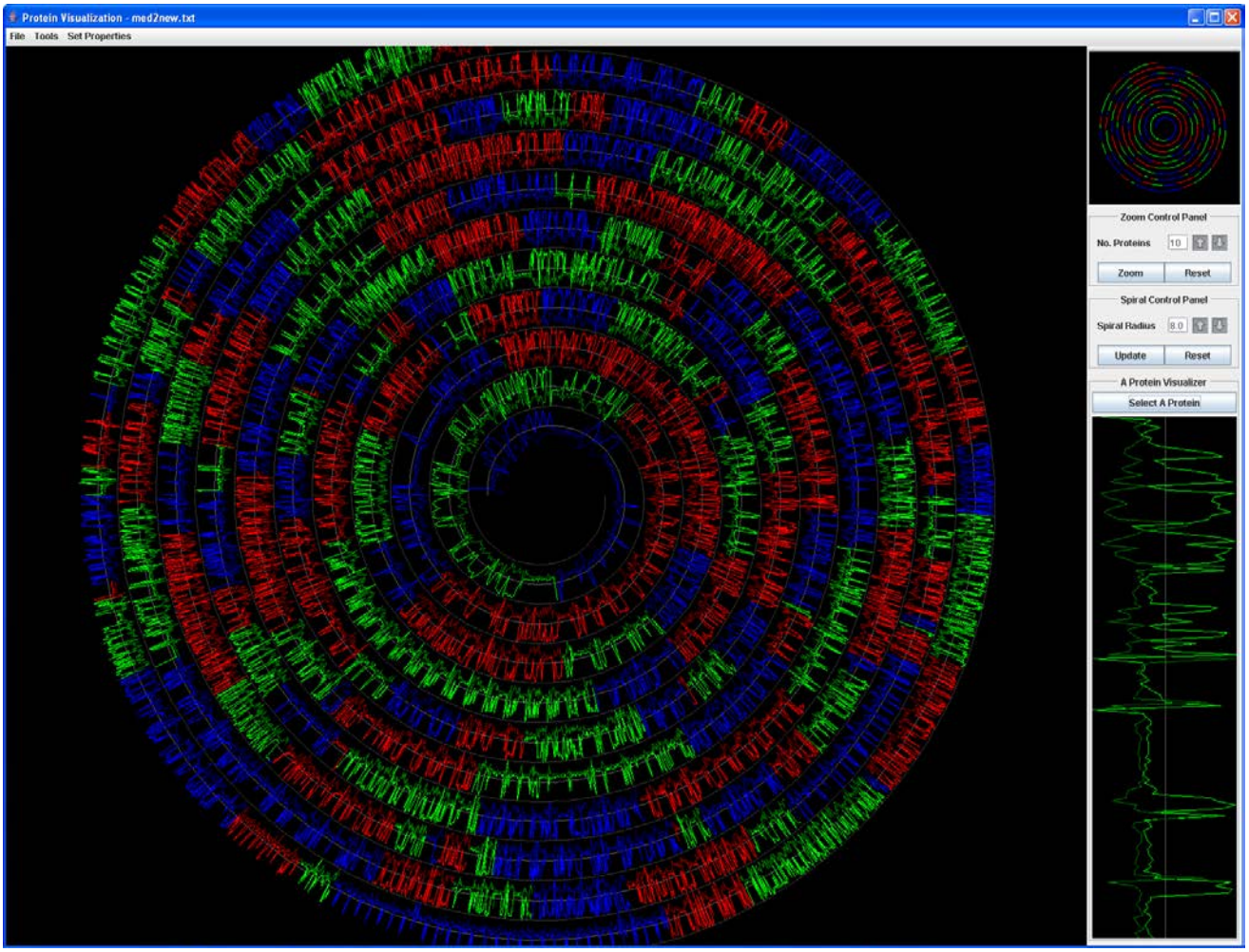

Figure 1. An example of the entire dihedral angles visualization.

4.1 Main view

The *main view* visualizes dihedral angle's value and clustered value of a selected number of proteins. The numbers of displayed proteins are conveniently selected from the 'Zoom Control Panel'. This visualization is able to display in detail a single protein as well as an entire view of several hundred of proteins regarding to the limitation of the display space. We use a concentric spiral line to visualize a large number of points from the dihedral angles sequence which aims to provide a space efficient, flexibility, as well as the visual continuity. Therefore, this method is capable of visualizing from a single protein to up to several hundreds of proteins depending on the screen resolution.

This panel provides the visualization based on not only the dihedral angles, but also the clustered results from section 3. We also use a rich graphic attribute to provide a clear visualization of each individual protein and its property. We next describe the technical detail of the *main view* visualization.

4.1.1. Point location

We use a concentric spiral, called Archimedes' spiral [14], to visualize the 1D sequence of points. Thus, at any point P with coordinate value (x, y) on this spiral line, this point (x, y) is defined by the formula:

$$\begin{cases} x = a\theta\cos(\theta) \\ y = a\theta\sin(\theta) \end{cases}$$

Where a is a constant and it defines the magnitude of spiral circles. This value can be interactively adjusted from the visualization. θ is the angle of the point P on the spiral line, in radian scale.

And the arc length s of the spiral line at the angle θ is calculated by the formula:

$$s = \frac{1}{2}a[\theta\sqrt{1+\theta^2} + \ln(\theta + \sqrt{1+\theta^2})]$$

In order to provide an equal distribution of points, the arc length or the distance between each two points is uniform. In addition, the calculation of the starting point is not at the center position in order to reduce the overcrowded at the center region.

Suppose that there is a sequence of points $\{P_1, P_2, P_3, \ldots, P_n\}$ with values of respectively $\{v_1, v_2, v_3, \ldots, v_n\}$. A value v_i of a point P_i can be defined by a dihedral angle or a clustered value of P_i. From the starting angle θ_0, the algorithm linearly calculates the position of each point. Suppose that at a point P_i ordered i in the sequence, we first find the nearest position $A(x_a, y_a)$ on the spiral line that the segment from the center point O to P_i cuts the spiral line. The point $A(x_a, y_a)$ is calculated linearly based on the previous point at the spiral line. As a result the point P_i is defined by the formula

$$\frac{\overrightarrow{AP_i}}{\overrightarrow{OA}} = \frac{v_i}{v_{\max}}\pi a$$

Where $v_{max} = max\{|v_1|, |v_2|, |v_3|, \ldots, |v_n|\}$, and a is the constant. Figure 2 shows an example of how to define the point P_i. Figure 3 shows an example of a spiral visualization showing the distribution of three proteins based on their dihedral angle's values.

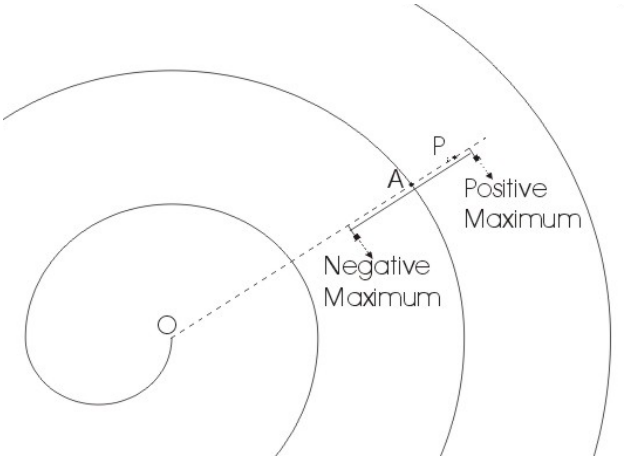

Figure 2. An example of a point location.

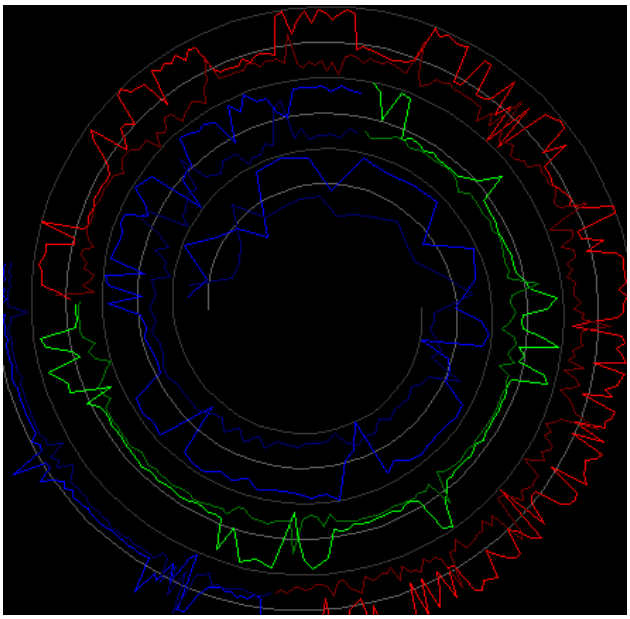

Figure 3. An example of a spiral visualization of a small dataset with three proteins.

4.1.2. Drawing properties

Each two consecutive dihedral angles have typically one positive value and one opposite negative value. The trend of these two sequences is quite similar. Therefore, we displays the two sets of *{0, 2, 4, …, 2n - 2}* and *{1, 3, 5, …, 2n – 1}* concurrently at two different sequences. At any time, one sequence is highlighted and the other is deemphasized using a darker color (see Figure 3). This property aims to reduce the overcrowded look of the visualization.

We use color attributes to provide a quick identification of each protein among the others. Nevertheless, too many colors might also cause the distraction of the view. And thus, only three standard colors (*red*, *green* and *blue*) were employed in our implementation in which the dihedral angles of each protein are orderly assigned to a color (see Figure 1 and Figure 4).

4.1.3. Clustering display

The *main view* provides not only the spiral display based on dihedral angle's values, but it also provides a *clustered view* and a *combined view*.

At the *cluster view*, the coordinate position of a point is calculated based on the normalized cluster value rather than its dihedral angle. Figure 4 shows the *clustered view* from three proteins where the closer to the spiral line the smaller clustered group of the point is. This figure indicates that most of dihedral angles of the first protein (in blue color) belong to cluster 1; most of dihedral angles of the second protein (in green color) belong to cluster 3; while the dihedral angles of the third protein (in red color) belong to either cluster 1 or cluster 3.

At the *combine view*, the position of a point is defined by its dihedral angle's value, and the clustered value is drawn as a small colored square. Each color is corresponding to a type of cluster. Nine clusters were found from the *FAÇADE* clustering algorithm, and thus they are mapped with nine colors respectively. These colors use different tone and brightness which makes them stand out from the default colors of the proteins. In addition, size of those colored squares can be adjusted to achieve the best view based on the user preference. Figure 5 shows an example of the *combine view* of the same three proteins. This view presents clearly the values of dihedral angle and the associative cluster.

Conclusion

We have presented a new technique for clustering and visualizing sequences of protein's dihedral angles of a large number of proteins as well as a few selected proteins. We apply *FAÇADE* clustering algorithm to discover the relevant dihedral angles for further pattern discovery analysis. The visualization combines three components including a *context view*, a *main view* and a *single protein view* which provide respectively a context view, a current focused proteins view, and a single proteins display. We use a concentric spiral coordinate system for positioning the dihedral angles along the spiral line. This visualization ensures the high capability of continuous and smooth display of a very long sequence of points depending on the screen resolution.

Although our work is still on progress, the initial result has shown its potential for visualizing continuously long sequences of dihedral angles up to hundred thousands points using a normal screen. Furthermore, this technique is also extendable for any one-dimensional sequences of different domains.

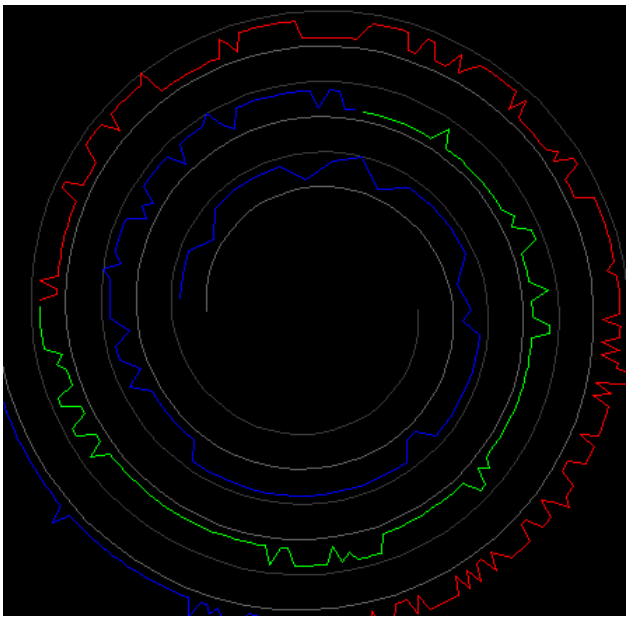

Figure 4. An example of the cluster view of dihedral angles of three proteins.

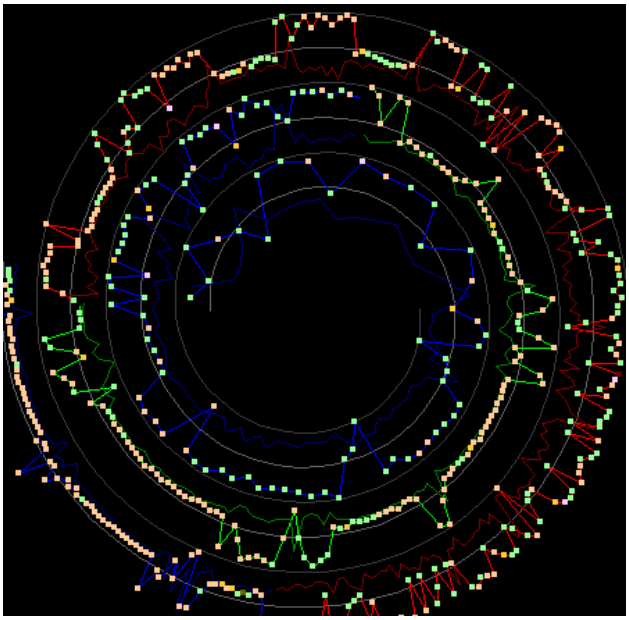

Figure 5. An example of the combined view of both dihedral angles and clustered values.

References

[1] O. Kreylos, N.L. Max, B. Hamann, S.N. Crivelli, and E.W. Bethel. Interactive Protein Manipulation. In Proceedings of *IEEE Visualization*. Application Track, 2003.

[2] F. Glaser, T. Pupko, I. Paz, R.E. Bell, D. Bechor, E. Martz, and N. Ben-Tal. ConSurf: Identification of Functional Regions in Proteins by Surface-Mapping of Phylogenetic Information. *Bioinformatics*. **19**(1), pp. 163-164, 2003.

[3] T. Can, Y. Wang, Y.F. Wang, and J. Su. FPV: Fast Protein Visualization Using Java 3D. *Bioinformatics*. **19**(8), pp. 913-922, 2003.

[4] C.A. Orengo, T.P. Flores, W.R. Taylor, and J.M. Thornton. Identification and Classification of Protein Fold Families. *Protein Engineering Design and Selection*. **6**, pp. 485-500, 1993.

[5] A.D. Michie, C.A. Orengo, J.M. Thornton. Analysis of Domain Structural Class Using an Automated Class Assignment Protocol. *Journal of Molecular Biology*. **262**, pp. 168-185, 1996.

[6] N.S. Boutonnet, A.V. Kajava, M.J. Rooman. Structural Classification of Alphabetabeta and Betabetaalpha Supersecondary Structure Units in Proteins. *Proteins*. **30**, 193-212, 1998.

[7] R. Ghai, T. Hain, and T. Chakraborty, “GenomeViz: Visualizing Microbial Genomes”, *BMC Bioinformatics*. **5**, pp. 198, 2004.

[8] [8] N. Sato, and S. Ehira, “GenoMap. A Circular Genome Data Viewer. *Bioinformatics*, **19**(12), pp. 1583-1584, 2003.

[9] A.K. Jain, M.N. Murty, and P.J. Flynn. Data Clustering: a Review. *ACM Computing Surveys*. **31**(3), 264-323, 1999.

[10] B. King. Step-Wise Clustering Procedures. *Journal of Am. Stat. Assoc*. **69**, pp. 86-101, 1967.

[11] Y. Qian, G. Zhang, and K. Zhang. FACADE: A Fast and Effective Approach to the Discovery of Dense Clusters in Noisy Spatial Data, In *Proceedings of ACM SIGMOD 2004 Conference*. ACM Press, pp. 921-922, 2004.

[12] Sneath, P. H. A. and Sokal, R. R. *Numerical Taxonomy*. Freeman, London, UK, 1973.

[13] Ward, J. H. JR. Hierarchical Grouping to Optimize an Objective Function. *Journal of Am. Stat. Assoc*, **58**, pp. 236-244, 1963.

[14] [9] E.W. Weisstein, "Archimedes' Spiral", *From MathWorld - A Wolfram Web Resource*, http://mathworld.wolfram.com/ArchimedesSpiral.html (accessed June 18, 2005)