Alignment of Protein Structures with a Memetic Evolutionary Algorithm

B. Carr, W. Hart[*]
Sandia National Laboratories
PO Box 5800, MS1110
Albuquerque NM87185
USA

N.Krasnogor, J. Hirst, E. Burke[†]
ASAP Group &
Computational Biophysics Group
University of Nottingham
Nottingham, NG72RD
United Kingdom

J. Smith[‡]
Intelligent Computer System Centre
University of the West of England
Bristol, BS161QY
United Kingdom

Abstract

CATEGORY: Real-World Applications

Structural comparison of proteins is a core problem in modern biomedical research. Identifying structural similarities is essential for the assessment of the relationship between structure and function in proteins, and structural comparison techniques play a key role in applications like rational drug design. In this paper we consider a technique for protein structure comparison known as the maximum contact map overlap problem. In this problem, the similarity between two protein structures is computed by aligning the proteins to maximize the number of shared contacts in their corresponding contact maps.

We present a new approach to this problem that uses a Multimeme evolutionary algorithm. The best solution found by our algorithm provides a lower bound on the value of the optimal structural alignment between the proteins. We have evaluated the Multimeme algorithm on a range of benchmark problems and compared with previous heuristics. We apply a linear programming method, which provides an upper bound, to assess the accuracy of our solutions. Our experiments show that the Multimeme evolutionary algorithm represents a significant improvement on the current state of the art in metaheuristics for this problem.

1 Introduction

Structural comparison of proteins is a central task in biomedical research. Identifying structural similarities can provide significant insights into the relation between structure and function in proteins. Reliable and efficient structural matching plays a key role in rational drug design and in assessing the quality of structure prediction methods. A variety of structure comparison methods have been developed, such as SCOP [14], DALI [6], and LGA [17, 18]. However, no one technique has proven robust across a wide range of applications.

One of the emerging approaches for solving this problem is to evaluate the *alignment (or overlap) of contact maps* between proteins [6, 8, 11]. In its simplest form, a contact map is a matrix of all pairwise distances within a protein's components [12, 7]; these components can be atoms, residues, etc, depending on the resolution of the model employed. The distances in a contact map typically are computed by considering either the distance between the C_α atoms in a pair of residues, or the minimum distance between *any* two atoms belonging to those residues. Thus a contact map provides a simple representation of a protein's native three dimensional structure.[1]

In this paper we reconsider the use of metaheuristics for the Contact Map Overlap (*Max CMO*) problem [11]. For this problem, the distances in the contact map are discretized to zero or one, depending on whether the pairwise distances between residues are within a specified threshold. Although this discretization would seem to be easier than aligning matrices with real values, the problem is in fact NP-complete [4, 5, 9]. We have previously proposed a

[*] {rdcarr,wehart}@sandia.gov

[†] {natalio.krasnogor,jhirst}@nottingham.ac.uk , ekb@cs.nottingham.ac.uk

[‡] james.smith@uwe.ac.uk

[1] A protein's native state is associated with its minimal free energy configuration. The biological function of a protein is achieved in this state.

rigorous approach to *Max CMO* [11]. This approach employs an integer programming (IP) formulation for *Max CMO*, which is solved using a branch-and-cut algorithm. The branch-and-cut algorithm uses a Linear Programming (LP) relaxation of the IP to produce the upper bounds, and a Genetic Algorithm (GA) is used to provide lower bounds at the branch nodes.

The aim of the present research is to investigate the use of more sophisticated evolutionary algorithms: Multimeme memetic evolutionary algorithms [9], which integrate multiple local search strategies with a standard evolutionary search. We employ the LP relaxation of the *Max CMO* IP to provide upper bounds on the quality of the alignment of two proteins' structures, and thus we can empirically evaluate the quality of the solutions that we generate. Further, we compare the results of the Multimeme algorithm with a standard GA as well as the LGA protein structure comparison algorithm.

2 The Maximum Contact Map Overlap Problem

2.1 0-1 Contact Maps

Although contact maps are generally represented as distance matrices, one way of simplifying this representation of a protein's structure is to define a contact as a pair of residues that are closer than a given threshold, θ. Typically, θ ranges between 2 and 9 Angstroms. This gives a 0-1 contact map, where the matrix has the form

$$S_{i,j} = \begin{cases} 1 & \text{if residue i and j are within distance } \theta \\ 0 & \text{otherwise} \end{cases} .$$

The advantage of this representation is that structural properties of proteins can be more easily visualized and compared [16, 15]. Figure 1 is the graphic representation of the 0-1 contact map for protein 1C7W shown in Figure 2(a).[2] In this figure, the α-helices are represented by wide bands along the main diagonal, while β-sheets manifest themselves as bands parallel or perpendicular to the diagonal.[3]

A 0-1 contact map can also be represented as an undirected graph. In this graph, each residue is a node and there exists an edge between nodes i and j if these residues are in contact (i.e. if $S_{i,j} = 1$). Figure 2 shows the native structures of two proteins, and Figure 3 shows the graphs corresponding to their contact maps. Note the long range interactions of residues that are far away in the sequence but close in the three dimensional structure adopted by the native state.

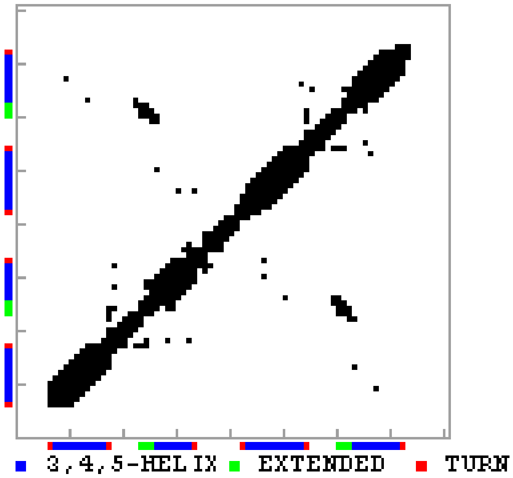


Figure 1: A 0-1 contact map comparing protein 1C7W with itself.

2.2 Problem Formulation

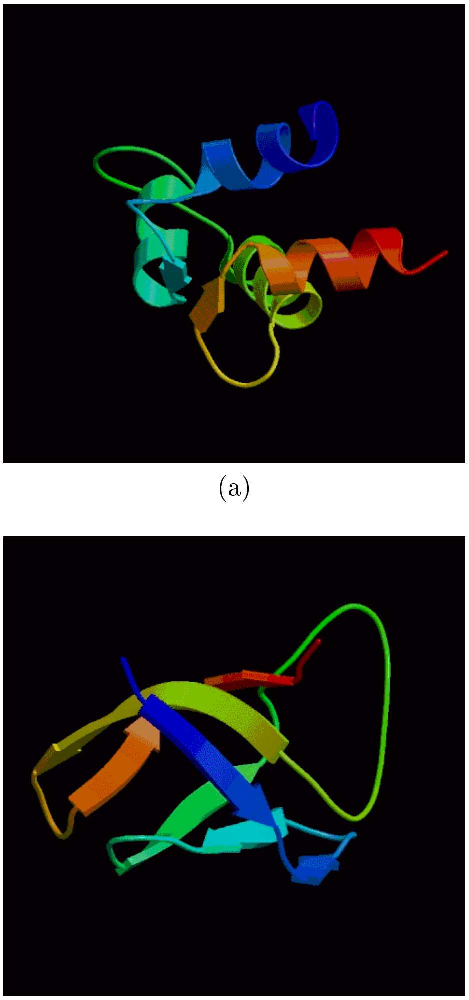
The *alignment* between two contact maps is an assignment of residues in the first contact map to residues on the second contact map. Residues that are thus aligned are considered equivalent. Further, consider a pair of contacts, one from each protein. We say that such a pair of contacts is equivalent if the pairs of residues that define the end-points of these contacts are equivalent. In the *Max CMO* problem, the value of an alignment between a pair of proteins is the number of equivalent contacts between these proteins. This number is called the *overlap* of the contact maps and the goal is to maximize this value. The *Max CMO* problem was first discussed by Godzik et al. [3], and it has been proven NP-complete [5, 9].

Lancia et al. [11] describe an IP approach for the MAX CMO problem, which builds upon a polynomial reduction from *Max CMO* to Maximum Independent Set (MIS). The size of the converted instances is the product of the number of contacts of the two maps (around 10000 nodes for a pair of proteins of 100 residues each). To solve MIS instances of this size, the authors exploit specific characteristics of the MIS instances.

Let $G_1 = (E_1, V_1)$ and $G_2(E_2, V_2)$ be the two graphs that correspond to two 0-1 contact maps, where E_i are the edges in these graphs and V_i the vertices. The IP

[2] The proteins used in this paper are taken from the Protein Data Bank [1], and the labels we use are the labels provided by the PDB.

[3] $\alpha-$helices and $\beta-$sheets are elements of a protein's secondary structure. See [2] for a description of protein secondary structure.



(a)

(b)

Figure 2: Ribbon representation of structures for proteins (a) 1C7W and (b) 1NMG. Arrow shows $\beta-$strand and spiral depicts helix.

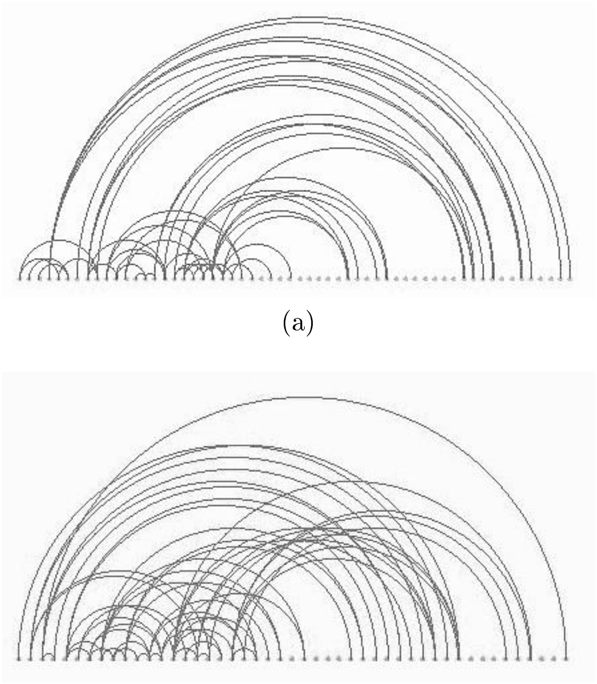
formulation proposed by Lancia et al. is:

$$\max \sum_{e\in E_1, f\in E_2} y_{e,f}$$

subject to the constraints

$$\begin{array}{ll}
x_{i,u}+x_{j,v}\leq 1 & \forall (i,u),(j,v) \text{ crossed} \\
\sum_{i<j} y_{(i,j)(u,v)} \leq x_{i,u} & \forall i\in V_1, (u,v)\in E_2 \\
\sum_{i>j} y_{(j,i)(u,v)} \leq x_{i,v} & \forall i\in V_1, (u,v)\in E_2 \\
\sum_{u<v} y_{(i,j)(u,v)} \leq x_{i,u} & \forall u\in V_2, (i,j)\in E_1 \\
\sum_{u>v} y_{(i,j)(v,u)} \leq x_{j,u} & \forall u\in V_2, (i,j)\in E_1 \\
x,y\in\{0,1\} &
\end{array}$$

The binary variable $x_{i,u}$ for $i\in V_1$ and $u\in V_2$ has a value of 1 if i is aligned with u and 0 otherwise. The binary variable $y_{e,f}$ has a value of 1 if the edges e,f are shared in a feasible solution and 0 otherwise.



(a)

(b)

Figure 3: Graphical representation of the contact maps of proteins (a) 1C7W and (b) 1NMG.

Hence the first equation is the statement of the goal of maximizing the shared edges (contacts). We say that (i,u) and (j,v) are *crossed* if both of these assignments are not feasible within a single alignment; these form crossed assignment lines in the alignment graphs below.

Lancia et al. [11] discuss the solution of this IP with a branch-and-cut algorithm. Note that if the last constraint in the IP is removed then this problem is an LP, so it can be solved in polynomial time. Further, the solution to this relaxation of the IP provides an upper bound on the globally optimal solution of the IP. These LP solutions are a critical element of the branch-and-cut algorithm described by Lancia et al. Further, they can be used to benchmark heuristic solvers like the EA we describe in the next section.

3 Multimeme Algorithms

Memetic algorithms [13] are evolutionary algorithms that include, as part of the "standard" evolutionary cycle of crossover-mutation-selection, a local search stage. They have been extensively used and studied on a wide range of problems. Multimeme evolutionary algorithms are introduced in Krasnogor et al. [10, 9]. The distinction between memetic and Multimeme algorithms is the use of a family of local searchers. A memetic algorithm employs a single local search heuristic, while a Multimeme algorithm relies on a set of simple local searchers. Multimeme algorithms self-adaptively select which heuristic to use for different instances, stages of the search or individuals in the population.

In a Multimeme algorithm an individual is composed of its genetic material (that represents the solution to the problem being solved) and its memetic material (that defines the kind of local searcher to use). The mechanisms of genetic exchange and variation are the usual crossover and mutation operators but tailored for the specific problem one wants to solve. Memetic transmission is done during crossover as follows. If the two parents use the same local searcher then the offspring will inherit that local searcher. However, if the local searchers are different then the offspring inherits the one associated with the fittest parent. Otherwise (the heuristics used by both parents are different but the fitnesses are the same) a random choice between both local searchers is made.

The rational behind this criterion is to propagate local searchers that are associated with fit individuals (as it is hoped that those individuals were improved by their respective local searchers). Also, during mutation, the meme of an individual can be overridden and a local searcher assigned at random (uniformly from the set of all available local searchers) with the probability specified by the innovation rate parameter.

4 A Multimeme Algorithm for *Max CMO*

We extend here the work on the *Max CMO* initiated by Lancia et.al. [11], who employed a standard GA with specially tailored genetic operators. We briefly describe those operators and explain how we enlarged that set for use in our Multimeme approach.

In a GA for *Max CMO* a chromosome is represented by a vector c of dimension n, for which each position can take values in the $[-1, \ldots, m-1]$ domain. Here, m is the length of the longer protein and n the length of the shorter. A position j in c, $c[j]$, specifies that the j^{th} residue in the longer protein is aligned to the $c[j]^{th}$ residue in the shorter. A value of -1 in that position will signify that residue j is not aligned to any of the residues in the other protein. Unfeasible configurations are not allowed, that is, if $i < j$ and $v[i] > v[j]$ or $i > j$ and $v[i] < v[j]$ (e.g. a crossing alignment) then the chromosome is discarded. That is, our algorithms work only with feasible solutions. It is simple to define genetic operators that preserve feasibilities based on this representation. Two-point crossover with boundary checks was used to mate individuals and create one offspring. Although both parents are feasible valid alignments, the newly created offspring can result in invalid (crossed) alignments. After constructing the offspring, feasibility is restored by deleting any alignment that crosses other alignments. Figure 4 shows a two point crossing over with an unfeasible intermediate offspring. At the later stage it is repaired and completed, i.e. all unassigned vertices are randomly assign to a vertex on the other protein if no new violations are produced (not shown in the picture).

The mutation move employed in the experiments is called a sliding mutation. It selects a consecutive region of the chromosome vector and adds, slides right, or subtracts, slides left, a small number. The phenotypic effect produced is the tilting of the alignments. In Figure 5 an example is shown. Again, alignments that violate the feasibility of the solution are discarded. Lancia et al. [11] describes a few variations on the sliding mutation.

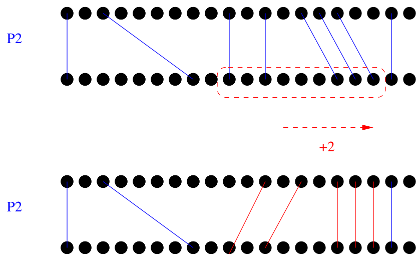


Figure 5: Sliding mutation under the vector representation for *Max CMO*. In this example a window size of 9 residues was chosen together with a right sliding by 2 residues.

In this paper we employ a Multimeme algorithm that, besides using the same mutation and crossover as the mentioned GA, has a set of 6 local search operators. Four of the local searchers implemented are parameterized variations of the sliding operator. The direction of movement, left or right sliding, and the tilting factor, i.e. the number added or subtracted, were chosen at random in each local search stage. The size of the window was taken from the set $\{2, 4, 8, 16\}$. Two new operators were also defined: a "wiper" move and a "split" move. The wiper move is depicted in Figure 6. At every iteration of the operator two alignments, represented by x and y in the lower protein of the picture, are chosen. The feasible regions of alignment for x and y are determined (marked with dotted line rectangles $R1$ and $R2$ in the graph). Subsequently all the residues within those regions are tested as candidate

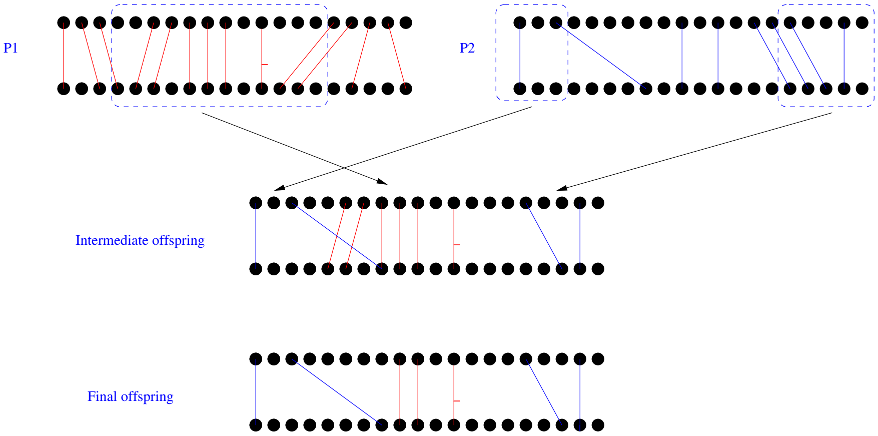


Figure 4: Two-point crossover with boundary checks for a vector representation of *Max CMO*.

alignments for x and y. The best alignment is chosen. In the graph this is represented by the vertices that are end points of the upper contact edge.

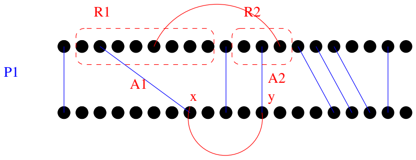


Figure 6: Wiper move under the vector representation for *Max CMO*. Two alignments of the lower protein are selected and tested exhaustively against all the possible feasible and compatible configurations around them.

During our investigations, it became evident that some sort of redistribution of consecutive alignments might be beneficial. We implemented a split operator to accomplish this. The split move, depicted in Figure 7, tries to rearrange regions of consecutive alignments. In the example, the first section of six consecutive alignments is broken into two regions of three alignments each. Note that the end points of the alignments are not changed in contrast with the sliding and wiper moves.

5 Experiments and Results

In order to evaluate our Multimeme algorithm we first implemented a GA, following as closely as possible the GA described by Lancia et al. [11].[4] We were able to reproduce Lancia et al.'s results and, although we found a small improvement of the final-values in our implementation, they were minor and we consider both GA's implementations to be very similar.

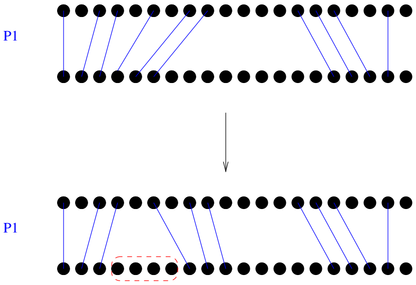


Figure 7: Split operator under the vector representation for *Max CMO*. This operator splits regions of consecutive alignments.

The GA used a population of size 300. The mutation rate was 0.15 per individual and crossover probability was set to 0.75. Fitness proportional selection was used to select the mating pool. An elitist (elite set size of 1) $(300, 300)$ selection strategy was employed. These parameters were selected after an initial assessment of parameter values with a few pairs of proteins. The Multimeme algorithm used the same basic GA setting and parameters, but also employed as memes the four variations of the sliding move, a split move and a wiper move as described in the previous section. The probability of local search was set to one, i.e., local search was applied to every individual in every generation. Each meme was iterated two times (short local searches). The values of mutation and crossover probabilities were not optimized for the Multimeme code but as mentioned before, taken from the GA setting. The innovation rate for the Multimeme algorithm was 1.0 and 0.15 (see below).

We performed two experiments on a set of 18 pairs of protein structures from the Protein Data Bank [1]. For these 18 pairs we had the upper bound values ob-

[4] Some extra experimental details were kindly given to us in private communications with the authors.

Instance	GA	Multimeme IR=1.0	LP
1a8o-1f22	25	23	28
1avy-1bct	19	22	25
1b6w-1bw5	23	23	24
1bct-1bw5	20	17	20
1bct-1f22	20	18	22
1bct-1ilp	18	18	23
1c7v-1c7w	62	62	62
1c9o-1kdf	29	29	40
1df5-1f22	21	22	27
1hlh-1hrf	19	21	24
1hlh-1nmf	22	22	27
1kst-2new	20	22	26
1nmf-2new	23	23	27
1nmg-1wdc	18	19	23
1pfn-1svf	16	16	16
1utr-1wdc	16	24	28
1vnb-1bhb	17	21	27
2new-3mef	21	19	26

Table 1: Maximum Contact Map Overlap values for several protein pairs. A GA, a Multimeme algorithm with innovation rate 1.0 are compared. The value of the LP results are also displayed to the right.

Instance	GA	Multimeme IR=0.15	LP
1a8o-1f22	25	25	28
1avy-1bct	22	22	25
1b6w-1bw5	23	24	24
1bct-1bw5	17	20	20
1bct-1f22	16	21	22
1bct-1ilp	18	19	23
1c7v-1c7w	62	62	62
1c9o-1kdf	31	34	40
1df5-1f22	24	24	27
1hlh-1hrf	20	22	24
1hlh-1nmf	22	23	27
1kst-2new	22	23	26
1nmf-2new	23	25	27
1nmg-1wdc	18	19	23
1pfn-1svf	16	16	16
1utr-1wdc	26	26	28
1vnb-1bhb	19	23	27
2new-3mef	23	22	26

Table 2: Maximum Contact Map Overlap values for several protein pairs. A GA, a Multimeme algorithm with innovation rate = 0.15 compared. The value of the LP results are also displayed to the right.

tained by the LP formulation described earlier. It is important to remark that, the LP gives estimations (i.e. upper bounds) on the possible maximum objective value for a particular instance of the problem. It does not produce (explicit) solutions to the problem instances.

The metric used in the experiments was the value of best alignment obtained out of 5 runs for each pair of proteins.

In the first experiment we assessed the performance of a Multimeme algorithm with an innovation rate set to 1 in a relatively fast experiment. For both the Multimeme and the GA the maximum number of function evaluations was $3 * 10^6$. The results are presented in Table 1.

From the table we can see that the two algorithms produce the same results in 7 cases, the GA outperforms the Multimeme in four cases and the Multimeme outperforms the GA in 7 cases. We can thus say that the Multimeme algorithm with innovation rate of 1.0 generates similar or better results than the GA (both algorithms using the same number of fitness evaluations) in 14 out of 18 cases.

The second experiment was meant to test the behavior of both the GA and the Multimeme algorithms in the same set of 18 pairs of proteins but employing more fitness evaluations, $5 * 10^6$ in this case. Also the innovation rate was reduced to 0.15. The alignment values obtained are presented in Table 2.

From inspection of the table, and comparing it with the previous one, we can see that both algorithms profit from longer runs. However, the difference between the two approaches is more noticeable in this case. Out of 18 protein pairs the GA outperformed the Multimeme in just one case, instance 2new-3mef, as opposed to four in Table 1. The Multimeme produced better results in 11 cases while for the remaining pairs, 6 instances, the values obtained with both algorithms were equivalent.

The Multimeme algorithm was able to match 4 of the optimum bounds produced by the LP. In the 4 instances where the GA and the Multimeme achieve similar results, i.e. pairs 1a8o-1f22, 1avy-1bct, 1df5-1f22 and 1utr-1wdc, the values obtained are below the LP bounds. However, we speculate that actually those alignments, i.e. the ones produced with the metaheuristics, are indeed optimal and that the LP program is able to obtain higher values by using fractional solutions that cannot possibly have physical meaning. Also, it is important to note that the gap between the Multimeme results and the LP bounds is in all cases smaller than 4 except in the case of the pair 1c9o-1kdf for which the gap is 6.

Other experiments were performed with different genetic operators, like DPX crossover and different mutation moves, but the results were not particularly better than the ones discussed here; hence they are omitted.

6 Comparison with LGA

In the previous section we verified that the Multimeme algorithm introduced in this paper produces optimal and almost optimal, i.e. with respect to the LP bounds, results. In this section we assess whether the alignments generated for CMO are qualitatively similar to other well known methods of structural alignment. To accomplish this aim, we will compare our alignments with those obtained with LGA [18, 17]. The later is a state of the art, publicly available program for the comparative analysis of protein structures.

LGA can be run in two modes, protein sequence aware mode and sequence independent mode. The former is suitable when the two proteins to be compared have the same number of residues and the later for the case when the two proteins are not necessarily of the same length. We use LGA in the sequence independent mode as the illustrative comparison we run was made with proteins of different size. The parameters used to run the LGA program were $-4 \;\; -sia \;\; -o1 \;\; -d\_6.5$. Please refer to Zemla [17] for details. The pair of proteins studied was 1c9o and 1kdf (from the Protein Data Bank). This pair is the one that produces the biggest gap between the solutions returned by the Multimeme algorithm and the LP upper bound[5]. Protein 1c9o is a cold shock protein from the genome of *Bacillus Caldolyticus* and 1kdf is an antifreeze protein from *Macrozoarses Americanus*. Because the functions are similar, it is expected that the structures of the two will have some resemblance and that either algorithm (LGA or the Multimeme) will be able to capture it.

Figure 8 plots the alignments obtained by our method and the LGA program. Axis X and Y are indexed by residues id, where X represents the residues of protein 1c9o and Y that of 1kdf. A mark, circle or square, in coordinates (x, y) should be interpreted as the alignment of residue number x in 1c9o with residue y in 1kdf. The closer to the diagonal the full alignment is the more similar are the structures. As it is possible to see from the graph there is only one mismatched region between the two alignments, the area between residue 14 and 20. In that window the difference between the two alignment is considerable. In the rest of the protein the two algorithms produce strikingly similar alignments where some perfect matching regions are visible. The overall shape of both alignments is also similar. To elucidate which of the two algorithms calculates the best alignment in the region of discrepancy (i.e better preserves secondary structure features like beta sheets, alpha helices, etc), we carried out a secondary structure analysis in this region. The analysis performed allows us to conclude that both algorithms produced results of similar quality as the proteins differ substantially on their secondary structure contents for the region studied.

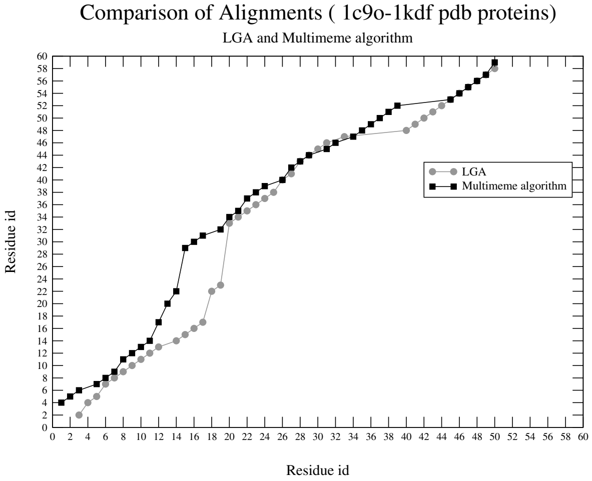


Figure 8: Comparison of the structural alignments obtained by LGA and the Multimeme algorithm for proteins 1c9o and 1kdf

7 Conclusion and Future Work

In this paper we reproduced the results of Lancia et al. for the *Max CMO* problem [11]. Their results provide the first application of a GA for this problem. We used a Multimeme algorithm with an architecture similar to that used in Krasnogor et al. [9, 10] to obtain results that improve over those produced by the standard GA. No exhaustive testing of parameters for the Multimeme algorithm was carried out, but rather the same setting as those produced for the GA were employed. Furthermore, our method gives results that are compatible with those obtained with a state of the art structure comparison algorithm [17, 18]. One immediate advantage of our method over, e.g., LGA is that being a population based approach it can potentially return not only one “best alignment” but a variety of alternative alignments. Moreover, these set of candidate alignments can be analyzed for biological

[5] It is a worst case comparison as it represents our poorest result.

relevance at a later stage by a human expert.

As an immediate follow up of this work a much larger set of protein pairs is being analyzed and the biological significance of the alignments obtained with our method will be assessed on those pairs. A Master-Worker parallel version of the LP-Multimeme integrated approach is under development. That platform will enable one to perform genome scale structures comparisons.

Acknowledgments We thank B. Walenz for his aid at several stages of this project and C. Papadimitriou for sending us several reprints. We also thank M. Collins for help performing the LP calculations. We acknowledge the funding of the British research council BBSRC (Bioinformatics Initiative, ref:42/BIO14458). This work was performed in part at Sandia National Laboratories. Sandia is a multiprogram laboratory operated by Sandia corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

References

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

[2] T.E. Creighton. *PROTEINS: Structures and Molecular Properties.* W.H. Freeman and Company, Publishers, 1993 (2nd edition).

[3] A. Godzik, J. Skolnick, and A. Kolinski. A topology fingerprint approach to inverse protein folding problem. *Journal of Molecular Biology*, 227:227–238, 1992.

[4] D. Goldberg. Phd thesis. *Department of Computer Sciences, UC Berkeley*, 2000.

[5] D. Goldman, S. Istrail, and C. Papadimitriou. Algorithmic aspects of protein structure similarity. *Proceedings of the 40th Annual Symposium on Foundations of Computer Sciences*, pages 512–522, 1999.

[6] L. Holm and C. Sander. Protein-structure comparison by alignment of distance matrices (DALI). *Journal of Molecular Biology*, 233:123–138, 1993.

[7] C. Hue-Sun and K.A. Dill. Origins of structure in globular proteins. In *Proc. Natl. Acad. Sci. USA*, volume 87, pages 6388–6392, August 1990.

[8] R.K. Kincaid. A molecular structure matching problem. *Computers Ops Res.*, 24:25–35, 1997.

[9] N. Krasnogor. *Studies on the Theory and Design Space of Memetic Algorithms.* Ph.D. Thesis, Faculty of Engineering, Computer Science and Mathematics. University of the West of England. Bristol, United Kingdom. http://dirac.chem.nott.ac.uk/~natk/Public/, 2002.

[10] N. Krasnogor and J.E. Smith. Emergence of profitable search strategies based on a simple inheritance mechanism. In *Proceedings of the 2001 Genetic and Evolutionary Computation Conference.* Morgan Kaufmann, 2001.

[11] G. Lancia, R. Carr, B. Walenz, and S. Istrail. 101 optimal PDB structure alignments: A branch-and-cut algorithm for the maximum contact map overlap problem. *Proceedings of The Fifth Annual International Conference on Computational Molecular Biology, RECOMB 2001*, 2001.

[12] S. Lifson and C. Sander. Antiparallel and parallel beta-strands differ in amino acid residue preferences. *Nature*, 282:109–11, 1979.

[13] P. A. Moscato. On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Technical Report Caltech Concurrent Computation Program Report 826, Caltech, Caltech, Pasadena, California, 1989.

[14] A.G. Murzin, S.E. Brenner, T. Hubbard, and C. Chothia. SCOP: A structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*, 247:536–540, 1995.

[15] I.N. Shindyalov and P.E. Bourne. WPDB a PC-based tool for analyzing protein structures. *Journal of Applied Crystallography*, 28:847–852, 1995.

[16] E.L.L. Sonnhammer and J.C. Wooton. Dynamic contact maps of protein structures. *Journal of Molecular Graphics and Modelling*, 16:1–5, 1998.

[17] A. Zemla. LGA program: A method for finding 3-D similarities in protein structures. *http://PredictionCenter.llnl.gov/local/lga*, 2000.

[18] A. Zemla, C. Venclovas, J. Moult, and K. Fidelis. Processing and analysis of CASP3 protein structure predictions. *PROTEINS: Structure, Function, and Genetics Suppl*, 3:22–29, 1999.