Mining Complex Patterns from Protein Surfaces[*]

Lorenzo Baldacci, Matteo Golfarelli
DEIS – University of Bologna, Viale Risorgimento 2, 40136 Bologna – Italy
{*lbaldacci, mgolfarelli*}*@deis.unibo.it*

Abstract

In the domain of bioinformatics, the role played in the biological process by proteins, that act as transmitters and receivers of information thus ruling the mechanisms that determine how organic systems function, has great importance. Recent studies produced evidence of a strict correlation between the surface characteristics of proteins and the way they interact. In this paper we propose an original approach for discovering protein similarities based on their surface characteristics represented in terms of surface patterns. *The approach starts from a detailed representation of the protein surfaces and determines a set of characteristic regions that defines a compact representation of the protein surface that is the input for an ad-hoc data mining technique used to find the frequent patterns. Tests, carried out on a benchmark dataset of molecules with suitably designed surface mutations, show that surface patterns can be used to correctly classify groups of similar proteins.*

1 Introduction

Understanding the different functions carried out by proteins is a basic issue in domains such as medicine, pharmacology, and chemistry and is the most challenging task of structural biology that tries to classify proteins according to their structures. The main approaches to classification devised so far are *sequential alignment*, that applies string matching techniques to the primary structure [1], and *structural alignment*, that is based on both the secondary and tertiary structures, thus considering the overall tridimensional structure of the protein [2].

It is a widely accepted assumption that the overall structure deserves the appropriate distribution of chemical properties on the surface that the protein presents to its molecular target. Thanks to its ”appearance” the protein could have the correct approach with the target or not. However, two proteins with similar structures may be divergent in their sequence, thus playing different functions. The structural classification as a tool for the individuation of a common physiological role is misleading in this case and a surface classification is considered necessary. Up to now, a successful strategy has not yet been developed to reach such a general goal: the most commonly exploited approach, adopted for example in *biomolecular docking* [3], consists in adopting a surface patch already recognized to play a key role for a certain function and using it as a probe to explore the surfaces of all the proteins with known structures [4]. This method, based on local features, is not always effective, since the properties of the whole protein could be determined by a set of not necessarily neighboring regions; furthermore, it will fail in all cases characterized by highly adaptable surfaces and when portions of surface far from the active site have a dominant allosteric effect on the functional region of the protein.

In this paper we propose an original approach for mining the set of *complex surface patterns* that occur frequently in a database (DB) of proteins. Common patterns will drive the search of unknown relationships between proteins and in protein classifications. The approach, which has the advantage of being based neither on local surface features nor on already known functional meanings, takes in as input, as shown in Figure1, a detailed representation of the protein surfaces and requires (1) adopting clustering techniques to determine a set of characteristic regions; (2) defining a compact and effective representation of the protein surface; and (3) applying data mining techniques to these representations to find the frequent patterns.

The main contributions of this paper are the following:

- The introduction of a compact representation for protein surfaces, based on graphs of homogeneous regions rather than on single regions, and the associated definition of surface pattern (Section 2);
- An original mining algorithm to find out frequent surface patterns (Section 3).
- A set of tests proving the effectiveness of the approach (Section 4).

[*] This work was partially funded by the IST programme of the European Community, Future and Emerging Technologies under the IST-2001-33058 PANDA project.

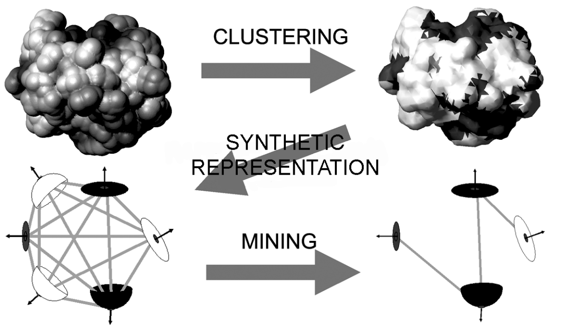

Figure 1. Approach overview: from protein surfaces to frequent patterns

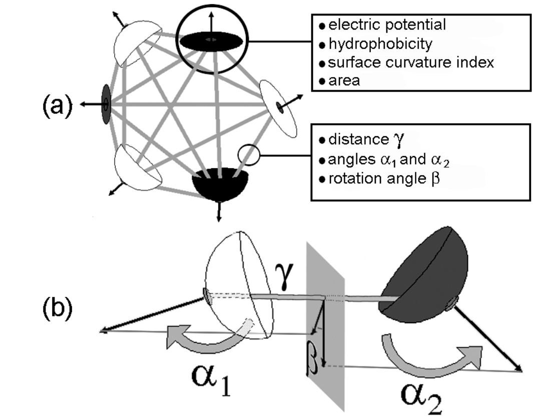

Figure 2. (a) Region graph for a protein; (b) Information used to compute the relative position of two regions

2 Surface Patterns

Let a collection of proteins be given; each one is associated with the vectorial representation of its surface in terms of triangular meshes, labeled with a set of local features such as electric potential and hydrophobicity. When studying the interactions between proteins, moving to a less detailed representation of the surface is encouraged not only by constraints on the computational complexity, but also by the following facts:

- *Extension of the interaction areas*. Both biologists and chemists agree that, normally, two proteins can interact only if the compatible surfaces cover at least 5% of the whole surface, while the meshes typically used cover an average area not larger than 0.01%.
- *Interaction mode*. During interaction, proteins show a high flexibility that allows the shape of their surface to be partially modified. Thus, exactly modelling those details that have no impact on interaction is useless.
- *Spatial information*. Each single triangular mesh does not carry the spatial information related to concavity and convexity, that instead characterizes larger regions and plays a basic role in interaction.
- *Homogeneity of features*. Interaction between proteins depends on regions showing homogeneous features.

These considerations suggest that a representation based on *homogeneous regions* of the surface, rather than on meshes, should be adopted. Such regions can be determined by means of clustering techniques based on the feature array associated to each single mesh. Discussing such techniques is outside the scope of this paper, thus we will assume a starting point a protein surface clusterized[1] in connected regions that are homogeneous with respect to a set of properties: in this work we considered electric potential, hydrophobicity, curvature and the size of the region.

The protein properties do not only depend on the set of regions characterizing their surface, but also on their relative positioning and orientation. The compact representation we propose for protein D_i is thus a completely connected graph whose nodes are the regions $d^i_1, \dots, d^i_m$ obtained by clustering, each described by the average value of features and by the area of the region (Figure 2.a). The arc connecting two regions d^i_1, d^i_2 expresses their relative position by means of: (1) the length γ of the vector connecting the barycenters of the two regions, (2) the angles α_1 and α_2 this vector forms with the versors of the regions, and (3) the angle β between the two versors, measured by projecting them on the plane orthogonal to the vector connecting them (Figure 2.b).

It is now possible to define a *pattern* P^i belonging to protein D_i as a completely connected subgraph of the graph that compactly represents its surface; we will call the *level* of a pattern the number of regions it comprises. Using a relative positioning system makes pattern matching easier since similarities can be computed independently of translations and rotations.

3 Mining Complex Patterns

The mining algorithm proposed in this section starts from the DB of proteins represented in their compact form, $\mathcal{D}$, to search for the frequent patterns including two or more

[1] The clustering process we adopted starts by estimating the geometric properties of the surface, which is carried out on the discrete mesh-based representation by computing indexes of average and Gaussian curvature on the surface points. Such indexes are then used together with the local features for determining, through a *boundary-based* approach [5].

```
for each $D_i \in \mathcal{D}$
{ $\mathcal{L}_1^i =$ GetFrequentRegions($D_i$);
  MineFrequentPatterns($\mathcal{L}_1^i$);
}
```

Figure 3. Main mining loop

regions. A pattern P is *frequent* if within $\mathcal{D}$ a set exists, whose cardinality is at least equal to a threshold $minsupp$, of patterns similar to P. This set is called the *support* of P. The similarity function should take into account both the local features of regions and their relative placement.

Finding frequent patterns whose level and composition are unknown generates an exponential search space. To this end, several techniques were proposed in the field of data mining; their applicability depends on the specific features of the domain [6]. In particular, our approach belongs to the level-wise class since it iteratively generates patterns made up of an increasing number of regions, however the problem we are facing presents the following specific aspects that make it impossible to directly apply the techniques known in the literature:

- *Similarity relationship between patterns* : the relationship used to compare patterns, and more specifically when determining the support, is not equality but similarity. Consequently, there is no set of reference regions, but rather each region is *unique* within the DB. Besides, similarity is not transitive, i.e. $P \sim Q \wedge Q \sim S \nRightarrow P \sim S$.
- *Presence of spatial constraints between the pattern regions*: part of the information that characterizes patterns is not associated to the single regions but to the arcs, i.e. patterns are also characterized by the spatial placement of regions.

Based on these considerations we can argue that *level-wise horizontal* algorithms (i.e. APriori) cannot be directly applied since the pattern supports cannot be computed simply by counting the “item labels”. On the other hand, even storing the pattern supports explicitly, as *level-wise vertical* algorithms do, is not sufficient due to intransitivity of similarity and consequently an access to the DB is still necessary. Finally, we emphasize that, despite the relevance of the information stored on the arcs, the problem does not need to be handled as subgraph mining [7] that requires automorphism to be computed thus making the problem untractable as soon as the pattern lengths increase [8].

Figure 3 shows the main loop of the proposed algorithm: procedure GetFrequentRegions(D_i) determines the set $\mathcal{L}_1^i$ of the single frequent regions (level-1 patterns) of protein D_i, while MineFrequentPatterns($\mathcal{L}_1^i$) triggers the true mining process. Though, on the one hand, the fact that the results of mining depend on what pattern is used as a prototype makes the problem computationally harder, on the other, it allows a larger number of additional data structures in MineFrequentPatterns, since the number of frequent patterns for each protein will obviously be small in comparison with that of the whole DB. Please note that the pseudo-code reported in Figure 3 is merely aimed at explanation: in fact, when implementing, groups of proteins can be processed at the same time in order to reduce the number of iterations, the size of the groups being a function of the available space in main memory.

```
1.  MineFrequentPatterns($\mathcal{L}_1^i$)
2.  { for ($k=2$, $\mathcal{L}_{k-1}^i \neq \emptyset$, $k++$) do
3.    { $\mathcal{C} =$ CandidatePatterns($\mathcal{L}_{k-1}^i$);
4.      for each $D_j \in \mathcal{D}$ do
5.        for each $P^i \in \mathcal{C}$; $D_j \in Prot(P^i)$ do
6.        { for each $Q^j \in Supp_j(P^i)$ do
7.            if Simil($Q^j$,$P^i$) $< \sigma$
8.              $Supp(P^i) \backslash = \{Q^j\}$;
9.        }
10.     $\mathcal{L}_k^i = \{P^i \in \mathcal{C}; \mid Supp(P^i) \mid \geq minsupp\}$;
11.   }
12. }
```

Figure 4. Mining algorithm

$\mathcal{D} = \{D_1, \ldots, D_n\}$	Protein database
$\mathcal{L}_i^k$	Frequent patterns of level k in protein D_i
$\mathcal{C}$	Set of candidate patterns
$P^i = (p_1^i, \ldots, p_k^i)$	Patterns of level k in protein D_i
$Supp(P)$	Patterns in the support of P
$Supp_j(P)$	Subset of patterns in $Supp(P)$ that belong to D_j
$Prot(P)$	Set of proteins to which at least one pattern in $Supp(P)$ belongs
σ	Similarity threshold for patterns

Table 1. Legend of the symbols used in the algorithm

Figure 4 reports the core of pseudo-code for the algorithm, while Table 3 summarizes the legend of symbols enabling its interpretation. We used uppercase letters from P to T to denote patterns, and the corresponding lowercase letters to denote their regions: in order to emphasize that a pattern/region belongs to protein D_i, it will be decorated with superscript i. The algorithm works iteratively by generating, at each step, the set $\mathcal{L}_i^k$ of the frequent patterns of increasing level k in protein D_i (row 2); the output of a step is the input of the next step. To increase performance, the algorithm works on a simplified representation of patterns, consisting of a sequence of pointers to the regions they include. For this reason, after the set $\mathcal{C}$ of potentially frequent patterns has been generated, the DB must be accessed to verify if the similarity constraint is actually met (row 7): function Simil(P,Q) loads P and Q from the DB and

```
1. CandidatePatterns($\mathcal{L}^i_{k-1}$)
2. { $\mathcal{C} = \emptyset$;
3.    for each $P^i, Q^i \in \mathcal{L}^i_{k-1}$;
            Mergeable($P^i, Q^i$)$\wedge$
            $\wedge \mid Prot(P^i) \cap Prot(Q^i) \mid \geq minsupp$
4.    { $T^i = (p^i_1, \cdots, p^i_{k-1}, q^i_{k-1})$;
5.       $Supp(T^i) = \{(r^l_1, \cdots, r^l_{k-1}, s^l_{k-1})$;
          $R^l \in Supp(P^i) \wedge S^l \in Supp(P^i) \wedge$
         $\wedge$Mergeable($R^l, S^l$)$\}$;
6.       if ($\mid Supp(T^i) \mid \geq minsupp$)
7.          $\mathcal{C} \cup = \{T^i\}$;
8.    }
9.    return $\mathcal{C}$;
10.}
```

Figure 5. Algorithm for generating candidate patterns

calculates their similarity, taking both the surface features and their spatial relationship into account. It is remarkable that the access to the DB is optimized by reading, exactly once, only the proteins that include at least one pattern belonging to one of the supports of candidate patterns (rows 4-6). The patterns whose support has cardinality higher than threshold $minsupp$ are inserted into the set of frequent patterns (row10).

The core of the algorithm is procedure CandidatePatterns (Figure 5), that generates the candidate patterns of level k by merging couples of patterns of level $k-1$.

The algorithm considers all possible couples of patterns in input (row 2) and carries out a pre-selection of candidates based on the constraints necessarily they must satisfy:

- *Region matching*: in order to obtain patterns of level k, two patterns that share $k-2$ regions must be merged. This constraint is verified within procedure Mergeable, whose code has been omitted for brevity. The procedure ensures non-redundant generation of patterns by posing a lexicographic ordering on regions.

- *Upper-bound of support cardinality*: if the number of proteins including patterns shared by the supports of the two generators is less than $minsupp$, the generated pattern cannot be frequent (row 3). Note that the value computed in this way is an upper bound since the actual matching of patterns within the protein is not checked. This check is computationally more complex, so it is carried out only for the patterns that passed the first check (rows 5-6).

The couples that meet the requirements above are merged (row 4) to generate a pattern of level k, and the corresponding support is computed (row 5). We explicitly represent the support, as typically done in vertical mining algorithms, to avoid accessing the DB even when generating candidates and to avoid computation of pattern isomorphisms. In this representation, we do not only represent the set of proteins that include a pattern similar to the one considered, but we directly specify the set of region sequences that makes them up, thus enabling verification of matching based on identifiers. Note, however, that accessing the DB is still necessary in MineFrequentPatterns, since the similarity between candidate patterns and their support, granted at level $k-1$, does not imply similarity at level k.

4 Tests

The tests we carried out are aimed at proving that surface patterns are useful to characterize the similarities between protein surfaces. For each couple of proteins similarity $Sim(D_i, D_j)$ is computed according to the number of shared patterns [9].

$$\frac{1}{k_{max}-1} \sum_{k=2}^{k_{max}} \frac{|\{P s.t. P \in \mathcal{L}^k_i \wedge Supp_j(P) \neq \varnothing\}|}{|\{P s.t. P \in \mathcal{L}^k_i\}|}$$

where k_{max} is the maximum level of patterns between D_i and D_j.

Our benchmark dataset is made up of 25 molecules obtained from the calmodulin-like protein (*Protein Data Bank* code 1cll) that has undergone two different chains of progressive surface mutations (introduced with the homology modelling technique) thus making mutants more and more different from the generating seed (labelled S). In the first chain the mutations applied are conservative (labelled c*_nr*) while in the second one they are non-conservative (labelled nc*_nr*), consequently the differences in the second chain will be more evident. Given the limited size of the dataset *minsupp* has been set equal to 2 and the mining algorithm discovered 270 patterns whose levels range from 2 to 4. Figure 6.(a) shows the similarities between proteins computed according to $Sim(D_i, D_j)$. It is evident that the similarity between the seed and a mutant decreases proportionally to the number of mutations, while the similarity between mutants that are close in the mutation chain is always high. On the other hand, similarity between proteins obtained in different chains is always low. Finally, the average similarity of the conservative mutations is higher, thus confirming that surface patterns are effective in representing surface similarities.

In order to further verify the effectiveness of the information stored in patterns we classified the proteins using the similarity function defined so far. We intentionally adopted a simple agglomerative hierarchical algorithm that, starting from clusters each containing one protein, iteratively joins the two most similar clusters. Cluster similarity has been

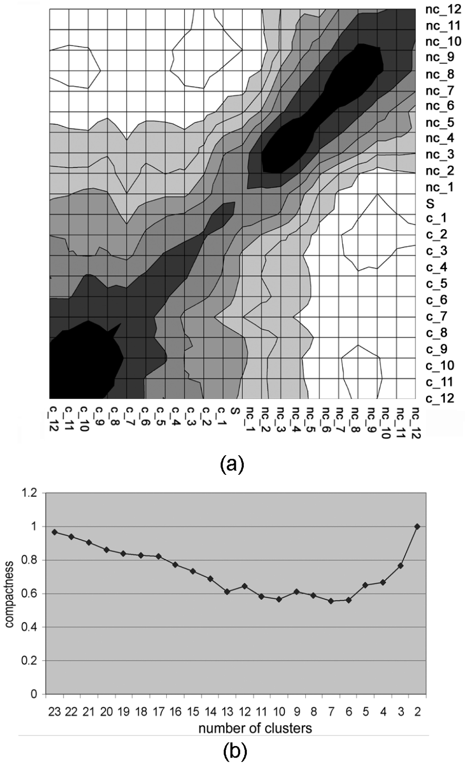

Figure 6. (a) Similarity map between couples of proteins in the dataset: darker colors mean higher similarity;(b) compactness of the clusters for different clustering levels

computed according to the complete linkage rule [10]. Figure 6.(b) shows the compactness of the clusters obtained at different levels of the clustering hierarchy. Assuming that an optimal clustering should divide the chains to fragments of sequential mutations, compactness is computed as the average density of each cluster with respect to the length of the corresponding fragment. The number of errors is particularly low for large clusters showing that the algorithm successfully creates a coarse-grained classification while it is more subject to errors when a higher number of classes (clusters) are present. It should be noted that the algorithm never creates clusters containing proteins from different chains.

5 Conclusions and Future Work

In this paper we described an approach for determining recurrent patterns on protein surfaces. Patterns model similarities between protein surfaces and can drive the search of unknown relationships between proteins as well as that for classification. This approach is the first attempt, known in the literature, to determine the similarities between proteins using global surface features and without exploiting any known functional meanings. The tests carried out show the effectiveness of the information extracted from the surfaces in classifying proteins and clear the way to a huge number of applications. Our future work includes the study of more sophisticated classification algorithms handling local maxima of the similarity function (that are at the origin of errors in classification) and overlapped classifications. As for applications, we will investigate cases in which protein functions are related to the overall surface characteristics, and when this type of analysis can overcome or complete the structural and sequential ones.

References

[1] A. Sagliano et al. A FastA based compilation of higher plant mitochondrial tRNA genes. *NAR*, 26:154–155, 1998.

[2] N.N. Alexandrov and R. Luethy. Alignment algorithm for homology modeling and threading. *Protein Science*, 7:254–258, 1998.

[3] T. Srinark and C. Kambhamettu. An approach for 3d segmentation on multiresolution surfaces. In *Proc. Int. Conf. Intelligent Technologies*, Chiang Mai, Thailand, 2003.

[4] F. Ferrè et al. Surface: a database of protein surface regions for functional annotation. *NAR*, 32:240–244, 2004.

[5] Y. Zhang et al. A simple and efficient algorithm for part decomposition of 3-d triangulated models based on curvature analysis. In *Proc. Int. Conf. Image Processing*, Rochester, NY, 2002.

[6] J. Boulicaut and B. Jeudy. Mining free itemsets under constraints. In *Proc. Int. Database Engineering & Applications Symposium*, Grenoble, France, 2001.

[7] M. Kuramochi and G. Karypis. Frequent subgraph discovery. In *Proc. Int. Conf. Data Mining*, San Jose, CA, 2001.

[8] S. Fortin. The graph isomorphism problem. Technical report, Dept. of Computer Science, University of Alberta, Canada, 1996.

[9] B. Fung et al. Hierarchical document clustering using frequent itemsets. In *Proc. 3rd SIAM Int. Conf. on Data Mining*, San Francisco, CA, USA, 2003.

[10] A. K. Jain et al. Data clustering: a review. *ACM Comput. Surv.*, 31(3):264–323, 1999.