DISCO: A New Algorithm for Detecting 3D Protein Structure Similarity

Nantia Iakovidou, Eleftherios Tiakas, Konstantinos Tsichlas

DISCO: a new algorithm for detecting 3D protein structure similarity.

Nantia Iakovidou, Eleftherios Tiakas and Konstantinos Tsichlas
{niakovid,tiakas,tsichlas}@csd.auth.gr

Aristotle University of Thessaloniki,
54124, Greece

Abstract. Protein structure similarity is one of the most important aims pursued by bioinformatics and structural biology, nowadays. Although quite a few similarity methods have been proposed lately, yet fresh algorithms that fulfill new preconditions are needed to serve this purpose. In this paper, we provide a new similarity measure for 3D protein structures that detects not only similar structures but also similar substructures to a query protein, supporting both multiple and pairwise comparison procedures and combining many comparison characteristics. In order to handle similarity queries we utilize efficient and effective indexing techniques such as M-trees and we provide interesting results using real, previously tested protein data sets.

Keywords: protein structure similarity, indexing technique

1 Introduction

The three dimensional (3D) structure of a protein is directly related to its functionality and could probably even determine it. This is why the comparison of protein structures became such an extremely important problem in structural biology and other scientific areas as well, such as medicine, agriculture and industry. Hundreds of algorithms that perform protein structure comparison have been proposed so far, with a view to compare and group proteins by structural similarity or identify distant homologues of protein families. The main reason for using 3D comparison algorithms, instead of using for example sequence comparison algorithms, is to detect functional similarities between proteins with low sequence similarity ([1], [2]).

In this paper we present a novel algorithm that performs 3D protein structure comparison at the level of C-alpha atoms and aims at detecting similarity between a specific query protein and a set of one or more known protein structures. For this purpose we use a simplified framework for representing the protein structures inspired by [3]. In our framework each protein molecule is represented as a sequence of weights, which come up from the combination of two measures. If we suppose for example that n is the number of C-alpha atoms in the protein molecule, then what we compute is a) the n-1 distances between consecutive C-alpha atoms and b) the n-2 cosines of their associated turning angles, which

according to [3] consist a sufficient angle descriptor and no use of torsion angles is needed. Then we combine these two quantities n-1 times in order to produce a series of the hybrid measure w $(w_1, w_2, ..., w_{n-1})$. In this way, local secondary structural information is maintained and also the use of the DIStance and COsine measures, from which our algorithm's name (DISCO) is produced, ensures that the algorithm is rotation and translation independent.

On the next step, we divide this sequence of weights into overlapping subsequences, which correspond to all possible consecutive protein substructures of the molecule. After repeating the aforementioned procedure for all proteins, we make use of indexing techniques (M-trees) in order to store these data and search for similarities either between pairs of proteins (Pair-DISCO Algorithm), either between a particular query protein and a set of other proteins (Multi-DISCO Algorithm). The use of overlapping subsequences enables us to detect not only similar structures but also similar substructures (motifs) in protein molecules, without thinking about sequentiality, which means about the order of appearance of the substructures into the molecule. This is very important because it means that we can capture both local and global similarity of protein structures and also because it enriches our algorithm with flexibility.

In fact, the use of indexing technique in our approach helps us to construct a database of protein structures in order to efficiently support and handle similarity queries of a specific protein across the totality of proteins that are stored in the determinate database. As will be shown in the paper, the indexing method calculates accurate similarities without having access to the whole database. In this way, the searching procedure becomes more efficient and rapid providing also dynamic data handling.

The rest of the paper is organised as follows. Section 2 presents previous related work in this area. In Section 3 our proposed approach and its experimental evaluation are presented. Finally some conclusions and future work are provided in Section 4.

2 Related Work

Numerous methods for performing protein structural searches have been proposed in recent years. Some of them perform alignment in rigid structures between amino acids at the level of C-alpha atoms. In general, all protein structure alignment programs optimize some mathematical definition of structural similarity. A typical example of this category is the root mean square deviation measure (RMSD). Although these methods provide a pragmatic definition of structural similarity, they fail to capture similar structures with extensive conformation changes such as internal rearrangements, which means that these methods ignore the flexibility of the polypeptide chains ([3]).

Several other algorithms align proteins at the level of secondary structure elements (SSEs). Such kind of approaches require a criterion for assigning secondary structures to proteins, as belonging to helices, strands or loops or not being part of an SSE at all. Tableau-based methods generally belong to this

category, such as SA Tableau Search ([4]) and IR Tableau ([5]). The truth about SSE methods is that there is not an exact procedure of assignment and also opinions vary about the precise beginning and end of SSEs ([4]). Furthermore, using only SSEs means that regions of protein structures that are not defined as being part of an SSE are not used at all and this fact can lead to less sensitive results.

A parameter that is also taken into account during the study of 3D protein structures is sequentiality, which means subsequent amino acids in one protein must correspond to subsequent amino acids in the partner protein. The majority of methods follow this restriction while the number of methods that are non-sequential is still limited ([6], [7]). The drawback of sequential approaches is that they can decrease the possibility to discover evolutionary relationships ([8]) as new protein structures can arise from the combination and permutation of substructures of a protein ([9]).

Methods that do not perform any kind of alignment have also been proposed, such as [3] and [10]. These methods use various ways to represent the protein molecules for example as paths or vectors, in order to use this representation to identify candidate sets of structural neighbours. The drawback of these approaches is that they perform an approximate representation of the protein molecule and this can lead to loss of secondary structure information and less accurate results.

Certain other issues also arise when structural comparison of proteins is studied such as pairwise and multiple comparison. The result of a pairwise alignment or comparison procedure concerns two particular proteins, while the multiple comparison procedure gives similarity results between a certain protein and a list of other known proteins. Another term called 'dynamics-based-alignment' has also been recently introduced ([11]) and is intended to compare the dynamic motions of different proteins. Furthermore, methods that perform structural alignment between a model protein and the true known structure of the protein have also been proposed ([12]) and they are known as model comparison methodologies, but these last two subjects are out of the scope of the current work.

3 The Proposed Approach

3.1 Protein Data Representation Algorithm

Let P be a set of proteins, and let a protein $P_a \in P$, and its *length* $n = |P_a|$, which is defined as the total number of C-alpha atoms that it contains. For the protein P_a let $x[1...n], y[1...n], z[1...n]$ be the corresponding 3D coordinates of the C-alpha atoms that it contains in the Euclidean three dimensional space, following the order of the protein structure.

We compute the $n-1$ Euclidean distances between consecutive C-alpha atoms using the formula:

$$d_i = \sqrt{(x[i] - x[i-1])^2 + (y[i] - y[i-1])^2 + (z[i] - z[i-1])^2}$$

for $i = 2, ..., n$. Note that $d_1 = 0$.

At the following, we compute the $n-2$ convex angles $\theta_i \in [0, \pi]$ between the linear vector segments $\boldsymbol{u}$, $\boldsymbol{v}$ associated to the secondary structures of proteins, using the cosine formula:

$$\cos\theta_i = \frac{\boldsymbol{u} \cdot \boldsymbol{v}}{|\boldsymbol{u}||\boldsymbol{v}|}$$

where,

$$\boldsymbol{u} \cdot \boldsymbol{v} = (x[i-1] - x[i])(x[i+1] - x[i]) + (y[i-1] - y[i])(y[i+1] - y[i])$$
$$+(z[i-1] - z[i])(z[i+1] - z[i])$$
$$|\boldsymbol{u}| = \sqrt{(x[i-1] - x[i])^2 + (y[i-1] - y[i])^2 + (z[i-1] - z[i])^2}$$
$$|\boldsymbol{v}| = \sqrt{(x[i+1] - x[i])^2 + (y[i+1] - y[i])^2 + (z[i+1] - z[i])^2}$$

for $i = 2, ..., n-2$. Note that $\theta_1 = \theta_{n-1} = 0$.

We compute the convex angles between all the protein atoms sequentially according to the Protein Data Bank file format ([13]). In this way we can capture the structural characteristics of all the recorded side-chains of the proteins in their regular backbone structure.

For the protein P_a we normalize the calculated distances d_i and angles θ_i in the interval $[0, 1]$:

$$nd_i = \frac{d_i}{max_i\{d_i\}}$$

$$n\theta_i = \frac{\theta_i}{\pi}$$

and we calculate the following combined linear measure for the C-alpha protein atoms:

$$w_i = a \cdot nd_i + (1 - a) \cdot n\theta_i$$

for $i = 1, ..., n-1$, and for a selected real number $a \in [0, 1]$, which defines how much the distances or the angles will affect the final calculated weights w_i between the protein C-alpha atoms.

Finally, we construct the protein's data as a sequence of the weights w_i:

$$P_a \longrightarrow (w_1, w_2, ..., w_{n-1})$$

Henceforth, all proteins of P will be represented by this sequence of weights.

3.2 Data Indexing

Among the existing indexing schemes for metric spaces, we use the Metric-Tree (M-tree) ([14]) to index the protein data, as it combines both efficient query processing and dynamic data handling. The M-Tree is a balanced tree that can index objects with attributes in a metric space, compared by distance functions which satisfy the metric properties (symmetry, positivity, triangular inequality). A simple example of such an indexing is depicted in Figure 1.

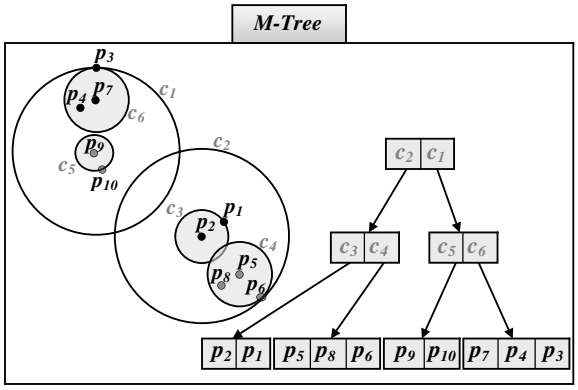

Fig. 1. Data Indexing with the M-Tree.

Using the M-tree, we have the advantage of inserting, updating and deleting dynamically new and old protein structures, and supporting efficiently nearest neighbors and range queries. Moreover, with the defined similarity (or distance) measures, which satisfy the metric space properties, we can support similarity queries between large protein structures, by partitioning the proteins into sub-proteins and using incremental nearest neighbor queries for ranking. The next subsections describe the proposed methodology for such similarity queries.

3.3 Top-k Protein-to-Protein Similarity Query (Multi-DISCO Algorithm)

Let P be a set of proteins. Each protein $P_i \in P$ is divided into $|P_i| - T + 1$ sub-proteins , with length T, which are indexed in an M-tree file. Let us consider an example. If a protein P_i has length 10, depicted by the series [1 2 3 4 5 6 7 8 9 10] and $T = 3$ then the set of $|P_i| - T + 1$ sub-proteins that our algorithm will create will be the following: ([1 2 3], [2 3 4], [3 4 5], [4 5 6], [5 6 7], [6 7 8], [7 8 9], [8 9 10]). We are interested in ranking all proteins in comparison with a given query protein P_q, and to retrieve the best top-k results. The idea is as follows.

For any sub-protein S_j of P_q we retrieve incrementally its nearest neighbor sub-proteins. For that purpose we use an *M-tree cursor*, which we initialize to the first nearest neighbor of S_j, and we retrieve the next nearest neighbors of S_j incrementally. An M-tree cursor (which has been implemented in the M-Tree project ([15])), provide sorted access from a specific object to the nearest neighbor order from that object and retrieve incrementally its nearest neighbors one by one. All existing sub-proteins into the M-tree will have a specific nearest neighbor order position from S_j. However, we do not leave the M-tree cursor to scan the entire indexed data-set by retrieving all nearest neighbors. When we retrieve at least one sub-protein from all existing proteins, we stop scanning the nearest neighbor order with the cursor. For any retrieved sub-protein we record its nearest neighbor order position as a ranking value of its corresponding

protein. We repeat this process for any existing sub-protein S_j of P_q and we keep the ranking results in a table. Then, the final rankings of the proteins are computed in another table R by taking the minimum or the maximum or the average of the recorded corresponding positions. The top-k similar proteins are extracted from the table R after an ascending sort. It is important to mention here that we can tune the parameter T, which is the length of each sub-protein, in order to have small or large sub-proteins.

3.4 Pairwise Similarity Query (Pair-DISCO Algorithm)

The pairwise similarity query is performed in a straightforward fashion. The idea is to take any sub-protein $S1_i$ from the first protein P_1 and to compute its distance to any sub-protein $S2_j$ from the second protein P_2. Then, using a specific rule we compute the average of the minimum or maximum or average distances and return this value as a final score. When this score is close to 0, the proteins are similar. When this score is close to 1 the proteins are dissimilar. It is important to note that we retrieve the data of the proteins P_1, P_2, from the M-tree into memory, before the distance computations. This leads to a significant improvement in the performance of the query.

3.5 Motif Extraction (Sub-structure query)

As already mentioned before, we divide each protein $P_i \in P$ into $|P_i| - T + 1$ sub-proteins, with length T. In this way, motif extraction is achieved as an intermediate step of Multi-DISCO algorithm, where we detect at least one sub-protein from all existing proteins to be the most similar to the each time examinant sub-protein. Because sub-structures of proteins are overlapping, for example if the first sub-structure is [1 2 3 4 5 6 7 8 9 10] then the second one will be [2 3 4 5 6 7 8 9 10 11], we have ranking results for all possible sub-proteins of the molecule. An example is shown in Table 1. Let us consider a dataset of 20 proteins. The numbers in the first column represent the protein and its sub-protein respectively. The second column contains the most similar sub-proteins from all twenty proteins of the example dataset in a row. That is to say, the first number of the second column corresponds to the most similar sub-protein of the first molecule, the second number (which is 120) corresponds to the most similar sub-protein of the second protein and so on.

3.6 Experimental Evaluation

The best comparison results were provided using the following parameters: sub-protein length equal to the 20% of the average length of the protein dataset, the weight of the hybrid measure equal to 0.05 and using the min rule to produce the final score. These numbers were extracted after extensive experimental procedures, the results of which will not be presented in the current paper due to lack of space. On the other hand, the number of proteins $|P|$ contained in

1-1:	1 120 6 45 103 1774 182 921 62 38 50 125 29 8 12 13 7 49 20 9
1-2:	1 80 21 4 48 1028 212 450 88 23 14 78 11 36 13 28 3 10 34 5
1-3:	1 56 16 4 52 1016 147 423 122 34 58 43 18 27 19 20 11 10 25 3
1-4:	1 86 3 33 43 1143 174 298 11 101 4 7 10 8 23 52 13 32 39 20
1-5:	1 66 22 25 19 866 67 136 10 74 13 15 16 5 24 30 4 96 61 8
1-6:	1 59 23 13 21 1166 61 258 24 117 5 16 26 4 14 17 6 15 60 20
1-7:	1 29 24 60 46 1126 144 226 32 193 13 47 58 9 10 6 5 12 27 18
1-8:	1 8 17 108 65 1399 96 438 66 194 48 11 146 22 12 7 5 4 21 19
1-9:	1 4 9 51 75 1461 55 406 77 165 33 11 159 32 28 12 2 8 10 40
1-10:	1 2 11 35 25 1081 20 303 24 157 22 7 91 46 10 5 8 9 3 21
1-11:	1 7 4 6 23 650 8 248 18 144 63 2 38 70 20 10 11 3 13 32
1-12:	1 7 5 3 10 473 11 347 103 194 37 4 8 34 12 16 30 2 19 55
1-13:	1 11 6 5 9 271 15 260 104 230 47 3 10 8 14 4 36 2 24 34
1-14:	1 11 6 5 8 263 13 282 127 339 53 4 15 10 16 2 41 3 24 44
1-15:	1 26 2 6 29 357 58 289 63 345 16 4 10 48 24 7 19 3 21 23
1-16:	1 78 3 59 42 468 56 376 8 282 22 6 7 37 29 13 2 12 33 27
1-17:	1 61 3 67 37 509 26 402 6 104 22 10 7 34 14 32 2 19 15 17
1-18:	1 84 4 75 81 330 44 446 20 95 49 16 8 40 5 9 2 12 15 23
1-19:	1 35 2 58 141 160 57 143 41 28 27 10 38 21 8 7 3 4 77 6
1-20:	1 22 17 69 127 858 49 160 40 14 59 15 28 11 4 3 12 2 71 10

Table 1. Example of sub-structure (motif) query.

the dataset and the total number of sub-proteins N that were created after the processing of the dataset, vary according to the each time tested protein dataset.

It is already known from biology that proteins that belong to the same family (a superfamily or a sub-family) typically have similar three-dimensional structures and functions ([16]). We first used Pair-DISCO algorithm to compare pairs of proteins that belong to the same protein families and to the same protein subfamilies. Similarly we applied the algorithm to compare proteins that belong to different protein families. By extensive experimentation we set a threshold for Pair-DISCO algorithm to distinguish significant similar proteins from simply similar ones. In general, values near zero indicate similar proteins. Significant similarity though is detected when the value of Pair-DISCO result is under 0.05.

We then applied Pair-DISCO algorithm in a representative sequence-independent dataset ([17]), obtained by using the Protein Data Bank ([13]) and taking into account only structural criteria. Each structural protein family is equally represented in this dataset and also every chain within it has less than 30% sequence identity. Table 2 summarizes the results of the Pair-DISCO algorithm. The first two columns contain the PDB id of the compared proteins and the third column shows the obtained similarity score. The forth column uses the aforementioned defined threshold in order to discriminate the significant similarities.

In our results we can see that proteins with PDB ids 1onc and 7rsa, which correspond to proteins P-30 and RNase-A respectively, are not signed to be significantly similar. This happens because the first protein performs functions and activities that the second one does not possess, even though both of them

Protein1	Protein2	Pair-DISCO	Signif. Similar
1npx	3grs	0.0198731	YES
1onc	7rsa	0.0809089	NO
1osa	4cpv	0.0295322	YES
2cmd	6ldh	0.0510457	NO
1aba	1ego	0.0567986	NO
1eaf	4cla	0.0234494	YES
2sga	4ptp	0.0321825	YES
1aaj	1paz	0.0259216	YES
5fd1	2fxb	0.0753062	NO
1gal	3cox	0.0240907	YES
1tlk	2rhe	0.0306053	YES
1omf	2por	0.0508227	NO
8i1b	4fgf	0.0645641	NO
1mup	1rbp	0.0321935	YES
1arb	4ptp	0.0330052	YES
2pia	1fnr	0.0267139	YES
3cd4	2rhe	0.0690796	NO
2mnr	4enl	0.0190777	YES
2gbp	2liv	0.0325059	YES
1fxiA	1ubq	0.03846	YES

Table 2. Pairwise results of Pair-DISCO algorithm.

are part of the same protein family (RNase) ([18]). On the other hand, proteins with PDB ids 1gal and 3cox were found to be significantly similar and this is true because they are enzymes that belong to the same category (oxidases), which constitutes a subclass of the oxidoreductases protein family. They also perform a similar function which is to catalyze an oxidation-reduction reaction, involving molecular oxygen (O_2) as the electron acceptor.

We also tested the performance of Multi-DISCO algorithm on numerous datasets, results of which are available, but we mainly focused on groups of proteins that were previously used as test cases ([19], [1]). In particular, we considered two datasets of proteins consisting of globins and serine proteinases, two of the most extensively studied protein families in biology, as shown in Table 3. Table 4 contains some representative results that derive from the application of the algorithm to the aforementioned dataset. The query protein is the one with PDB id 3est that belongs to the serine proteinases family and the ranking results using the min rule are listed on column two. Observe that the algorithm ranks first correctly the protein with PDB id 3est, which is in fact the query protein. This happens because we include every time the query proteins in the examined datasets for reasons of verification. Our algorithm also ranks correctly in the first places, proteins that belong to the same family and particularly to the same sub-family with that of the query protein. Correspondingly, the algorithm ranks to the last places the proteins that belong to the globins family.

Data set		PDB codes
Globins	Set1	1hhoA, 2dhbA, 1hhoB, 2dhbB, 1mdb
	Set2	1hhoA, 2dhbA, 1hhoB, 2dhbB, 1mdb, 2lhb, 1hbg, 1eco, 2lh7
	Set3	1hhoA, 2dhbA, 1hhoB, 2dhbB, 1mdb, 1eco, 2lh7, 4vhb, 1dlw, 1dly, 1idr
Serine Proteinases	Set1	3est, 2pka, 1ton, 3rp2, 4ptp, 5cha, 1ppb
	Set2	3est, 2pka, 1ton, 3rp2, 4ptp, 5cha, 1ppb, 1sgt, 1arb, 2sga, 3sgb, 2alp, 2snv

Table 3. Globin and Serine Proteinase data sets.

Query Protein	Ranking Results
3est	3est, 4ptp, 5cha, 3rp2, 2pka, 1ton, 1ppb, 3sgb, 2alp, 2snv, 1sgt, 1arb, 2sga, 2lhb, 1hhoB, 2dhbB, 1mdb, 2dhbA, 1hhoA, 1dly, 1idr, 1dlw, 2lh7, 4vhb, 1hbg, 1eco

Table 4. Ranking results using the Multi-DISCO algorithm.

4 Conclusion

In this paper we proposed a methodology for detecting similarity among or between 3D protein structures using a robust indexing technique. Our proposed scheme allows both pairwise and multiple comparisons and also it is the first time that a residue-based method allows motif (substructure) queries, as well. Furthermore, the proposed algorithms are capable of capturing both global and local similarity by tuning the appropriate parameters and combine many attributes such as translations-rotations independence, non-sequentiality and flexibility.

The results of the paper constitute a first step in our research. As a future work we will include comparison of our methodology to other well-known algorithms. We also plan to investigate how our algorithms behave in a parallel system with multiple disks. Additionally, we plan to see how our algorithms could be implemented in distributed environments and in particular we aim at the Map-Reduce framework ([20]). In this way, we will be able to allow for a larger database that will maintain thousands of proteins and will support queries and updates with good response times.

References

1. Micheletti, C. and Orland, H.: MISTRAL: a tool for energy-based multiple structural alignment of proteins. Bioinformatics 25, 2663-2669 (2009).

2. Lichtarge, O., Sowa, M.E.: Evolutionary predictions of binding surfaces and interactions. Current Opinion in Structural Biology, 12, 21-27 (2002).
3. Zhi, D., Krishna, S., Cao, H., Pevzner, P., Godzik, A.: Representing and comparing protein structures as paths in three-dimensional space. BMC Bioinformatics, 7, 460-475 (2006).
4. Stivala, A.D., Stuckey, P.J., Wirth, A.I.: Fast and accurate protein substructure searching with simulated annealing and GPUs. BMC Bioinformatics, 11, 446-463 (2010).
5. Zhang, L., Bailey, J., Konagurthu, A.S., Ramamohanarao, K.: A fast indexing approach for protein structure comparison. BMC Bioinformatics 2010, 11, S46 (2010).
6. Yuan, X., Bystroff, C.: Non-sequential structure-based alignments reveal topology-independent core packing arrangements in proteins. Bioinformatics, 21, 1010-1019 (2005).
7. Kolbeck, B., May, P., Schmidt-Goenner, T., Steinke, T., Knapp, E.W.: Connectivity independent protein-structure alignment. BMC Bioinformatics, 7, 510-510 (2006).
8. Xie, L., Bourne, P.E.: Detecting evolutionary relationships across existing fold space. PNAS USA, 105, 5441-5446 (2008).
9. Fong, J.H., Geer, L.Y., Panchenko, A.R., Bryant, S.H. Modeling the evolution of protein domain architectures using maximum parsimony. Journal of Molecular Biology, 366, 307-315 (2007).
10. Budowski-Tal, I., Nov, Y., Kolodny, R.: FragBag, an accurate representation of protein structure, retrieves stuctural neighbours from the entire PDB quickly and accurately. PNAS USA, 107, 3481-3486 (2010).
11. Zen, A., Carnevale, V., Lesk ,A.M., Micheletti, C.: Correspondences between low-energy modes in enzymes: dynamics-based alignment of enzymatic functional families. Protein Science, 17, 918-929 (2008).
12. Ortiz, A.R., Strauss, C.E.M., Olmea, O.: MAMMOTH: an automated method for model comparison. Protein Science, 11, 2606-2621 (2002).
13. Berman, H.M. et. al. The Protein Data Bank. Nucleic Acids Research, 28, 235-242 (2000).
14. Ciaccia, P., Patella, M., Zezula, P.: M-tree: An efficient access method for similarity search in metric spaces. In: Proceedings of the 23rd International Conference on Very Large Databases (VLDB), pp. 426-435. Morgan Kaufmann, Athens, Greece (1997).
15. The M-tree Project, `http://www-db.deis.unibo.it/Mtree/`
16. Needleman, S.B., Wunsch,C.D.: A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of Molecular Biology, 48, 443-453 (1970).
17. Fischer, D., Elofsson, A., Rice, D., Eisenberg, D.: Assessing the performance of fold recognition methods by means of a comprehensive benchmark. In Pacific Symposium on Biocomputing, 300-318 (1996).
18. Mosimann, S.C., Ardelt, W., James, M.N.G.: Refined 1.7 A X-ray crystallographic structure of P-30 protein, an amphibian ribonuclease with anti-tumor activity. Journal of Molecular Biology, 236, 1141-1153 (1994).
19. Konagurthu, A.S., Whisstock, J.C., Stuckey, P.J., Lesk,A.M.: MUSTANG: A multiple structural alignment algorithm, Proteins: Structures, Function and Bioinformatics, 64, 559-574 (2006).
20. Gaggero, M., Leo, S., Manca, S., Santori, F., Schiaratura, O., Zanetti, G.: Parallelizing bioinformatics applications with MapReduce. In CCA-08: Cloud Computing and its Applications, Poster presentation, Chicago (2008).