Mining Protein Contact Maps

Jingjing Hu[†], Xiaolan Shen[†], Yu Shao[‡], Chris Bystroff[‡], Mohammed J. Zaki[†] [*]
[†]Computer Science Department
[‡]Biology Department
Rensselaer Polytechnic Institute, Troy, NY 12180
huj5@cs.rpi.edu, xiaolan.shen@oracle.com
shaoy@rpi.edu, bystrc@rpi.edu, zaki@cs.rpi.edu

ABSTRACT

The 3D conformation of a protein may be compactly represented in a symmetrical, square, boolean matrix of pairwise, inter-residue contacts, or "contact map". The contact map provides a host of useful information about the protein's structure. In this paper we describe how data mining can be used to extract valuable information from contact maps. For example, clusters of contacts represent certain secondary structures, and also capture non-local interactions, giving clues to the tertiary structure.

In this paper we focus on two main tasks: 1) Given the database of protein sequences, discover an extensive set of non-local (frequent) dense patterns in their contact maps, and compile a library of such non-local interactions. 2) Cluster these patterns based on their similarities and evaluate the clustering quality. We show via experiments that our techniques are effective in characterizing contact patterns across different proteins, and can be used to improve contact map prediction for unknown proteins as well as to learn protein folding pathways.

Keywords

Protein Contact Map, Dense Patterns, Clustering required for Proceedings

1. INTRODUCTION

Bioinformatics is an emerging field undergoing rapid, exciting growth. This has been mainly fueled by advances in DNA sequencing and mapping techniques. The Human Genome Project has resulted in an exponentially growing database of genetic sequences, while the Structural Genomics Initiative is doing the same for the protein structure database. One of the grand challenges in bioinformatics is protein structure prediction, where one is interested in determining the 3D structure of a protein given its amino acid sequence. It is well known that proteins fold spontaneously and reproducibly to a unique 3D structure in aqueous solution.

Today we are witnessing a paradigm shift in predicting protein structure from its known amino acid sequence $(a_1, a_2, \cdots, a_n)$. The traditional or Ab initio folding method employed first principles to derive the 3D structure of proteins. However, even though considerable progress has been made in understanding the chemistry and biology of folding, the success of ab initio folding has been quite limited.

[*]This work was supported in part by RPI Exploratory Seed Grant, NSF CAREER Award IIS-0092978, and NSF Next Generation Software Program grant EIA-0103708.

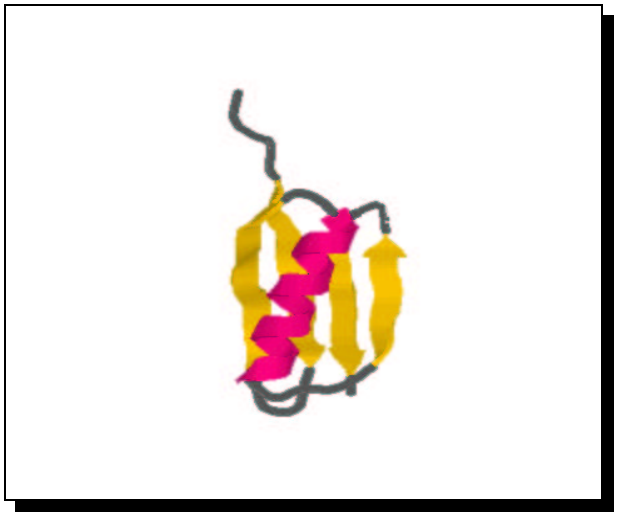
Instead of simulation studies, an alternative approach is to employ learning from examples using a database of known protein structures. For example, the Protein Data Bank (PDB) [2] records the 3D coordinates of the atoms of thousands of protein structures. Most of these proteins cluster into around 700 fold-families based on their similarities. It is conjectured that there will be on the order of 1000 fold-families for the natural proteins [8]. The PDB thus offers a new paradigm to protein structure prediction by employing data mining methods like clustering, classification, association rules, hidden Markov models, etc.

The ability to predict protein structure from the amino acid sequence will do no less than revolutionize molecular biology. All genes will be interpretable as three-dimensional, not one-dimensional, objects. The task of assigning a predicted function to each of these objects (arguably a simpler problem than protein folding) would then be underway. In the end, combined with proteomics data (i.e. expression arrays), we would have a flexible model for the whole cell, potentially capable of predicting emergent properties of molecular systems, such as signal transduction pathways, cell differentiation, and the immune response.

1.1 Protein Folding Pathway

Proteins are chains of amino acids residues. The early work of Anfinsen [1] and Levinthal [4] established that a protein chain folds spontaneously and reproducibly to a unique three dimensional structure when placed in aqueous solution. The sequence of amino acids making up the polypeptide chain contains, encoded within it, the complete building instructions. Levinthal also proved that the folding process cannot occur by random conformational search for the lowest energy state, since such a search would take millions of years, while proteins fold in milliseconds. As a result, Anfinsen proposed that proteins must form the structure in a time-ordered sequence of events, now called a "pathway". The nature of these events, whether they are restricted to "native contacts" (defined as contacts that are retained in the final structure) or whether they might include non-specific interactions, such as a general collapse in size at the very beginning, were left unanswered. Over time, the two main theories for how proteins fold became known as the "molten globule" or "hydrophobic collapse" (invoking non-specific interactions) and the "framework" or "nucleation/condensation" model (restricting pathways to native contacts only).

Over the years, the theoretical models for folding have converged somewhat, in part due to a better understanding of the structure of the so-called "unfolded state" and due to a more detailed description of kinetic folding intermediates. The "folding funnel" model [6] has reconciled hydrophobic collapse with the nucleation-condensation model by envisioning a distorted, funicular energy landscape and a "minimally frustrated" pathway. The view remains



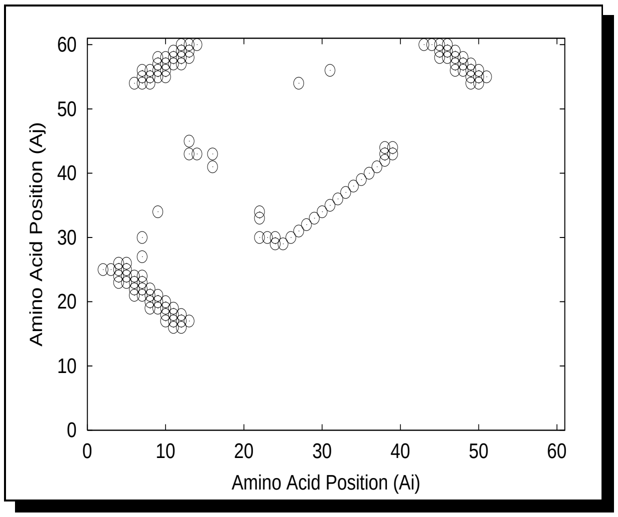


Figure 1: Left: 3D structure for protein G (PDB file 2igd, $N = 61$), Right: its contact map showing parallel (top left cluster) and anti parallel sheets (bottom left and top right cluster), and helix features (thin cluster close to main diagonal).

of a gradual, counter-entropic search for the hole in the funnel as the predominant barrier to folding.

1.2 Protein Contact Map

The 3D conformation of a protein may be compactly represented in a symmetrical, square, boolean matrix of pairwise, inter-residue contacts, or "contact map." The contact map of a protein (see Fig. 1) is a particularly useful representation of protein structure. The contact map provides useful information about the protein's secondary structure, and it also captures non-local interactions giving clues to its tertiary structure.

Two amino acids in a protein that come into contact with each other form a non-covalent interaction (hydrogen-bonds, hydrophobic effect, etc.). More formally, we say that two residues (or amino acids) a_i and a_j in a protein are in *contact* if the 3D distance $\delta(a_i, a_j)$ is at most some threshold value t (a common value is $t = 7\mathring{A}$), where $\delta(a_i, a_j) = |\mathbf{r_i} - \mathbf{r_j}|$, and $\mathbf{r_i}$ and $\mathbf{r_j}$ are the coordinates of the α-Carbon atoms of amino acids a_i and a_j (an alternative convention uses beta-carbons for all but the glycines). We define *sequence separation* as the distance between two amino acids a_i and a_j in the amino acid sequence, given as $|i - j|$. A contact map for a protein with N residues is an $N \times N$ binary matrix C whose element $C(i, j) = 1$ if residues i and j are in contact, and $C(i, j) = 0$ otherwise.

The contact map provides a host of useful information. For example, clusters of contacts represent certain secondary structures: α-Helices appear as bands along the main diagonal since they involve contacts between one amino acid and its four successors; β-Sheets are thick bands parallel or anti-parallel to the main diagonal (see Fig. 1). Tertiary structure may also be obtained by reverse projecting into 3D space using the MAP algorithm [7] or other distance geometry methods. Vendruscolo et al [7] have also shown that it is possible to recover the 3D structure from even corrupted contact maps. For predicting and characterizing the elusive global fold of a protein we are usually only interested in those contacts that are far from the main diagonal. In this paper we thus ignore any pair of residues whose sequence separation $|i - j| < 4$.

1.3 Contributions

In this paper we describe how data mining can be used to extract valuable information from contact maps. More specifically we focus on two main tasks: 1) Use contact maps to discover an extensive set of non-local dense patterns and compile a library of such non-local interactions. 2) Cluster these patterns based on their similarities and evaluate the clustering quality. We further highlight promising directions of future work. For example, how mining can help in generating heuristic rules of contacts, and how one can generate plausible folding pathways in contact map conformational space.

The protein folding problem will be solved gradually, by many investigators who share their results at the bi-annual CASP (Critical Assessment of protein Structure Prediction) meeting [5], which offers a world-wide blind prediction challenge. Here, we investigate how mining can uncover interesting knowledge from contact maps.

2. CHARACTERIZING CONTACT MAPS

Proteins are self-avoiding, globular chains. A contact map, if it truly represents a self-avoiding and compact chain, can be readily translated back to the three-dimensional structure from which it came. But, in general, only a small subset of all symmetric matrices of ones and zeros have this property. Previous work [9] has generated a method to output a contact map that both satisfies the geometrical constraints and is likely to represent a low-energy structure. Interactions between different subsequences of a protein are constrained by a variety of factors. The interactions may be initiated at several short peptides (initiation sites) and propagate into higher-order intra- or inter-molecular interactions. The properties of such interactions depend on (1) the amino acid sequence corresponding to the interactions, (2) the physical geometry of all interacting groups in three dimensions, and (3) the immediate contexts (linear, and secondary components for tertiary structural motifs) within which such interactions occur.

We describe below in detail the method that we use for mining frequent dense patterns or structural motifs in contact maps. All protein sequences used are from Protein Data Bank (PDB). Briefly, there are four major stages in our approach: (1) Mining dense patterns,(2) Pruning mined patterns, (3) Clustering the dense patterns, and (4) Integration of these patterns with biological data.

2.1 Mining Dense Patterns in Contact Maps

To enumerate all the frequent 2D dense patterns we scan the database of contact maps with a 2D sliding window of a user specified size. Across all proteins in the database, any sub-matrix under the window that has a minimum "density" (the number of '1's or contacts) is captured. For a $N \times N$ contact map (N is the length of the protein), using a 2D $W \times W$ window, there are $(N - W) \times (N - W)/2$ possible sub-matrices. We have to tabulate those which are dense, using different window sizes. We choose window sizes from 5 to 10 to capture denser contacts close to the diagonal (i.e., short-range interactions), as well as the sparser contacts far from the diagonal (i.e., long-range interactions).

Due to the intrinsic constraints in protein secondary and tertiary structures, the density of the contacts naturally decreases with chain separation distance (in the contact map, the distance from the main diagonal). In order to also capture these less dense but possibly significant patterns, we scale the minimum density cutoff as a function of the chain separation distance. The density weighing function we used is as follows:

$$min\_d = minDensity * (1 - (|i - j|)/N)$$

where *minDensity* is the user specified density threshold, i and j are the starting indices of a window in the 2D contact map (here it represents the top left position of a sub-matrix).

2.1.1 Counting Dense Patterns

As we slide the $W \times W$ window, the sub-matrix under the window will be added to a dense pattern list if its density exceeds the *min_d* threshold. However, we are interested in those dense patterns that are frequent, i.e., when adding a new pattern to the list of dense patterns we need to check if it already exists in the list. If yes, we increase the frequency of the pattern by one, and if not, we add it to the list initialized with a count of one.

The main complexity of the method stems from the fact that there can be a huge number of candidate windows. For instance, with a window size of $W = 5$, and for $N = 60$, we have 1485 windows per contact map. This translates to roughly 28 million possible windows for a database with 18,455 contact maps (equal to the number of proteins stored in the PDB database). Of these windows only relatively few will be dense, since the number of contacts is a lot less than the number of non-contacts. Still we need an efficient way of testing if two sub-matrices are identical or not. We assume that P is the number of current dense patterns of size $W \times W$. The naive method to add a new pattern is to check equality against all P patterns, where each check takes $O(W^2)$ time, giving a total time of $O(W^2 P)$ per equality check. A better approach is to use a hash table of dense patterns instead of a list. This can cut down the time to $O(W^2)$ per equality check if a suitable hash function is found. We will describe below how we can further improve the time to just $O(W)$ per check.

2.1.2 Counting Dense Patterns via Hashing

For fast hashing and equality checking, we will encode each sub-matrix in the following way: each row of the contact map, i.e., the $\{0, 1\}$ sequence, will be converted into a number corresponding to the binary value represented by the sequence, and all the numbers computed this way will be concatenated into a string. For example the 5×5 submatrix below is encoded as the string: 0.12.8.8.0.

```
submatrix   binary value of row
00000          0
01100          12
01000          8
01000          8
00000          0
stringId(concatenate row values) = 0.12.8.8.0
Hashing of a Dense Pattern
```

According to our sub-matrix encoding scheme, each dense $W \times W$ window M is encoded as the string $stringId(M) = v_1.v_2.\cdots.v_W$, where v_i is the value of the row treated as a binary string. For fast counting we will employ a 2-level hashing scheme. For the first level we use the sum of all the row values as the hash function:

$$h_1(M) = \sum_{i=1}^{W} v_i$$

The second level hashing uses the *stringId* as the hash key and therefore is an exact hashing, i.e., $h_2(M) = stringId(M)$. The use of this 2-level hashing scheme allows us to avoid many unnecessary checks. The first level hashing (h_1) narrows the potential matching sub-matrices to a very small number. Then the second level hashing (h_2) computes the exact matches. Computing h_1 and h_2 both take $O(W)$ time; thus the equality check of a sub-matrix takes $O(W)$ time.

After all dense areas are hashed into the second level slot, the support counts for each unique *stringId* of the dense patterns are collected, and those patterns that have support counts more than a user specified *minSupport* will be considered frequent dense patterns and will be output for further analysis.

2.2 Pruning

After obtaining mined patterns that are frequent and are relatively dense, we pruned them using a number of heuristics in order to extract biologically meaningful structural motifs. Due to the intrinsic characteristics and constraints of the secondary structures, alpha helices form contact patterns that line along the main diagonal of the contact matrix, whereas beta-sheets form contact patterns that are either perpendicular (anti-parallel beta sheets) or parallel (parallel beta sheets) to the main diagonal. The positions at which these patterns could occur are also constrained. In contrast, the contact patterns that belong to a tertiary structure (interactions between two secondary structural components) are more likely to be less dense and distant from the main diagonal. Furthermore, they do not have definitive contact shapes compared to the well defined secondary structure groups. Thus it is difficult to extract these and isolate them from other patterns. We took several approaches to attack the difficulty: first, as described in the previous section, we weighed the minimum density according to the distance of each sub-matrix to the main diagonal, such that distal regions have smaller density threshold than proximal regions; second, by varying window-size until an appropriate size is reached, we can differentiate the tertiary interactions from the rest by measuring the density.

The next step is to prune redundant patterns. As described above we used a sliding window scheme to capture all possible areas in a matrix, however, there are a few factors in the scheme that cause redundancy in the mined frequent dense patterns. For example, the following patterns identify the same non-local motif but are treated as different patterns because their string IDs are different:

```
00000    00000    00000
01000    00100    00010
01000 -> 00100 -> 00010
01000    00100    00010
00000    00000    00000
window slides to right by one position
```

We addressed this problem by recognizing the stringIDs for all horizontally- and vertically-shifted forms as equal.

After the pruning step, we generated the possible dense patterns with *minSupport* 1, i.e., the exhaustive set of dense patterns that appear in our database. We also varied the amino acid contact threshold while creating the contact map (recall that two amino acids are in contact if they are at most t distance apart in 3D; we used $t =$ 5,6 and 7 $\AA$ in our experiments). When using sliding window size less than 5, the dense patterns generated are trivial and didn't show enough structural meaning. With window size 6 and above, we generated only slightly more dense patterns than with window size 5. We consider 5 an important window size to generate existing dense patterns. In the following study, only data with sliding window size 5 will be listed. An example dense pattern with associated information is shown below (its support count is 5 and its volume, the number of 1's, is 10):

```
Sup:5 Str:0.28.12.15.1. Vol:10
00000
11100
01100
01111
00001

a dense pattern example
```

The numbers of non-redundant dense patterns extracted using different contact thresholds is shown in the second column of Table 1 (it also shows other clustering information which will be explained in the next section).

Contact Threshold	# Patterns	# Clusters	Cluster Quality
5 $\AA$	2508	83	0.8931
6 $\AA$	9929	99	0.8633
7 $\AA$	21231	367	0.8367

Table 1: Clustering of Dense Patterns

2.3 Clustering Dense Patterns

In the mining step, a large number of possible dense patterns are generated even after pruning. Instead of analyzing these non-local patterns directly it is beneficial to group them into groups of similar interactions. To characterize all the dense patterns that we have mined, clustering provides an effective way to obtain a gross view.

There are two main approaches to clustering. 1) Partition-based clustering tries to divide the data of N objects into k partitions or groups using heuristic search or iterative methods (e.g., k-means clustering). 2) Hierarchical clustering comes in two flavors. a) Agglomerative clustering technique starts with each object in its own cluster. At each step pairs of clusters with minimum distance between them are successively merged. b) Divisive clustering takes the opposite approach, it starts with all the records in one cluster, and then successively splits clusters into small pieces.

In this paper, we used agglomerative clustering to group the mined dense patterns to find the dominant non-local interactions, using the methodology described below.

1. *Calculating distance*: First, the distance between every pair of patterns is calculated using the formula:

$$Distance(M_i, M_j) = \sum_{k=1}^{W^2} |M_i[k] - M_j[k]| \qquad (1)$$

where M_i and M_j are dense patterns, and k is the position in the $W \times W$ matrix taken as a linear array (top left corner is position 0 and bottom right is $W \times W$). Thus $M_i[k]$ is either 0 or 1, indicating a non-contact and contact, respectively. The smaller the distance between two patterns, the more likely the two patterns are similar to each other.

We also need to define the distance between two clusters, say c_1 and c_2. Let the size of c_1 be n and the size of c_2 be m patterns. Then the distance between the pair of clusters is given as: $\sum_{i=1}^{n}\sum_{j=1}^{m} Distance(M_i, M_j)$ (with pattern $M_i \in c_1$ and $M_j \in c_2$), i.e., the sum of all pair-wise distances between patterns in a cluster.

2. *Clustering*: Before we start the clustering, we need to determine a threshold distance for a cluster, namely, the maximum average distance among the patterns in one cluster. Once this is done, the procedure is as follows: 1) Compare all pairs of clusters and mark the pair that is closest. 2) The distance between this closest pair of clusters is compared to the threshold value. If the distance is less than the threshold distance, these clusters become linked and are merged into a single cluster. Return to Step 1 to continue the clustering. If the distance between the closest pair is greater than the threshold, the clustering stops. If the threshold value is too small, there will still be many groups present at the end, and many of them will be singletons. Conversely, if the threshold is too large, objects that are not very similar may end up with the same cluster. We used distance 4 as the threshold for clustering.

```
Cluster No.1, Count = 59
Contact Probabilities:
0:0.05  1:0.05  2:0.68  3:0.85  4:0.71
5:0.03  6:0.02  7:0.14  8:0.07  9:0.09
10:0.05 11:0.05 12:0.12 13:0.09 14:0.03
15:0.03 16:0.05 17:0.15 18:0.27 19:0.85
20:0.25 21:0.10 22:0.59 23:0.92 24:0.83

Representative contact pattern:
         00111
         00000
         00000
         00001
         00011
```

After the agglomerative clustering step, for each cluster, we need a way to compactly describe the dominant interactions represented by all members of the cluster. For this we calculated the contact probability at each of the $W \times W$ positions in the submatrix. Assume that there are n patterns grouped in cluster c. Contact probability at position k is defined as the ratio of the number of contacts at that position divided by the cluster cardinality, and is given as $p_k^c = (1/n) \times \sum_{i=1}^{n} M_i[k]$. Based on these probability values, a representative contact pattern is generated for each cluster. In a representative contact pattern, we record a '1' at position k whenever p_k^c is greater than some probability threshold r and a '0' otherwise. An example cluster is shown below with associated information. Count is the number of patterns in the cluster and the notation 0:0.05 means that the probability of contact at position 0 is 0.05. The representative contact pattern for the cluster with a probability threshold $r = 0.65$ is also shown.

The number of clusters generated using different amino acid contact threshold are listed in Table 1, Column 2 (with clustering threshold of 4 and window size 5). For instance at $6\AA$ contact threshold we obtained 99 clusters from the 9929 mined patterns.

3. *Evaluating Clustering Quality*: After clustering is finished, we need a method to evaluate how effective it is. One way is to define an objective notion of clustering quality. While this may be hard in general, the contact probabilities for a cluster gives a good indication about how good the cluster is. For example, a cluster with very high values at some positions and very low values at some positions is a good cluster, while a cluster that has contact probabilities close to 0.5 is not very good. In other words, if a majority of the cluster members agree on a position (mostly 0's or mostly 1's), that indicates a good clustering.

   We use the formula below to generate the sum of contact probabilities at each position in the window within a cluster c:

$$S_c^1 = \sum_{k=1}^{W^2} p_k^c, \text{ if } p_k^c > 0.5 \tag{2}$$

$$S_c^0 = \sum_{k=1}^{W^2} (1 - p_k^c), \text{ if } p_k^c \leq 0.5 \tag{3}$$

   The quality of a cluster c is then given by the sum $Q_c = S_c^1 + S_c^0$. A high Q_c value close to 1 indicates a good cluster, while a value close to 0.5 indicates a poor cluster.

   The final clustering quality across all the clusters is given as the weighted sum of individual cluster quality values, as shown in the formula below:

$$Q = \frac{\sum_{i=1}^{|C|} |c_i| \times Q_{c_i}}{N}, (0.5 \leq Q \leq 1) \tag{4}$$

   where C is the set of clusters, $|C|$ is the number of clusters, $c_i \in C$ is one cluster in the set, $|c_i|$ is the numbers of patterns in the cluster, and N is the total number of patterns. Note how the clustering quality Q varies from 0.5 to 1, with a higher value suggests better clustering quality because it clusters patterns which share similar occurrence positions for '1's and '0's. A cluster with the same contact pattern has a $Q = 1$. The clustering quality corresponding to clusters generated in our experiments is listed in Table 1, Column 3 (with clustering threshold of 4 and window size 5). For example, given window size of 5, contact distance threshold of 6 $\AA$ and clustering threshold of 4, the clustering quality of the 99 clusters is 0.865753.

2.4 Integration and Visualization

After dense patterns are found and clustered, the final step is to incorporate the protein sequence/structure information with them. That is, for each dense pattern and its occurrences in the different contact maps (that is in different protein segments in PDB), we note the protein id, the start positions of the window (given as (X, Y) coordinates of the top left corner), and the type of interaction. This information is then used to visualize the mined patterns or interactions. An example of a dense pattern with associated information is shown below. This pattern with 11 contacts, occurs only once in PDB file with id $1vjs\_.1$, at position $(134, 109)$, i.e., it represents a non-local interaction between protein segment at positions 134-138 (the X axis) and the protein segment at positions 109-113 (the Y axis), in this case an interaction between two beta-strands.

```
Sup:1 Str:1.5.31.24.16 Vol:11

00001
00101
11111
11000
10000
pdb- x_start y_start interaction
1vjs_.1  134   109   beta strand-beta strand
```

2.5 Experimental Results

We used a non-redundant set of 2702 proteins from the PDB for our experiments. Preliminary distance maps for protein were produced based on the 3D coordinates of the α-Carbon atoms of each amino acid Based on these distance maps, binary contact maps were generated using several contact thresholds As described above, we discovered 9929 dense patterns when using a sliding window of size 5, maximum amino acid contact threshold of $6\AA$ and a minimum density of 0.125. When agglomerative clustering is applied, 99 clusters are generated using a clustering threshold of 4. The clustering quality is 0.8633. Two example clusters with their four associated patterns and corresponding interactions are given below:

Fig. 2 shows an example of the structures of four different patterns from one of the mined clusters. Beta strand interacting with beta strand is the dominant non-local motif in this cluster. The corresponding dense patterns are shown below:

```
  00011    00001    00010    00011
  00011    00101    00000    00101
  01111    11111    11000    11100
  11000    11000    10000    10000
  10000    10000    10000    00000

No.1355    3496     6282     7980
```

Fig. 3 shows an example of the structures of four different patterns from another cluster. Beta turn interacting with beta turn is the main motif in this cluster. The corresponding dense patterns are shown below:

```
  11010    01000    11000   11010
  01111    01110    01100   01110
  01000    01000    01110   01100
  01000    01000    01000   01100
  11000    11000    01000   01000

No.196      503     2834    8697
```

In other clusters, different dominant interactions were discovered. Fig. 4 shows some example interactions. These patterns can be further divided into sub-classes according to the number of contacts involved in each component, multiplicity of interacting atoms (one to one, one to many, or many to many), sequence specificities, and the linear/secondary structural contexts of the interaction.

These experiments shows that we efficiently clustered patterns according to their similarities both in sub-matrix level and structure level. With our clustering method, we can compile a library of possible dense patterns for further application in extracting valuable

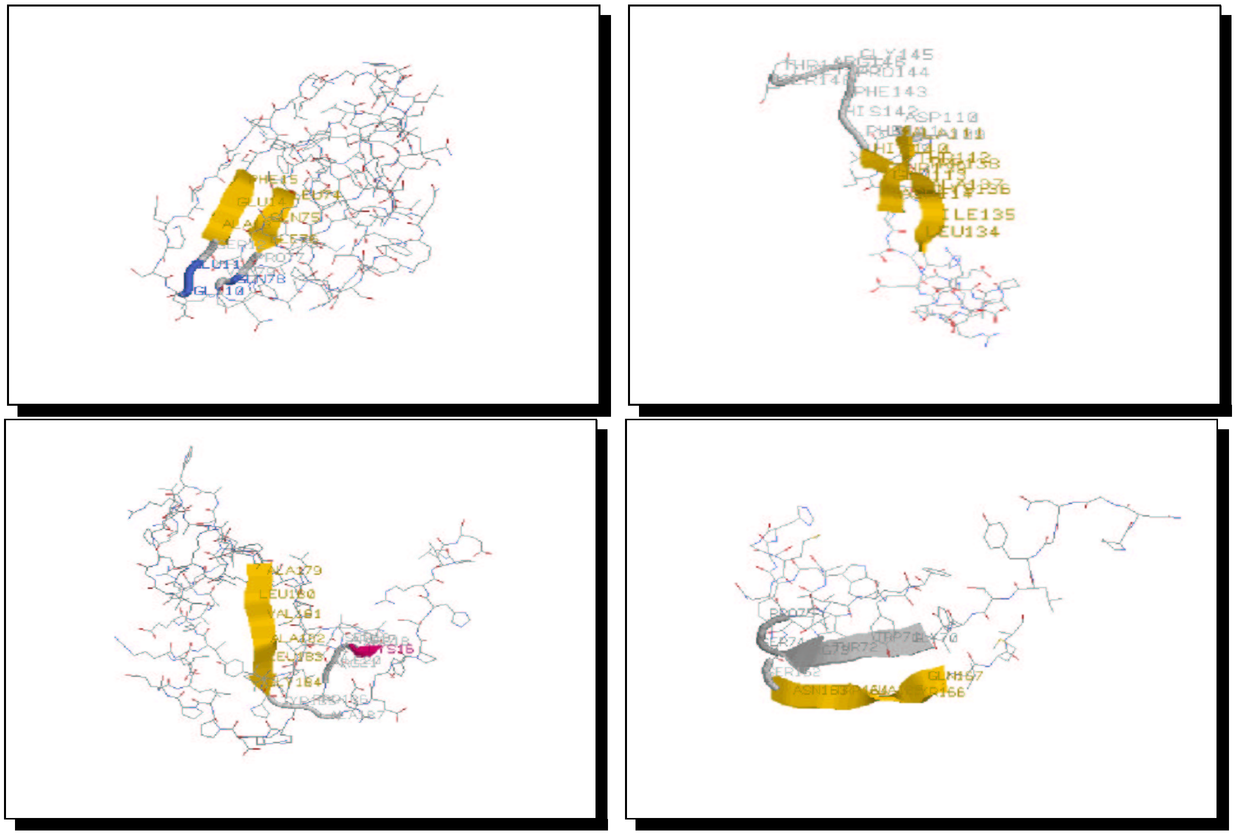


Figure 2: Secondary Structures of four different patterns from one cluster–Beta Strand vs. Beta Strand: Upper left: pattern 1355, Upper right: pattern 3496, Lower left: pattern 6282, Lower right: pattern 7980

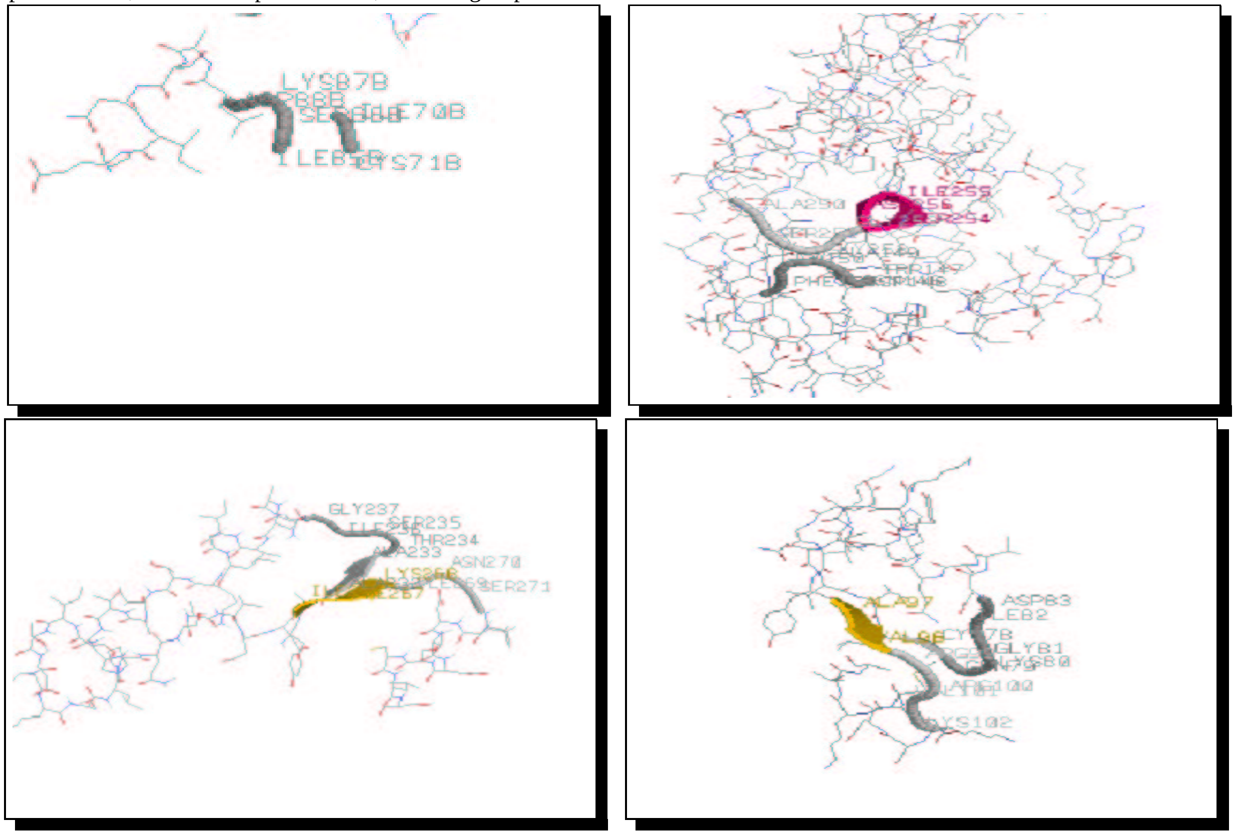


Figure 3: Secondary Structures of four different patterns from one cluster–Beta turn vs. Beta turn: Upper left: pattern 196, Upper right: pattern 503, Lower left: pattern 2834, Lower right: pattern 8697

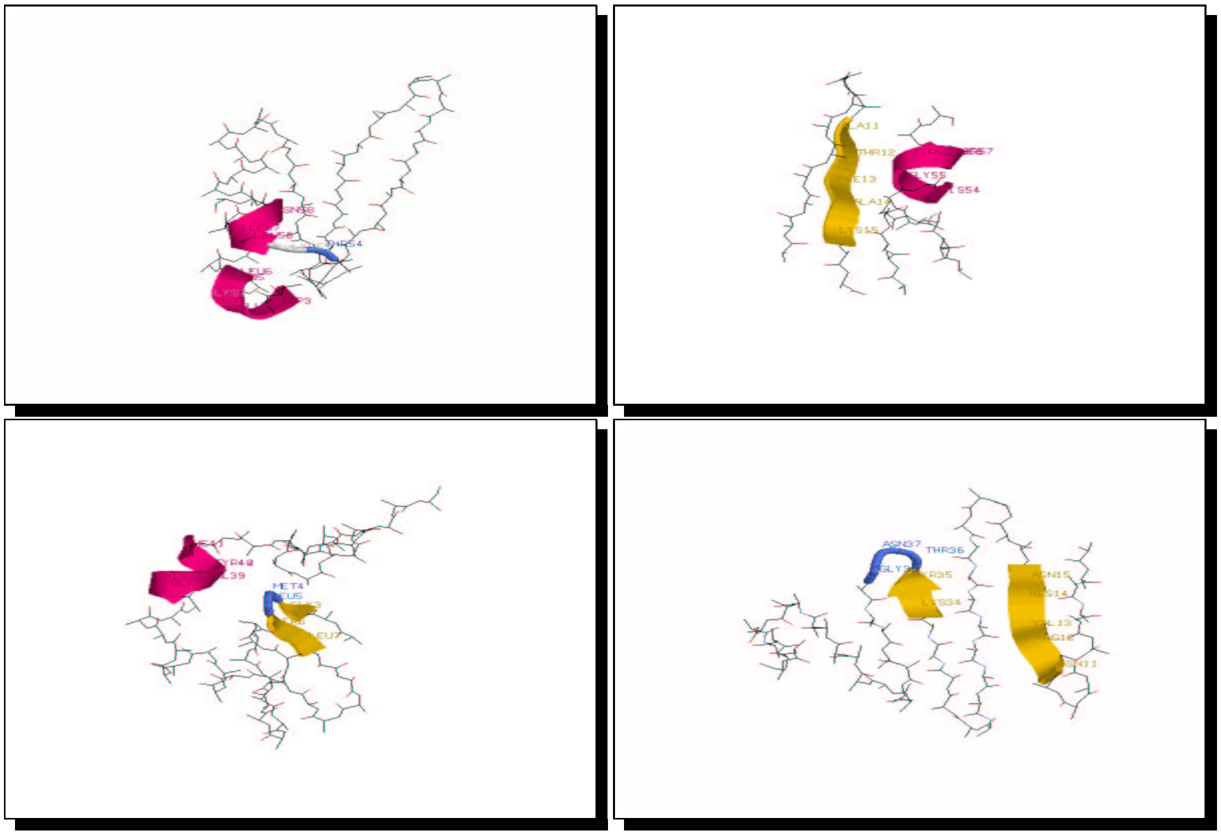


Figure 4: Frequent Patterns between Secondary Structures: 1) Alpha Helix - Alpha Helix 2) Alpha Helix - Beta Sheet, 3) Alpha Helix - Beta Turn, 4) Beta Sheet - Beta Turn

information to improve the accuracy of protein structure and pathway prediction. We are currently creating a library of all possible non-local interactions in “real” contact maps.

3. FUTURE DIRECTIONS

Many interesting questions still remain to be answered in the context of contact map mining. The ultimate goal would be to use the mined results for better structure prediction.

3.1 Improving Prediction of Contact Maps

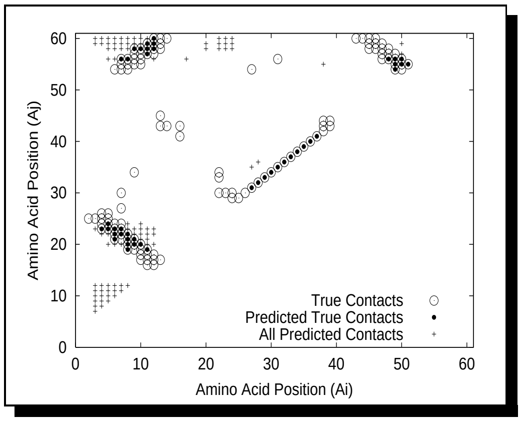


Figure 5: Predicted Contact Map (PDB file 2igd)

We applied a hybrid method based on hidden Markov Models and association rule mining to predict the contact map for a given protein sequence (see [9] for details). Fig. 5 shows the predicted contact map for the protein $2igd$ from Fig. 1. We got 35% accuracy and 37% coverage for this protein. The figure shows the true contacts, the contacts correctly predicted, and all the contacts predicted (correctly or incorrectly). Our prediction was able to capture true contacts representing portions of all the major interactions. For example, true contacts were found for the alpha helix, the two anti-parallel beta sheets and the parallel beta sheet. However, some spurious clusters of contacts were also discovered, such as the triangle in the lower left corner or the block of contacts in top left and middle regions of the contact map. Using the extensive library of non-local motifs, one can eliminate such false contacts by recognizing the fact that they never occur in real proteins, and thus these blocks of contacts are physically impossible. In future work we will describe the effectiveness of this post-processing approach (by filtering out physically impossible blocks) in improving the prediction of contact maps.

3.2 Mining Heuristic Rules for “Physicality”

Simple geometric considerations may be encoded into heuristics that recognize physically possible and protein-like patterns within contact maps, C. For example, we may consider the following to be rules that are never broken in true protein structures: a) If $C(i,j) = 1$ and $C(i+2, j+2) = 1$, then $C(i, j+2) = 0$, and $C(i+2, j) = 0$. b) If $C(i+2, j) = 1$ and $C(i, j+2) = 1$), then $C(i,j) = 0$, and $C(i+2, j+2) = 0$. These rules encode the observation that a beta sheet (contacts in a diagonal row) is either parallel or anti-parallel (respectively), but not both.

Another example may be drawn from contacts with alpha helices: If $C(i, i+4) = 1$ and $C(i, j) = 1$ and $C(i+4, j) = 1$, then $C(i+2, j) = 0$. This follows from the fact that $i+2$ lies on the opposite side of the helix from i to $i+4$, and therefore cannot share contacts with non-local residue j. Local structures may be used in the definition of the heuristics. For example, if an unbroken set of $C(i, i+4) = 1$ exists, the local structure is a helix, and therefore, for all $|j - i| > 4$ in that segment, $C(i, j) = 0$. The question is whether one can mine these rules automatically.

One approach is to discover "positional" rules, i.e., the heuristic geometric rules by considering an appropriate neighborhood around each contact $C(i, j)$ and noting down the relative coordinates of the other contacts and non-contacts in the neighborhood, conditional on the local structure type(s). For instance, consider a lower 1-layer (denoted LL1) neighborhood for a given point, $C(i, j)$. LL1 includes all the coordinates within $i+1$ and $j+1$, i.e. each point has 3 other points in its LL1 neighborhood, namely $C(i, j+1)$, $C(i+1, j)$ and $C(i+1, j+1)$. From the LL1 region around each point we obtain a database which can be mined for frequent combinations. Other rules can be found by defining an appropriate neighborhood and by incorporating sequence information. We are currently developing techniques to mine such heuristic rules of contact automatically.

3.3 Rules for Pathways in Contact Map Space

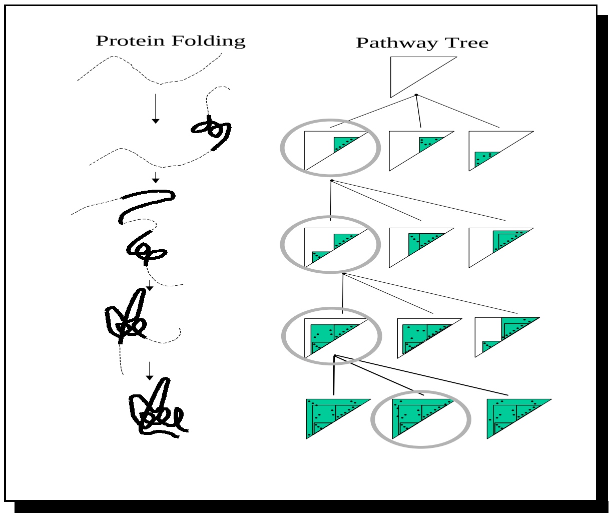


Figure 6: Folding Pathways in Contact Map

A pathway in contact map space consists of a time-ordered series of contacts. The pathway is initiated by high-confidence Initiation-sites [3], and thereafter it follows a tree-search format (Figure 6: triangles represent intermediate contact maps). We may impose a "condensation rule" onto our pathway model by assuming that any new contact must occur within S_{max} residues of a contact that is already formed. In other words we assume that $U(i, j) \leq S_{max}$, where $U(i, j)$ is the number of "unfolded" residues between i and j. Intervening residues are "folded" when a contact forms. Each level of the tree is the addition of a contact that satisfies the condensation rule. A maximum of k branches can be selected based on the energy. In addition, contacts that are not physically possible can be rejected, using the mined clusters of dense patterns or using the heuristics rules for physicality. Identical branches (same set of contacts, different order) can of course be merged.

We are currently developing methods to discover the folding pathways in the contact map space. It is worth observing that while the structure prediction problem has attracted a lot of attention, the pathway prediction problem has received almost no attention. However, the solution of either task would greatly enhance the solution of the other, hence it is natural to try to solve both of these problems within a unifying framework. Our current work is a step toward this unified approach.

4. CONCLUSIONS

In this paper we tackled two problems: discovering the extensive set of (non-local) dense contact patterns from the existing protein sequences and clustering the mined patterns. For the first problem, we developed a novel string encoding and hashing technique to extract all the dense submatrices by sliding a 2D window across the contact map. We discovered common (non-local) dense patterns using a dynamic density threshold and several pruning techniques. Using our approach we were able to extract some typical interactions that occur in existing proteins' contact maps. We compiled a library based on such non-local interactions of secondary structures. This library would be analogous to the I-sites library, but while the I-sites library records the common motifs for short contiguous segments (3-19 residues), the new library will record interactions between non-contiguous segments. For the latter problem, we used agglomerative clustering method and clustered patterns according to their similarities and evaluated our clustering quality. We believe that this pattern mining and clustering results are helpful for protein structure predictions and discovering protein folding pathway.

5. REFERENCES

[1] C. Anfinsen and H. Scheraga. Experimental and theretical aspects of protein folding. *Adv. Protein Chem*, 29:205–300, 1975.

[2] H. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. Bhat, H. Weissig, I. Shindyalov, and P. Bourne. The protein data bank. *Nucleic Acids Research*, 28:235–242, 2000.

[3] C. Bystroff, V. Thorsson, and D. Baker. HMMSTR: A hidden markov model for local sequence-structure correlations in proteins. *Journal of Molecular Biology*, 301:173–190, 2000.

[4] C. Levinthal. Are there pathways for protein folding? *J. Chem. Phys.*, 65:44–45, 1968.

[5] J. Moult, J. Pedersen, R. Judson, and K. Fidelis. A large-scale experiment to assess protein structure prediction methods. *Proteins*, 23(3):ii–v, 1995.

[6] B. Nolting, R. Golbik, J. Neira, A. Soler-Gonzalez, G. Schreiber, and A. Fersht. The folding pathway of a protein at high resolution from microseconds to seconds. *Proc. Natl. Acad. Sci. USA*, 94(3):826–830, 1997.

[7] M. Vendruscolo, E. Kussell, and E. Domany. Recovery of protein structure from contact maps. *Folding and Design*, 2(5):295–306, September 1997.

[8] Y. Wolf, N. Grishin, and E. Koonin. Estimating the number of protein folds and families from complete genome data. *Journal of Molecular Biology*, 299(4):897–905, 2000.

[9] M. Zaki, S. Jin, and C. Bystroff. Mining residue contacts in proteins using local structure predictions. *IEEE International Symposium on Bioinformatics and Biomedical Engineering*, November 2000.