
Protein Structure Comparison: Algorithms and Applications

Giuseppe Lancia¹ and Sorin Istrail²

¹ Dipartimento Elettronica e Informatica, Via Gradenigo 6/A, Padova, Italia
lancia@dei.unipd.it

² Celera Genomics, 45 West Gude Dr., Rockville, MD
sorin.istrail@celera.com

1 Introduction

A protein is a complex molecule for which a simple linear structure, given by the sequence of its aminoacids, determines a unique, often very beautiful, three dimensional shape. Such shape (3D structure) is perhaps the most important of all protein's features, since it determines completely how the protein functions and interacts with other molecules. Most biological mechanisms at the protein level are based on shape-complementarity, so that proteins present particular concavities and convexities that allow them to bind to each other and form complex structures, such as skin, hair and tendon. For this reason, for instance, the drug design problem consists primarily in the discovery of ad hoc peptides whose 3D shape allows them to "dock" onto some specific proteins and enzymes, to inhibit or enhance their function.

The analysis and development of models, algorithms and software tools based on 3D structures are heceforth very important fields of modern Structural Genomics and Proteomics. In the past few years, many tools have emerged which allow, among other things, the comparison and clustering of 3D structures. In this survey we cannot possibly recollect all of them, and even if we tried, our work would soon be incomplete, as the field is dynamically changing, with new tools and databases being introduced constantly. At the heart of all of them, there is the Brookhaven Protein Data Bank, the largest world's repository of 3D protein structures.

Two typical problems related with protein structure are *Fold Prediction* and *Fold Comparison*. The former problem consists in trying to determine the 3D structure of a protein from its aminoacid sequence alone. It is a problem of paramount importance for its many implications to, e.g., medicine, genetic engineering, protein classification and evolutionary studies. We will not discuss

the Fold Prediction problem here, if not marginally, but we will instead focus on algorithms and applications for protein fold (structure) comparison. The following are some of the reasons why the Fold Comparison problem is also extremely important:

- *For determining function.* The function of a new protein can be determined by comparing its structure to some known ones. That is, given a set of proteins whose fold has already been determined and whose function is known, if a new one has a fold highly similar to a known one, then its function will be similar as well. This type of problems imply the design of search algorithms for 3D databases, where a match must be based on structure similarity. Analogous problems have already been studied in Computational Geometry and Computer Vision, where a geometric form or object has to be identified by comparing it to a set of known ones.
- *For clustering.* Given a set of proteins and their structures, we may want to cluster them in families based on structure similarity. Furthermore, we may want to identify a consensus structure for each family. In this case, we would have to solve a multiple structure alignment problem.
- *For assessment of fold Predictions.* The Model Assessment Problem is the following: Given a set of “tentative” folds for a protein, and a “correct” one (determined experimentally), which of the guesses is the closest to the true? This is, e.g., the problem faced by the CASP (Critical Assessment of Structure Prediction) jurors, in a biannual competition where many research groups try to predict protein structure from sequence. The large number of predictions submitted (more than 10,000) makes the design of sound algorithms for structure comparison a compelling need. In particular, such algorithms are at the base of CAFASP, a recent Fully Automated CASP variant.

The problem of Structure Similarity/Alignment determination is in a way analogous to the Sequence Alignment problem, but the analogy is only superficial and it breaks down when it comes to their complexity. There is a dramatic difference between the complexity of sequence alignment and structure alignment. As opposed to the protein sequence alignment, where we are certain that there is a unique alignment to a common ancestor sequence, in structure comparison the notion of a common ancestor does not exist. Similarity in folding structure is due to a different balance in folding forces, and there is not necessarily a one-to-one correspondence between positions in both proteins. In fact, for two homologous proteins that are distantly related, it is possible for the structural alignment to be entirely different from the correct evolutionary alignment [14].

By their nature, three-dimensional computational problems are inherently more complex than the similar one-dimensional ones for which we have more effective solutions. The mathematics that can provide rigorous support in understanding models for structure prediction and analysis is almost nonexistent, as the problems are a blend of continuous, geometric- and combinato-

rial, discrete-mathematics. Not surprisingly, various simplified versions of the structure comparison problems were shown NP-complete [17, 18].

The similarity of two structures can be accessed on whole proteins or on local domains (e.g., in a clearly defined multi-domain target). This is analogous to global vs local sequence alignments, and different techniques and similarity scores should be used for the two cases.

In this article we focus on the algorithmical aspects and applications of the problem of structure comparison. We will pay particular attention to the measures based on *contact map* similarity. For a more general treatment of different algorithms and measures, the reader is referred to a good survey by Lemmen and Lengauer [30].

An inherent difficulty in pairwise structure comparison is that it requires a structure similarity scoring scheme that captures biological relevance of the chemical and physical steric constraints involved in molecular recognition. In this respect, contact maps seem to have many advantages over other representations. The information of contacts is relatively simple but complete, and it provides a reliable first-approximation to the overwhelming complexity of the problem.

Among the classical measures (non contact map-based) for structure comparison, there is the RMSD (Root Mean Square Deviation) which tries to determine an optimal rigid superpositioning of a set of residues of one structure into a predetermined set of corresponding residues of another. The RMSD measure has many recognized flaws. Most notably, it is a global measure which can be a very poor indicator of the quality of a model when only parts of the model are well predicted. In fact, the wrongly predicted regions produce such a large RMSD that it is impossible to see if the model contains "well predicted" parts at all. Despite its drawbacks, the RMSD measure is widely used, and many other measures are based on it. We will see, e.g., the algorithms Max-Sub and GDT, used in CAFASP, the algorithms MNYFIT and STAMP, used by the database HOMSTRAD, 3dSEARCH, used in the database SCOP, the server DALI, and others.

With some notable exception, (e.g. 3dSEARCH) of algorithms whose complexity and properties were analyzed rigorously and in detail, these algorithms are usually heuristic procedures using some sort of simple local search to optimize a certain objective function. These heuristics are typically based on reasonable assumptions on the properties that the optimal solution should have in order to heavily prune the search space. From a theoretical computer scientist's perspective, the analysis and foundation of these algorithms, if present at all, are often unsatisfactory, and we remain in doubt that different, more sophisticated approaches (e.g. Branch-and-Bound, or Metaheuristics ad hoc) may be able to compute solutions with a better value of the particular objective function.

Every optimization model set up for a problem in molecular biology is only an arbitrary *representation* of the natural process, and hence it is arguable that its optimal solution is in fact the "right" one. This argument is

usually employed to justify the use of simple heuristics instead of any rigorous techniques or provable property of the model. However, after a heuristic optimization, the so-called “global optimum” found is then used to draw conclusions of biological meaning or support some theory on the biological process. Hence, it is important for the quality of the solution to be as good as possible, which justifies the use of sophisticated optimization techniques. It is auspicious therefore an ever increasing cooperation of the community of theoretical computer scientists, applied discrete mathematicians and molecular biologists to address these problems starting from their computational complexity, devising exact and approximation algorithms, studying lower and upper bounds, and designing effective heuristics of many sorts.

The remainder of the paper is organized as follows. We start with a short account on 3D structures, their properties and how they are determined. The rest of the article is divided roughly in two parts. The first part focuses on applications of structure alignment problems, while the second part is centered on algorithms. This second part is in turn divided in two, first talking about measures which are not based on contact maps, and then focusing on contact map problems. Among the applications of structure comparison, we describe some databases based on structure similarity (PDB, SCOP, HOMSTRAD, SLoop, CAMPASS, FSSP) and the problem of protein folding prediction assessment, relevant, e.g., in the CASP and CAFASP competitions. We then describe some of the main algorithms for structure comparison, and in particular those used by some of the applications previously mentioned. We survey RMSD and its variants, DALI, MaxSub, Lgscore, GDT, Geometric Hashing (3dSEARCH), MNYFIT and STAMP. In the final part of the article, we talk about novel algorithms and problems based on the contact map representation of structures. We review the literature and some of our work on exact and heuristic algorithms for the maximum contact map overlap problem. The chapter is closed by a section on possible generalizations and directions for future work.

2 Preliminaries

A *protein* consists of a chain of *aminoacids*, of length varying from about 50 to 3,000 and more. The chemical structure of a single aminoacid is comprised of a carbon atom (called C_α) connected to a carboxyl group and an amine group, a hydrogen atom and a part depending on the specific aminoacid, called the *residue*. The amine group of one aminoacid is linked to the carboxyl group of the next one, giving rise to the linear chain. The *backbone* of the protein is the sequence of C_α atoms, to which the rest of the atoms are attached.

The chemical properties and forces between the aminoacids are such that, whenever the protein is left in its natural environment, it folds to a specific 3-dimensional structure, called its *native*, which minimizes the total free energy. Several experiments have confirmed that two proteins with the same

aminoacid sequence have the same 3D structure under natural conditions, and that, if a folded protein is artificially stretched to a chain and then released, it will fold again to the same native.

The 3D structure of a protein is fully specified by the set of (x, y, z) -coordinates of its atoms, with respect to an arbitrary origin. An alternative, weaker description is the set of distances between either all atoms or only the C_α s. The main procedures used to determine the 3D structure of a protein are *X-ray crystallography* and *Nuclear Magnetic Resonance*. In X-ray crystallography the 3D structure of a molecule is determined by X-ray diffraction from crystals. This technique requires the molecule to be first *crystalized*, at perfect quality; then, its *diffraction pattern*, produced by X-irradiation is studied. This involves the analysis of thousands of spots, each with a position and an intensity. Obtaining good crystals and studying the phases of the waves forming each spot are very complex problems. For these reasons, X-ray crystallography is a very long and delicate process, which may take several months to complete. The result is the set of spacial coordinates of each atom in the structure. For a good description of this technique, see [47] or [5].

Nuclear Magnetic Resonance (NMR) is performed in an aqueous solution, where the molecules tumble and vibrate from thermal motion. NMR detects the chemical shift of atomic nuclei with nonnull spin. In order to get an adequate resolution, the molecule must tumble rapidly, which typically limits the size of the proteins to which this technique is applicable. The result of NMR is a set of (estimates of) pairwise distances between the atoms and hence it yields a collection of structures (namely all those compatible with the observed atom-to-atom distances) rather than a single one. According to some studies, the results of NMR are “not as detailed and accurate as that obtained crystallographically” [8].

For a protein we distinguish several levels of structure. At a first level, we have its *primary structure*, given by the monodimensional chain of aminoacids. Subsequent structures depend on the protein fold. The *secondary structure* describes the protein as a chain of structural elements, the most important of which are α -*helices* and β -*sheets*. The *tertiary structure* is the full description of the actual 3D fold, while the *quaternary structure* describes the interaction of several copies of the same folded protein. Correspondingly, when comparing two proteins, one may use algorithms that highlight differences and similarities at each of these levels. Algorithms for the comparison of primary structures are known as *sequence alignment* methods and are beyond the scope of this paper. Here, we will focus mainly on algorithms based on the secondary and tertiary structures.

The input to a generic structure alignment algorithm contains the description of the n -ary structures of two proteins (e.g., for $n = 3$, the set of 3D atomic coordinates of two different molecules). Loosely speaking, the goal of the algorithm is to find a geometrical transformation (typically rigid, made of rotation and translation) which maps a “large” number of elements of the first molecule to corresponding elements of the second. Such algorithms can

be of two types, i.e. *sequence order dependent* or *sequence order independent*. In the sequence dependent situation, the mapping must take into account the “identity” of the atoms, i.e. an atom of one structure can only be mapped to the same atom in the other. This way the problem is reduced to a 3D curve matching problem, which is essentially a monodimensional case, and so computationally easier. This approach can be useful for finding motifs preserving the sequence order. A sequence order independent algorithm does not exploit the order of atoms, that is, treats each atom as an anonymous “bead” indistinguishable from the others. This is a truly 3D task, which can detect non-sequential motifs and binding sites. It can be used to search structural databases with only partial information and it is robust to insertion and deletions. Consequently, it is also more challenging computationally.

In the remainder of this survey we will discuss algorithms of both types. The most important sequence order dependent similarity measure is certainly the *Root Mean Square Deviation* (RMSD). This is the problem of rigid superposition of n points (a_i) and n points (b_i), by means of a rotation R and a translation t which minimize $\sum_i |Ra_i + t - b_i|^2$, and can be solved in $O(n)$ time [55] by Linear Algebra techniques. As already mentioned, this measure is not flawless: in fact, the RMSD for a pair of structures almost identical, except for a single, small region of dissimilarity, can be very high.

3 Applications of Structure Comparisons

3.1 Databases Organized on Structure

The *Protein Data Bank* (PDB, [3]) is the most comprehensive single worldwide repository for the processing and distribution of 3D biological macromolecular structure data. As soon as a new 3D structure has been determined, it can be deposited in the PDB, this way making it readily available to the scientific community. Currently, the 3D structures of proteins in the PDB are determined mainly by X-ray crystallography (over 80%) and Nuclear Magnetic Resonance (about 16%). The remaining structures are determined by other methods, such as theoretical modeling, which are not experimental in nature, but rather based on computation and/or similarity to known proteins.

In this archive, one can find the primary and secondary structure information, atomic coordinates, crystallographic and NMR experimental data, all annotated with the relevant bibliographic citations, in a suitable text file format. Several viewers are available which interpret this format to give a 3D graphic rendering of the protein.

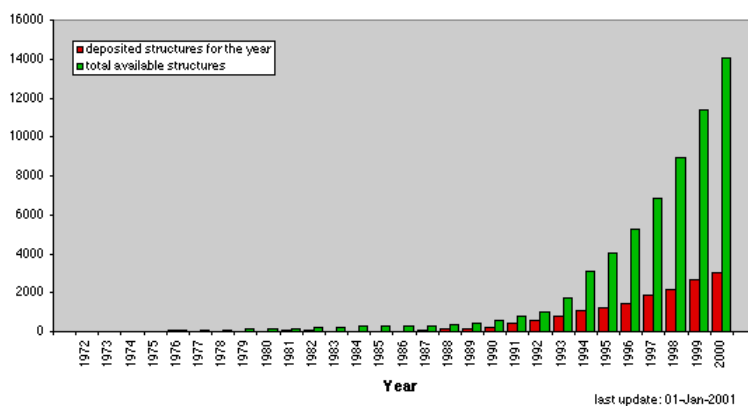


Fig. 1. The growth of the PDB.

The amount of information in the PDB has increased rapidly over the past years, passing from the roughly one thousand structures of 1990 to over 14,000 of today (see Figure 3.1³). In order for this information to be usable and accessible, it must be organized into databases which can be queried, e.g., for global/local 3D similarity and/or complementarity, or for finding evolutionary related homologs. A number of such databases has been created in the last decade, each with its own special features. Common to all of them is the use of structure comparison algorithms as the basic tool for clustering and information retrieval. In the remainder of the section we survey a few of the most known, i.e. SCOP, FSSP, HOMSTRAD, SLoop and CAMPASS.

SCOP

In the SCOP (Structural Classification Of Proteins, [42]) database, all proteins of known structure are related to each other according to their structural similarity and evolutionary relationships. The database was created by both manual inspection and the use of algorithms and automated methods. It contains detailed information on all known protein folds, and on all structurally close relatives to a given protein. The SCOP database is also a starting point for the construction of more specific databases, of which we will mention a few later on in the survey.

As of 2000, the database contained information on some 12,000 entries from the PDB, with relative folds, families and superfamilies statistics. A hidden Markov Model for SCOP superfamilies has recently been proposed by Gough *et al* [19]. One of the programs used in the partially automatic classification process in SCOP is 3dSEARCH [51], a procedure based on Geometric Hashing, which is a technique originally developed in the field of Computer Vision. We will shortly describe the technique and the algorithm later in this paper.

³ Figure from <http://www.rcsb.org/pdb/holdings.html>

HOMSTRAD

The HOMologous STRucture Alignment Database (HOMSTRAD, [39]) is a curated database of structure-based alignments for homologous protein families. It originated from a small database of seven family alignments [44], which has been gradually extended and updated over the years. This database provides structural alignments in various formats, annotated and displayed by using the program JOY [38] which was developed to highlight structural features. These alignments have been carefully examined by manual editing and are organized in homologous families, i.e., either having a common evolutionary origin, or a high percentage of sequence identity (most of the families have on average more than 30 percent identity). The classification scheme adopted by HOMSTRAD is a combination of many others, among which those adopted by SCOP. The sequence alignment programs used to help in the classification are BLAST and FUGUE. In HOMSTRAD, the known protein structures are clustered in evolutionary related families, together with a sequence representative of each family. These representatives are computed on the basis of common 3D features by using structure alignment programs such as MNYFIT and STAMP, described later in this article. The database also provides superimposed structures and links to other databases and alignment/comparison programs. The use of HOMSTRAD and JOY is suitable for comparative modelling and accurate structure alignments, while SCOP is more suited for the hierarchical classification of protein structures.

SLoop

The basis of the SLoop database [7] is in the work of Donate *et al.* [11], who described a classification of protein loops and loop regions. SLoop is a database of super secondary fragments, in which proteins are clustered according to the similarity of their secondary structures, in particular the length and type of loop regions. The clustering of structures derives from a two-stage process. First, the program SSTRUC is used to identify the loop regions, and each loop is separated into groups according to the length and type of the bounding secondary structures. Then, the loops within each group are pairwise superimposed, the relative similarities are computed and then used in a classical clustering scheme.

CAMPASS

The CAMbridge database of Protein Alignments organised as Structural Superfamilies (CAMPASS, [53]) database is a collection of structure-based sequence alignments of protein(domain)s belonging to a set of superfamilies. Currently, it is restricted to proteins sharing the same topology, and an arbitrary cut-off of 25% sequence identity has been used to eliminate homologous entries within a superfamily. The superfamilies have been chosen according

to the SCOP database and the literature. Individual domains within multi-domains proteins have been considered separately. Although this database is founded on primary structure similarity, it fits within the scope of this survey since for the distantly related members of each superfamily, simple multiple sequence alignment procedures are not appropriate. Hence, the alignment of superfamily members is based on the conservation of structural features like the presence of secondary structures, hydrogen bonding and solvent accessibility. For this type of alignments, the program COMPARE [49] has been used, which takes into account structural information. In the present version, the database consists of 52 superfamilies for which structure-based sequence alignments are available.

FSSP

The Fold classification based on Structure–Structure alignment of Proteins (FSSP, [26]) database provides all–against–all structure comparison of all current PDB structures with more than 30 residues. The alignment and clusters are updated continuously via the DALI search engine (described later in this article). The classification is fully automatic. As of this writing, the database contains some 2,700 sequence families, out of a population of roughly 25,000 protein structures.

The clustering of all proteins in the PDB is done by first identifying a set of “representative” folds (which are highly dissimilar) and partitioning all the remaining proteins into homologs of the structures in the representative set (where a sequence is considered a homolog of another if they have at least 25% sequence identity). Then, an all–against–all comparison is performed on the sequences in the representative set, this way inducing a clustering of all structures and, implicitly, their homologs. The FSSP entries include the resulting alignments as well as structure alignments of the representatives and their homologs.

3.2 Fold Prediction and Assessment

The problem of protein fold prediction is perceived as one of the core questions, albeit extremely complex, for the molecular biology and gene therapy of the 21st century. The amount of time and work required to determine a protein structure experimentally makes the design of faster yet highly reliable methods a compelling need. One of the possibilities which is extensively studied is the creation of algorithms capable of computing the final fold of a protein from its aminoacid sequence. This is a challenging computational problem and the prize for its solution includes, beyond the obvious scientific and medical implications, also relevant economical interests. For this reasons, some companies among which IBM, are devoting a good deal of research into this problem. In 1999 IBM announced a \$100 million research project, named *Blue gene* [2], for the development of a system capable of more than one

petaflop (10^{15} operations per second) which will tackle the problem of protein folding by simulating the actual physical phenomenon in a massive, parallel, computation.

The computational problem of protein fold prediction is beyond the scope of this paper. For a good introduction to the topic see, e.g., [34]. What is relevant for our purposes is the assessment of the prediction quality, i.e., how can we judge if a prediction is “close enough” to a model? Clearly, we need tools of structure comparison which, given a model determined experimentally, and a prediction of the former, are capable to come up with a *score* (possibly, a real number in the range $[0, 1]$, with 0 meaning completely wrong, 1 meaning completely right).

We now describe some of the situations where this problem occurs and how it is currently solved.

CASP and CAFASP

The Critical Assessment of techniques for protein Structure Prediction (CASP [40, 41]) is an international bi-annual exercise in which various research groups try to predict the 3D structure of some proteins from their aminoacid sequences. The experiment consists of three phases: First the prediction targets are collected from the experimental community; then the predictions are collected from the modeling community; finally, the assessment and discussion of the results takes place. Depending on the method used to predict, there are different categories, i.e. Comparative Modeling, Fold Recognition or Threading, Docking and *ab initio* Folding. The rules of the exercise do not forbid human intervention in making the predictions. In fact, the most successful teams so far are those who can annoverate some of the best experts in the field. Similarly, the assessment of the results is not done automatically, but, again, human expertise is required. Thus, for each prediction category, a group of evaluators is asked to compile a ranking of the teams. Some automatic methods are employed as well, with the caveat that measures such as RMSD and other rigid-body superposition methods have been found unsuited for the recognition of well predicted substructures or domains. Since it has been so far unclear even if a single, objective, quantitative assessment based on just one measure can exist, in the CASP experiments a large number of scores have been employed, but the final word is left to the expert evaluation and human visualization. Some of the assessment techniques are described in [35]. The large number of models submitted (11136 in CASP4) underlines the ever increasing importance of automatic measures.

CAFASP [12] is a Fully Automated CASP, in which human intervention is forbidden. In this competition, held in parallel with CASP, both predictions and evaluations must be performed by unsupervised programs. Although it is widely believed that human-expert predictions and assessments can be significantly more accurate than their automatic counterparts, one of the goals of CAFASP is to push the research for reducing the gap between the two. The

main algorithm used in CAFASP for scoring the predictions is MaxSub, described later in this paper. This algorithm was partially validated by scoring some of the CASP3 Fold-Recognition (FR) models and comparing the rankings obtained to the human-expert assessment carried out by CASP jurors. Even if some differences were observed, the top six predicting groups ranked by MaxSub were the same as those ranked by CASP3. On the other hand, this algorithm had very little success in comparing hard FR targets to their models, because it fails to detect local structural similarities. In the words of CAFASP organizers [13], *even for a target with a known fold, the fact that MaxSub scored a prediction with zero does not necessarily imply that the prediction is totally wrong. (...) Because of the relatively low sensitivity of our automated evaluation, a “human expert” is required to learn more about the prediction capabilities of the servers.*

Live Bench

The Live Bench Project [6] is an effort aimed at the continuous benchmarking of protein structure prediction servers. This project implements an automatic large-scale assessment of the performance of publicly available fold-recognition/prediction servers, such as PDB-Blast, GenTHREADER, FFAS, T98-lib, 3D-PSSM and INBGU. The goal is to provide a consistent, automated framework in which such servers can be evaluated and compared. Every week, all new PDB protein structures are submitted to all participating fold recognition servers. For instance, in the period October 1999–April 2000, 125 targets were used for comparisons, divided into 30 “easy” structures and 95 “hard” ones. The results of the predictions are compared using similar algorithms as in CAFASP, i.e., MaxSub and Lgscore. Note that, differently from CASP and CAFASP, these “predictions” are in fact done after the structures are made known and available on the PDB, and hence the reliability of LiveBench is based on the assumption that the evaluated servers do not use any hidden feature directing the prediction towards the correct answer. One of the main results of the LiveBench project was to recognize that the servers largely disagree in many cases (they were able to produce structurally similar models for only one half of the targets, and for only one third of the targets such models were significantly accurate) but that a “combined consensus” prediction, in which the results of all servers are considered, would increase the percentage of correct assignments by about 50%.

4 Software and Algorithms for Structure Comparison

In this section we review some of the existing algorithms for protein structure comparison. We start with the RMSD measure, which is at the basis of other algorithms described, such as MaxSub, GDT, 3dSEARCH and MNY-FIT. The section includes both sequence order dependent and sequence order

independent algorithms. In the latter we find DALI, Lgscore, 3dSEARCH and STAMP.

A general comment which applies to almost all of the algorithms described is a lack of rigorousness from a Theoretical Computer Science point of view. That is, these algorithms are usually heuristic procedures using some sort of local search to optimize a certain objective function. The algorithms that we surveyed are not “approximate” as intended in Combinatorial Optimization (i.e., there is no guarantee that the solution found will be within a certain factor of the optimum, or of a bound to the optimum, which is never studied), but are mostly fairly simple heuristics which use “common sense” to prune the search space. That is, they exploit some reasonable assumptions on the properties that the optimal solution should have (which, however, are not proved to always hold) to discard partial solutions from further considerations. This way of proceeding makes certainly sense for hard problems such as these, and in fact, the ideas on which Branch-and-Bound is based are somewhat similar. However, Branch-and-Bound is an *exact* and *rigorous* method, which guarantees an optimal solution or, if stopped before reaching one, returns the maximum error percentage in the best solution found. We think that Branch-and-Bound should also be considered, at least to tackle the smaller instances, as a viable option for these problems. Also, in the realm of local search heuristic procedures, there are several sophisticated paradigms that have proved effective for many hard combinatorial problems. In particular, Genetic Algorithms, Tabu Search with its variants, Simulated Annealing, Randomized Algorithms and various Metaheuristics. Given the relatively small effort required in implementing such searches (most of these procedures are available as general purpose libraries and the user has to implement only one or two core modules) we think that more work should be devoted into investigating which of these techniques is more suitable for a particular problem at hand.

RMSD

Given n points (a^1, \dots, a^n) , with $a^i = (a_1^i, a_2^i, a_3^i)$ and n points (b^1, \dots, b^n) , a “rigid body superposition”, is a composition of a rotation and a traslation, that takes the points a^i into a'^i and makes the sum of differences $|a'^i - b^i|$ as small as possible.

More specifically, let Δ be defined by

$$\Delta^2 = \min_{R,t} \sum_i |Ra_i + t - b_i|,$$

where R is a rotation matrix (i.e., $\det R = 1$) and t a translation vector. The Root Mean Square Deviation of a and b is Δ/\sqrt{n} .

The optimal traslation t is determined easily, since it can be proved that it must take the center of mass of the a^i points into the center of mass of the

b^i s. To determine the optimal R one must consider the correlation matrix A defined by $A_{ij} = \sum_{h=1}^n a_i^h b_j^h$, as it can be shown that the optimal rotation maximizes $\text{Tr}(RA)$. One way to maximize $\text{Tr}(RA)$, exploits the fact that the problem is three dimensional. Represent the matrix R as

$$R = 1 + \sin \theta M + (1 - \cos \theta) M^2$$

where

$$M = \begin{pmatrix} 0 & n & -m \\ -n & 0 & l \\ m & -l & 1 \end{pmatrix}$$

and θ represent the rotation angle around an axis in the direction of the unit vector $u = (l, m, n)$. It then follows that

$$\text{Tr}(RA) = \text{Tr}(A) + \text{Tr}(MA) \sin \theta + (\text{Tr}(A) - \text{Tr}(M^2 A)) \cos \theta$$

and hence

$$\max_{\theta} \text{Tr}(RA) = \text{Tr}(A) + \sqrt{(\text{Tr}(MA))^2 + (\text{Tr}(A) - \text{Tr}(M^2 A))^2}.$$

So, for a fixed axis of rotation, the calculation of the optimal angle θ is immediate [36].

Alternative ways of determining the optimal rotation matrix R , based on the computation of the eigenvalues of $A^T A$, or on the *quaternion* method, are described in [31].

As previously mentioned the RMSD is widely used as a sequence dependent measure of 3D similarity, i.e., to find an optimal mapping of points of a structure into a predetermined set of corresponding points in another structure. Despite some known flaws, the RMSD measure is at the hearth of many other measures. The main problem with RMSD is that it is more sensitive to small regions of local differences than to large regions of similarity. In fact, small distances between well-matched atoms have a much lesser impact on the RMSD than do the very large distances between poorly matched C_{α} s.

Another issue with RMSD is that it is not designed to be equivalent between different targets. For instance, the quality of a model with an RMSD of a 4 Å for a 40 residues long target is not the same as that of a model of 4 Å RMSD over 400 residues.

The problem of local different regions affecting the total score is common to other global measures, but for the RMSD value the effect is larger than for other scores, e.g., the contact map similarity. In this situation, a solution is typically to first calculate the similarity for segments of the protein and then define a normalized score based on the number of residues in the segments. However, the right relationship between the length of the segment and the

value must be defined, and this is not a trivial problem. There are two conflicting issues: we are interested in large segments that are highly similar, but the similarity and the length of the segments are inversely proportional.

Similar problems are faced by “trimming” methods which, given a current alignment, re-define iteratively a core of pairs of residues that are matched within a small distance, work on these only, and then try to extend the alignment. The threshold and the degree of trimming are to some extent arbitrary, and this choice affects the final outcome.

DALI

In DALI [25], the structural alignment of two proteins is done indirectly, not by comparing the actual structures, but their *distance matrices*. The distance matrix of a folded protein P is the matrix $D^P = (d_{ij}^P)$ of the Euclidean distances between all pairs (i, j) of its residues. The matrix provides a 2D representation of a 3D structure, and contains enough information for retrieving the actual structure, except for overall chirality [22]. The idea underlying DALI is that if two structures are similar, then their distance matrices must be similar too. An analogous idea is used to compare structures via their *contact maps* and is described later in this article. In order to find similarities between two distance matrices, D^A and D^B , DALI uses a heuristic trick, and looks for all 6×6 submatrices of consecutive rows and columns in D^A and D^B (that is, it maps $i^A, i^A + 1, \dots, i^A + 5$ into $i^B, i^B + 1, \dots, i^B + 5$, and $j^A, j^A + 1, \dots, j^A + 5$ into $j^B, j^B + 1, \dots, j^B + 5$) to find patterns of similarity. This is done in the first step of the algorithm. The set $\{i^A, \dots, i^A + 5, j^A, \dots, j^A + 5\}$ with its counterpart in B , is called a *contact pattern*. The second step attempts at merging the contact patterns into larger, consistent, alignments. An alignment of two proteins A and B is a one-to-one function M of some residues M_A (the *matched* ones) of A into some residues of B . DALI tries to determine the alignment that maximizes the global objective function

$$S(M) = \sum_{i,j \in M_A} \phi(i, j, M(i), M(j)).$$

This objective function, which looks only at matched residues, depends on the particular ϕ used, which in turn depends on the C_α - C_α distances d_{ij}^A and $d_{M(i),M(j)}^B$. The two main forms for ϕ used by DALI are a *rigid* and an *elastic* score. The rigid score is defined by $\phi(i, j, M(i), M(j)) := \theta - |d_{ij}^A - d_{M(i),M(j)}^B|$, where $\theta = 1.5\text{\AA}$ is the zero level of similarity. The more complex elastic score uses relative, other than absolute deviations, and an envelope function to weigh down the contribution of long distance matched pairs.

Some heuristic rules are employed to speed up the search. For instance, only a subset of all contact patterns is considered. Also, overlapping contact patterns (they could possibly overlap by 11 out of 12 residues) are suppressed by partitioning the protein in nonoverlapping structural elements and merging

repetitive segments. Further rules to prune the search involve the suppression of pairs of patterns for which the sum of distances of one is not within a given interval of the same value in the other. Other rules of the same nature are employed which we do not describe here. The final optimization is done via a Monte Carlo algorithm, which is basically the local search strategy known as Simulated Annealing. In this search a *neighborhood* of a solution is explored for a good *move*, leading to a new solution. A move which takes into a solution worse than the current one can be accepted, but with probability inverse to the degradation in the solution quality. The *temperature* of the system is also used to favour or make more difficult such non-improving moves. DALI uses two basic moves, named *expansion* and *trimming* ones. Expansion moves try to extend a current partial alignment by adding to it a contact pattern compatible with one of the currently matched residues (i.e., containing the same match), and then possibly removing some matches that have become noncompatible. Trimming moves simply remove from a current alignment any subalignment of 4 elements that gives negative contribution to the total similarity score. These moves are alternated as one trimming cycle every five expansion cycles.

DALI was used to carry out an all-against-all structure comparison for 225 representative protein structures from the PDB, providing the basis for classification in the FSSP database. Also, the DALI server can be used to submit the coordinates of a query protein structure which is then compared against the PDB.

MaxSub

MaxSub [50] is an algorithm explicitly developed to be used for the automatic assessment of protein structure similarity (in particular within the CAFASP exercise). Therefore it was written with the goals of (a) being simple and objective, (b) producing one single number (score) that measures the amount of similarity between two structures and (c) performing similarly to human-expert evaluations. These goals were partly met, and the measure was partially validated when, on a particular category and subset of predictions, it ranked in the top the same groups that were ranked in the top by the CASP human evaluators. A prediction can be evaluated in different aspects. For example, in the Homology Modeling category one may want to score the accuracy of the loops or the side chain packing. Or, in the Fold Recognition category, one may want to evaluate whether a compatible fold was recognized, regardless of the quality of the alignment obtained. MaxSub, in particular, focuses on the quality of the alignment of the models to the target. It is a *sequence-dependent* assessment, i.e., only corresponding residues are compared. The result is a single scalar in the range of 0 (completely wrong) to 1 (perfect model). The scalar is a normalization of the size of the largest "well-predicted" subset, according to a variation of a formula devised by Levitt and Gerstein [32]. MaxSub is also similar to GDT, a method developed by Zemla et al. [59] for being used in CASP3, which attempts to find a large set of residues which

can be superimposed over the experimental structure within a certain error. GDT is briefly described later in this survey.

The input to MaxSub are the 3D coordinates $A = \{a_1, \dots, a_n\}$ and $B = \{b_1, \dots, b_n\}$ of the C_α atoms of a prediction and a target. A solution is a subset $M \subset \{1, \dots, n\}$ (called a *match*) such that the pairs (a_j, b_j) for $j \in M$ can be superimposed "well-enough" (i.e., below a given threshold d) by a transformation T (rotation and translation). The objective is to maximize the size of M , i.e., find the largest subset of residues which superimpose well upon their correspondents.

Given M , the value of the best superposition can be easily found, (analogously to the RMSD determination previously described), by using the transformation T_M which minimizes, over all T ,

$$RMS(M) = \sqrt{\frac{\sum_{j \in M} \|a_j - T(b_j)\|}{|M|}}$$

where $\|\cdot\|$ is the Euclidean norm.

The optimization of M is performed by a simple heuristic algorithm. This algorithm is based on the reasonable (although not rigorous) assumption that a good match must contain at least $L \geq 4$ consecutive pairs, i.e. $\{(a_i, b_i), i = j, j+1, \dots, j+L-1\}$ for some j . So the idea is, for a fixed L , to look, in time $O(n)$, for all matches of L consecutive elements, and try to extend each of them. The extension of a match M is also done heuristically, in 4 iterations, where at each iteration k the pairs (a_i, b_i) for which $\|a_i - T_M(b_i)\| < kd/4$ are added to M .

Let M^* be the best match found by the above procedure. In order to return a single, normalized score, a final value S is computed as

$$S(M^*) = \left(\sum_{i \in M^*} \frac{1}{1 + (\frac{d_i}{d})^2} \right) / n,$$

where $d_i = \|a_i - T_{M^*}(b_i)\|$. In our opinion, there are several non rigorous steps that should be addressed in evaluating MaxSub and its reliability. For instance, although the basic objective optimized by the algorithm is *maximize* $|M|$ *such that* $d_i \leq d$ *for all* $i \in M$, the optimization of S (which is the score eventually returned) calls ultimately for *maximize over all* M , $\sum_{i \in M} \frac{1}{d^2 + d_i^2}$ which is a different objective with possibly a different optimum. Also, there are many arbitrary parameters, such as L , the number 4 of iterations in the extension phase, and the threshold d . Some experiments have shown that the algorithm does not depend on the choice of d too much, while the dependance on the other parameters is not described with the same accuracy. As far as how close the solution is to the optimum, a comparison to the best of a random set of 70,000 matches was used. Given the astronomical number of possible solutions, it is arguable if such method can be used to conclude anything about the actual performance of the algorithm.

Lgscore

The measure Lgscore [9] is used in LiveBench and CAFASP for the automatic assessment of fold recognition problems. This measure is statistically based, and relies on the following formula, by Levitt and Gerstein [32], for the similarity of two structures, after a superposition in which M denotes the aligned residues:

$$S_{\text{str}}(M) = K \left(\sum_{i \in M} \frac{1}{1 + (d_i/d_0)^2} - \frac{N_g}{2} \right),$$

where K and d_0 are constants (usually set to 10 and 5\AA), d_i is the distance between the aligned pairs i of C_α atoms, and N_g is the number of gaps in the alignment. The P-value of a similarity is the likelihood that such a similarity could be achieved by chance. Levitt and Gerstein showed how to compute the P-values for a distribution of S_{str} depending on the length of the alignment. Lgscore is the negative log of the P-value for the most significant subpart of the alignment. In order to find such most significant segment, two heuristic algorithms are used, i.e. “top-down” and “bottom-up”. The top-down approach consists of a loop in which (1) a superposition is done of all residues that exist in the current model and the target; (2) the P-value of this superposition is stored and the residues that are furthest apart are deleted in the model and the target. The loop is repeated as long as there are at least still 25 residues. The alignment with the best P-value is returned. The bottom-up approach essentially tries to match a fragment $i, \dots, i+j$ of $j \geq 25$ consecutive residues in the model and the target, for different values of i and j and returns the best P-valued fragment. None of the two approaches dominates the other, although the authors report that the bottom-up algorithm found the best subset in most cases. The arbitrary value of 25 residues as a threshold for fragment length is justified in [9] since “short segments are given unrealistic good P-values”.

GDT

GDT (Global Distance Test, [59]) is an algorithm for identifying large sets of residues (not necessarily continuous) in a prediction and a target which do not deviate more than a given cutoff value. The algorithm is heuristic, and similar to Lgscore. It consists of a loop, started from the alignment between model and target of the longest continuous fragments within the cutoff value, found by enumeration, and all three-residue segments. At each iteration, an RMS transform is computed for the given alignment, and the pairs of residues whose distance is greater than the threshold are removed. The loop is ended as soon as no residues are removed by one iteration.

Geometric Hashing and 3dSEARCH

Comparing protein structures and, more generally, querying databases of 3D objects, such as the PDB, can be regarded as special cases of Object Recognition problems in Computer Vision. Therefore, algorithms originally developed in the field of Computer Vision have now found a suitable application in Structural Genomics as well. *Geometric hashing* [28] is an example of such a technique. Geometric Hashing can be used to match a 3D object (a protein structure) against one or more similar objects, called “models” (e.g. a database organized on structure). The key idea is to represent each model under different systems of coordinates, called *reference frames*, one for each triplet of non collinear points of each model. In any such system, all points of the model (which are vectors of reals) can be used as keys (i.e., indices in a hash, or look-up, table) to store the information about the model they belong to. In a *preprocessing phase*, all models are processed this way, and an auxiliary, highly redundant, hash table is created. In the *querying phase*, executed each time a target is to be matched against the models, the following steps are taken. For each reference frame of the target, the points of the target (in the coordinate system defined by the reference frame) are used to access the hash table and hence retrieve some of the models. These models define a *match list* for the particular reference frame. A match list gives a set of alignments of the points in the target and in each of the models retrieved. The match lists are then merged in order to find larger alignments. The final step is to compute an RMSD transformation for each alignment obtained, so as to find the model most similar to the target.

The program 3dSEARCH [51], used by the database SCOP, implements the strategy of geometric hashing on the secondary structure representation of each protein.

MNYFIT

The program MNYFIT [54], which is used by the database HOMSTRAD, belongs to a suite of programs called COMPOSER, for homology modeling of protein structures. With COMPOSER one can work out the construction of a predicted 3D structure based on one or more known homologous ones. MNYFIT is a core module within this software collection, and can handle up to 10 molecules of about 500 residues each. It performs the superposition of two or more related protein structures, individues structurally equivalent residues and their location, and with this information is able to define a common framework for a set (family) of proteins.

The main algorithm used by MNYFIT is the least squares fitting procedure of McLaughlan [36] for rigid body superposition, akin to the RSMD procedure discussed above.

The superposition is performed initially using at least three atoms from the C_α backbones of each molecule, which occupy topologically equivalent

positions in all the molecules to be fitted. For two or more superimposed structures, a threshold is used to define which of the C_α atoms are considered structurally equivalent (i.e. those whose distance in the superposition is smaller than the threshold). Then, the alignment is iteratively updated with the double objective of increasing the number of structurally equivalent atoms while at the same time keeping a low Root Mean Square Deviation between the equivalenced atoms in the superimposed structures. When more than two molecules are superimposed, the objective is to determine a consensus structure (called “framework”), which should be as close as possible to all involved structures, and which represents the average positions of the structurally equivalent atoms of all superimposed molecules. The framework returned from the final step of the procedure is used by COMPOSER for further model–building operations.

STAMP

One of the programs used by the database HOMSTRAD for the detection of homologous protein families is STAMP [48], a program for the alignment of protein sequences based on their 3D structures. The core of this program contains an implementation the basic Smith and Waterman dynamic programming procedure for sequence alignment [52], but using suitable similarity scores which express the probability of residue–residue structural equivalence. These scores are computed according to the equation by Argos and Rossmann [1]

$$P_{ij} = \exp \frac{d_{ij}^2}{-2E_1^2} \exp \frac{s_{ij}^2}{-2E_2^2},$$

where d_{ij} is the distance between the C_α atoms of residues i and j , and s_{ij} measures the local main chain conformation. The Smith and Waterman procedure is embedded in a loop as follows: at each iteration, the output of the dynamic program is a residue–residue equivalence (the sequence alignment). This list of equivalences is used to compute a best fit transformation minimizing the RMSD, via the least square method of McLaughlan. The new set of coordinates and residue–residue distances, obtained under this transformation, are then used to recompute the similarity score values, which are then used for another round of the Smith Waterman procedure. The loop is repeated iteratively until the alignment becomes stable.

The method has proved experimentally effective, allowing the generation of tertiary structure based multiple protein sequence alignments for a variety of protein structural families. However, the method is only effective for proteins which share a good deal of global topological similarity, while fails if applied to, e.g., proteins with common secondary structures but with different connectivity, orientation or organization.

STAMP allows for the specification of a minimum number of equivalent residues to be matched in two structures, the reversals of strand directions, the

swapping of sequence segments and more. The output contains two measures of alignment confidence: a “structural similarity score”, which can also be used to measure the functional and evolutionary relationship, and an “individual residue accuracy” which is intended to measure the quality of the topological equivalence of the pairs of aligned residues.

5 Problems Based on Contact Map Representations

A *contact map* [33, 21] is a binary version of the distance matrix representation of a protein structure. More specifically, the contact map of a folded protein of n residues is a 0-1, $n \times n$ matrix C , whose 1-elements correspond to pairs of amino acids in three-dimensional “contact”. A contact can be defined in many ways. Typically [37], one considers $C_{ij} = 1$ when the distance of two heavy atoms, one from the i -th aminoacid and one from the j -th aminoacid of the protein, is smaller than a given threshold (e.g., 5\AA). The framework of the contact map representation of proteins is very appealing, since this intuitive and fairly simple representation is already complex enough to capture the most important properties of the folding phenomenon. It has been shown that it is relatively easy to go from a map to a set of possible structures to which it may correspond [21, 56]. This result has opened the possibility of using contact maps to predict protein structure from sequence, by predicting contact maps from sequence instead. Vendruscolo and collaborators, among others, have looked at the problem of devising an energy function based on contacts, which should be minimized by the protein’s native state contact map [57, 45]. For this purpose, they have set up a system of linear inequalities, with 20×20 variables C_{ab} for all pairs of aminoacids a and b , which represent the weight to give to a contact between the aminoacids a and b . The inequalities are built as follows. Given the contact map of a correct structure r , there is an inequality for any alternative structure w over the same sequence of aminoacids, imposing that the energy of r is lower than that of w . Alternative structures are obtained, e.g., by threading through other known folds. The results are that “a simple pairwise contact energy function is not capable of assigning the lowest energy to an experimentally determined structure” [45], but by using corrective factors, such as a distinction between contacts on the surface or in the core, and simple distance-dependant interaction weights, one can achieve contact potentials which are in fact often stabilized by the native contact maps. The use of energy function minimization to predict contact maps is just one possible way. To the best of our knowledge, very little success has been met so far in the contact map prediction problem. It is possible that the research on this question will be boosted by the fact that the competition CAFAPS has recently introduced the new “contacts” prediction category.

The statistics of contact maps have been studied as well, and it has been shown that the number of contact maps corresponding to the possible configurations of a polypeptide chain of n residues, represented by a self-avoiding

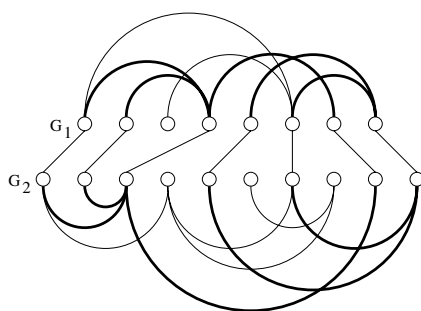


Fig. 2. An alignment of value 5

walk in the d -dimensional lattice, grows exponentially with n for all $d \geq 2$ [58].

5.1 Using Contact Maps for Structure Comparison

Besides their use for protein fold prediction, contact maps can be exploited to compare 3D structures. The basic idea is fairly obvious: if two structures are similar, we should expect their contact maps to be similar as well, and conversely. Hence, we can use an indirect method for structure comparison, i.e., contact map comparison instead. In our previous work [29] we have designed an exact algorithm based on an Integer Programming formulation of this problem, which we will now review.

We can regard the contact map of a protein p as the adjacency matrix of a graph G_p . Each residue is a node of G_p , and there is an edge between two nodes if the corresponding residues are in contact. The *Contact Map Overlap* (CMO) problem, calls for determining a sequence-independent alignment of some residues in the first protein (nodes in G_1) with residues of the second protein (nodes in G_2) which highlights the largest set of common contacts as follows: The value of the alignment is the number of contacts (i.e., edges) in the first map whose endpoints are aligned with residues that are also in contact in the second map. This value is called the *overlap* for the two proteins, and the optimization problem is to find the maximum overlap. Figure 5.1 shows two contact maps and a feasible alignment.

This measure was introduced in [15], and its optimization was proved NP-hard in [18], thus justifying the use of sophisticated heuristics or Branch-and-Bound methods.

Some of the most powerful algorithms for finding exact solutions of combinatorial optimization problems are based on *Integer Programming* (IP) [43]. The IP approach consists in formulating a problem as the maximization of a linear function of some integer variables and then solving it via Branch-and-Bound. The upper bound comes from the *Linear Programming* (LP) *relaxation*, in which the variables are not restricted to be integer, and is poly-

mially solvable. When the LP-relaxation value is close to the value over the integers, then the bound, and hence the pruning of the search space, is effective. In order to obtain good bounds, the formulation is often reinforced by the use of additional constraints, called *cuts* (from which the approach name, *Branch-and-Cut*): these are constraints that do not eliminate any feasible integer solution, but make the space of fractional solutions smaller, this way decreasing the value of the LP bound. In many cases a good IP formulation requires an exponential number of constraints and/or cuts. This would make its practical solution impossible, unless there is a way to include all of them only implicitly. This way exists, and works as follows. Given an LP fractional solution x^* and an inequality $ax \leq b$, we say that the inequality is *violated* by x^* if $ax^* > b$. If we have an exponential number of inequalities, we can solve the LP with only a (small) subset of them, obtain a solution x^* and then check if any of the (exponentially many) inequalities that were left out is violated by x^* . If not, the current solution is optimal with respect to the full formulation, otherwise, we can add the violated inequality to the current LP and iterate the process. The check for a violated inequality is called *separation* and is carried out by a *separation algorithm*. By a fundamental result of Grötschel, Lovász and Schrijver [20], the existence of a polynomial-time separation algorithm is a necessary and sufficient condition for solving the whole exponential-sized LP relaxation in polynomial time.

The CMO problem can be reduced to a (very large) *Maximum Independent Set* (MIS) problem on a suitable graph. An *independent set* is a set of vertices such that there is no edge between any two of them. The MIS is a classic problem in combinatorial optimization which has a definition nice and simple, but is one of the toughest to solve exactly. However, by exploiting the particular characteristics of the graphs derived from the CMO problem, we can in fact solve the MIS on graphs of 10,000 nodes and more.

The natural formulation of the MIS as an IP problem has a binary variable x_v for each vertex v , with the objective of maximizing $\sum_v x_v$, subject to $x_u + x_v \leq 1$ for all edges $\{u, v\}$. This formulation gives a very weak bound, but it can be strengthened by the use of *clique inequalities* cuts, such as $\sum_{v \in Q} x_v \leq 1$, which say that any clique Q can have at most one node in common with any independent set. The addition of these constraints can lead to tight formulations. This is exactly the case for the CMO problem. In [29] we formulated the CMO problem as an IP and solved it by Branch-and-Cut, where the cuts used are mainly clique-inequalities. Although there is an exponential number of different clique inequalities, we can characterize them completely and separate over them in fast ($O(n^2)$) polynomial time. Note that finding cliques in a graph is in general a difficult problem, but in this case we can solve it effectively since it can be shown that the underlying graph is perfect. The following section gives some details on the formulations and results from [29]. Further details and full proofs can be found in the original paper.

IP Formulation

We can phrase the CMO problem in graph-theoretic language as follows: We are given two undirected graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, with $n_i = |V_i|$ for $i = 1, 2$. A total order is defined on $V_1 = \{a_1 < \dots < a_{n_1}\}$ and $V_2 = \{b_1 < \dots < b_{n_2}\}$. It is customary to draw such a graph with the vertices arranged increasingly on a line. We denote an edge by an ordered pair (i, j) , with a tail in the left endpoint and a head in the right endpoint.

A non-crossing alignment of V_1 in V_2 is defined by any two subsets of the same size k , $\{i_1, \dots, i_k\} \subseteq V_1$ and $\{u_1, \dots, u_k\} \subseteq V_2$, where $i_1 < i_2 < \dots < i_k$ and similarly for the u_h 's. In this alignment, u_h is aligned with i_h for $1 \leq h \leq k$. Two edges (contacts) $(i, j) \in E_1$ and $(u, v) \in E_2$ are *shared* by the alignment if there are $l, t \leq k$ s.t. $i = i_l, j = i_t, u = u_l$ and $v = u_t$ (see Figure 5.1). Each pair of shared edges contributes a *sharing* to the objective function. The problem consists in finding the non-crossing alignment which maximizes the number of sharings.

An alignment corresponds to a set of lines connecting nodes of V_1 and V_2 in the usual drawing with V_1 drawn on the top and V_2 on the bottom. We denote such a line for $i \in V_1$ and $j \in V_2$ by $[i, j]$. We say that two lines *cross* if their intersection is a point. The sharings (e_1, f_1) and (e_2, f_2) can be both achieved by an alignment if and only if they are *compatible*, i.e. no two of the lines between the tails of e_1 and f_1 , the tails of e_2 and f_2 , the heads of e_1 and f_1 and the heads of e_2 and f_2 cross. A set of sharings is *feasible* if the sharings are all mutually compatible, otherwise it is *infeasible*. Similarly we define a feasible and infeasible set of lines. If we draw the lines connecting the endpoints of an infeasible set of sharings, we have an infeasible set of lines.

We denote by y_{ef} a binary variable for $e \in E_1$ and $f \in E_2$, which is 1 iff the edges e and f are a sharing in a feasible solution. The objective function of CMO is

$$\max \sum_{e \in E_1, f \in E_2} y_{ef}, \quad (1)$$

and the constraints are

$$y_{e_1 f_1} + y_{e_2 f_2} \leq 1, \quad (2)$$

for all $e_1, e_2 \in E_1, f_1, f_2 \in E_2$ s.t. (e_1, f_1) and (e_2, f_2) are not compatible.

It can be shown that the formulation made of constraints (2) gives a very weak LP bound, and also contains too many constraints. To strengthen the bound, we use a new set of variables and strong cuts. We introduce a new set of binary variables x_{iu} for $i \in V_1$ and $u \in V_2$, and constraints forcing the nonzero x variables to represent a non-crossing alignment. We then bound the y variables by means of the x variables, so that the edges (i, j) and (u, v) can be shared only if i is mapped to u and j to v . For $i \in V_1$ (and analogously for $i \in V_2$), let $\delta^+(i) = \{j \in i + 1, \dots, n_1 : (i, j) \in E_1\}$ and $\delta^-(i) = \{j \in 1, \dots, i - 1 : (j, i) \in E_1\}$. Then we have the following constraints:

$$\sum_{j \in \delta^+(i)} y_{(i,j)(u,v)} \leq x_{iu}, \quad \sum_{j \in \delta^-(i)} y_{(j,i)(u,v)} \leq x_{iv} \quad (3)$$

for all $i \in V_1$, $(u, v) \in E_2$, and analogous constraints for $i \in V_2$ and $(u, v) \in E_1$. We call these *activation* constraints. Finally, the *noncrossing* constraints are of the form:

$$x_{iu} + x_{jv} \leq 1 \quad (4)$$

for all $1 \leq i \leq j \leq n_1$, $1 \leq v \leq u \leq n_2$ s.t. $i \neq j \vee u \neq v$.

Our IP formulation for the max CMO problem is given by (1), (3), and (4), where x and y are all binary variables. The cuts in the x variables are described next.

First, we make clear the connection to the independent set problem. We define two graphs G_x and G_y . In G_x there is a node N_{iu} for each line $[i, u]$ with $i \in V_1$ and $u \in V_2$ and two nodes N_{iu} and N_{jv} are connected by an edge iff $[i, u]$ and $[j, v]$ cross. Similarly, in G_y there is a node N_{ef} for each $e \in E_1$ and $f \in E_2$ and two nodes N_{ef} and $N_{e'f'}$ are connected by an edge iff the sharings (e, f) and (e', f') are not compatible. Then, a selection of x variables feasible for all noncrossing constraints corresponds to an independent set in G_x and a feasible set of sharings is an independent set in G_y . The maximum independent set in G_y is the optimal solution to CMO. All cuts valid for the independent set problem can be applied to the x and y variables, and, most notably, the clique inequalities.

The separation of the clique inequalities works as follows. Given a fractional solution x^* , we look for the *maximum weight clique* in G_x , where x_{ij}^* is the weight of N_{ij} . If the maximum weight clique weighs less than 1, then there are no violated clique inequalities, otherwise, we can add at least one such cut to the current LP.

Finding maximum weight cliques is a polynomial problem when the underlying graph is perfect. We can prove that

Theorem 5.1. *The graph G_x is perfect.*

There are algorithms for finding a max weighted clique in a weakly triangulated graph (as G_x can be shown to be) of time $O(n^5)$, due to Hayward, Hoang, Maffray [23] and Raghunathan [46]. However, for this specific graph we can do better and find max weighted cliques in time $O(n^2)$, thus making a huge difference in the practical solution of the problem.

We start by characterizing all cliques in G_x , i.e. sets of alignment lines which are all mutually crossing. We define the following notion of a *triangle*: this is a set of lines with one common endpoint and a second endpoint in a range of consecutive nodes, like $T(i, j|u) := \{[i, u], [i+1, u], \dots, [j-1, u], [j, u]\}$ where $i \leq j \in V_1$ and $u \in V_2$, and $T(i|j, u) := \{[i, j], [i, j+1], \dots, [i, u-1], [i, u]\}$ where $i \in V_1$ and $j \leq u \in V_2$. For S a set of lines, by $x(S)$ we denote the value $\sum_{[i,j] \in S} x_{ij}$.

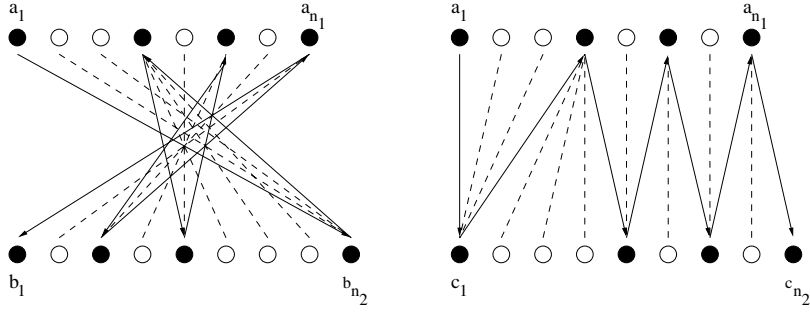


Fig. 3. Left: A zigzag path P (bold) and the set $T(P)$. Right: Same path after flipping V_2 .

Call $a_1, a_{n_1}, b_1,$ and b_{n_2} the set of *terminal* nodes. Consider a path P which passes through all the terminal nodes, and alternates nodes of V_1 and V_2 in a zig-zag fashion: That is, we can orient the path so that a_1 is the first of the nodes of V_1 visited by the path, and if a_k has been visited by the path, then all of the nodes in V_1 visited after a_k are to its right. Similarly, b_{n_2} is the first of the nodes of V_2 visited by the path, and if b_h has been visited by the path, then all of the nodes in V_2 visited after b_h are to its left. Note that any such path must start and end at a terminal node (see figure 5.1, Left), and must always include the lines $[a_1, b_{n_2}]$ and $[a_{n_1}, b_1]$. For each node of degree two in P , a triangle is defined by considering the set of lines incident on the node and contained within the two lines of the path. Let $T_A(P)$ ($T_B(P)$) be the set of triangles defined by P with tip in the nodes of V_1 (V_2) having degree two in P . We define $T(P) := T_A(P) \cup T_B(P)$. The following theorem characterizes the cliques of G_x .

Theorem 5.2. *A set Q of lines is a maximal clique in G_x if and only if there exists a zigzag path P such that $Q = T(P)$.*

The inequalities $x(T(P)) \leq 1$ for all zigzag paths P are therefore the strongest clique cuts for this particular independent set problem. We now show that to find the most violated such inequality in time $O(n^2)$. To make the following argument easier, we rename the nodes of V_2 as $\{c_1, \dots, c_{n_2}\}$, so that the leftmost node c_1 is b_{n_2} and the rightmost, c_{n_2} , is b_1 (i.e., we flip the nodes of V_2 with respect to the usual drawing). A zigzag path P now looks as a path which goes from left to right both in V_1 and V_2 . We call such a path a *leftright* path (see figure 5.1, Right).

With respect to the new drawing of W , orient each line $[a, c]$ in the two possible ways and, given a real vector x^* , define the length for each arc $(a, c) \in V_1 \times V_2$ and $(c, a) \in V_2 \times V_1$ as follows: $l(a, c) = x^*(T(a|1, c)) - x^*(T(1, a|c))$ and $l(c, a) = x^*(T(1, a|c)) - x^*(T(a|1, c))$. The lengths of four special arcs are defined separately, as $l(a_1, c_1) = 0$, $l(c_1, a_1) = 0$, $l(a_{n_1}, c_{n_2}) = x^*(T(a_{n_1}|1, c_{n_2}))$

and $l(c_{n_2}, a_{n_1}) = x^*(T(1, a_{n_1} | c_{n_2}))$. Now, consider a leftright path P starting with either the arc (a_1, c_1) or (c_1, a_1) and ending with either the arc (a_{n_1}, c_{n_2}) or (c_{n_2}, a_{n_1}) . Call $l(P)$ the standard length of this path, i.e. the sum of arcs lengths. We then have the following lemma.

Lemma 5.1. *For a leftright path P , $l(P) = x^*(T(P))$.*

Hence, to find the max-weight clique in G_x , we just need to find the longest leftright path. This can be computed effectively, by using Dynamic Programming. Call $V_{\setminus}(i, j)$ the length of a longest leftright path starting at a_i and using nodes of V_2 only within $c_j, c_{j+1}, \dots, c_{n_2}$. Also, call $V_{\succ}(i, j)$ the length of a longest zigzag path starting at c_j and using nodes of V_1 only within $a_i, a_{i+1}, \dots, a_{n_1}$. Then we have the following recurrences:

$$\begin{aligned} V_{\setminus}(i, j) &= \max\{l(a_i, c_j) + V_{\succ}(i+1, j), V_{\setminus}(i, j+1)\}, \\ V_{\succ}(i, j) &= \max\{l(c_j, a_i) + V_{\setminus}(i, j+1), V_{\succ}(i+1, j)\}. \end{aligned}$$

These recurrences can be then solved backwards, starting at (n_1, n_2) , in time $O(n^2)$.

Genetic Algorithms and Local Search

In a Branch-and-Cut algorithm, pruning of the search space happens whenever the upper bound ub to the optimal solution of a subproblem is smaller than a lower bound lb to the global optimum. As described before, ub is the value of the LP-relaxation. It is important then to have good (i.e., large) lower bounds as well. Any feasible solution gives a lower bound lb to the optimum. In our work we have developed heuristics of two types for generating good feasible solutions, i.e. *Genetic Algorithms* and *Steepest Ascent Local Search*.

A generic Genetic Algorithm (GA, [24, 16]) mimics an evolutionary process with a population of individuals that, by the process of mutation and recombination, gradually improve over time. For optimization problems, an individual is a candidate solution, mutation is a slight perturbation of the solution parameters, and recombination is a “merging” of two candidate solutions. The use of GAs for Structural Genomics problems is not new, e.g., GAs were used for protein fold prediction [27, 10].

Our application of the GA encodes the solutions as sets of alignment edges, associating a residue in one contact map graph with a residue in the other. A mutation will slightly shift one edge, and randomly add new edges in any available space, while recombination will pick edges out of two candidate solutions and create a new candidate solution using those edges.

Figure 5.1 shows two mutations. The first mutation shifts the alignment edges on the circled nodes to the right by one position, causing the right-most edge to be removed and a new edge to be inserted. The second mutation shifts the circled edges to the left by one position, causing the left-most shifted

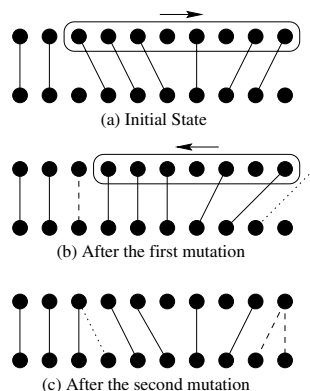


Fig. 4. The mutation operator. The top alignment shows the state before any mutations, the middle alignment shows the state after the first mutation but before the second mutation, the bottom alignment shows the state after both mutations. Dotted lines are edges that have been removed by a mutation, dashed lines are edges that have been added after a mutation is performed.

edge to be removed and a new edge to be randomly inserted on the right end – exactly one of the dashed lines is inserted.

The recombination operator is used to create a new candidate solutions (the child) from existing solutions. This is done by randomly selecting two existing solutions (the parents) by a standard GA method, i.e., randomly, but biased towards good solutions. A set of contiguous edges is randomly selected from one of the parents and is copied directly to the child. Next, all edges from the second parent that do not cross edges in the child are copied to the child. Finally, new edges are added in any available positions, exactly as was done with mutation.

Our Steepest Ascent Local Search heuristic algorithms follow the standard approach: Let s be a feasible current solution. The *neighborhood of s* is the set of solutions which can be obtained by applying a *move* to s . If all solutions in the neighborhood are no better than the current solution s , s is a local optimum and the search is terminated. Otherwise, the move that results in the best solution value is applied to s , and the search continues. Since converging to a local optimum is very fast, the search can be repeated many times, each time starting from a random feasible solution.

A feasible contact map alignment solution is identified by a pair of lists of the same size, (A, B) , where $A = (a_1 < \dots < a_k)$ are nodes from G_1 and $B = (b_1 < \dots < b_k)$ are nodes from G_2 . The alignment maps residue a_i in the first graph to residue b_i in the second graph (see Figure 5.1(a)).

Our two local search algorithms differ only in the definition of the moves that generate a neighborhood. The first algorithm uses moves that add a single specific line to the solution, removing any lines that cross the new line.

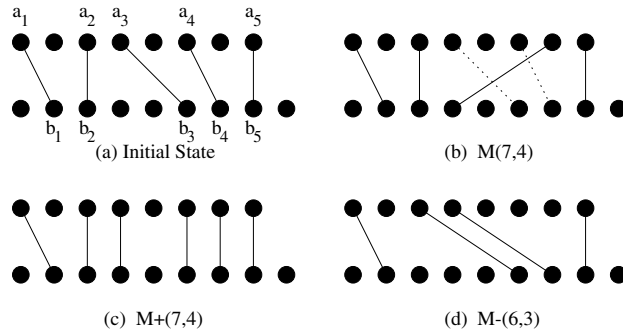


Fig. 5. Moves for the two algorithms. (a) Starting solution s . (b) A move from the first algorithm applied to s . (c) An increasing move applied to s . (d) A decreasing move applied to s .

Formally, the move $M(a, b)$ is defined for all $a \in G_1 - A$ and $b \in G_2 - B$ and will add the line $[a, b]$, removing all lines $[a_j, b_j]$ such that $a < a_j \wedge b > b_j$ or $a > a_j \wedge b < b_j$ (see Figure 5.1(b)). This move allows for big “jumps” in the solution space, by introducing very skewed lines and by removing many lines at once, and is suitable for instances in which the contact maps are not similar.

The second algorithm uses two types of moves, a *decreasing* move and an *increasing* move. The decreasing move, $M^-(a, b)$, defined for $a \in A$ and $b \in B$, simply removes a from A and b from B . Figure 5.1(d) shows the decreasing move $M^-(6, 3)$, which removes a_4 and b_2 from their respective lists. The increasing move, $M^+(a, b)$, where a and b are as defined for $M(a, b)$ in the first algorithm, adds a to A and b to B (see Figure 5.1(c)). These moves do not introduce very skewed lines easily and so are suited for similar proteins, in which good solutions are made of many parallel lines.

Computational Experiments

Our program has been implemented and run on some proteins from the PDB. This is the first time that exact solutions have been found for real instances of this problem. We have run our procedure on a set of 269 proteins, with sizes ranging from 64 to 72 residues and 80 to 140 contact each. The set was chosen to contain both a large number of similar proteins, as well as a large number of dissimilar proteins. An all-against-all computation would have resulted in 36046 alignments; we selected a subset of 597 alignments, so that there would be an equal number of similar pairs and dissimilar pairs.

In order to perform such a massive computation on a standard, single-processor Pentium PC, we have limited the time devoted to each individual instance to one hour (details can be found in [29]). Therefore, some problems have not been solved to optimality. However, even within the time limit

	Fold	Family	Residues	Seq. Sim.	RMSD	Proteins
1	Flavodoxin-like	CheY-related	124	15-30%	< 3Å	1b00, 1dbw, 1nat, 1nr, 1qmp, 1rnl, 3cah, 4tmy
2	Cupredoxins	Plastocyanin/ azurin-like	99	35-90%	< 2Å	1baw, 1byo, 1kdi, 1nin, 1pla, 2b3i, 2pcy, 2plt
3	TIM beta/ alpha-barrel	Triosephosphate isomerase	250	30-90%	< 2Å	1amk, 1aw2, 1b9b, 1btm, 1hti, 1tmh, 1tre, 1tri, 1ydv, 3ypi, 8tim
4	Ferritin-like	Ferritin	170	7-70%	< 4Å	1b71, 1bcf, 1dps, 1fha 1ier, 1rcd

Fig. 6. The Skolnick set.

set, we have been able to solve 55 problems optimally and for 421 problems (70 percent) the gap between the best solution found and the current upper bound was less than or equal to 5, thereby providing a strong certificate of near-optimality. The results also show the effectiveness of our lower bounding heuristic procedures, and in particular, of our genetic algorithm. The GA heuristic turned out to be superior to the others, finding 52 of the 55 optimal solutions.

In a second test, we used our programs to cluster proteins according to their Contact Map Overlap. I.e., given a set of proteins, we compute a normalized score based on their best CMO alignment, and check if pairs with high score are actually coming from the same family.

5.2 The Skolnick Clustering Test

A test set was suggested to us by Jeffrey Skolnick. The set contains 33 proteins classified by SCOP into four families: CheY-related, Plastocyanin/azurin-like, Triosephosphate isomerase and Ferritin (see Figure 5.1). Since these proteins are relatively large (beyond the capability of Branch-and-Cut exact solution, the problem is NP-hard after all), we used the heuristics that worked well for providing lower bounds in the Branch-and-Cut algorithm, i.e. GA and Local Search.

We applied the heuristics to all 528 contact map pairs, and clustered the proteins based on the following similarity score:

$$s_{ij} = \frac{c_{ij}}{\min\{m_i, m_j\}}$$

where m_i and m_j are the number of contacts in proteins i and j , and c_{ij} is the number of shared contacts in the solution found. Note that $0 \leq s_{ij} \leq 1$. A score of 1.0 would indicate that the smaller contact map is completely contained in the larger contact map. We found that 409 alignments with score between 0.0 and 0.313 were almost exclusively between contact maps in different families; 1.3% of these were alignments within the same family. The remaining 119 alignments, with score between 0.314 and 0.999, were all

between contact maps in the same family. Hence, we validated the CMO score for clustering with 98.7% accuracy (1.3% false negatives).

5.3 Conclusions

In order to promote the use of contact map overlap for structure comparison, more work should be devoted into designing effective algorithms for proteins of larger size than our current limit. Armed with such an algorithm, one could run an all against all alignment of the PDB structures, and cluster them according to their contact map similarity.

Our Branch-and-Bound and heuristic algorithms can be extended to new contact map-based similarity measures. For instance, one can consider the introduction of weights w_e on the contacts $e = \{i_1, i_2\}$. These weights can be based on the residues appearing at positions i_1 and i_2 , or on the distance between i_1 and i_2 in the folded protein. This would allow to model a situation in which some contacts are “more important” to preserve than others. Similarly, penalties u_{ij} can be used to weigh each residue-residue x_{ij} alignment. Finally, the objective function could be made parametric in the number of residues considered, so that the measure can be used, e.g., for local alignments. The problem would then read: for a given k find the best contact map alignment which maps exactly k residues of the first proteins in k of the second.

These and similar measures will be the object of our future research.

6 Acknowledgements

Support for Lancia was provided in part by the Italian Ministry of University and Research under the National Project “Bioinformatics and Genomics Research”, and by the Research Program of the University of Padova.

References

1. P. Argos and M. Rossmann. Exploring structural homology of proteins. *J. Mol. Biol.* **105** (1976) 75–95
2. F. Allen *et al.* Blue Gene: A vision for protein science using a petaflop supercomputer. *IBM System Journal* **40(2)** (2001) 310–321
3. H.M.Berman, J.Westbrook, Z.Feng, G.Gilliland, T.N.Bhat, H.Weissig, I.N.Shindyalov, P.E.Bourne, The Protein Data Bank. *Nucl. Ac. Res.* **28** (2000) 235–242
4. T. L. Blundell. Structure-based drug design. *Nature* **384** (1996) 23–26
5. C. Branden and J. Tooze, *Introduction to Protein Structure*. Garland, 1999
6. J. M. Bujnicki, A. Elofsson, D. Fischer and L. Rychlewski. LiveBench-1: Continuous benchmarking of protein structure prediction servers. *Prot. Sc.* **10** (2001) 352–361

7. D. F. Burke, C. M. Deane, and T. L. Blundell. Browsing the SLoop database of structurally classified loops connecting elements of protein secondary structure. *Bioinformatics* **16(6)** (2000) 513–519
8. T. E. Creighton, *Proteins: Structures and Molecular Properties*. Freeman, New York, 1993
9. S. Cristobal, A. Zemla, D. Fischer, L. Rychlewski and Arne Elofsson. How can the accuracy of protein models be measured? (2000) submitted
10. J. Devillers (ed), *Genetic Algorithms in Molecular Modeling*, Academic Press, London, 1996
11. L. E. Donate, S. D. Rufino, L. H. Canard, and T. L. Blundell. Conformational analysis and clustering of short and medium size loops connecting regular secondary structures: a database for modeling and prediction. *Protein Sci.* **5(12)** (1996) 2600–2616
12. D. Fischer et al.. CAFASP-1: Critical Assessment of Fully Automated Structure Prediction Methods. *Proteins* **Suppl 3** (1999) 209–217
13. <http://www.cs.bgu.ac.il/~dfisher/CAFASP2>
14. A. Godzik. The structural alignment between two proteins: Is there a unique answer?. *Prot. Sc.* **5** (1996) 1325–1338
15. A. Godzik, J. Skolnick and A. Kolinski, A topology fingerprint approach to inverse protein folding problem. *J. Mol. Biol.* **227** (1992) 227–238
16. D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, 1989
17. D. Goldman. *PhD Thesis*, U.C. Berkeley, 2000
18. D. Goldman, S. Istrail and C. Papadimitriou, Algorithmic Aspects of Protein Structure Similarity. *Proc. of the 40th IEEE Symposium on Foundations of Computer Science*, 512–522, 1999.
19. J. G. Gough, C. Chothia C., K. Karplus, C. Barrett and R. Hughey. Optimal Hidden Markov Models for all sequences of known structure. *Currents in Computational Molecular Biology* (Miyano S., Shamir R., Toshihisa T. eds.), Univ. Acad. Press Inc. Tokyo, 2000.
20. M. Grötschel, L. Lovász and A. Schrijver. The Ellipsoid Method and its Consequences in Combinatorial Optimization. *Combinatorica* **1** (1981) 169–197
21. T. F. Havel, G. M. Crippen and I. D. Kuntz, *Biopolymers* **18** (1979), 73
22. T. F. Havel, I. D. Kuntz and G. M. Crippen. The theory and practice of distance geometry. *Bull. Math. Biol.* **45** (1983) 665–720
23. R. B. Hayward, C. Hoang and F. Maffray. Optimizing Weakly Triangulated Graphs. *Graphs and Combinatorics* **5** (1987) 339–349
24. J. H. Holland. *Adaptation in Natural and Artificial Systems*. MIT Press, 1992
25. L. Holm and C. Sander. Protein Structure Comparison by Alignment of Distance Matrices. *J. Mol. Biol.* **233** (1993) 123–138
26. L. Holm and C. Sander. Mapping the protein universe. *Science* **273** (1996) 595–602
27. M. Khimasia and P. Coveney, Protein Structure prediction as a hard optimization problem: the genetic algorithm approach, 1997
28. Y. Lamdan, J. T. Schwartz and H. J. Wolfson. Object Recognition by Affine Invariant Matching. *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 335–344, 1988
29. 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. *Proc. of the 5th ACM REsearch in COMputational Biology*, 193–202, 2001
30. C. Lemmen and T. Lengauer, Computational methods for the structural alignment of molecules. *Journal of Computer–Aided Molecular Design* **14** (2000) 215–232
 31. A. M. Lesk. Computational Molecular Biology. In: *Encyclopedia of Computer Science and Technology* (A. Kent, J. Williams, C. M. Hall and R. Kent eds.) **31** (1994) 101–165
 32. M. Levitt and M. Gerstein. A Unified Statistical Framework for Sequence Comparison and Structure Comparison. *Proc. Natl. Acad. Sc.* **95** (1998) 5913–5920
 33. S. Lifson and C. Sander, *Nature* **282** (1979), 109
 34. G. M. Maggiora, B. Mao, K. C. Chou and S. L. Narasimhan. Theoretical and Empirical Approaches to Protein–Structure Prediction and Analysis. In: *Methods of Biochemical Analysis* **35** (1991) 1–60
 35. A. Marchler–Bauer and S. H. Bryant. Comparison of Prediction Quality in the Three CASPS. *Proteins Suppl* **3** (1999) 218–225
 36. McLaughlan, *Acta Crystallogr.* (1979)
 37. L. Mirny and E. Domany. Protein fold recognition and dynamics in the space of contact maps. *Proteins* **26** (1996) 391–410
 38. K. Mizuguchi, C. M. Deane, T. L. Blundell, M. S. Johnson and J. P. Overington. JOY: protein sequence–structure representation and analysis. *Bioinformatics* **14** (1998) 617–623
 39. K. Mizuguchi, C. M. Deane, T. L. Blundell, and J. P. Overington. HOMSTRAD: a database of protein structure alignments for homologous families. *Protein Sci.* **7(11)** (1998) 2469–2471
 40. J. Moult, T. Hubbard, S. Bryant, K. Fidelis, J. Pedersen and Predictors. Critical Assessment of Methods of Proteins Structure Prediction (CASP): Round II. *Proteins Suppl* **1** (1997) dedicated issue
 41. J. Moult, T. Hubbard, K. Fidelis, and J. Pedersen. Critical Assessment of Methods of Protein Structure Prediction (CASP): Round III. *Proteins Suppl* **3** (1999) 2–6
 42. 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. *J. Mol. Biol.* **247** (1995) 536–540
 43. G. L. Nemhauser and L. Wolsey. *Integer and Combinatorial Optimization*, John Wiley and Sons, 1988
 44. J. P. Overington, M.S. Johnson, A. Sali, and T.L. Blundell, Tertiary structural constraints on protein evolutionary diversity; Templates, key residues and structure prediction. *Proc. Roy. Soc. Lond.* **B 241** (1990) 132–145
 45. K. Park, M. Vendruscolo and E. Domany, Toward an Energy Function for the Contact Map Representation of Proteins. *PROTEINS: Structure, Function and Genetics* **40** (2000) 237–248
 46. A. Raghunathan. Algorithms for Weakly Triangulated Graphs, U.C. Berkeley Tech. Rep., CSD-89-503 (1989)
 47. G. Rhodes, *Crystallography Made Crystal Clear*. Academic Press, 2nd Ed., 1999
 48. R. B. Russell, G. J. Barton. Multiple protein sequence alignment from tertiary structure comparison. *PROTEINS: Struct. Funct. Genet.* **14** (1992) 309–323
 49. A. Sali and T. L. Blundell. Definition of general topological equivalence in protein structures. A procedure involving comparison of properties and relationships

- through simulated annealing and dynamic programming. *J. Mol. Biol.* **212**(2) (1990) 403-428
50. N. Siew, A. Elofsson, L. Rychlewski and D. Fischer, MaxSub: An Automated Measure for the Assessment of Protein Structure Prediction Quality. *Bioinformatics* **16** (2000) 776-785
 51. A. P. Singh and D. L. Brutlag 1998. Protein Structure Alignment: A comparison of methods. *Bioinformatics* (2000), Submitted.
 52. T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *J. Mol. Biol.* **147** (1981) 195-197
 53. R. Sowdhamini, D. F. Burke, J. F. Huang, K. Mizuguchi, H. A. Nagarajaram, N. Srinivasan, R. E. Steward, and T. L. Blundell. CAMPASS: a database of structurally aligned protein superfamilies. *Structure* **6**(9) (1998) 1087-1094
 54. M. J. Sutcliffe, I. Haneef, D. Carney and T. L. Blundell. *Prot. Eng.* **1** (1987) 377-384
 55. S. Umeyama. Least-Squares Estimation of Transformation Parameters Between Two Point Patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-13(4) (1991) 376-386
 56. M. Vendruscolo, E. Kussell and E. Domany, Recovery of protein structure from contact maps. *Fold. Des.* **2** (1997) 295-306
 57. M. Vendruscolo, R. Najmanovic and E. Domany, Protein Folding in Contact Map Space. *Phys. Rev. Lett.* **82**(3) (1999) 656-659
 58. M. Vendruscolo, B. Subramanian, I. Kanter, E. Domany and J. Lebowitz, Statistical Properties of Contact Maps. *Phys. Rev. E* **59** (1999) 977-984
 59. A. Zemla, C. Venclovas, J. Moult and K. Fidelis. Processing and Analysis of CASP3 Protein Structure Predictions. *Proteins Suppl* **3** (1999) 22-29