PROTEIN STRUCTURE COMPARISON VIA CONTACT MAP ALIGNMENT

FELIPE LEAL VALENTIM

2010
PROTEIN STRUCTURE COMPARISON VIA CONTACT MAP ALIGNMENT

Dissertação apresentada à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Biotecnologia Vegetal, área de Concentração em Biotecnologia Vegetal, para obtenção do título de “Mestre”.

Orientador
Prof. Ricardo Martins de Abreu Silva

LAVRAS
MINAS GERAIS -BRASIL
2010
Valentim, Felipe Leal.
Protein Structure Comparison via Contact Map Alignment / Felipe Leal Valentim. – Lavras : UFLA, 2010.
77 p. : il.

Orientador: Ricardo Martins de Abreu Silva.
Bibliografia.


CDD – 574.0285
FELIPE LEAL VALENTIM

PROTEIN STRUCTURE COMPARISON VIA CONTACT MAP ALIGNMENT

Dissertação apresentada à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Biotecnologia Vegetal, área de Concentração em Biotecnologia Vegetal, para obtenção do título de “Mestre”.

APROVADA em 05 de Fevereiro de 2010

Prof. Luciano Vilela Paiva - UFLA

Prof. Wagner Meira Junior - UFMG

Prof. Gloria Regina Franco - UFMG

Prof. Ricardo Martins de Abreu Silva
UFLA
(Orientador)

LAVRAS
MINAS GERAIS - BRASIL
# SUMÁRIO

ABSTRACT ................................................................. i
RESUMO ................................................................. ii
CHAPTER 1: General Introduction ................................. 1
1 Protein structures and Contact Maps ......................... 2
1.1 Availability of structural data of proteins ............... 3
1.2 Protein Contact Maps ........................................... 3
2 Protein Structure Comparisons ................................... 5
2.1 Contact Map Alignment ......................................... 6
3 Objectives and structure of the thesis ....................... 7
4 References ........................................................... 8

CHAPTER 2: GRASP with Path-Relinking for the MAX-CMO problem 10
1 Abstract ............................................................. 11
2 Resumo ............................................................... 12
3 Introduction .......................................................... 13
4 GRASP with Path-relinking for MAX-CMO ................. 20
4.1 Greedy randomized construction ............................ 23
4.2 Approximate local search ..................................... 24
4.3 Path-relinking ..................................................... 35
5 Computational experiments and Results ..................... 36
5.1 Test environment and Datasets ............................... 37
5.2 Comparison of the GRASP-PR heuristic with other algorithms 37
5.3 Time-to-target plots for GRASP-PR against other heuristics 44
5.4 GRASP-PR and the Skolnick Clustering test set .......... 47
5.5 GRASP-PR as a structural alignment tool .................. 49
6 Concluding remarks ................................................ 56
7 References .......................................................... 57

CHAPTER 3: GOIC-Biocomp: a web-based tool for protein alignment 61
1 Abstract ............................................................. 62
2 Resumo ............................................................... 63
3 GRASP with Path-relinking for MAX-CMO .................... 64
4 GOIC-Biocomp server ............................................. 67
5 References .......................................................... 69

CHAPTER 4: Summary, Conclusions and Future Work .......... 71
1 Abstract ............................................................. 72
2 Resumo ............................................................... 73
3 Summary, General Conclusions and Future Work .......... 74
4 References .......................................................... 77
ABSTRACT

VALENTIM, Felipe Leal. **Protein Structure Comparison via Contact Map Alignment.** 2010. 77 p. Master thesis (Master in Plant Biotechnology) - Universidade Federal de Lavras, Lavras.

Proteins are primary components in almost all biological processes in living organisms. It is known that the variety of protein functions is a result of the differences in protein structures. Therefore, understanding and comparing the structure of proteins is a major challenge in modern molecular biology. The structural alignment and comparison of proteins became an essential task, whose solution is instrumental in aiding other problems such as drug design, protein structure/function prediction, and protein clustering. One promising class of approaches for measuring protein similarity relies on the alignment of the protein contact maps. The most common mathematical statement of the contact map comparison problem is called the Maximum Contact Map Overlap (MAX-CMO). In this context, in this Master’s thesis it has been proposed the hybrid heuristic Greedy Random Adaptive Search Procedure with Path-relinking for the Maximum Contact Map overlap problem which have been revealed able to find improved solutions. Another proposal which has been presented in this work is the implementation of a computational tool that allows the structural alignment of proteins through the proposed heuristic. The chapters 2 and 3 of this dissertation represent the manuscripts describing these two proposals and a final chapter that contains the conclusion and outlines the possibilities for future work.

**Keywords:** Proteins, Structural alignment, Contact Maps, MAX-CMO.

---

*Advisor: Ricardo Martins de Abreu Silva - UFLA*
RESUMO

VALENTIM, Felipe Leal. **Protein Structure Comparison via Contact Map Alignment.** 2010. 77 p. Dissertação (Mestrado em Biotecnologia Vegetal) - Universidade Federal de Lavras, Lavras.*

As proteínas são componentes primários em quase todos os processos biológicos nos organismos vivos. Sabe-se que a variedade de funções de proteínas é um resultado das diferenças nas estruturas de proteínas. Portanto, compreender e comparar a estrutura das proteínas é um desafio importante na biologia molecular moderna. O alinhamento estrutural e comparação das proteínas tornaram-se tarefas essenciais, cuja solução é fundamental para auxiliar outros problemas, tais como o desenho racional de novos fármacos, predição de função/estrutura de proteínas, e clusterização de proteínas. Uma classe promissora de abordagens para medir a similaridade da proteína depende do alinhamento dos mapas de contato da proteína. A formalização mais comum para o problema matemático de alinhamento de mapas de contato é chamado o Maximum Contact Map Overlap problem (MAX-CMO). Neste contexto, esta dissertação de mestrado propõe a heurística híbrida Greedy Random Adaptive Search Procedure com Path-relinking para o Maximum Contact Map Overlap problem, que tem se revelado capaz de encontrar soluções promissoras. Outra proposta apresentada neste trabalho é a implementação de uma ferramenta computacional que permite o alinhamento estrutural de proteínas através da heurística proposta. Os capítulos 2 e 3 desta dissertação representam os artigos que descrevem estas duas propostas. Um capítulo final descreve experimentos adicionais realizados com a heurística e a ferramenta computacional.

**Palavras-chave:** Proteínas, Alinhamento estrutural, mapas de contato, MAX-CMO.

---

*Orientador: Ricardo Martins de Abreu Silva - UFLA*
CHAPTER 1

GENERAL INTRODUCTION
1 PROTEIN STRUCTURES AND CONTACT MAPS

Proteins are primary components in almost all biological processes in living organisms. It is known that the variety of protein functions is a result of the differences in protein structures. Therefore, understanding and comparing the structure of proteins is a major challenge in modern molecular biology (Eidhammer et al., 2004).

Despite the large amount of diversity in their functions, all proteins are made of the same components, amino acids. And all amino acids share the same basic structure. Each amino acid consists of a central carbon atom \( (C_{\alpha}) \), an amino group \( (NH_3) \), at one end, a carboxyl group \( (COOH) \) at the other end, and a side-chain \( (R) \) that characterizes the amino acids. This side-chain is usually referred to as an amino acid residue, or simply a residue (Eidhammer et al., 2004).

In order to form a protein molecule, the carboxyl group of one amino acid forms a peptide bond with the amino group of another amino acid and an \( (H_2O) \) molecule is revealed. The sequence of peptide bonds forms the protein backbone. There are 20 different side-chains specified by genetic code each of which is addressed by a letter of the alphabet. Since each protein is a sequence of amino acids, it can be described by a string over this set of 20 letters (Eidhammer et al., 2004).

The structure of proteins is organized in four structural levels: primary, secondary, tertiary, and quaternary structures. The linear sequence of amino acids that contribute to the formation of a protein molecule is called its primary structure. Many proteins contain roughly 100-1000 amino acids (Eidhammer et al., 2004) (some even more than 4000). Local arrangement of a few or a few dozen amino acid residues (Hunter, 1993) is seen in particular patterns repeatedly in many different proteins. These patterns are formed because of the interactions mediated by hydrogen bonds mainly within the backbone. The three-dimensional fold of the
protein molecule - which is a result of connecting secondary structures together - is called tertiary structure of the protein (Eidhammer et al., 2004). There are many proteins in nature which form from combinations of two or more protein chains. The spatial arrangement of these proteins is called quaternary structure. In this work, we are particularly interested in the tertiary structure of proteins, and when we refer to the protein structure in what follows, we will be referring to their tertiary ones.

1.1 Availability of structural data of proteins

The Protein Data Bank (PDB) (Berman et al., 2000) is the standard repository for collecting information on determined three-dimensional structures of proteins and other large biological molecules which are found in all of the living organisms. The coordinates of the structures in the PDB are determined by some experimental methods such as X-ray crystallography and Nuclear Magnetic Resonance (NMR) spectroscopy*. A rapid increase in the number of protein structures deposited in the PDB has been observed in recent years, and because of this growth, protein structure comparison has become a key problem in bioinformatics. This rapid growth has been related to the recent emergence of large scale protein structure determination projects (Nair et al., 2009), called structural genomics (Westbrook et al., 2003).

1.2 Protein Contact Maps

Three-dimensional structure of proteins can be represented by their distance maps. A distance map is an \( N \times N \) matrix, where \( N \) is the number of amino acids in the sequence of a protein. Each element \( d_{i,j} \) in the matrix \( D \) represents the distance between the \( i^{th} \) and the \( j^{th} \) amino acids, usually in Angstrom (Å). The distance between two residues can be defined in different ways, such as the

*More information about the PDB can be found in http://www.wwpdb.org/documentation/
FIGURE 1 At the top, the protein PDB-ID:1am6 as taken from the PDB (Berman et al., 2000). On the left below, the extracted Distance Map of one chain of the protein. And on the right, Contact map of the same protein by applying the threshold of 6.5Å. Each pixel in this map indicates that the two corresponding amino acids are within the distance of 6.5Å of each other.

Contact maps are a thresholded version of distance maps. The contact map of a folded protein with $N$ residues is a binary matrix $N \times N$ of all pairwise distances within that protein. Two residues are said to be in contact if the distance between their $C\alpha$ is not greater than a presumed threshold (typically in range 5Å-12Å) (Vullo & Frasconi, 2003) - see Figure 1.

Different secondary structures can be recognized in contact maps through their special patterns. In particular, $\alpha$-helices appear as thick bands along the
main diagonal, while $\beta$-sheets appear as thin bands parallel or perpendicular to the main diagonal (Glasgow et al., 2006). Therefore, the contact map is a minimalist representation of a protein native three-dimensional structure (Krasnogor, 2004). This property leads to the idea that if two protein contact maps are similar to each other, their corresponding proteins have similar structures as well.

2 PROTEIN STRUCTURE COMPARISONS

Protein structure comparison has become a key problem in bioinformatics, improving researches that seek to find functional/evolutionary relationship among proteins and leading scientists in tasks such as protein function determination (Wolfon et al., 2005), rational drug (Wieman et al., 2004) design, the assessment of fold prediction (Goldsmith-Fischman & Honig, 2003), or protein clustering and classification (Dietmann et al., 2001). Moreover, structural alignment is a valuable tool for the comparison of proteins with low sequence similarity, where evolutionary relationships between proteins can not be easily detected by standard sequence alignment techniques (CAPRARA et al., 2000; Balaji & Srinivasan, 2007). A basic problem in pairwise protein structure comparison is finding a scoring scheme for similarity. Currently, most of the scoring schemes use the information about three-dimensional coordinates of protein structures, or their two-dimensional representations as distance maps. Another large class of approaches for measuring protein similarity relies on mutual comparison of contact maps. These methods are based on the hypothesis that similarity in protein contact maps results in similarity in protein structures. In this work, we focus on this approach of alignment of contact maps.
2.1 Contact Map Alignment

A large class of methods for protein structure comparison scores the similarity of proteins by comparing their binary contact maps. These approaches are based on the hypothesis that contact maps capture important information about the native structure of proteins (Krasnogor et al., 2003). Thus, the similarity between contact maps results in similarity between protein structures. The most common mathematical statement of the contact map comparison problem is called the Maximum Contact Map Overlap (Greenberg et al., 2004) (MAX-CMO). In the formulation of this problem, contact maps are interpreted as adjacency matrices of graphs. Each protein is represented by a graph whose nodes correspond to one of the amino acids of that protein. There is an edge between two nodes of the graph whenever their corresponding amino acids are in contact, i.e., their positions in the three-dimensional structure of the protein are within a specified distance of one another. The problem is now to calculate the similarity of proteins by aligning the two contact map graphs. The alignment value (i.e. the amount of similarity) is determined by the size of the common subgraph, which is identified by the alignment, that is, the number of edges connecting two equivalent nodes in both graphs. Chapter 2 details this problem.

3 OBJECTIVES AND STRUCTURE OF THE THESIS

This project main objective, which proposal is detailed in Chapter 2, is the development of a novel and efficient heuristic for the Maximum Contact Map Overlap problem. The chapter 2 represents the manuscript describing this proposal - introduction, review, formalizations and mathematical modeling of the MAX-CMO problem; methodology development; our proposal presented as the GRASP-PR algorithm for the MAX-CMO problem; as well as presenting the
results of performed analyses. This manuscript meets the requirements of the
journal “IEEE/ACM Transactions on Computational Biology and Bioinformatics”
(TCBB) for publishing an article of type “Regular Paper”.

A secondary objective – presented in Chapter 3 as a brief manuscript – is
the implementation of a computational tool that allows the structural alignment
of proteins through the proposed heuristic and also other successful algorithms
described in subsequent chapters. This second manuscript meets the requirements
and restrictions of the journal “Bioinformatics” (Oxford journals) to publish an
article of type "Application Note".

The fourth and final Chapter contains the conclusion and outlines the possi-
bilities for future work.
4 REFERENCES


CHAPTER 2

GRASP WITH PATH-RELINKING FOR THE MAXIMUM CONTACT MAP OVERLAP PROBLEM
1 ABSTRACT

Structural alignment emerged as a valuable tool for the comparison of proteins with low sequence similarity, since structurally similar but sequentially unrelated proteins have been discovered and rediscovered by many researchers. Recently, the growth of the Protein Data Bank has been accelerated by a large scale structure determination projects, and thus, fast and efficient algorithms for protein structure comparison has become more important to take advantage of the huge amount of structural data. There exist several approaches to perform the structural alignment, being the solution of the Maximum Contact Map Overlap problem one efficient available alternative. Although Maximum Contact Map Overlap problem may be solved using exact algorithms, simple approximate algorithms that obtains good quality solutions using less computational resources and time are still required. This paper proposes a variant of the greedy randomized adaptive search procedure with path-relinking (GRASP-PR) for MAX-CMO. Computational experiments are performed comparing a GRASP-PR heuristic with other algorithms from literature on real and simulated data. The GRASP-PR heuristic effectiveness is analyzed, demonstrating that our approach is a promising strategy to resolve the problem.

Keywords: Structural alignment, maximum contact map overlap problem, GRASP with Path-Relinking.
O alinhamento estrutural emergiu como uma valiosa ferramenta para comparação de proteínas com baixa similaridade de sequência, visto que proteínas com estruturas semelhantes mas sequencialmente não relacionadas têm sido descobertas e redescobertas por muitos pesquisadores. Recentemente, o crescimento do banco de dados da proteínas foi acelerado por uma grande escala de projetos de determinação estrutural e, assim, algoritmos rápidos e eficientes para a comparação da estrutura da proteína têm se tornado mais importantes para tomarem vantagem sobre a enorme quantidade de dados estruturais. Existem diversas abordagens para realizar o alinhamento estrutural, sendo a solução do Maximum Contact Map Overlap problem uma eficiente alternativa disponível. Embora o Maximum Contact Map Overlap problem possa ser resolvido utilizando algoritmos exatos, simples algoritmos aproximados que obtêm soluções de boa qualidade utilizando menos recursos computacionais e tempo continuam necessários. Este artigo propõe uma variação da heurística greedy randomized adaptive search procedure com path-relinking (GRASP-PR) para o MAX-CMO. Experimentos computacionais são realizados comparando o GRASP-PR contra outros algoritmos da literatura em dados simulados e reais. A eficiência da heurística GRASP-PR é analizada, demonstrando que nossa proposta é uma estratégia promissora para resolver o problema.

**Palavras-chave:** Alinhamento estrutural, maximum contact map overlap problem, Greedy Random Adaptive Search Procedure com Path-relinking.
3 INTRODUCTION

Proteins are organic compounds that play an important role in nearly all cell processes, including metabolic, immunological, cell signaling, and regulation of the cell cycle. Proteins are made up of amino acids arranged in a linear chain and folded in a three-dimensional form. An amino acid is a molecule containing an amino group, a carboxyl group, and a side chain usually referred to as an amino acid residue, or simply a residue. One can think of a protein as being made up of a backbone with hanging residues (see Figure 1). Note that although two residues may be far apart in the backbone, because of the three-dimensional form of the protein, they may actually be close together. From the 20 standard amino acid building blocks, perhaps millions of proteins exist in nature, most of which currently have unknown function.

Protein structure alignment has become a standard structural analysis tool providing similarity measures between the structures. Protein structure similarity may indicate functional/evolutionary relationship that usually leads scientists in tasks such as protein function determination (Wolfson et al., 2005), rational drug (Wieman et al., 2004) design, assessment of fold (Goldsmith-Fischman & Honig, 2003) prediction or protein clustering and classification (Dietmann et al., 2001). Recently, the growth of the Protein Data Bank (PDB) (Berman et al., 2000) has been accelerated by a large scale structure determination projects, called structural genomics (Westbrook et al., 2003). As a result, fast and efficient algorithms for protein structure comparison have become more important to take advantage of the huge amount of structural data.

One promising way of accomplishing the structural alignment is to evaluate the alignment of their contact maps. A protein contact map is used to represent the distances between every pair of residues in a three-dimensional protein structure.
The distance between two residues is usually defined as being either the smallest Euclidean distance between the points where the two residues connect to the backbone structure or as the smallest Euclidean distance between any pair of atoms in the residues.

FIGURE 1 A protein can be viewed as a chain of hanging residues.

A contact map (see Figure 2) consists of either a graph or a two-dimensional matrix (binary or real). The graph representation (Figure 2.c) shows the contact map as a graph with a sequence of nodes corresponding to the sequence of residues and an edge for each pair of non-consecutive residues whose distance is below a given threshold. The length of a contact map in the graph representation is defined by the number of nodes in the graph.

For a protein with \( n \) residues \( \{1, 2, \ldots, n\} \), the binary matrix representation (Figure 2.a) is a square \((0, 1)\) \( n \times n \) matrix \( C \) where the element \( C_{i,j} \), for \( i, j = 1, \ldots, n \ (i \neq j) \), indicates whether the distance \( \delta_{ij} \) between non-consecutive residues \( i \) and \( j \) is less than a predefined distance threshold \( t \), i.e.

\[
C_{i,j} = \begin{cases} 
1 & \text{if } \delta_{ij} < t \text{ and } |j - i| > 1; \\
0 & \text{otherwise.}
\end{cases}
\]
With respect to the real matrix representation (shown in Figure 2.b), a contact map is a square real-valued matrix $C$, where $C_{i,j} = \delta_{ij}$.

Contact maps provide a more compact representation of the protein structure than its corresponding three dimensional atomic coordinates. The advantage is that contact maps are invariant to rotations and translations, both favorable properties for the comparison of protein structures. For more detail, the reader is referred to (Bartoli et al., 2008). In the remainder of this paper, we restrict our attention to the contact map graph representation. The terms nodes in a contact map and residues in a protein will be interchangeable.

To determine the similarity of two proteins requires the definition of a metric. In this paper we use two similarity metrics. This first, called contact map overlap (Goldman et al., 1999) is illustrated in Figure 3. This figure is derived from a
similar figure in (Lancia et al., 2001). The alignment shows the residues selected in the subgraphs (nodes 1, 2, 4, 5, 6, 7, and 8 from $V_A$ and nodes 1, 2, 3, 5, 7, 9, and 10 from $V_B$). The linear ordering is preserved by associating $1 \leftrightarrow 1$, $2 \leftrightarrow 2$, $4 \leftrightarrow 3$, $5 \leftrightarrow 5$, $6 \leftrightarrow 7$, $7 \leftrightarrow 9$, and $8 \leftrightarrow 10$. The corresponding edges in the isomorphic graphs are solid and color matched. These edges satisfy the condition that their endpoints are associated. For example, edge $(1, 4)$ in $E_A$ corresponds to edge $(1, 3)$ in $E_B$ because of node associations $1 \leftrightarrow 1$ and $4 \leftrightarrow 3$.

![Contact map alignment diagram](image)

**FIGURE 3** Example of contact map alignment. Isomorphic subgraphs have five (solid color-matched) edges each (Lancia et al., 2001).

Figure 4 shows an optimal alignment (overlapped edges are identified by red lines) between the contact maps for distinct proteins $1hlm^*$ (Figure 4.a) and $1ahs^†$ (Figure 4.b). The alignment value is defined as the number of edges of each subgraph identified by the alignment, i.e., the set of corresponding edges in each graph.

[^*]: [http://www.rcsb.org/pdb/explore/expore.do?structureId=1HLM](http://www.rcsb.org/pdb/explore/expore.do?structureId=1HLM)

[^†]: [http://www.rcsb.org/pdb/explore/expore.do?structureId=1AHS](http://www.rcsb.org/pdb/explore/expore.do?structureId=1AHS)
FIGURE 4 Three dimensional native structures for proteins (a) 1ash and (b) 1hlm as taken from the Protein Data Bank (PDB) (Berman et al., 2000), and (c) an optimal alignment of value 279 of two 6.5Å threshold contact maps of the proteins. The optimal value was determined in (Xie & Sahinidis, 2007) with a branch and bound algorithm. This alignment was generated with a pure GRASP (without path-relinking) heuristic. The figure was created with a modified version of the java program BuildContactMapFromPDB available at http://www.cs.nott.ac.uk/~nxk/USM/protocol.html.

Given two contact maps $G_A = (V_A, E_A)$ and $G_B = (V_B, E_B)$ such that $|V_A| = n$ and $|V_B| = m$, the CONTACT MAP OVERLAP PROBLEM (Goldman et al., 1999) is to find two subsets $S_A \subseteq V_A$ and $S_B \subseteq V_B$ with $|S_A| = |S_B|$ and an order preserving bijection $f$ between $S_A$ and $S_B$ such that the cardinality of the overlap set

$$\mathcal{L}(S_A, S_B, f) = \{(u, v) \in E_A : u, v \in S_A, (f(u), f(v)) \in E_B\}$$
is maximized. A solution \((S_A, S_B, f)\) for the contact map overlap problem can be represented as an assignment vector \(p\) of size \(n\) such that

\[
p_u = \begin{cases} 
v & \text{if } (u, v) \in \mathcal{L}(S_A, S_B, f) \\
\text{nil} & \text{otherwise.}
\end{cases}
\]

We later refer to the cardinality \(|\mathcal{L}(S_A, S_B, f)|\) of the overlap set defined by \(p\) as \(\Lambda(p)\).

The contact map overlap problem was shown by (Goldman et al., 1999) to be NP-hard and can be formulated as a \((0, 1)\) integer program (Greenberg et al., 2004). Define the binary variable \(x_{ij} = 1\) if and only if node \(i \in V_A\) is associated with node \(j \in V_B\), and define the binary variable \(y_{(i,k)(j,l)} = 1\) if and only if \((i, k) \in E_A\) and \((j, l) \in E_B\) are corresponding edges in the isomorphic subgraphs.

The objective of the CMO problem is to maximize

\[
\sum_{(i,k) \in E_A, (j,l) \in E_B} y_{(i,k)(j,l)}.
\]

The selected edges must have their endpoints associated in such a way that

\[
y_{(i,k)(j,l)} = 1 \Rightarrow x_{ij} = x_{kl} = 1,
\]

i.e.

\[
y_{(i,k)(j,l)} \leq x_{ij}, \\
y_{(i,k)(j,l)} \leq x_{kl},
\]

for all \((i, k) \in E_A\) and \((j, l) \in E_B\). Furthermore, at most one node in one graph
can be associated with a node in the other graph, i.e.:

\[
\sum_{i \in V_A} x_{ij} \leq 1, \forall j \in V_B, \\
\sum_{j \in V_B} x_{ij} \leq 1, \forall i \in V_A.
\]

Finally, any two associations (edges with one endpoint in \(V_A\) and the other in \(V_B\)) cannot cross, i.e.:

\[
x_{ij} + x_{kl} \leq 1, \text{ for } 1 \leq i < k \leq |V_A| \text{ and } 1 \leq l < j \leq |V_B|.
\]

The second similarity measure used in this paper is the root mean square deviation (RMSD). This measure is not used directly in our heuristic. Instead, it is used only to verify the quality of the solutions found by the heuristic. The RMSD is the average distance between the backbones of superimposed proteins. We use the tool Biopython of (Cock et al., 2009) to compute the RMSD values.

The contact map overlap problem was introduced in (Godzik et al., 1992). Several exact algorithms as well as heuristics have been since proposed for this problem. (Lancia et al., 2001) describe a branch and cut strategy that employs lower-bounding heuristics at the branch nodes. (Caprara & Lancia, 2002) proposed a Lagrangian relaxation approach, where the optimal Lagrange multipliers are found by subgradient optimization. (Carr et al., 2002) proposed a memetic evolutionary algorithm. (Xie & Sahinidis, 2007) used dynamic programming as tool to design a branch-and-bound algorithm with several reduction techniques to eliminate inferior residue-residue pairs early in the search procedure. (Pelta et al., 2008) proposed three versions of a multi-start variable neighborhood search heuristic for solving MAX-CMO.
The remainder of the paper is organized as follows. In Section 4 we present the GRASP with path-relinking heuristic for the MAX-CMO problem. Computational experiments are described in Section 5.2. Finally, concluding remarks are made in Section 6.

4 GRASP WITH PATH-RELINKING FOR MAX-CMO

A GRASP heuristic (Feo & Resende, 1995; Resende & Ribeiro, 2003) is a multi-start procedure in which a greedy randomized solution is constructed to be used as a starting solution for local search for all iterations. Local search repeatedly substitutes the current solution by a better solution in the neighborhood of the current solution. If there is no better solution in the neighborhood, the current solution is declared a local maximum and the search stops. The best local maximum found over all GRASP iterations is output as the solution.

GRASP iterations are independent, i.e. solutions found in previous GRASP iterations do not influence the algorithm in the current iteration. The use of previously found solutions to influence the procedure in the current iteration can be thought of as a memory mechanism.

One way to incorporate memory into GRASP is with path-relinking (Glover, 1996). In GRASP with path-relinking (Laguna & Martí, 1999; Resende & Ribeiro, 2005), an elite set of diverse good-quality solutions is maintained to be used in all GRASP iterations. After a solution is produced with greedy randomized construction and local search, that solution is combined with a randomly selected solution from the elite set using the path-relinking operator. The combined solution is then considered apt to be included in the elite set. Ultimately, it is added to the elite set if it meets quality and diversity criteria.
procedure GRASP+PR-CMOP
Input: $C^A, C^B$
Output: solution $p^*$

\[ P \leftarrow \emptyset; \]

\[ \text{while stopping criterion not satisfied do} \]

\[ p \leftarrow \text{GreedyRandomized}(); \]

\[ p \leftarrow \text{ApproximateLocalSearch}(p); \]

\[ \text{if } P \text{ is full then} \]

\[ \text{Randomly select a solution } q \in P; \]

\[ r \leftarrow \text{PathRelinking}(p, q); \]

\[ r \leftarrow \text{ApproximateLocalSearch}(r); \]

\[ \text{if } c(r) > \max\{c(s) : s \in P\} \text{ then} \]

\[ t \leftarrow \arg\min\{\Delta(r, s) : s \in P\}; \]

\[ P \leftarrow P \cup \{r\} \setminus \{t\}; \]

\[ \text{else if } c(r) > \min\{c(s) : s \in P\} \text{ and } r \notin P \text{ then} \]

\[ t \leftarrow \arg\min\{\Delta(r, s) : s \in P : c(s) < c(r)\}; \]

\[ P \leftarrow P \cup \{r\} \setminus \{t\}; \]

\[ \text{end} \]

\[ \text{else} \]

\[ \text{if } P = \emptyset \text{ then} \]

\[ P \leftarrow \{p\}; \]

\[ \text{else if } p \notin P \text{ then} \]

\[ P \leftarrow P \cup \{p\}; \]

\[ \text{end} \]

\[ \text{end} \]

\[ \text{return } p^* = \arg\max\{c(s) : s \in P\}; \]

\textbf{Algorithm 1}: Pseudo-code of the GRASP-PR heuristic for MAX-CMO.
Algorithm 1 shows pseudo-code for the GRASP with path-relinking heuristic for the MAX-CMO problem. The algorithm takes two contact maps as input $C^A$ and $C^B$ of proteins $A$ and $B$, with $n$ and $m$ residues ($m > n$), respectively. It outputs an array $p^*$ of length $n$, with $p^*_i = \text{nil}$, if node $i \in C^A$ representing residue $i \in A$ is not aligned, and $p^*_i = j$, if node $i \in C^A$ is aligned with node $j \in C^B$.

After initializing the elite set $P$ as empty in line 1, the GRASP with path-relinking iterations are computed in lines 2 to 26 until a stopping criterion is satisfied. This criterion could be, for example, a maximum number of iterations, a target solution quality, or a maximum number of iterations without improvement. During all iterations, a greedy randomized solution $p$ is generated in line 3 and tentatively improved in line 4 with an approximate local search. The greedy randomized construction and the approximate local search are described in Subsections 4.1 and 4.2, respectively.

If the elite set $P$ is empty, solution $p$ is added to it in line 20. If the elite set is not empty, then while it is not full, solution $p$ is added to it in line 23 if it is sufficiently different from the solutions already in the elite set. To define the term “sufficiently different” more precisely, let $\Delta(p, q)$ denote the number of assignments in $p$ that are different from those in $q$. For a given level of difference $\delta$, we say $p$ is sufficiently different from all elite solutions in $P$ if $\Delta(p, q) > \delta$ for all $q \in P$, which we indicate with the notation $p \not\approx P$.

If the elite set $P$ is full, then path-relinking is applied in line 7 between $p$ and some elite solution $q$ randomly chosen from $P$ in line 6, resulting in solution $r$. In line 8, $r$ is updated by an approximate local minimum in its neighborhood. Path-relinking is described in Subsection 4.3.

If $r$ is the best solution found so far, then in line 11 it replaces $t$, the solution
most similar to it, computed in line 10. Otherwise, if \( r \) is better than the worst solution in \( P \) and \( r \neq P \), then in line 15 it replaces \( t \), the solution most similar to it, computed in line 14.

### 4.1 Greedy randomized construction

Greedy randomized construction in a GRASP heuristic combines elements of a greedy algorithm with randomization to produce a series of starting solutions for local search. Pseudo-code for the greedy randomized procedure for the MAX-CMO problem is shown in Algorithm 2, referred to in line 3 of Algorithm 1 as GreedyRandomized.

```
procedure GreedyRandomized
    Input: \( C^A = (V_A, E_A) \), \( C^B = (V_B, E_B) \), \( \alpha \)
    Output: Assignment vector \( p \)
    1 Randomly select \( q \in \text{UNIF}[\lceil \alpha \times n \rceil, n] \);
    2 Initialize \( R^A \leftarrow \emptyset \); \( R^B \leftarrow \emptyset \);
    3 for \( i = 1, \ldots, q \) do
        4 Randomly select \( r_A \in V_A \) with \( \text{prob}(r_A) \sim \deg(r_A) \);
        5 \( R^A \leftarrow R^A \cup \{r_A\} \);
        6 Randomly select \( r_B \in V_B \) with \( \text{prob}(r_B) \sim \deg(r_B) \);
        7 \( R^B \leftarrow R^B \cup \{r_B\} \);
    8 end
    9 Sort \( R^A \) and \( R^B \) in increasing order;
    10 for \( k = 1, \ldots, q \) do
        11 i \leftarrow k-th element of \( R^A \); \( j \leftarrow k \)-th element of \( R^B \);
        12 \( p_i \leftarrow j \);
    13 end
    14 return \( p \);
```

**Algorithm 2**: Greedy randomized construction procedure

Given two proteins \( A \) and \( B \), the construction procedure takes as input their contact maps \( C^A = (V_A, E_A) \) and \( C^B = (V_B, E_B) \) of lengths \( n \) and \( m \) (\( m > n \)).
respectively, and outputs a vector $p$ of length $n$, where $p_i = \text{nil}$ if residue $i \in V_A$ is not aligned, and $p_i = j$ if residue $i \in V_A$ is aligned with residue $j \in V_B$.

In line 1 of Algorithm 2, the number $q$ of residues to be aligned is randomly selected with uniform probability from the interval $[\lceil \alpha \times n \rceil, n]$, where $\alpha \in (0, 1]$ is a positive real valued input parameter. Let $R^A \subseteq V_A$ and $R^B \subseteq V_B$ be the sets of residues from proteins $A$ and $B$, respectively, that will be aligned. They are initialized empty in line 2. In the for loop in lines 3 to 8, $q$ residues are randomly selected from $V_A$ and $V_B$, greedily favoring nodes with high degree. High degree nodes have a greater chance of being endpoints of isomorphic edges than do low degree nodes. These residues are added, respectively, to sets $R^A$ and $R^B$. In line 9, the elements of sets $R^A$ and $R^B$ are sorted in increasing order. Finally, in the for loop in lines 10 to 14 the $k$-th residue of $R^A$ is aligned with the $k$-th element of $R^B$, for $k = 1, \ldots, q$.

### 4.2 Approximate local search

Since there is no guarantee that the construction procedure presented in Section 4.1 produces a local maximum number of overlaps, a local improvement procedure can be applied starting at the constructed solution $p$ to attempt to increase the number of overlaps. Given a starting solution $p$, a local improvement strategy examines solutions in the neighborhood $N(p)$ of $p$ and replaces $p$ by some solution $p' \in N(p)$ with $\Lambda(p') > \Lambda(p)$.

Given a solution $p$, Figure 5 shows four quadrilateral structures that can potentially occur in the solution. In the figures, nodes $r^L_A, r^R_A \in S_A \subseteq V_A$ and $r^L_B, r^R_B \in S_B \subseteq V_B$, while $r_A \in V_A \setminus S_A$ and $r_B \in V_B \setminus S_B$. Each structure has two edges, $(r^L_A, r^L_B)$ and $(r^R_A, r^R_B)$ defined by the bijection $f$ of $\mathcal{L}(S_A, S_B, f)$.

We now describe the neighborhoods of each of the four quadrilateral structures.
• **Type I:** Each quadrilateral structure of Type I in a solution admits two neighbors shown in Figure 6. The first neighbor is obtained by removing edge $(r_L^A, r_L^B)$. This move corresponds to setting $p_i = \text{nil}$, where $i = r_L^A$. Likewise, the second neighbor is obtained by removing edge $(r_R^A, r_R^B)$. This move corresponds to setting $p_i = \text{nil}$, where $i = r_R^A$. With respect to its contact map, there are $O(n)$ quadrilateral structures of this type.

• **Type II:** Each quadrilateral structure of Type II in a solution admits two neighbors shown in Figure 7. The first neighbor is obtained by replacing edge $(r_L^A, r_L^B)$ by $(r_A, r_L^B)$, where $r_L^A < r_A < r_R^A$. This move corresponds to setting $p_i = \text{nil}$ for $i = r_L^A$ and $p_i = j$, where $i = r_A$ and $j = r_L^B$. Likewise, the second neighbor is obtained by replacing edge $(r_R^A, r_R^B)$ by $(r_A, r_R^B)$, where $r_L^A < r_A < r_R^A$. This move corresponds to setting $p_i = \text{nil}$ for $i = r_R^A$ and $p_i = j$, where $i = r_A$ and $j = r_R^B$. In the worst case, there are $O(n)$ quadrilateral structures of this type.

• **Type III:** Each quadrilateral structure of Type III in a solution admits two neighbors shown in Figure 8. The first neighbor is obtained by replacing edge $(r_L^A, r_L^B)$ by $(r_L^A, r_B)$, where $r_L^B < r_B < r_R^B$. This move corresponds to setting $p_i = j$, where $i = r_L^A$ and $j = r_B$. Likewise, the second neighbor

![FIGURE 5 Quadrilateral structures of solution.](image)
is obtained by replacing edge $(r^R_A, r^R_B)$ by $(r^R_A, r_B)$, where $r^L_B < r_B < r^R_B$. This move corresponds to setting $p_i = j$, where $i = r^R_A$ and $j = r_B$. In the worst case, there are $O(m)$ quadrilateral structures of this type.

• **Type IV**: Each quadrilateral structure of Type IV in a solution admits five neighbors shown in Figure 9. The first neighbor (middle structure in the figure) is obtained by adding edge $(r_A, r_B)$, where $r^L_A < r_A < r^R_A$ and $r^L_B < r_B < r^R_B$. This move corresponds to setting $p_i = j$, where $i = r_A$ and $j = r_B$. The second neighbor (top left structure in the figure) is obtained by replacing edge $(r^L_A, r^L_B)$ by $(r_A, r^L_B)$, where $r^L_A < r_A < r^R_A$ and $r^L_B < r_B < r^R_B$. This move corresponds to setting $p_i = j$, where $i = r_A$ and $j = r_B$. The third neighbor (top right structure in the figure) is obtained by replacing edge $(r^R_A, r^R_B)$ by $(r_A, r^R_B)$, where $r^R_A < r_A < r^R_A$ and $r^R_B < r_B < r^R_B$. This move corresponds to setting $p_i = j$, where $i = r_A$ and $j = r_B$. The fourth neighbor (bottom left structure in the figure) is obtained by replacing edge $(r^L_A, r^L_B)$ by $(r_A, r^L_B)$, where
This move corresponds to setting \( p_i = \text{nil} \) for \( i = r^L_A \) and \( p_i = j \), where \( i = r_A \) and \( j = r^L_B \). Finally, the fifth neighbor (bottom right structure in the figure) is obtained by replacing edge \((r^R_A, r^R_B)\) by \((r_A, r^R_B)\), where \( r^L_A < r_A < r^R_A \). This move corresponds to setting \( p_i = \text{nil} \) for \( i = r^R_A \) and \( p_i = j \), where \( i = r_A \) and \( j = r^R_B \). In the worst case, there are \( O(mn) \) quadrilateral structures of this type.

In a standard local search, one explores the neighborhood of a solution and moves either to the first or to the best improving solution. In either situation, in the worst case, the entire neighborhood will need to be explored at least once. Such large neighborhoods are expensive to explore with a standard local search method. To avoid exploring the entire neighborhood, we propose an approximate local search scheme similar to the one introduced in (Mateus et al., 2009).

The idea of the approximate local search is to sample the neighborhood of the current solution at most \( \text{MaxItr} \) times or until \( \text{MaxCLS} \) improving neighbors.
are identified. The search then moves from the current solution $p$ to an improving sampled neighbor $p'$ that is randomly chosen with probability proportional to $\Lambda(p')$. The search repeats until no improved neighbors are found after $MaxItr$ probes. Algorithm 3 shows pseudocode for the approximate local search procedure. The procedure takes as input: the starting solution $p$, the maximum size $MaxCLS$ of the candidate local search set $CLS$, the maximum number $MaxItr$ of times the neighborhood of the current solution is sampled, and a parameter $k$ that determines the maximum number of consecutive moves from the current solution $p$. The loop in lines 1 to 28 is repeated until no improving sampled solution is found, i.e. $CLS = \emptyset$. In line 2, the sampled solution counter $count$ and set $MaxCLS$ are initialized. The loop in lines 3 to 24 is repeated until either the set $CLS$ is full or $MaxItr$ neighbors of $p$ are sampled. In line 4 the current solution is saved as $p'$.

Starting from $p$, the for loop in lines 5 to 19 performs a number of moves. This number of moves is chosen uniformly at random in the range $[1, \ldots, k]$. Recall
that a quadrilateral structure is made up of four related nodes, \( r^L_A, r^R_A \in S_A \) and \( r^L_B, r^R_B \in S_B \), such that \( r^L_A \) and \( r^R_A \) are, respectively, aligned with \( r^L_B \) and \( r^R_B \), and no other alignment exists between these two edges. Therefore, given one of the four nodes, the other three are entirely determined. We call this seed node an anchor.

In line 6, position of the anchor is chosen at random to be one of the following: left-\( C^A \), right-\( C^A \), left-\( C^B \), or right-\( C^B \). Once the anchor position is fixed, line 7 determines the set \( R \) of residues for the anchor. Let \( \Gamma_A \) and \( \Gamma_B \) be the set of aligned residues in \( C^A \) and \( C_B \), respectively. Let \( i^A_L = \inf \Gamma_A \) and \( i^A_R = \sup \Gamma_A \) be, respectively, the leftmost and the rightmost aligned residue of \( C^A \). Similarly, let \( i^B_L = \inf \Gamma_B \) and \( i^B_R = \sup \Gamma_B \) be, respectively, the leftmost and the rightmost aligned residue of \( C^B \). If the anchor position is

- left-\( C^A \), then \( R = \Gamma_A \setminus \{i^A_R\} \),
- right-\( C^A \), then \( R = \Gamma_A \setminus \{i^A_L\} \),
- left-\( C^B \), then \( R = \Gamma_B \setminus \{i^B_R\} \),
Let \( \text{deg}(r) \) be the degree of \( r \in \mathcal{R} \) with respect to its contact map. In line 8, the anchor residue \( r \) is selected at random from \( \mathcal{R} \) with probability proportional to \( \text{deg}(r) \). In line 9, the quadrilateral structure \( \mathcal{Q} = \{r^L_A, r^R_A, r^L_B, r^R_B\} \) is determined. If the anchor position is

- right-\( C^B \), then \( \mathcal{R} = \Gamma_B \setminus \{i_B^L\} \).

Depending on which type of quadrilateral structure \( \mathcal{Q} \) is, the appropriate move updates solution \( p' \) in lines 10 to 18. In lines 20 to 22, if \( p' \) is better than the current solution \( p \), it is added to the set \( \text{CLS} \). In line 23, the sampled solution counter \( \text{count} \) is incremented. After completing the \textbf{repeat} loop in lines 3 to 24, if the set \( \text{CLS} \) is not empty, then in line 26, the new current solution \( p \) is randomly selected from set \( \text{CLS} \) with probability proportional to \( \Lambda(p) \).

Procedures \texttt{MoveTypeI}, \texttt{MoveTypeII}, \texttt{MoveTypeIII}, and \texttt{MoveTypeIV} are described next. Each procedure takes as input a solution \( p \) and a compatible quadrilateral structure \( \mathcal{Q} \) and output a solution in the its neighborhood.
procedure ApproximateLocalSearch
Input: p, MaxCLS, MaxItr, k
Output: Approximate local maximum p
repeat
  count ← 0; CLS ← Ø;
repeat
  p' ← p;
  for i = 1, . . . , UNIF{1, 2, . . . , k} do
    Randomly select anchor position;
    Determine allowable set \( \mathcal{R} \) of residues for anchor;
    Select anchor residue \( r \in \mathcal{R} \) with \( \text{prob}(r) \sim \deg(r) \);
    Determine \( Q = \{ r_L^A, r_L^B, r_R^A, r_R^B \} \);
    case \( Q \) is of Type I
    | \( p' \leftarrow \text{MoveTypeI}(p') \);
    case \( Q \) is of Type II
    | \( p' \leftarrow \text{MoveTypeII}(p') \);
    case \( Q \) is of Type III
    | \( p' \leftarrow \text{MoveTypeIII}(p') \);
    otherwise
    | \( p' \leftarrow \text{MoveTypeIV}(p') \);
end
if \( \Lambda(p') > \Lambda(p) \) then
  CLS ← CLS \cup \{p'\};
end
count ← count + 1;
until |CLS| ≥ MaxCLS or count ≥ MaxItr;
if CLS ≠ Ø then
  Randomly select a solution \( p \in CLS \);
end
until CLS = Ø;
return \( p \);

Algorithm 3: Pseudo-code for ApproxLocalSearch: Approximate local search procedure.

Pseudocode for procedure MoveTypeI is shown in Algorithm 4. This procedure simply moves to one of the two Type I neighbors of \( p \) with equal probability. In line 1, a coin toss is simulated to select the edge to be removed from \( p \). If the outcome of the coin toss is heads, then in lines 3 and 7, edge \((r_L^A, r_L^B)\) is removed. Otherwise, in lines 5 and 7, edge \((r_R^A, r_R^B)\) is removed.

Pseudocode for procedure MoveTypeII is shown in Algorithm 5. This procedure moves to one of the two Type II neighbors of \( p \) with equal probability. In line 1, a residue \( r_A \) located between \( r_L^A \) and \( r_R^A \) is randomly selected with proba-
probability proportional to $\deg(r_A)$. In line 2, a coin toss is simulated to select the edge to be removed from $p$. If the outcome of the coin toss is heads, then in lines 4 and 8 edge $(r_A^L, r_B^L)$ is replaced by edge $(r_A, r_B^L)$. Otherwise, in lines 6 and 8, edge $(r_A^R, r_B^R)$ is replaced by edge $(r_A, r_B^R)$.

\begin{algorithm}
\begin{algorithmic}
\Procedure{MoveTypeI}{Input: $r_A^L$, $r_A^R$, $r_B^L$, $r_B^R$, $p$}
\State Output: Assignment vector $p$
\State Randomly select $\pi \in \text{UNIF}(0, 1)$;
\If{$\pi < 0.5$}
\State $i \leftarrow r_A^L$;
\Else
\State $i \leftarrow r_A^R$;
\EndIf
\State $p_i \leftarrow \text{nil}$;
\State \Return $p$;
\EndProcedure
\end{algorithmic}
\caption{Type I move in approximate local search}
\end{algorithm}

\begin{algorithm}
\begin{algorithmic}
\Procedure{MoveTypeII}{Input: $r_A^L$, $r_A^R$, $r_B^L$, $r_B^R$, $p$}
\State Output: Assignment vector $p$
\State Randomly select $r_A^L < r_A < r_A^R$, with $\text{prob}(r_A) \sim \deg(r_A)$;
\State Randomly select $\pi \in \text{UNIF}(0, 1)$;
\If{$\pi < 0.5$}
\State $j \leftarrow r_B^L$;
\Else
\State $j \leftarrow r_B^R$;
\EndIf
\State $i \leftarrow r_A$; $p_i \leftarrow j$;
\State \Return $p$;
\EndProcedure
\end{algorithmic}
\caption{Type II move in approximate local search}
\end{algorithm}
Pseudocode for procedure \texttt{MoveTypeIII} is shown in Algorithm 6. This procedure moves to one of the two Type III neighbors of \( p \) with equal probability. In line 1, a residue \( r_B \) located between \( r_B^L \) and \( r_B^R \) is randomly selected with probability proportional to \( \deg(r_B) \). In line 2, a coin toss is simulated to select the edge to be removed from \( p \). If the outcome of the coin toss is heads, then in lines 4 and 8 edge \((r_A^L, r_B^L)\) is replaced by edge \((r_A^L, r_B)\). Otherwise, in lines 6 and 8, edge \((r_A^R, r_B^R)\) is replaced by edge \((r_A^R, r_B)\).

\begin{verbatim}
procedure MoveTypeIII
    Input: \( r_A^L, r_A^R, r_B^L, r_B^R, p \)
    Output: Assignment vector \( p \)
1    Randomly select \( r_B \) such that \( r_B^L < r_B < r_B^R \);
2    Randomly select \( \pi \in \text{UNIF}(0, 1) \);
3    if \( \pi < 0.5 \) then
4        \( i \leftarrow r_A^L \);
5    else
6        \( i \leftarrow r_A^R \);
7    end
8    \( j \leftarrow r_B \); \( p_i \leftarrow j \);
9    return \( p \);
\end{verbatim}

Algorithm 6: Type III move in approximate local search

Pseudocode for procedure \texttt{MoveTypeIV} is shown in Algorithm 7. This procedure moves to one of the five Type IV neighbors of \( p \). In line 1, residue \( r_A \), located between \( r_A^L \) and \( r_A^R \) is randomly selected with probability proportional to \( \deg(r_A) \). Likewise, in line 2, residue \( r_B \), located between \( r_B^L \) and \( r_B^R \) is randomly selected with probability proportional to \( \deg(r_B) \). In line 3, a coin toss is simulated to determine whether a new edge will be added or if one will be replaced. If the outcome of the coin toss is heads, then in lines 18 and 20 edge \((r_A, r_B)\) is added. Otherwise two simultaneous coin tosses are simulated to determine which
of the other four neighborhood solutions will be selected for the move. If is the
outcome is two heads, then in lines 7 and 20, edge \((r_A^L, r_B^L)\) is replaced by edge edge \((r_A^L, r_B)\). If is the outcome of the first coin is heads and of the second tails, then in lines 9 and 20, edge \((r_A^R, r_B^R)\) is replaced by edge edge \((r_A^R, r_B^R)\). If is the outcome of the first coin is tails and of the second heads, then in lines 11, 12, and 20, edge \((r_A^R, r_B^R)\) is replaced by edge edge \((r_A^L, r_B^L)\). Finally, if is the outcome is two tails, then in lines 14, 15, and 20, edge \((r_A^L, r_B^L)\) is replaced by edge edge \((r_A^L, r_B)\).

4.3 Path-relinking

Path-relinking (Glover, 1996) is an intensification scheme that explores paths in the solution space connecting two good-quality (or elite) solutions. Given two solutions of the MAX-CMO problem denoted by their respective assignment vectors \(s\) and \(t\), let \(\Delta(s, t) = \{i = 1, \ldots, n : s_i \neq t_i\}\). Path-relinking examines each solution in the path \(s = p_1(s, t), p_2(s, t), \ldots, p_k(s, t) = t\) connecting \(s\) and \(t\), where \(k = |\Delta(s, t)|\) and \(p_j(s, t)\) is the \(j\)-th solution in the path from \(s\) to \(t\). Solution \(p_j(s, t) \in \mathcal{N}(p_{j-1}(s, t))\) such that \(|\Delta(p_j(s, t), t)| = |\Delta(p_{j-1}(s, t), t)| - 1\), i.e. \(p_j(s, t)\) is obtained by reassigning a residue \(\ell \in \Delta(p_{j-1}(s, t), t)\) of \(p_{j-1}(s, t)\) to \(t\). It is easy to verify that for any pair of solutions \(s\) and \(t\) there will be at least one path of the type described above from \(s\) to \(t\).

Algorithm 8 illustrates the pseudo-code of the path-relinking procedure applied to a pair of solutions \(s\) (starting solution) and \(t\) (target solution).

The procedure starts in line 1 by computing set \(\Delta(s, t) = \{i = 1, \ldots, n : s_i \neq t_i\}\) comprised of the set of indices for which the residues in \(s\) and \(t\) differ. In
procedure MoveTypeIV
Input: $r_L^A, r_A^R, r_L^B, r_B^R, p$
Output: Assignment vector $p$

1. Randomly select $r_A$ such that $r_L^A < r_A < r_A^R$.
2. Randomly select $r_B$ such that $r_L^B < r_B < r_B^R$.
3. Randomly select $\pi_1 \in \text{UNIF}(0, 1)$;
4. if $\pi_1 < 0.5$ then
5. | Randomly select $\pi_2 \in \text{UNIF}(0, 1)$;
6. | case $\pi_2 \leq 0.25$
7. | | $i \leftarrow r_L^A; j \leftarrow r_B$;
8. | case $0.25 < \pi_2 \leq 0.5$
9. | | $i \leftarrow r_A^R; j \leftarrow r_B$;
10. | case $0.5 < \pi_2 \leq 0.75$
11. | | $i \leftarrow r_A^R; p_i \leftarrow \text{nil}$;
12. | | $i \leftarrow r_A; j \leftarrow r_B^R$
13. | otherwise
14. | | $i \leftarrow r_A^R; p_i \leftarrow \text{nil}$;
15. | | $i \leftarrow r_A; j \leftarrow r_B^R$
16. end
17. else
18. | $i \leftarrow r_A; j \leftarrow r_B$;
19. end
20. $p_i \leftarrow j$;
21. return $p$;

Algorithm 7: Type IV move in approximate local search
**procedure** PathRelinking

**Input:** Pair of solutions $s$ and $t$

**Output:** Best solution $x^*$ in path from $s$ to $t$

1. Compute set $\Delta(s, t) \leftarrow \{ i = 1, \ldots , n : s_i \neq t_i \}$;
2. $x^* \leftarrow \arg\max \{ \Lambda(s), \Lambda(t) \}$;
3. $\Lambda^* \leftarrow \Lambda(x^*)$;
4. $x \leftarrow s$;
5. **while** $\Delta(x, t) \neq \emptyset$ **do**
6. Define $\Delta'(x, t) \subseteq \Delta(x, t)$ to be the set of feasible residues;
7. $i^* \leftarrow \arg\max \{ \Lambda(x \oplus i) : i \in \Delta'(x, t) \}$;
8. $\Delta(x \oplus i^*, t) \leftarrow \Delta(x, t) \setminus \{ i^* \}$;
9. $x \leftarrow x \oplus i^*$;
10. **if** not ISFeasible($x$) **then**
11. Repair($x, i$);
12. **end**
13. **if** $\Lambda(x) > \Lambda^*$ **then**
14. $\Lambda^* \leftarrow \Lambda(x)$;
15. $x^* \leftarrow x$;
16. **end**
17. **end**
18. **return** $x^*$;

**Algorithm 8:** Path-relinking between solutions $s$ and $t$.

lines 2 and 3, the best solution $x^*$ among $s$ and $t$ and its cost $\Lambda^*$ are determined.

In line 4 the current solution $x$ is initialized to $s$. The loop in lines 5 to 17 is repeated until the path is traversed, i.e. the current solution $s$ is a neighbor of the target $t$. In line 6, the set of feasible indices $\Delta'(x, t)$ is defined to be the set of indices $i \in \Delta(x, t)$ for which the assignment $y = (x_1, \ldots , x_{i-1}, t_i, x_{i+1}, \ldots , x_n)$ is feasible. We use the shorthand notation $y = x \oplus i$ to represent this move. If the constraints of this new solution are not violated, the new solution is feasible. Otherwise, a repair procedure is applied in an attempt to make it feasible in lines 10 to 12. The repair procedure simply removes all alignments that are violating any of the constraints, maintaining the alignment $i$ of the solution target.

In line 7, the feasible index $i^*$ that results in the highest valued assignment is determined and in line 8 set $\Delta(x \oplus i^*, t) \leftarrow \Delta(x, t) \setminus \{ i^* \}$. In line 14 the move is made and in lines 13 to 16 the best solution in the path and its value are updated if necessary. The best solution $x^*$ found in the path from $s$ to $t$ is returned in line 18.

36
5 COMPUTATIONAL EXPERIMENTS AND RESULTS

In this section, we report on computational experiments carried out with the GRASP-PR heuristic introduced in this paper in order to analyze its effectiveness and to compare our algorithm with existing ones. First, we describe our test environment and use datasets, next we analyze and compare our implementation with other heuristics from the literature on a suite of test problems.

5.1 Test environment and Datasets

All experiments with the GRASP-PR were carried out on a Dell PE1950 computer with dual quad core 2.66 GHz Intel Xeon processors and 2 Gb of memory, running Red Hat Linux nesh version 5.1.19.6 (CentOS release 5.2, kernel 2.6.18-53.1.21.el5). The GRASP-PR heuristic was implemented in Java and compiled into bytecode with javac version 1.6.0 05. The random-number generator is an implementation of the Mersenne Twister algorithm (Matsumoto & Nishimura, 1998) from the COLT‡ library.

TABLE 1 Dataset information. Pairs stand for the number of pairwise comparisons performed. The values for Avg.Contacts corresponds to contact maps at 7Å.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Pairs</th>
<th>Avg. Residues</th>
<th>Avg. Contacts</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lancia</td>
<td>2702</td>
<td>57.07</td>
<td>95.91</td>
<td>(Caprara &amp; Lancia, 2002)</td>
</tr>
<tr>
<td>Skolnick</td>
<td>161</td>
<td>158.23</td>
<td>470.93</td>
<td>(Caprara &amp; Lancia, 2002)</td>
</tr>
<tr>
<td>Chew-Kedem</td>
<td>145</td>
<td>201.91</td>
<td>928.75</td>
<td>(Chew &amp; Kedem, 2002)</td>
</tr>
</tbody>
</table>

For the test bed analysis, we used three datasets§ (see table 1): The Lancia dataset (Caprara & Lancia, 2002) with 269 proteins and 2702 different instances of protein pairs, the Skolnick dataset (Caprara & Lancia, 2002) with 40 proteins and 161 different instances.

‡COLT is a open source library for high performance scientific and technical computing in Java. See http://acs.lbl.gov/~hoschek/colt/
§The datasets can be downloaded at http://goic.dcc.ufla.br/Biocomp/Datasets.html
161 different instances of protein pairs, the Chew-Kedem dataset (Chew & Kedem, 2002) with 40 proteins and 145 different instances.

5.2 Comparison of the GRASP-PR heuristic with other algorithms

In the first experiment, we use all instances of the three datasets, totaling 3008 instances of protein pairs.

Caprara & Lancia (2002) proposed a Lagrangian Relaxation (LR) approach for solving MAX-CMO, where the optimal Lagrange multipliers are found by subgradient optimization. Besides yielding an upper bound on the optimal solution of the original problem, the Lagrangian multipliers are used to drive a heuristic to construct the MAX-CMO solutions. Krasnogor et al. (2003) have demonstrated that this LR algorithm was able to find better results than the previous algorithms for the MAX-CMO over the Chew-Kedem dataset, and after that, the LR was reviewed over different datasets (Caprara et al., 2004; Xie & Sahinidis, 2007) confirming its effectiveness.

Pelta et al. (2008) argued that in order to compare a set of algorithms properly, all of them should be ideally compiled and run in the same computational environment. Hence, the experiments carried out in this study met such a requirement. We make use of the source codes of LR and VNS algorithms to perform the comparison.

The VNS heuristic is presented as “a simple and fast” heuristic for protein structure comparison and its original stopping criterion for each run is 100 iterations or 20 iterations without improvements (whatever comes first). Three versions of VNS heuristic are available: MSVNS1, MSVNS2 and MSVNS3. In all computational experiments in this paper, we set the best VNS version and parameter

---

*The LR algorithm was implemented in C programming language and it was that was kindly provided to us by one of the authors of Caprara & Lancia (2002), Alberto Caprara.

*The VNS algorithm was fully implemented in C++ programming language and the software is available for download at [http://modo.ugr.es/jrgonzalez/msvns4maxcmo](http://modo.ugr.es/jrgonzalez/msvns4maxcmo).
looking at the paper: “The best alternative is MSVNS3 with windows sizes of 10-30-50 leading to an average error below 3.6% for Lancia’s dataset with 2702 pairs, and below 1.7% for the Skolnick’s one.” (Pelta et al., 2008).

There are also three different modes of the LR algorithm, which differ by the heuristic used to build the MAX-CMO solutions from the lagrange multipliers. We applied one denoted by LAGR-R which uses local search at the root node only to accomplish it, because according to Xie & Sahinidis (2007) they “recorded the best solutions and CPU time for LAGR-R”.

We also adjusted the best combination of parameters for the GRASP-PR heuristic. To accomplish it, we tested 268 combinations of parameters on 4 randomly selected instances (2 from Lancia, 1 from Chew-Kedem and 1 from Skolnick). We performed 10 runs per parameter combination on each one of the 4 instances, with a running time limit of 1 second and a calculation of the average error of each round. For the four analyzed instances, the parameters that showed a lower average error are: \( \alpha \) (delimiter of greedy randomized construction): 0.7, \( MaxCLS \) (Local Search parameter) : 20, \( MaxItr \) (Local Search parameter) : 10, Local Search moves : 2, \( |P| \) (elite set \( P \) size) : 8, \( \delta \) (sufficiently different level) : 2.

After selecting the best versions and parameters for each algorithm, we performed 30 runs on each instance of the selected datasets using MSVNS3 with windows size of 30. For each instance, the minimum, the average and the maximum overlap values of the 30 runs were calculated, as well as the minimum and maximum running times. The average running time of the runs subset that reached the maximum overlap value is also calculated. This last measure of time was set as the stopping criterion for the LR and GRASP-PR algorithms, and then we carried out 30 runs on each instance using these two others algorithms and the minimum, average and maximum overlap values of the 30 runs of these ones.
are also calculated. As result of this experiment, complementary data containing the values for each instance of the three datasets is available for download at [http://goic.dcc.ufla.br/Biocomp/ResultsCMOP.xls](http://goic.dcc.ufla.br/Biocomp/ResultsCMOP.xls) as a benchmark for MAX-CMO algorithms comparison.

For instances of Lancia and Skolnick datasets, the error per instance of each algorithm was calculated using overlap values** given by the exact algorithm from the algorithm presented in (Xie & Sahinidis, 2004). For instances of Chew-Kedem dataset, for which the exact overlap values were not available, the error(%) was calculated using the Upper Bound values returned by the LR algorithm. In both cases, the error is calculated with respect to the maximum overlap value of the 30 runs per instance. The results are summarized in Tables 1 and 2.

In the experiment conducted by us, the results for the VNS algorithm showed in Table 1 closely corroborate with results presented by Pelta et al. (2008). They report an average error of 3.6% for Lancia dataset and below 1.7% for the Skolnick’s. We obtained 3.398% and 1.708%, respectively.

For Lancia dataset, Table 2 shows that the VNS heuristic presents the lowest average error (3.398%) and the highest number of optimally solved instances (1578 - 58.4%). The LR algorithm obtained the second lowest average error (8.438%) although it has optimally solved fewer instances than the GRASP-PR (970 and 1114 number of optimally solved instances for the LR algorithm and GRASP-PR, respectively). Considering only the instances not optimally solved (Near-Optimally Solved), the average errors increase to 8.160%, 15.512% and 13.150% for VNS, LR and GRASP-PR, respectively. If we rank the algorithms considering the value of average error, for Lancia dataset, we observe that the VNS

**These optimal overlap values were kindly provided by one of the authors of reference Pelta et al., (2008), Juan González, who make use of same values for computing the error of the VNS algorithm in respect to the optimal overlap values.
algorithm is the best one, followed by LR and finally the GRASP-PR. Considering the number of instances optimally solved, the VNS algorithm, the GRASP-PR and the LR, we found out that the Lancia dataset has protein pairs with a smaller number of residues and contacts, containing also the easiest number of instances, despite it having the largest number of them (see table 1).

For the Skolnick dataset, considering the average error of all instances, we rank the algorithms with the VNS (1.708%) as the best heuristic followed by GRASP-PR (3.821%). And considering the number of optimally solved instances, we rank the GRASP-PR - 89 (55.28%) optimally solved instances - as the best one followed by VNS - 63 (39.13%) optimally solved instances. The results for this dataset show that the results of GRASP-PR are competitive with the other two heuristics analyzed.

Table 3 shows results for the Chew-Kedem dataset, where the error is calculated based on the best Upper Bound algorithm value obtained from the LR algorithm. In this table, the results are grouped according to the protein families in order to investigate whether any specific type of structural folding is favoring some enhancement that could further improve the proposed heuristic, which could be attained through analyses and proposition of specialized constructive methods, designed for different families of proteins. Additional result analyses in different protein families of this datasets will be performed. In general, the GRASP-PR algorithm showed the best results, with an average error of 12.412%, followed closely by the LR (12.839%). These results encourage the use of the proposed GRASP-PR heuristic, for the Chew-Kedem is considered the most challenging dataset according to to Krasnogor et al. (2003).
<table>
<thead>
<tr>
<th>Algorithm</th>
<th>N</th>
<th>Error (%)</th>
<th>Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Avg.</td>
<td>SD</td>
</tr>
<tr>
<td>Lancia’s Dataset</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lim</td>
<td>2702 (100%)</td>
<td>3.398</td>
<td>0.010</td>
</tr>
<tr>
<td>Optimally Solved</td>
<td>VNS</td>
<td>1578 (58.4%)</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGR</td>
<td>2702 (100%)</td>
<td>8.438</td>
<td>0.010</td>
</tr>
<tr>
<td>Near-Optimally Solved</td>
<td>VNS</td>
<td>1732 (64.1%)</td>
<td>13.150</td>
</tr>
<tr>
<td>GRASP-PR</td>
<td>1124 (41.6%)</td>
<td>8.160</td>
<td>0.010</td>
</tr>
<tr>
<td>LAGR</td>
<td>2702 (100%)</td>
<td>4.328</td>
<td>0.010</td>
</tr>
<tr>
<td>Skolnick’s Dataset</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lim</td>
<td>161 (100%)</td>
<td>1.708</td>
<td>0.010</td>
</tr>
<tr>
<td>Optimally Solved</td>
<td>VNS</td>
<td>63 (39.13%)</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGR</td>
<td>161 (100%)</td>
<td>4.920</td>
<td>0.010</td>
</tr>
<tr>
<td>Near-Optimally Solved</td>
<td>VNS</td>
<td>98 (60.87%)</td>
<td>2.806</td>
</tr>
<tr>
<td>LAGR</td>
<td>106 (65.84%)</td>
<td>7.124</td>
<td>0.010</td>
</tr>
</tbody>
</table>

TABLE 2 Results over 2702 pairs from Lancia’s dataset, and 161 pair from Skolnick’s dataset. The error is measured with respect to the optimum value.
TABLE 3 Results over 145 pairs from Chew-Kedem dataset. The error is measured based on the Upper Bound value given by the Lagrangian Relaxation algorithm.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>N</th>
<th>Error (%)</th>
<th></th>
<th>Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Avg.</td>
<td>SD</td>
<td>Median</td>
</tr>
<tr>
<td>All Pairs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VNS</td>
<td>145 (100%)</td>
<td>12.839</td>
<td>11.104</td>
<td>9.626</td>
</tr>
<tr>
<td>GRASP-PR</td>
<td>145 (100%)</td>
<td>12.412</td>
<td>12.850</td>
<td>6.497</td>
</tr>
<tr>
<td>LAGR</td>
<td>145 (100%)</td>
<td>9.756</td>
<td>11.885</td>
<td>5.398</td>
</tr>
<tr>
<td>Globin Pairs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VNS</td>
<td>102 (70.34%)</td>
<td>9.039</td>
<td>3.348</td>
<td>8.867</td>
</tr>
<tr>
<td>GRASP-PR</td>
<td>102 (70.34%)</td>
<td>6.648</td>
<td>3.416</td>
<td>8.867</td>
</tr>
<tr>
<td>LAGR</td>
<td>102 (70.34%)</td>
<td>5.209</td>
<td>2.090</td>
<td>4.881</td>
</tr>
<tr>
<td>Alpha-Beta Pairs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VNS</td>
<td>13 (8.97%)</td>
<td>16.918</td>
<td>21.046</td>
<td>2.667</td>
</tr>
<tr>
<td>GRASP-PR</td>
<td>13 (8.97%)</td>
<td>17.612</td>
<td>22.121</td>
<td>2.000</td>
</tr>
<tr>
<td>LAGR</td>
<td>13 (8.97%)</td>
<td>18.074</td>
<td>22.763</td>
<td>2.000</td>
</tr>
<tr>
<td>Beta Pairs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VNS</td>
<td>15 (10.34%)</td>
<td>14.320</td>
<td>4.890</td>
<td>14.079</td>
</tr>
<tr>
<td>GRASP-PR</td>
<td>15 (10.34%)</td>
<td>27.355</td>
<td>9.707</td>
<td>29.130</td>
</tr>
<tr>
<td>LAGR</td>
<td>15 (10.34%)</td>
<td>11.500</td>
<td>5.030</td>
<td>11.905</td>
</tr>
<tr>
<td>TIM-Barrel Pairs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VNS</td>
<td>6 (4.14%)</td>
<td>35.474</td>
<td>11.756</td>
<td>39.383</td>
</tr>
<tr>
<td>GRASP-PR</td>
<td>6 (4.14%)</td>
<td>32.766</td>
<td>5.720</td>
<td>34.374</td>
</tr>
<tr>
<td>LAGR</td>
<td>6 (4.14%)</td>
<td>31.046</td>
<td>12.518</td>
<td>35.821</td>
</tr>
<tr>
<td>Mixed Pairs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VNS</td>
<td>9 (4.14%)</td>
<td>32.448</td>
<td>15.905</td>
<td>30.631</td>
</tr>
<tr>
<td>LAGR</td>
<td>9 (4.14%)</td>
<td>32.175</td>
<td>16.751</td>
<td>30.040</td>
</tr>
</tbody>
</table>
5.3 Time-to-target plots for GRASP-PR against other heuristics

In the following experiment, we used eight randomly selected instances: 4 from dataset Lancia, 2 from Skolnick and 2 from the Chew-Kedem one. Information on these instances is summarized in the table 4.

<table>
<thead>
<tr>
<th>ID</th>
<th>Prot. 1</th>
<th>Res. Contacts</th>
<th>Prot. 2</th>
<th>Res. Contacts</th>
<th>Target Value</th>
<th>Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1gzi</td>
<td>58</td>
<td>9msi</td>
<td>59</td>
<td>106</td>
<td>Lancia</td>
</tr>
<tr>
<td>2</td>
<td>1ekl</td>
<td>58</td>
<td>1msj</td>
<td>59</td>
<td>114</td>
<td>Lancia</td>
</tr>
<tr>
<td>3</td>
<td>1eqt</td>
<td>58</td>
<td>1hcc</td>
<td>47</td>
<td>75</td>
<td>Lancia</td>
</tr>
<tr>
<td>4</td>
<td>1fh3</td>
<td>54</td>
<td>1pxt</td>
<td>54</td>
<td>93</td>
<td>Lancia</td>
</tr>
<tr>
<td>5</td>
<td>1cby</td>
<td>128</td>
<td>1hmy</td>
<td>118</td>
<td>366</td>
<td>Skolnick</td>
</tr>
<tr>
<td>6</td>
<td>1pla</td>
<td>97</td>
<td>1pca</td>
<td>99</td>
<td>282</td>
<td>Skolnick</td>
</tr>
<tr>
<td>7</td>
<td>1babA</td>
<td>142</td>
<td>1mba</td>
<td>146</td>
<td>439</td>
<td>Chew-Kedem</td>
</tr>
<tr>
<td>8</td>
<td>1aa9</td>
<td>171</td>
<td>1ct9A</td>
<td>497</td>
<td>1508</td>
<td>Chew-Kedem</td>
</tr>
</tbody>
</table>

Time-to-target (TTT) plots display on the ordinate axis the probability that an algorithm will find a solution at least as good as a given target value within a given running time, shown on the abscissa axis. TTT plots were used by Feo et al. (1994) and have been advocated thenceforward as a way to characterize the running times of stochastic algorithms for combinatorial optimization.

In this analysis, each heuristic is run $n$ times on the fixed instance and using the given target solution value. For each of the $n$ runs, the random number generator is initialized with a distinct seed and, therefore, the runs are assumed to be independent. For each instance/target pair, the running times are sorted in increasing order. We associate with the $i$-th sorted running time $t_i$ a probability...
\[ p_i = \left( i - 1/2 \right)/n \] and plot the points \( z_i = [t_i, p_i] \), for \( i = 1, ..., n \). Then, this cumulative probability distribution is plotted allowing to infer and compare details and information about the performance of the heuristics.

For each of the selected instances, we made 200 independent runs of the GRASP-PR, 200 runs of the VNS and 200 runs of the LR algorithm. Each of these runs stopped when the target-valued solution (Target Value - Table 4) was found, and we record the time taken for each run. Figure 10 and 11 shows TTT plots for all algorithms analyzed on all instances from Table 4. These plots display the empirical probability distributions of the random variable time to target solution. Each heuristic was run a total of 1600 times in the experiments.

![Protein pair: pdb1gzi pdb9msi Target = 106](image)

![Protein pair: pdb1ekl pdb1msj Target = 103](image)

![Protein pair: pdb1eqt pdb2hcc Target = 55](image)

![Protein pair: pdb1fh3 pdb1ptx Target = 57](image)

FIGURE 10 Time-to-target plots. Plots of cumulative probability distributions of GRASP-PR, VNS and LR running times for instances 1, 2, 3, and 4 (see table 4).

The relative position of the curves to the left implies that, given a fixed amount
of computing time, the algorithm referred to that curve has a higher probability than grasp of finding a target solution. The relative position of the curves to the right implies that, given a fixed probability of finding a target solution, the expected time taken by the algorithm referred to that curve to find a solution with that probability is greater than the time taken by the other ones. For example, consider instance 6 in Figure 11. The probability of finding a target at least as good as the target value 253 in 3.5 seconds is approximately of 40% for all algorithms. In 4.0 seconds at the most, these probabilities increase to approximately 90% for GRASP-PR and 85% for VNS and LR.

**FIGURE 11** Time-to-target plots. Plots of cumulative probability distributions of GRASP-PR, VNS and LR running times for instances 5, 6, 7, and 8 (see table 4).

Figure 10 show that the algorithm VNS has a small dominance over the GRASP-PR in two cases (instances 1 and 4), and the opposite in the other two: for in-
stances 2 and 3 the GRASP-PR demonstrates to be better. These figures are plots over instances of the dataset Lancia with smaller number of residues and contacts, then easier instances, and maybe favoring simple heuristics as the VNS. Figure 11 shows the dominance of GRASP-PR for instances 5 and 8, and for instances 6 and 7 there are two cases where it is not possible to identify a clear dominance. These results demonstrate that the proposed algorithm is very competitive compared the other algorithms analyzed.

5.4 GRASP-PR and the Skolnick Clustering test set

The aim of the Skolnick clustering test originally suggested by Skolnick and described in (Lancia et al., 2001) is to classify 40 proteins into four families according to their cluster membership. The proteins belonging to this dataset are shown in Table 1. In the following, we use their assigned indexes to refer to the respective proteins.

<table>
<thead>
<tr>
<th>ID</th>
<th>PDB</th>
<th>CID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1b00</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>1dbw</td>
<td>A</td>
</tr>
<tr>
<td>3</td>
<td>1nat</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>1intr</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>1qmp</td>
<td>A</td>
</tr>
<tr>
<td>6</td>
<td>1qmp</td>
<td>B</td>
</tr>
<tr>
<td>7</td>
<td>1qmp</td>
<td>C</td>
</tr>
<tr>
<td>8</td>
<td>1qmp</td>
<td>D</td>
</tr>
<tr>
<td>9</td>
<td>1rn1</td>
<td>A</td>
</tr>
<tr>
<td>10</td>
<td>1rn1</td>
<td>B</td>
</tr>
</tbody>
</table>

Table 2 describes the proteins and their families. Its fourth column with iden-
tifier *Seq. Sim.* indicates that sequence alignment fails for clustering the protein according to their family membership. This motivates structural alignment for solving the Skolnick clustering test.

**TABLE 6** Protein domains of the Skolnick test set and their categories as taken from (Caprara & Lancia, 2002). Shown are the characteristics of the four families, the mean number of residues, the range of similarity obtained by sequence alignment and the identifiers of the proteins.

<table>
<thead>
<tr>
<th>Family</th>
<th>Style</th>
<th>Residues</th>
<th>Seq-Sim.</th>
<th>Proteins</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>alpha-beta</td>
<td>124</td>
<td>15-30%</td>
<td>1-14</td>
</tr>
<tr>
<td>2</td>
<td>beta</td>
<td>99</td>
<td>35-90%</td>
<td>15-23</td>
</tr>
<tr>
<td>3</td>
<td>alpha-beta</td>
<td>250</td>
<td>30-90%</td>
<td>24-34</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>170</td>
<td>7-70%</td>
<td>35-40</td>
</tr>
</tbody>
</table>

The GRASP-PR algorithm and the server’s one are applied in an all-against-all fashion to the dataset and a distance matrix is calculated. The GRASP-PR running time limit is adjusted to 0.5 seconds, thereby, for the 780 pairwise structural alignments the process required about 10 minutes. As overlap values are not adequate *per se* for classification purposes because such values depend on the size of the proteins being compared, it is applied a normalization scheme, according to (Pelta et al., 2008), it may play a crucial role in protein classification. There is no general agreement on how to do normalization, so we use two of the available alternatives - first and second alternatives were proposed in (Lancia & Istrail, 2004) and (Xie & Sahinidis, 2004), respectively:

\[
Norm1(P_i, P_j) = \frac{\text{overlap}(P_i, P_j)}{\min(\text{contacts}_{P_i}, \text{contacts}_{P_j})} \quad (1)
\]

\[
Norm2(P_i, P_j) = \frac{2 \cdot \text{overlap}(P_i, P_j)}{\text{contacts}_{P_i} + \text{contacts}_{P_j}} \quad (2)
\]

Finally, with the values of the distance matrix normalized by *Norm1* and
FIGURE 12 Hierarchical Clustering based on the normalized overlap values among proteins in Skolnick’s dataset. The upper dendrograms (a, b) correspond to the average and complete linkage clustering using Norm1 and the lower ones (c, d) to the average and complete linkage clustering using Norm2.
Norm2, we apply complete and average linkage hierarchical clustering as implemented in R software package with the final objective of evaluating if the strategy is able to detect similarity at SCOP’s fold level (Murzin et al., 1995). We can see the dendograms generated by Goic-Biocomp server in Figure 1. For visualization purposes, the class number is displayed at the right of the protein name.

The results indicate an agreement with the SCOP categories as shown in Table 3. The GRASP-PR heuristic is able to perfectly recover the original grouping independently of the normalization and clustering algorithms, since it successfully classified the Skolnick proteins into five families according to the SCOP classification levels.

TABLE 7 Descriptions of the proteins clusters from the Skolnick’s test. Fold, family, and superfamily are according to SCOP.

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Proteins</th>
<th>Fold/Superfamily/Family</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1-8, 12-14</td>
<td>Flavodin-like, Che Y-like, Che Y-related</td>
</tr>
<tr>
<td>2</td>
<td>9-11</td>
<td>Microbial ribonucleases, Microbial ribonucleases, Fungi ribonucleases</td>
</tr>
<tr>
<td>3</td>
<td>15-23</td>
<td>Cuperdoxin-like, Cuperdoxins, Plastocyanin/Plastoazurin-like, TIM-beta alpha-barrel</td>
</tr>
<tr>
<td>4</td>
<td>24-34</td>
<td>Triosephosphate isomerase (TIM), Triosephosphate isomerase (TIM)</td>
</tr>
<tr>
<td>5</td>
<td>35-40</td>
<td>Ferritin-like, Ferritin</td>
</tr>
</tbody>
</table>

5.5 GRASP-PR as a structural alignment tool

For a protein structure alignment algorithm, it is important to investigate whether it produces biologically meaningful alignments. In this computational experiment we aim to investigate whether our heuristic provides biologically meaningful align-
ments for practical instances, besides to investigate whether GRASP-PR can replicate the results obtained by exact methods but with less computational effort and a simple strategy. To accomplish it, we use the MAX-CMO solution to guide the superimposition of proteins pairs and we make a visual inspection of these ones, as well as we perform analysis of the RMSD value, comparing the results of GRASP-PR against the successful exact algorithm “Exact-Reduction Based (RB) algorithm” proposed by (Xie & Sahinidis, 2007).

To superpose two protein structures one must have a mapping between equivalent amino acids in the two proteins. We can use the MAX-CMO solution as this required mapping which is used to guide the superimpositioning. So, after calculating the mapping between equivalent amino acids of the two proteins via MAX-CMO solution, we make use of a Biopython (Cock et al., 2009) script to create a PDB file with the two structures superposed.

The root mean square deviation (RMSD) is the measure of the average distance between the backbones of superimposed proteins. Unlike the number of overlaps found by MAX-CMO algorithms, the lower the RMSD to superpose the protein pair given the residues mapping, and also to calculate the RMSD value calculated for two superimposed proteins, more similar they are.

In this experiment, we use five instances selected at random from different datasets. The contact maps are constructed with a threshold of 6.5Å - see Table 4. Based on this test set, we compare GRASP-PR against the RB algorithm in terms of overlaps value and RMSD. The running time limit for the GRASP-PR heuristic is set to 5 seconds, while the RB, as a exact algorithm, just stop running as the optimal value is found. Table 5 compare the two algorithms in terms of overlaps value of MAX-CMO solutions and RMSD resultant of the superimposition.

As we can see in table 5, in terms of the overlaps and RMSD values, our
TABLE 8 Test set description. In the following, we use their assigned indexes to
refer to the respective protein pair.

<table>
<thead>
<tr>
<th>Instance</th>
<th>Protein 1</th>
<th>Protein 2</th>
<th>Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1ash</td>
<td>1hlm</td>
<td>Chew-Kedem (Chew &amp; Kedem, 2002)</td>
</tr>
<tr>
<td>2</td>
<td>1qfo</td>
<td>1neu</td>
<td>Chew-Kedem (Chew &amp; Kedem, 2002)</td>
</tr>
<tr>
<td>3</td>
<td>2ach</td>
<td>7api</td>
<td>Leluk-Konieczny-Roterman (Leluk et al., 2003))</td>
</tr>
<tr>
<td>4</td>
<td>1rcd</td>
<td>1ier</td>
<td>Skolnick (Caprara &amp; Lancia, 2002)</td>
</tr>
<tr>
<td>5</td>
<td>4tmt</td>
<td>1qmpB</td>
<td>Skolnick (Caprara &amp; Lancia, 2002)</td>
</tr>
</tbody>
</table>

TABLE 9 Test results of the five selected proteins pairs. In the table is shown the
running time, overlaps value/ the error (%) with respect to the optimal
value of MAX-CMO, the RMSD value found by each algorithm over
each instance.

<table>
<thead>
<tr>
<th>Instance</th>
<th>GRASP-PR</th>
<th>RB</th>
<th>Overlap/Error(%)</th>
<th>GRASP-PR</th>
<th>RB</th>
<th>RMSD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.00</td>
<td>754.40</td>
<td>271(2.87%) 279(0.00%)</td>
<td>5.21</td>
<td>3.19</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>5.00</td>
<td>185.8</td>
<td>156(13.33%) 180(0.00%)</td>
<td>3.01</td>
<td>2.77</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>5.00</td>
<td>3746.3</td>
<td>700(0.28%) 702(0.00%)</td>
<td>1.31</td>
<td>1.39</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>5.00</td>
<td>48.58</td>
<td>448(0.00%) 448(0.00%)</td>
<td>0.65</td>
<td>0.65</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5.00</td>
<td>55.25</td>
<td>255(0.00%) 255(0.00%)</td>
<td>1.18</td>
<td>1.18</td>
<td></td>
</tr>
</tbody>
</table>

algorithm is very competitive with the exact algorithm from (Xie & Sahinidis,
2007), since the error with respect to the optimal value is 0.00 for instances 4 and
5, and very low for the other instance 1, 2 and 3 - 2.87%, 13.33% and 0.28%,
respectively. For instances 4 and 5, the RMSD value found by GRASP-PR is
as lower as the ones found by the RB algorithm, and for instances 1 and 2 this
difference is very small. These results are very impressive particularly considering
that the running time of GRASP-PR heuristic is about 10x to 750x lower than the
running time of the RB algorithm.

For instance 3, we have an interesting particular case: although the overlap
value found by the GRASP-PR is worse (the higher the overlap value, the better
is the result of the MAX-CMO problem, and the lower the value of RMSD, the
better is the algorithm in terms of structure superimposition) the RMSD value of
GRASP-PR is better (smaller) than the values found by the RB algorithm. Such ambiguity is already known and discussed since the work entitled “The structural alignment between two proteins: is there a unique answer?” (Godzik, 1996), justifying the use of heuristics capable of giving fast near-optimal solutions of meaningful alignments.

The superimpositions of this experiment are shown in Figures 2-6, in which we can make a visual inspection of alignment. A key observation made from these figures is that the GRASP-PR was able to align all the selected protein pairs, and the resultant superimpositions had a high similarity with the superimposition resultant from the alignment of the RB algorithm with much less effort of time and a simple strategy.

FIGURE 13 Protein backbones superimposition generated from: the GRASP-PR solution on the left side and the RB one on the right side. Instance 1 - 1ASH (Blue) and 1HLM (Green).

Figures of instances 2, 3, 4 and 5 show that the alignment is identical to the naked eye for the two algorithms. For instances 4 and 5, this high similarity was expected, since both algorithms reached the exact value of overlaps. But for instance 2, even without reaching the optimal value of overlap, the GRASP-PR showed an alignment very similar to the RB algorithm, in addition to obtaining a better value of RMSD. In this figure 1, alignment of GRASP-PR differs somewhat
FIGURE 14 Protein backbones superimposition generated from: the GRASP-PR solution on the left side and the RB one on the right side. Instance 2 - 1QFO (Blue) and 1NEU (Green).

FIGURE 15 Protein backbones superimposition generated from: the GRASP-PR solution on the left side and the RB one on the right side. Instance 3 - 1RCD (Blue) and 1IER (Green).
FIGURE 16 Protein backbones superimposition generated from: the GRASP-PR solution on the left side and the RB one on the right side. Instance 4 - 4TMT (Blue) and 1QMPB (Green).

FIGURE 17 Protein backbones superimposition generated from: the GRASP-PR solution on the left side and the RB one on the right side. Instance 5 - 2ACH (Blue) and 7API (Green).
from the alignment of the RB algorithm, but both alignments seem to be significant as the difference in the RMSD between the algorithms value is very low.

The study performed in this dataset shows that our strategy can replicate the results obtained using exact methods but with less computational effort and a simple strategy. Moreover, this experiment illustrates that non-exact MAX-CMO values may have solutions as meaningful as the exact ones. Both elements are important results per se. We should mention that all experiments done in this final chapter can be done through Biocomp-Server described in Chapter 2.

6 CONCLUDING REMARKS

In this work, we tested a straight and simple GRASP implementation with Path-Relinking for the MAX-CMO problem, which obtains encouraging results. Computational results demonstrate that the heuristic is a well-suited approach for the MAX-CMOP, and comparisons with other successful heuristics from literature show that the proposed heuristic produces results very competitive. GRASP-PR obtained results that were very well applied to real problems using less computational effort than exact algorithms. Moreover, we mention that all experiments conducted in this paper can be reproduced using the Goic-Biocomp server at http://goic.dcc.ufla.br/Biocomp.
7 REFERENCES


HOLM, L. A fully automatic evolutionary classification of protein folds: dali 

FEO, T.; RESENDE, M.; SMITH, S. A greedy randomized adaptive search 

FEO, T. A.; RESENDE, M. G. Greedy randomized adaptive search procedures. 

GLOVER, F. Tabu search and adaptive memory programming: advances, 
applications, and challenges. In: BARR, R. S.; HELGASON, R. V.; 
KENNINGTON, J. L. (Ed.). *Interfaces in computer science and operations 
research*: advances in metaheuristics, optimization, and stochastic modeling 

GODZIK, A. The structural alignment between two proteins: is there a unique 
1996.

GODZIK, A.; KOLINSKI, A.; SKOLNICK, J. Topology fingerprint approach to 

GOLDMAN, D.; ISTRAIL, S.; PAPADIMITRIOU, C. Algorithmic aspects of 
protein structure similarity. In: *ANNUAL SYMPOSIUM ON FOUNDATIONS 

GOLDSMITH-FISCHMAN, S.; HONIG, B. Structural genomics: computational 
methods for structure analysis. *Protein Science*, Cold Spring Harbor, v. 12, n. 9, 

GREENBERG, H. J.; HART, W. E.; LANCIA, G. Opportunities for 
combinatorial optimization in computational biology. *Informs Journal on 


CHAPTER 3

GOIC-BIOCOMP: A WEB-BASED TOOL FOR PROTEIN ALIGNMENT
1 ABSTRACT

Protein structure comparison and clustering are key problems in bioinformatics. An available and efficient alternative to perform protein structural comparison is to align the contact maps of protein pairs. This approach can be formulated as a well known mathematical problem called Maximum Contact Map Overlap Problem (MAX-CMO). Tools are still required to solve the problem by using algorithms that obtain good quality solutions using less computational and time resources. This paper presents a web-based tool for protein structure alignment based on the greedy randomized adaptive search procedure with path-relinking (GRASP-PR) for MAX-CMO problem. Experiments can be performed via web comparing the GRASP-PR heuristic with other algorithms from literature. The tool is available: http://goic.dcc.ufla.br/Biocomp.
Comparação de estruturas de proteínas é um problema chave em bioinformática. Uma alternativa eficiente disponível para realizar a comparação estrutural de proteínas é alinhar os mapas de contatos de pares de proteínas. Essa abordagem pode ser formulada como um problema matemático bem conhecido chamado Maximum Contact Map Overlap (MAX-CMO). Ferramentas que resolvem esse problema usando algoritmos que obtêm soluções de boa qualidade usando menos recursos computacionais e tempo ainda são requeridos. Esse artigo apresenta uma ferramenta web para o alinhamento estrutural de proteínas baseado na heurística greedy randomized adaptive search procedure com path-relinking (GRASP-PR) para o MAX-CMO. Experimentos podem ser realizados via web comparando a heurística GRASP-PR com outros algoritmos da literatura. A ferramenta é disponível em http://goic.dcc.ufla.br/Biocomp.
3 GRASP WITH PATH-RELINKING FOR MAX-CMO

Recently, the growth of the Protein Data Bank (PDB) (Berman et al., 2000) has been accelerated by a large scale structure determination project, called structural genomics (Westbrook et al., 2003). As a result, fast and efficient algorithms for protein structure comparison have become more important to take advantage of the huge amount of structural data.

One promising way of accomplishing the structural alignment is to evaluate the alignment of their contact maps, which are used to represent the distances between every pair of residues in a three-dimensional protein structure.

In the graph representation, the contact map $G = (V, E)$ is a graph with a set of nodes $V$ corresponding to the sequence of residues and a set of edges $E$ corresponding to the edges for each pair of non-consecutive residues whose distance is below a given threshold.

Given two contact maps $G_A = (V_A, E_A)$ and $G_B = (V_B, E_B)$ such that $|V_A| = n$ and $|V_B| = m$, the CONTACT MAP OVERLAP PROBLEM (Goldman et al., 1999) is to find two subsets $S_A \subseteq V_A$ and $S_B \subseteq V_B$ with $|S_A| = |S_B|$ and an order preserving bijection $f$ between $S_A$ and $S_B$ such that the cardinality of the overlap set

$$\mathcal{L}(S_A, S_B, f) = \{(u, v) \in E_A : u, v \in S_A, (f(u), f(v)) \in E_B\}$$

is maximized. A solution $(S_A, S_B, f)$ for the contact map overlap problem can be represented as an assignment vector $p$ of size $n$ such that

$$p_u = \begin{cases} v & \text{if } (u, v) \in \mathcal{L}(S_A, S_B, f) \\ \text{nil} & \text{otherwise.} \end{cases}$$
Algorithm 1 shows pseudo-code for the GRASP (Feo & Resende, 1995) with path-relinking (Glover, 1996) heuristic for the MAX-CMO problem. The algorithm takes as input two contact maps $C^A$ and $C^B$ of proteins $A$ and $B$, with $n$ and $m$ residues ($m > n$), respectively. It outputs an array $p^*$ of length $n$, with $p^*_i = \text{nil}$, if node $i \in C^A$ representing residue $i \in A$ is not aligned, and $p^*_i = j$, if node $i \in C^A$ is aligned with node $j \in C^B$.

After initializing the elite set $P$ as empty, the GRASP with path-relinking iterations are computed until a stopping criterion is satisfied. This criterion could be, for example, a maximum number of iterations, a target solution quality, or a maximum number of iterations without improvement. In all iterations, a greedy randomized solution $p$ is generated and tentatively improved with a local search.

If the elite set $P$ is empty, solution $p$ is added to it. If the elite set is not empty, then while it is not full, solution $p$ is added to it if it is sufficiently different from the solutions already in the elite set. In order to define the term “sufficiently different” more precisely, let $\Delta(p, q)$ denote the number of assignments in $p$ that are different from those in $q$. For a given level of difference $\delta$, we say $p$ is sufficiently different from all elite solutions in $P$ if $\Delta(p, q) > \delta$ for all $q \in P$, which we indicate with the notation $p \not\approx P$.

If the elite set $P$ is full, then path-relinking is applied between $p$ and some elite solution $q$ randomly chosen from $P$, resulting in solution $r$. Next, $r$ is updated by a local minimum in its neighborhood. If $r$ is the best solution found so far, then it replaces $t$, the solution most similar to it. Otherwise, if $r$ is better than the worst solution in $P$ and $r \not\approx P$, then it replaces $t$, the solution most similar to it.

Several exact algorithms as well as heuristics have been since proposed for MAX-CMO problem. In addition to GRASP-PR heuristic, BIOCOMP web tool provides some algorithms from literature as follows. A Lagrangian relaxation ap-
proach proposed by Caprara & Lancia (2002), where the optimal Lagrange multipliers are found by subgradient optimization. A dynamic programming as tool to design a branch-and-bound algorithm with several reduction techniques to eliminate inferior residue-residue pairs early in the search procedure proposed as a reduction based exact algorithm by Xie & Sahinidis (2007). A multi-start variable neighborhood search heuristic for solving MAX-CMO developed by Pelta et al. (2008).

**Procedure GRASP+PR-CMOP**

**Input:** $C^A, C^B$

**Output:** solution $p^*$

$P \leftarrow \emptyset$;

while stopping criterion not satisfied do

$p \leftarrow \text{GreedyRandomized}(\cdot)$;

$p \leftarrow \text{LocalSearch}(p)$;

if $P$ is full then

Randomly select a solution $q \in P$;

$r \leftarrow \text{PathRelinking}(p, q)$;

$r \leftarrow \text{LocalSearch}(r)$;

if $c(r) > \max\{c(s) : s \in P\}$ then

$t \leftarrow \arg\min\{\Delta(r, s) : s \in P\}$;

$P \leftarrow P \cup \{r\} \setminus \{t\}$;

else if $c(r) > \min\{c(s) : s \in P\}$ and $r \not\approx P$ then

$t \leftarrow \arg\min\{\Delta(r, s) : s \in P : c(s) < c(r)\}$;

$P \leftarrow P \cup \{r\} \setminus \{t\}$;

else

if $P = \emptyset$ then

$P \leftarrow \{p\}$;

else if $p \not\approx P$ then

$P \leftarrow P \cup \{p\}$;

end

end

end

return $p^* = \arg\max\{c(s) : s \in P\}$;

Algorithm 1: GRASP-PR for MAX-CMO algorithm.
4 GOIC-BIOCOMP SERVER

Goic-Biocomp server (Figure 1) takes as input the selected algorithm for the MAX-CMO problem and the PDB files (Westbrook et al., 2003) corresponding to the proteins to be aligned. The protein contact maps used as input for the algorithm are generated from the input PDB files. The algorithm outputs the residue alignment and the cardinality of the overlap set. Depending on the comparison approach, a different set of information are reported in the output web page. In pairwise comparison approach, besides the algorithms results, we have the binary and real two-dimensional matrix representation of the contact maps and also the superimposed proteins plot. While these matrixes are generated by PConPy tool (Ho et al., 2008) from protein PDB files, the superimposed PDB plots are created by Jmol tool. (Jmol, 2009) takes as input superimposed PDB files created by BioPython tool from the residue alignment. Biopython (Cock et al., 2009) is also responsible to compute the root mean square deviation (RMSD), the average distance between the backbones of superimposed proteins.

In multiple comparison approach, three or more proteins are compared all-against-all through the pairwise approach. The result is a symmetric two-dimensional matrix having as distance metric the cardinality of the overlap set of each pairwise structure alignment. This matrix is normalized according to two normalization schemes reported by (Lancia & Istrail, 2004) and (Xie & Sahinidis, 2004). Next, the normalized matrix is used as input for three clustering methods (Everitt & Dunn, 1992): single, complete and average linkage clustering. At the end, the clustering results are plotted as dendograms using R statistical tool (R Development Core Team, 2005).
**FIGURE 1** Goic-Biocomp architecture

**Acknowledgement:** The authors wish to thank Alberto Caprara, Giuseppe Lancia, Juan R Gonzalez and Nick Sahinidis for the programs provided and the authorization to make them available on the site. The AT&T Labs Research, Florham Park, New Jersey, USA. Support for this effort was provided through National Council of Scientific and Technological Development (CNPq), Brazil and Fundação à Pesquisa do Estado de Minas Gerais (FAPEMIG), Brazil.
5 REFERENCES


CHAPTER 4

SUMMARY, GENERAL CONCLUSIONS AND FUTURE WORK
1 ABSTRACT

The final chapter contains the summary, general conclusions and outlines the possibilities for future work with the GRASP-PR algorithm presented in Chapter 2 and the Goic-Biocomp web tool described in Chapter 3.
2 RESUMO

O último capítulo contém o resumo, as conclusões gerais e descreve as possi-
bilitades de trabalho futuro com o algoritmo GRASP-PR apresentado no capítulo
2 e a ferramenta web Goic-Biocomp descrita no Capítulo 3.
3 SUMMARY AND GENERAL CONCLUSION AND FUTURE WORK

Protein structure comparison is one of the most important problems in bioinformatics (see Chapter 1, Section 2). One approach for solving this problem is to first, extract protein binary contact maps from the protein tertiary structure (see Chapter 1, Section 1), and next, align these contact maps. In order to guide the design and development of a new algorithm to solve this problem, we used the Maximum Contact Map Overlap (MAX-CMO), one of the most common mathematical statements of the contact map alignment problem. Our purpose is based on the hybridization of two promising heuristics, called Greedy Random Adaptive Search Procedure with Path-Relinking (GRASP-PR) for the MAX-CMO.

Initially, we aimed to validate our proposal by comparing the results obtained by other popular algorithms described in the literature that solve the MAX-CMO problem. Based on the performed comparisons and analysis, we can observe that the GRASP-PR is very competitive when compared against some of the most successful algorithms: the Variable Neighborhood Search heuristic and the Lagrangian Relaxation algorithm. In addition, from an optimization point of view, we can mention at least two ways to obtain further improvements to our propose (Chapter 2, Section 2): a) by trying more specialized Greedy randomized construction procedures and b) by better tuning the parameters’ values chosen.

An important element in several bioinformatics problems is the relation between the optimum value of the objective function and the biological relevance of the corresponding solution. In protein structure comparison, we should remember that we are dealing with a mathematical model that captures some aspects of the biological problem, being possible to measure protein structure similarity in several ways. For example, up to 37 measures are reviewed in (May, 1999).

Then, in a second moment, we aimed at investigating whether our heuristic
provides biologically meaningful alignments for practical instances; further investigating whether GRASP-PR can replicate the results obtained by applying exact methods with less computational effort. In order to accomplish this, we used three different assessment criteria for the algorithms: the overlap value, RMSD value and a visual inspection of the superimposition of the structures guided by the correspondence between residues of two proteins obtained by the MAX-CMO. We showed that our strategy can replicate the results obtained using exact methods but with less computational on practical instances. Furthermore, we observed that not always the different similarity measures and criteria for evaluating structural alignment algorithms agree with each other, a question much discussed since 1996 in the work entitled “The structural alignment between two proteins: is there a unique answer?” (Godzik, 1996). In this sense, a promising strategy would be to combine two of the most used similarity metrics in an attempt to reshape the MAX-CMO problem with a multi-objective function: minimize the RMSD and to maximize the number of overlaps.

Besides obtaining the highest overlap values, it is also critical to develop strategies able to obtain a proper similarity ranking of proteins. Our experiments showed that in terms of SCOP’s family the (normalized) overlap values given by the GRASP-PR seemed to be good enough to capture the similarity.

After the GRASP-PR algorithm was implemented and validated, we built an interface that allows easy use of MAX-CMO algorithms in order to perform the pairwise structural alignment or structural clustering of proteins. This web interface is available at http://goic.dcc.ufla.br and we call it Goic-Biocomp web tool (Chapter 3). Goic-Biocomp server produces up to six kinds of output when the task of pairwise structural alignment is requested: (i) the distance map plots; (ii) the contact map plots; (iii) a PDB file containing the coordinates of the
superimposed molecules; (iv) a sequence alignment corresponding to the equivalent residues found by the MAX-CMO; (v) an RMSD report that contains the calculated RMSD values (in Angstroms) between the superimposed molecules; (vi) a Jmol (Jmol, 2009) applet view of the superimposed molecules. For the structural clustering, the Goic-Biocomp server produces the dendogram as output.

In summary, Goic-Biocomp web tool provides a simple-to-use, web-accessible approach to performing the structural pairwise alignment and the structural clustering, via MAX-CMO using different successful algorithms reported in literature, allowing several studies on protein structures with different purposes - The interested reader is referred to (Wolfon et al., 2005) for a detailed description of methods and applications for the determination of protein function.
4 REFERENCES


