FEDERAL UNIVERSITY OF TECHNOLOGY – PARANÁ
GRADUATE PROGRAM IN ELECTRICAL AND COMPUTER ENGINEERING

CÉSAR MANUEL VARGAS BENÍTEZ

CONTRIBUTIONS TO THE STUDY OF THE PROTEIN FOLDING PROBLEM USING BIOINSPIRED COMPUTATION AND MOLECULAR DYNAMICS

DOCTORAL THESIS

CURITIBA

2015

CÉSAR MANUEL VARGAS BENÍTEZ

CONTRIBUTIONS TO THE STUDY OF THE PROTEIN FOLDING PROBLEM USING BIOINSPIRED COMPUTATION AND MOLECULAR DYNAMICS

Doctoral Thesis presented to the Graduate Program in Electrical and Computer Engineering of the Federal University of Technology – Paraná as partial fulfillment of the requirements for the degree of "Doctor of Science (D.Sc.)" – Area of concentration: Computer Engineering.

Advisor: Prof. Dr. Heitor Silvério Lopes

CURITIBA
2015

Dados Internacionais de Catalogação na Publicação

B467c Benítez, César Manuel Vargas
2015 Contributions to the study of the protein folding problem using bioinspired computation and molecular dynamics / César Manuel Vargas Benítez.-- 2015.
193 p.: il.; 30 cm

Texto em inglês, com resumo em português.
Tese (Doutorado) - Universidade Tecnológica Federal do Paraná. Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial, Curitiba, 2015.
Bibliografia: p. 159-180.

1. Dobramento de proteínas. 2. Bioinformática. 3. Computação evolutiva. 4. Processamento paralelo (Computadores). 5. Autômato celular. 6. Dinâmica molecular. 7. Métodos de simulação. 8. Engenharia elétrica - Teses. I. Lopes, Heitor Silvério, orient. II. Universidade Tecnológica Federal do Paraná - Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial. III. Título.

CDD 22 -- 621.3

Biblioteca Central da UTFPR, Câmpus Curitiba

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
Câmpus Curitiba

Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial

Título da Tese Nº. 112

Contributions to the study of the protein folding problem using bioinspired computation and molecular dynamics.

por

César Manuel Vargas Benítez

Orientador: Prof. Dr. Heitor Silvério Lopes

Esta tese foi apresentada como requisito parcial à obtenção do grau de DOUTOR EM CIÊNCIAS – Área de Concentração: Engenharia deq Computação, pelo Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial – CPGEI – da Universidade Tecnológica Federal do Paraná – UTFPR, às 13h do dia 27 de abril de 2015. O trabalho foi aprovado pela Banca Examinadora, composta pelos doutores:

Prof. Dr. Heitor Silvério Lopes
(Presidente – UTFPR)

Prof^{a}. Dr^{a}. Karina dos Santos Machado
(FURG)

Prof^{a}. Dr^{a}.Denise Tsunoda
(UFPR)

Prof. Dr. Fabrício Martins Lopes
(UTFPR-CP)

Prof. Dr. Rafael Bertolini Frigori
(UTFPR-TD)

Visto da Coordenação:

Prof. Emilio Carlos Gomes Wille, Dr.
(Coordenador do CPGEI)

A Folha de Aprovação assinada encontra-se na Coordenação do Curso (ou Programa).

In memory of my son Ravi, my sister Fatima and my uncle Victor, who passed away during the period of my doctorate course.

I also dedicate this work to my family, especially to my wife Karina and my cat Fourier.

ACKNOWLEDGEMENTS

First of all, I would like to express my deepest gratitude to my advisor, Prof. Heitor Silvério Lopes, for his support and guidance over the years.

My sincere appreciation is extended to Profs. Denise Tsunoda, Karina Machado, Fabrício Lopes, Rafael Frigori, Elaine Benelli, André Kashiwabara and Rafael Barreto for their willingness to be referees for the doctoral examinations and for their advice.

I would like to thank my colleagues of the BIOINFO Lab and the CPGEI staff, especially the professors and Terezinha.

Finally, I would like to extend my deepest gratitude to my family. Without their support, I probably would not have a chance to be at UTFPR. My heartfelt gratitude to my wife, Karina, for her inestimable comprehension. To my cat, Fourier, for his affection while working at night.

Words cannot express how thankful I am.

"You can't cross the sea merely by standing and staring at the water."
Rabindranath Tagore

"Physics is mathematical not because we know so much about the physical world, but because we know so little; it is only its mathematical properties that we can discover"
Bertrand Russell

"We can't solve problems by using the same kind of thinking we used when we created them"
Albert Einstein

"There's always some further action to take"
Pierre Boulle

ABSTRACT

VARGAS BENÍTEZ, CÉSAR MANUEL. Contributions to the study of the Protein Folding Problem using Bioinspired Computation and Molecular Dynamics. 193 p. Doctoral Thesis – Graduate Program in Electrical and Computer Engineering, Federal University of Technology – Paraná. Curitiba, 2015.

The Protein Folding Problem (PFP) is considered one of the most important open challenges in Biology and Bioinformatics. In this thesis, a novel approach for simulating the protein folding pathways is proposed where, instead using the three-dimensional structure of the protein, the folding states are represented by Contact Maps (CM). A two-dimensional Cellular Automata (2D-CA) evolver is used to simulate the folding process, where each configuration represents a folding state and it is obtained according to its predecessor and a transition rule. Since finding transition rules for simulating a dynamic behavior is a very difficult task, it is proposed a distributed Gene-Expression Programming (GEP)-based approach, called pGEP-CA. Specific fitness functions, based on similarity and symmetry measures, are proposed. Futhermore, a heterogeneous parallel Ecology-inspired algorithm is proposed. This algorithm, called pECO, is used for reconstructing the structures from the CMs, using the 3D-AB *off-lattice* model. Moreover, to the best of our knowledge, it is presented the first application of Molecular Dynamics (MD) to the PFP, using the same model of proteins. Experiments were done to evaluate the adequacy of the proposed approaches. Also, a brief analysis of the load balancing of the parallel architectures is presented. Results show that the approaches obtained coherent results, suggesting their adequacy for the problem. The induced transition rules by the pGEP-CA are able to generate 2D-CA that represent CMs correctly. Concerning the pECO approach, results show that the combination of concurrent evolutionary approaches took advantage of both the coevolution effect and the different search strategies. In addition, it can be observed that the MD approach is capable of displaying biological features such as the hydrophobic core formation and the protein breathing motion. Furthermore, it is observed that parallel processing was not only justified but also essential for obtaining results in reasonable processing time. Finally, concluding remarks and several research directions for future works are presented.

Keywords: Protein Folding, Bioinformatics, Evolutionary Computation, Parallel Computing, Cellular Automata, Molecular Dynamics

RESUMO

VARGAS BENÍTEZ, CÉSAR MANUEL. CONTRIBUTIONS TO THE STUDY OF THE PROTEIN FOLDING PROBLEM USING BIOINSPIRED COMPUTATION AND MOLECULAR DYNAMICS. 193 f. Tese de doutorado – Programa de Pós-graduação em Engenharia Elétrica e Informática Industrial, Universidade Tecnológica Federal do Paraná. Curitiba, 2015.

O Problema de Dobramento de Proteínas (PDP) é considerado um dos desafios abertos mais importantes da Biologia e Bioinformática. Nesta tese, uma nova abordagem para simular os *pathways* de dobramento de proteínas é proposta onde, ao invés de utilizar a estrutura tridimensional da proteína, os estados de dobramento são representados por Mapas de Contatos (MC). Autômatos Celulares bidimensionais (2D-CA) são utilizados para simular o processo de dobramento, onde cada configuração representa um estado de dobramento e é obtida em relação ao seu estado predecessor e uma regra de transição. Determinar uma regra de transição para um dado comportamento dinâmico representa uma tarefa complexa. Portanto, é apresentada uma abordagem distribuida baseada em Programação de Expressão Gênica, chamada pGEP-CA. Funções de *fitness* específicas, baseadas em medidas de similaridade e simetria, são propostas. Também, um algoritmo heterogêneo paralelo Ecologicamente-inspirado é proposto. Este algoritmo, chamado pECO, é utilizado na reconstrução de estruturas a partir de MCs, usando o modelo 3D-AB *off-lattice*. De acordo com o nosso conhecimento, é apresentada a primeira aplicação de Dinâmica Molecular (DM) ao PFP, usando o mesmo modelo de proteínas. Experimentos foram realizados para verificar a adequabilidade das abordagens propostas. Além disto, uma breve análise sobre o balanceamento de carga de processamento das arquiteturas paralelas é apresentada. Os resultados mostram que as abordagens obtiveram resultados coerentes, sugerindo que são adequadas para o problema. As regras de transição induzidas pelo pGEP-CA são capazes de gerar 2D-CA que representam MCs corretamente. Sobre a abordagem pECO, os resultados demonstram que a combinação de abordagens evolucionárias concorrentes se beneficia do efeito da coevolução e das diferentes estratégias de busca. Além disto, pode ser observado que a abordagem de DM é capaz de levar a conformações que mimetizam propriedades biológicas, como a formação do núcleo hidrofóbico e os movimentos de respiração (*breathing*) das proteínas. Também foi observado que o processamento paralelo é essencial, permitindo a obtenção de resultados em tempos de processamento razoáveis. Finalmente, as conclusões e diversas direções de pesquisa são apresentadas.

Palavras-chave: Dobramento de proteínas, Bioinformática, Computação Evolucionária, Computação Paralela, Autômatos Celulares, Dinâmica Molecular

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVIATIONS

3DHP-SC	3DHP Side-Chain model
ABC	Artificial Bee Colony
ACMC	Annealing Contour Monte Carlo
ACO	Ant Colony Optimisation
AIS	Artificial Immune Systems
AMBER	Assisted Model Building with Energy Refinement
ANFIS	Adaptative Neuro-Fuzzy Inference System
ANOVA	Analysis of Variance
BCC	Body-Centered-Cubic
CA	Cellular Automata
CAO	Contact Appearance Order
CASP	Critical Assessment of Structure Prediction
CATH	Class, Architecture, Topology, Homology
CD	Circular Dichroism
CG	Coarse-Grained
CGE	Charge Graph Embedding model
CM	Contact Maps
COMAR	COntact MAp Reconstruction
CPSP	Constraint-based Protein Structure Prediction
cryo-ME	Cryo-electron Microscopy
DMD	Discrete Molecular Dynamics
DNM	Damped Network Model
EC	Evolutionary Computation
ENM	Elastic Network Model
FASTER	Fast and accurate Side-Chain Topology and Energy Refinement
FCC	Face-Centered-Cubic
FPGA	Field-Programmable Gate Arrays
FR	Folding Recognition
FRET	Fluorescence Resonance Energy Transfer
FT-COMAR	Fault Tolerant Contact Map Reconstruction
GA	Genetic Algorithms
GP	Genetic Programming
HPGA	Hierarchical Parallel Genetic Algorithm
HP-TSSC	Hydrophobic-Polar Tangent Spheres Side Chain Model
HMM	Hidden Markov Models
KNN	K Nearest Neighbors
LCG	Linear Congruential Generator
LPE	Lattice Polymer Embedding
MC	Monte-Carlo
MCFI	Monte Carlo Fragment Insertion
MD	Molecular Dynamics

MIMD	Multiple Instruction stream Multiple Data stream
MJ	Miyazawa-Jerningan
MPI	Message Passing Interface
ML	Machine Learning
NMR	Nuclear Magnetic Resonance
NN	Neural Networks
NOE	Nuclear Overhauser Effect
PCLF	Protein Chain Lattice Fitting
PCN	Protein Contact Networks
PDF	Probability Distribution Function
PH	Perturbed Homopolymer
PRIMO	Protein Intermediate Model
PRIMONA	Protein/nucleic-acid version of PRIMO
RAPTOR	RApid Protein Threading predictOR
RMSD	Root-Mean-Square Deviation
Rn	Rank of Native structure
SABBAC	Structural Alphabet based protein Backbone Builder from Alpha Carbon trace
SC	Side Chain
SCM	Side chain Center of Mass
SCOP	Structural Classification of Proteins
SICHO	Side Chain Only
SSE	Secodary Structure Elements
SVD	Singular Value Decomposition
SVM	Support Vector Machine
TASSER	Threading/ASSembly/Refinement
TS	Tabu Search
TSE	Transition State Ensemble
UNRES	United-residue
WSG	Weighted Secondary structure element Graph

CONTENTS

1 INTRODUCTION

"Proteins don't have a folding problem. It's we humans that do."

—Ram Samudrala

1.1 MOTIVATIONS

Nowadays, one of the most important and open challenging problems in Molecular Biology and Bioinformatics is to obtain a better understanding of the protein folding process. In this process, under physiological conditions, every protein folds into a unique three-dimensional structure, that determines their specific biological function.

It is known that ill-formed proteins can be completely inactive or even harmful to the organism. Several diseases are believed to be the result of the aggregation of ill-formed proteins, such as Alzheimer's disease, cystic fibrosis, Huntington's disease and some types of cancer (LUHESHI; DOBSON, 2009; BROADLEY; HARTL, 2009; CHEN et al., 2008). Therefore, acquiring knowledge about the tertiary structure of proteins is an important issue, since such knowledge can lead to important medical and biochemical advancements and even to the development of new drugs with specific functionality (RÖTHLISBERGER et al., 2008; BROGLIA; TIANA, 2003).

Due to its great importance for Medicine and Biochemistry, researchers have been focusing on the study of this process and, consequently, many information is available. Thanks to the several genome sequencing projects being conducted in the world, a large number of new proteins have been discovered. However, only a small amount of such proteins have its 3-dimensional structure known. For instance, the UniProtKB/TrEMBL repository [1] of protein sequences has currently around 92 million records (as in March/2015), and the Protein Data Bank – PDB (BERMAN et al., 2000)

[1] Available at http://www.ebi.ac.uk/uniprot/

has the structure of only 107,620 proteins (as in March/2015). This fact is due to the cost and difficulty in unveiling the structure of proteins, from the biochemical point of view. It is here that Computer Science plays an important role, developing computational models and approaches for the Protein Folding Problem (PFP). The simplest computational model for the PFP is known as Hydrophobic-Polar (HP) model, both in two (2D-HP) and three (3D-HP) dimensions (DILL et al., 1995). However, the computational approach for searching a solution for the PFP using simple HP models was proven to be *NP*-complete (ATKINS; HART, 1999).

Several methods have been developed for protein structure prediction from sequence. Moreover, template-based homology and threading/taxonomic methods use information of previously solved native structures. However, *ab initio* methods perform the protein structure prediction without using any previously obtained structural information. They are based on the principles of physics, where conformations are evaluated according to an energy or a scoring function. To the best of our knowledge, the Molecular Dynamics (MD) approach (including its variations) is the only computational methodology that really provides a time-dependent analysis of the folding process (LIWO; KHALILI; SCHERAGA, 2005). Generally, it involves the three-dimensional coordinates of the particles that form the protein and numerical integration of the classical equations of motion.

Despite the great advances in recent years, MD simulations have been limited by the current computer hardware and their computationally expensive force calculation. Currently, to overcome such drawback, two solutions are to use coarse-grained models and faster hardware (HARDER et al., 2012). For instance, there are proposed methods that use distributed computing (i.e. Folding@home (LANE et al., 2013)) and custom built special-purpose hardware.

An alternative way to reduce the inherent complexity of the simulations with three-dimensional structures is to use Contact Maps (CMs), which are minimalistic two-dimensional representations (VENDRUSCOLO; KUSSELL; DOMANY, 1997). Notwithstanding, the use of CMs to study the protein folding have been sparsely explored. In recent literature, methods have been developed for their prediction from sequence. Furthermore, few research groups proposed heuristic approaches for protein structure reconstruction from native CMs (VASSURA et al., 2011; DUARTE et al., 2010). Also, they have been used in mining of secondary local structures (HU et al., 2002).

A particular class of computer programs for reproducing complex behaviors are the Cellular Automata (CA), in which a configurational state is determined according to its predecessor and a *transition rule*. The two-dimensional version of the CA (2D-CA) is suitable to model CMs. Therefore, we propose a novel approach for simulating the protein folding using them, aiming at taking advantage of the previously cited benefits of both approaches.

Despite being computationally simple, it is very hard to find a transition rule for a given dynamic behavior. Therefore, a novel approach based on Gene Expression Programming (GEP) for inducting *transition rules* of 2D-CA for simulating the protein folding is presented in this thesis.

A reconstruction procedure of the three-dimensional structure of proteins is need after the CM prediction, which has been proved to be *NP*-hard (VASSURA et al., 2008b). Consequently, metaheuristic approaches seem to be the most reasonable algorithmic choice for dealing with the problem. Regarding this issue, this thesis also presents a parallel ecological-inspired approach. It is based on Evolutionary Computation algorithms working cooperatively, featuring the 3D-AB *off-lattice* model of proteins.

1.2 OBJECTIVES

The main objective of this thesis is to propose and apply computational methods to the protein folding problem that can be used to reproduce folding pathways, aiming at obtaining the three-dimensional structure of proteins, based on a *coarse-grained* model of proteins. The specific objectives are:

- To elaborate a study of the theoretical issues related to the Protein Folding Problem;
- To present an approach based on Cellular Automata for predicting Contact Maps of proteins;
- To propose a Evolutionary Computation approach for finding *transition rules* of Cellular Automata applied to the simulation of the protein folding process, using Contact Maps;
- To develop a heterogeneous Bio-inspired Computation approach applied to the reconstruction of protein structures from Contact Maps;

- To study and develop parallel approaches in a Beowulf Cluster to speed up processing time;
- To implement a Molecular Dynamics approach using a coarse-grained model of proteins in order to simulate the folding process and generate the Contact Maps that can be used for evaluating the performance of the proposed approaches;
- To present new benchmarks for the 3D-AB *off-lattice* model;

1.3 OUTLINE

This thesis is divided into five chapters. Chapter 2 presents a review of the literature, starting with the description of proteins. The reader will find an overview about the protein folding problem (PFP), including descriptions of protein models and computational methods. In addition, a thorough description of the computational methods related to this thesis, such as Molecular dynamics, Cellular Automata, Evolutionary Computation and Parallel Computing. Chapter 3 describes in details the proposed methodology, starting with an overview and a detailed description of each building block. Chapter 4 presents the experiments and results achieved in this work. Finally, Chapter 5 presents some final considerations and future research directions. An appendix comprising some aspects is given as well.

1.4 CONTRIBUTIONS

During the development of this thesis, several subjects were addressed. Some of them have resulted in scientific publications with relevant contributions, that are presented below:

Chidambaram et al. (2011) present the application of Machine Learning and Evolutionary Computation methods to define suitable classifiers for predicting the secondary structure of proteins, starting from their primary structure (that is, their linear sequence of amino acids).

Benítez et al. (2011) present Reconfigurable Hardware Computing approaches for accelerating protein folding simulations using the Harmony Search algorithm and the 3D-HP-SC model of proteins.

Benítez, Parpinelli and Lopes (2012) report the hybridization of the Artificial

Bee Colony (ABC) and Genetic Algorithm (GA), in a hierarchical topology.

The application of Molecular Dynamics to the PFP using the 3D-AB *off-lattice* model of proteins is presented in (BENÍTEZ; LOPES, 2013) and (BENÍTEZ; LOPES, 2012). They also offered new reference values for synthetic and globular protein sequences.

A heterogeneous parallel ecology-inspired algorithm (pECO) applied to search low energy conformations for the PSP, concerning the 3D-AB *off-lattice* model, is presented in (BENÍTEZ; PARPINELLI; LOPES, 2013).

Scalabrin et al. (2014) present a new evolutionary algorithm based on the standard Harmony Search strategy, called population-based Harmony Search (PBHS). It also provides a parallelization method for the proposed algorithm, using Graphical Processing Units (GPU).

A performance comparison of Swarm Intelligence algorithms for the Protein Structure Prediction Problem (PSP) is presented in (PARPINELLI et al., 2014).

Benítez, Weinert and Lopes (2015) present a novel distributed bio-inspired approach that uses Gene Expression Programming (GEP) to evolve transition rules for two-dimensional Cellular Automata (2D-CA) applied to the protein Contact Map Prediction.

2 THEORETICAL BACKGROUND

"The only real wisdom is knowing you know nothing"

— Socrates

2.1 PROTEINS

Proteins are essential for life and they have countless biological functions. Proteins are synthesized in the ribosome of cells following a template given by the messenger RNA (mRNA). During the synthesis, the protein folds into a unique three-dimensional structure, known as native conformation. This process is called protein folding. The biological function of a protein depends on its three-dimensional conformation, which in turn, is a function of its primary and secondary structures.

All proteins are composed by a chain of amino acids (also called residues) that are linked by means of peptide bonds. Each amino acid is characterized by a central carbon atom (also known as Cα) to which are attached (as shown in Figure 1) a hydrogen atom, an amine group (NH_2), a carboxyl group (COOH) and a side-chain that gives to each amino acid a distinctive function (also known as radical R). Two amino acids form a peptide bond when the carboxyl group of one molecule reacts with the amino group of the other. This process of amino acids aggregation is known as dehydration by releasing a water molecule (GRIFFITHS et al., 2000). All amino acids have the same backbone and they differ from each other only by the side-chain, which can range from just a hydrogen atom (in glycine) to a complex heterocyclic group (in tryptophan). The side-chain defines the physical and chemical properties of the amino acids of a protein (NELSON; COX, 2008; COOPER, 2000).

There are numerous amino acids in the nature, but only 20 are proteinogenic. They are shown in Table 25 – Annex A. The first to be discovered was asparagine, in

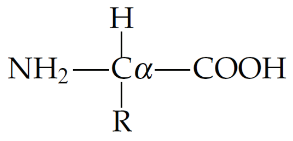


Figure 1: General structure of an α-amino acid. The side-chain (R element) attached to the Cα defines the function of the amino acid

Source: Adapted from (NELSON; COX, 2008).

1806. The last, threonine, was identified in 1938 (NELSON; COX, 2008).

In order to understand the structures and functions of proteins, it is of fundamental importance to have knowledge about the properties of the amino acids, defined by their side-chain. Thus, the amino acids can be grouped into five categories: hydrophobic (also called non-polar), hydrophilic (also called polar), neutral, basic and acid (COOPER, 2000).

2.1.1 HYDROPHOBICITY SCALES

There have been proposed several hydrophobicity scales. Some of them are presented below:

- **Kyte-Doolittle scale (KYTE; DOOLITTLE, 1982)**: this scale is based on a combination of statistical data on distribution of amino acids between surface-exposed and buried locations in globular protein structures, and of water-vapor partition coefficients for amino acid analogues.
- **Wimley scale (WIMLEY; WHITE, 1996)**: this scale includes the contribution of the peptide bond determined from the partitioning of two series of small model peptides into the interfaces of neutral phospholipid membranes.
- **Hessa scale (HESSA et al., 2005)**: this "biological" hydrophobicity scale was obtained by measuring the ratio of *glycosylated Lep* molecules that insert into the membrane bilayer mediated by the Sec61 *translocon* (SNAPP et al., 2004).
- **Einsenberg scale (EISENBERG et al., 1984)**: this scale is a consensus scale, derived from five other scales and normalized to a mean of zero and a standard deviation of one.
- **Janin scale (JANIN, 1979)**: this scale also provides information about the accessible and buried amino acid residues of globular proteins.

- **Engelman scale (ENGELMAN; STEITZ; GOLDMAN, 1986)**: this scale was obtained through the calculation of free transfer energies using hydrophobic terms derived from accessible surface area calculations, and hydrophilibc terms derived from calculations of Born and H-bonding energies.

Table 1 shows the hydrophobicity scales, where positive values represent hydrophobic amino acids in the Kyte-Doolittle, Wimley, Eisenberg, Janin and Engelman scales. In the Hessa scale, unlike the other scales, the more negative values reflect the larger hydrophobicity.

Table 1: Hydrophobicity scales

Aminoacid		Kyte & Doolittle	Wimley	Hessa	Eisenberg	Janin	Engelman (GES)
Alanine	Ala	1.8	-0.17	-0.11	0.62	0.3	1.6
Cysteine	Cys	2.5	0.24	0.13	0.29	0.9	2
Aspartic acid	Asp	-3.5	-1.23	-3.49	-0.9	-0.6	-9.2
Glutamic acid	Glu	-3.5	-2.02	-2.68	-0.74	-0.7	-8.2
Phenylalanine	Phe	2.8	1.13	0.32	1.19	0.5	3.7
Glycine	Gly	-0.4	-0.01	-0.74	0.48	0.3	1
Histidine	His	-3.2	-0.96	-2.06	-0.4	-0.1	-3
Isoleucine	Ile	4.5	0.31	0.6	1.38	0.7	3.1
Lysine	Lys	-3.9	-0.99	-2.71	-1.5	-1.8	-8.8
Leucine	Leu	3.8	0.56	0.55	1.06	0.5	2.8
Methionine	Met	1.9	0.23	0.1	0.64	0.4	3.4
Asparagine	Asn	-3.5	-0.42	-2.05	-0.78	-0.5	-4.8
Proline	Pro	-1.6	-0.45	-2.23	0.12	-0.3	-0.2
Glutamine	Gln	-3.5	-0.58	-2.36	-0.85	-0.7	-4.1
Arginine	Arg	-4.5	-0.81	-2.58	-2.53	-1.4	-12.3
Serine	Ser	-0.8	-0.13	-0.84	-0.18	-0.1	0.6
Threonine	Thr	-0.7	-0.14	-0.52	-0.05	-0.2	1.2
Valine	Val	4.2	-0.07	0.31	1.08	0.6	2.6
Tryptophan	Trp	-0.9	1.85	-0.3	0.81	0.3	1.9
Tyrosine	Tyr	-1.3	0.94	-0.68	0.26	-0.4	-0.7

Source: Adapted from (KYTE; DOOLITTLE, 1982), (WIMLEY; WHITE, 1996), (HESSA et al., 2005), (EISENBERG et al., 1984), (JANIN, 1979) and (ENGELMAN; STEITZ; GOLDMAN, 1986).

Table 2 shows the amino acid classification according to the hydrophobicity scales and the classification by (ALBERTS et al., 2002). In this table, the amino acids are classified into two classes, according to their affinity to water (hydrophobicity): 'A' (hydrophobic) and 'B' (hydrophilic or polar).

2.1.2 STRUCTURE OF PROTEINS

From the chemical point of view, proteins are structurally complex and functionally sophisticated molecules. The structural organization of proteins is commonly described into four levels of complexity, in which the upper cover the properties of lower: primary, secondary, tertiary and quaternary structures (LODISH et al., 2000; GRIFFITHS et al., 2000; NELSON; COX, 2008; COOPER, 2000).

Table 2: Amino acid classification according to the hydrophobicity scales

Amino acid		Alberts	Kyte & Doolittle	Wimley	Hessa	Eisenberg	Janin	Engelman (GES)
Alanine	Ala	A	A	B	A	A	A	A
Cysteine	Cys	A	A	A	A	A	A	A
Aspartic acid	Asp	B	B	B	B	B	B	B
Glutamic acid	Glu	B	B	B	B	B	B	B
Phenylalanine	Phe	A	A	A	A	A	A	A
Glycine	Gly	A	B	B	B	A	A	A
Histidine	His	B	B	B	A	B	B	B
Isoleucine	Ile	A	A	A	A	A	A	A
Lysine	Lys	B	B	B	B	B	B	B
Leucine	Leu	A	A	A	A	A	A	A
Methionine	Met	A	A	A	A	A	A	A
Asparagine	Asn	B	B	B	B	B	B	B
Proline	Pro	A	B	B	B	A	B	B
Glutamine	Gln	B	B	B	B	B	B	B
Arginine	Arg	B	B	B	A	B	B	B
Serine	Ser	B	B	B	B	B	B	A
Threonine	Thr	B	B	B	B	B	B	A
Valine	Val	A	A	B	A	A	A	A
Tryptophan	Trp	A	B	A	A	A	A	A
Tyrosine	Tyr	B	B	A	A	A	B	B

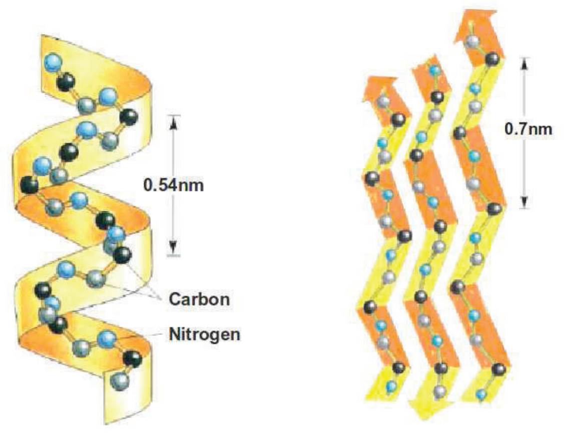
Source: Own work, based on Table 1.

The primary structure is the linear sequence of amino acids. This is the simplest level of organization, it represents only the peptide bonds between amino acids.

The secondary structure of a protein refers to the local conformation of some part of a three-dimensional structure. There are, basically, three main secondary structures: α-helices (PAULING; COREY; BRANSON, 1951a), β-sheets (PAULING; COREY; BRANSON, 1951b) and turns (LEWIS; MOMANY; SCHERAGA, 1973). In the structure of an α-helix, the backbone is tightly turned around an imaginary helix (spiral) and the side-chains of the amino acids protrude outwards the backbone (Figure 2(a)). The α-helix is stabilized by hydrogen bonds between the carboxyl oxygen of the amino acid residue at the position n in the polypeptide chain with the amide group (NH) of the residue $n+4$ (NÖLTING, 2006). The β-sheet is formed by two or more polypeptide segments of the same molecule, or different molecules, arranged laterally and stabilized by hydrogen bonds between the NH and CO groups (Figure 2(b)). Adjacent polypeptides in a β-sheet can have same direction (parallel β-sheet) or opposite directions (antiparallel β-sheet). Functionally, the antiparallel β-sheets are present in various types of proteins, for example, enzymes, transport proteins, antibodies and cell-surface proteins (BRANDEN; TOOZE, 1999). Hydrogen bonds between carboxyl oxygens and amide groups of adjacent strands stabilize β-sheets (NÖLTING, 2006). Turns are composed by a small number of amino acids and they are usually located in the surface of proteins forming folds that redirect the polypeptide chain into the protein. They allow large proteins to fold into highly compact structures. Turns are stabilized by a hydrogen bond between the carboxyl oxygen of the residue at the position n with the amide group (NH) of the residue $n+3$ (NÖLTING, 2006).

Secondary structures can be associated through side-chain interactions to motifs (BRANDEN; TOOZE, 1999; GRIFFITHS et al., 2000; NÖLTING, 2006). Motifs are patterns often found in three-dimensional structures that perform specific functions. For instance, the helix-turn-helix motif is important in DNA-protein interactions.

Different probabilities are observed for the incorporation of the amino acid residues into different types of secondary structure (CREIGHTON, 1993; NÖLTING, 2006). Table 26 in Annex A shows the conformational preference parameters which are based on the occurrence of a specific amino acid in a specific type of secondary structure. For instance, alanine is considered as an α-helix former. (HARPER; ROSE, 1993) pointed that the conformational preference depends on the relative position in the secondary structure element.



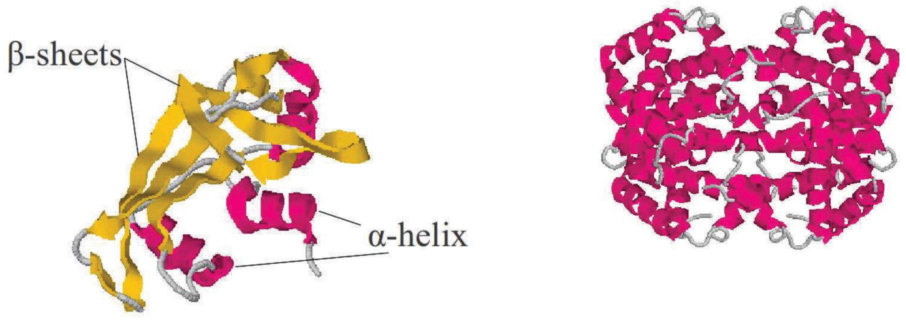
(a) (b)

Figure 2: α-helix (a) and β-sheet (b) structures.

Source: Adapted from (ALBERTS et al., 2002).

The tertiary structure represents the conformation of a polypeptide chain, i.e. the three-dimensional arrangement of the amino acids. The tertiary structure is the folding of a polypeptide as a result of interactions between the side chains of amino acids that are in different regions of the primary structure. Figure 3(a) shows an example of tertiary structure, where the presence of two secondary structures: α-helix and β-sheet can be observed.

Finally, the quaternary structure consists of the interactions between different polypeptide chains in proteins composed of more than one polypeptide (COOPER, 2000), as shown in Figure 3(b).



(a) (b)

Figure 3: Tertiary structure of Ribonuclease-A (a) and quaternary structure of Hemoglobin (b).

Source: Adapted from figures that were generated using RasMol [1] from PDB files (BERMAN et al., 2000).

Proteins can be classified into two major groups, considering higher levels of structure (NELSON; COX, 2008): fibrous and globular proteins. Both groups are structurally different: fibrous proteins consist of a single type of secondary structure; globular proteins have a nonrepetitive sequence and often contain several types of secondary structures. Helices are the most abundant form of secondary structure in globular proteins, followed by sheets, and in the third place, turns (NÖLTING, 2006).

2.2 THE PROTEIN FOLDING PROBLEM

The *protein folding* is the process by which polypeptide chains are transformed into compact structures that perform biological functions. These functions include control and regulation of essential chemical processes for the living organisms. Under physiological conditions, the most stable three-dimensional structure is called the native conformation and actually allows a protein to perform its function.

In vitro experiments carried out by Christian Anfinsen and colleagues (ANFINSEN, 1973) show that proteins can be denaturated by modifications in the environment where they are. Most proteins can be denaturated by temperature and pH variations, affecting weak interactions between residues (i.e.: hydrogen bonds). During the denaturation process, proteins lose their native shape and, consequently, their function. Anfinsen showed that some denatured (misfolded or unfold) proteins can refold into their native conformation. However, the spontaneous refolding only occurs in single-domain proteins. Failure to fold into the intended three-dimensional conformation usually leads to proteins with different properties that simply become inactive. In the worst case, such misfolded (incorrectly folded) proteins can be harmful to the organism.

Better understanding the protein folding process could help to: (a) accelerate drug discovery by replacing slow, expensive structural biology experiments with faster computational simulations, and (b) infer protein function from genome sequences. With the fast exponential growth of experimentally determined structures available in the PDB (BERMAN et al., 2000) (see Section 2.3), protein structure prediction has become as much a problem of inference and machine learning as it is of protein physics (DILL et al., 2008).

Despite the considerable theoretical and experimental effort expended to study the protein folding process, there is not yet a detailed description of the mechanisms that govern the folding process.

Although the concept of the folding process arose in the field of Molecular Biology, this problem is clearly interdisciplinary, requiring support of many knowledge areas, and it is considered to be one of the most important open challenges in Biology and Bioinformatics (NICOSIA; STRACQUADANIO, 2008). In contemporary Computational Biology, there are two protein folding problems. The first problem, called Protein Structure Prediction (PSP), is to predict the protein structure (conformation) from sequence (primary structure). The second one is the Protein Folding Problem (PFP). It is to predict the protein folding pathways, which consists in determining the sequence of folding events that leads from the primary structure of a protein (its linear sequence of amino acids) to its native structure (LOPES, 2008).

There are many computational methods to deal with the folding problem. However, the Molecular Dynamics (MD) approach (including all its variants) is the only computational methodology that really provides a time-dependent analysis of a system in Molecular Biology and, consequently, it can be employed to solve the PFP (LIWO; KHALILI; SCHERAGA, 2005).

Several computational models have been proposed for representing protein structures with different levels of complexity and, consequently, computational feasibility. Ideally, both the protein and the solvent should be represented at atomistic level because this approach is the closest to reality (DAY; DAGGETT, 2003). However, the simulation of computational models that take into account all the atoms of a protein is frequently unfeasible due to the dimensionality of the system ($> 10^4$ degrees of freedom) (LIWO; KHALILI; SCHERAGA, 2005), even with the most powerful computational resources (in nature, proteins can rapidly and reliably find their way into well-defined folded configurations). Generally, atomistic simulations of real-size proteins are usually limited to unfolding the native conformation of the proteins followed by refolding (DAY; DAGGETT, 2003). The dimensionality of a system containing the protein and the solvent can be reduced when the solvent is treated implicitly and a reduced coarse-grained model of proteins is used. In this scenario, several reduced (mesoscopic) models have been proposed. Although such reduced models are not realistic, their simulation can show some characteristics of real proteins. Computational studies of reduced models have provided several valuable insights into the folding process.

The prediction of the structure of a protein is modelled as the minimization of the corresponding free-energy, following the Anfinsen's Thermodynamic Hypothesis (see Section 2.2.1). It is also known that the native conformation of a protein represents the folding state with minimal free-energy. According to (PEDERSEN, 2000), a computational model that obeys this principle must have the following features:

- a model of the protein, defined by a set of entities representing atoms and connections among them;
- a set of rules defining the possible conformations of the protein;
- a computationally feasible function for evaluating the free-energy of each possible conformation.

Whereas the protein structure prediction problem (PSP) is widely acknowledged as an open problem, the protein folding problem (PFP) has received little attention. It is important to note that the ability to predict the folding pathways can improve methods for predicting the native structure of proteins.

The total number of possible conformations of a real protein is huge and it would take an astronomical length of time to find the native conformation by means of exhaustive search of all conformational space (KARPLUS, 1997). Currently, it is known that the folding process does not include mandatory steps between unfolded and folded states, but a search of many accessible conformations (KARPLUS, 1997).

One approach to enumerate folding pathways is to start with an unfolded protein and consider the various possibilites for the protein to fold. The protein folds from a denatured conformation with a high free energy to its native conformation, following an energy landscape (GRUEBELE, 2002).

2.2.1 THE THERMODYNAMIC HYPOTHESIS OF PROTEIN FOLDING

As commented before, it is believed that the native conformation of a protein obeys the Thermodynamic Hypothesis, which was proposed by Anfinsen (1973) [2]. This theory is a postulate in Molecular Biology and it is based on the observation that the native conformation is determined by the amino acid sequence of the protein and that, at physiological conditions, the protein folding process occurs spontaneously, where the protein tends to an stable and kinetically accessible three-dimensional structure with minimal free-energy, (PEDERSEN, 2000). This structure is also known as the native conformation, and it is thermodynamically stable in the surrounding environment where the protein is. This means that the protein stays with the same shape during an indeterminate time in the absence of further external disturbance.

Following this theory, several discrete models (further explored in Section 2.4) have been proposed to determine the native conformation of a protein from its primary structure (amino acid sequence). This is accomplished from the thermodynamic point of view, simulating the conformational space of a protein using a free-energy function also known as the Gibbs free-energy (BOERIO-GOATES, 2000). The Gibbs free-energy (also known as the *free enthalpy*), is one of the most important concepts of thermodynamics, and defines the spontaneity of a chemical reaction according to the first and second laws of thermodynamics (NELSON; COX, 2008). Equation 1 shows the Gibbs free-energy function (ΔG) defined by entalphy (ΔH), entropy (ΔS) and temperature (T). Entalphy represents the energy of a system, including the internal energy given by the reactions between particles of the system. On the other hand, Entropy is an expression of randomness or disorder of the system.

$$\Delta G = \Delta H - T\Delta S \tag{1}$$

In the context of protein structure, the variation of the Gibbs free-energy discriminates between folded and unfolded proteins (NELSON; COX, 2008), where unfol-

[2] Nobel Prize Laureate in 1972

ded proteins are characterized by a high level of entropy. This entropy and hydrogen-bond interactions between amino acids and the solvent (water) tend to maintain the unfolded state. It is also known that the chemical reactions, which stabilize the native conformation of a protein, include hydrogen bonds, Van Der Waals interactions and hydrophobic interactions (KAUZMANN, 1959; DILL, 1990). In addition, it is also known that the hydrophobic interactions have more influence on the protein folding process than the other kind of interactions, contributing to the formation of a hydrophobic core inside the proteins (DILL, 1999; NELSON; COX, 2008).

2.2.2 THE LEVINTHAL PARADOX

Levinthal (1968) proposed an objection concerning the idea that the search of the native conformation can be done using a random search approach, which, in the worst case becomes an exhaustive search, including all possible conformations of a given protein.

The main argument of Levinthal was based on thought experiments (*gedanken experiments*), where the complexity and size of the conformational (or search) space can be estimated from the time to find the native conformation of a protein. The time to find the native state is given by the number of possible configurations of the protein structure multiplied by the time required to find one configuration. For example, a number of 10^{70} possible configurations (for a 100-residue protein) and 10^{-11}s to find one configuration lead to an astronomical folding time (about 10^{52} years), which is larger than the age of the Universe (about 1.4×10^{10} years). However, it is known that proteins fold spontaneously in times ranging from milliseconds to seconds (i.e.: small globular proteins often fold very quickly, in tens of μs (PANDE et al., 1998; ENGLANDER, 2000)), and this represents the Levinthal's paradox (LEVINTHAL, 1968; KARPLUS, 1997).

2.2.3 FUNNELS, THE ENERGY LANDSCAPE AND FOLDING PATHWAYS

The term "folding funnel" was introduced by (LEOPOLD; MONTAL; ONUCHIC, 1992) and consists of a a kinetic mechanism for understanding the self-organizing principle of the sequence-structure relationship, in the framework of a conceptually novel suggestion that some native structures may be kinetically accessible while others may be not.

They studied two 27-residue sequences: one that folded into a special structure and a random sequence. The first one was able to fold in 500,000 Monte-Carlo (MC) iterations while the second one was not. (LEOPOLD; MONTAL; ONUCHIC, 1992) explained this difference by the lack of kinetic accessibility for the second protein. Similarly, several other researchers view folding funnel as a kinetic concept (SHAKHNOVICH, 2006). For instance, Ozkan, Dill and Bahar (2002) studied a simple two-dimensional lattice Gō model (UEDA; TAKETOMI; Gō, 1978) which, in turn, was proposed for the simulation of the protein folding by means of simple non-bonded interactions between amino acids. They concluded that the folding process in that model is fast, multichannel and funnel-like in the sense that conformations start by higher energy conformations and converge into lower energy ones.

It has been argued that new experimental observations of the existence of "hidden intermediates" (HIs) are not consistent with the funnel model (ENGLANDER, 2000; RUMBLEY et al., 2001), which represents a relaxation process. (OZKAN; DILL; BAHAR, 2002) observed the existence of multiple transition states, which are the barriers between the HIs. They also state that denaturants and temperature affect the time of appearance of HIs. The main conclusion of their work is the demonstration that increasingly structured nonnative states can contribute to two-state protein folding kinetics, even when not occurring along a single sequential pathway.

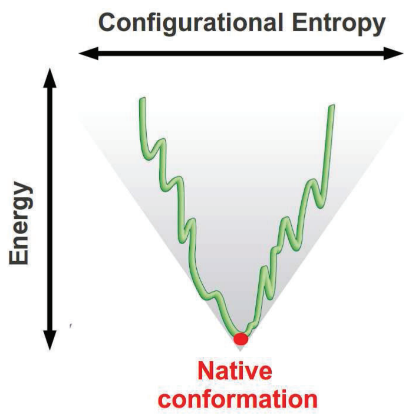
The folding funnel diagram, which was introduced by (WOLYNES; ONUCHIC; THIRUMALAI, 1995), provides a pictorial representation of how the Levinthal's paradox is resolved. Figure 4(a) shows a schematic representation of the funnel diagram, where the energy is plotted vertically and the configurational entropy horizontally. The increase in configuration entropy, which gives the diagram its funnel-like shape, aids the protein in finding the native state. That is, the decrease in the number of available configurations, as the native state approaches, tends to slow the folding process. The difficulty in finding these configurations is essentially the origin of the Levinthal paradox. To understand the folding process it must be realized that the major determinant of the folding rate is the free-energy surface (also known as energy landscape) of the protein, rather than the energy shown in the funnel diagram. The free-energy considers the potential energy, that decreases towards the native state (it represents a favorable folding contribution) and the decrease of the configurational entropy (unfavorable contribution) (KARPLUS, 2011). The balance between the potential energy and the configurational entropy leads to an energy barrier, resulting in a two-state folding behavior, which is observed for most small proteins (FERSHT, 1999).

The search for the native state goes through a huge conformational space by a folding pathway, which is characterized by several intermediate folding states and energy barriers (LEVINTHAL, 1968) between the two significantly folded states, the denatured and the native states (the energy barrier separating these two states is larger than a few kilocalories (KARPLUS, 2011)).

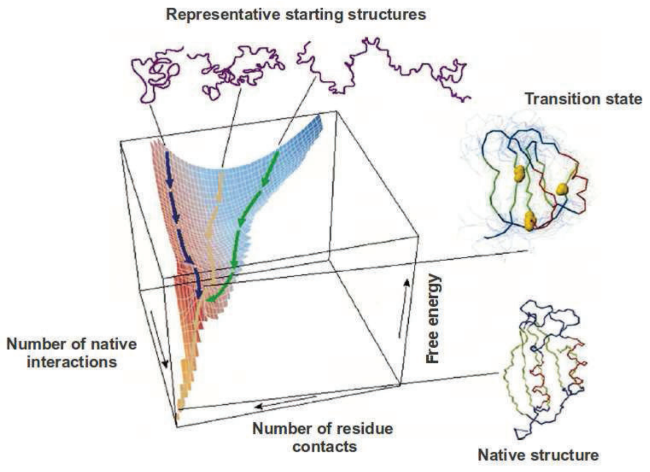
More recently, Frigori, Rizzi and Alves (2013) argue, based on a simple force field, that the absence of a free-energy barrier favors the presence of partially unfolded conformations in the hPrP heteropolymer. It could explain why it is an aggregation-prone heteropolymer.

The conceptual basis of the energy landscape is shown in Figure 4(b), where it incorporates the accepted assumption that the native state of a protein has the lowest free-energy. In this figure, the surface is derived from a computer simulation of the folding of a highly simplified model of a small protein (DINNER et al., 2000). The surface shows a funnel-like behavior, where denatured conformations converge to the unique native structure. The critical region on the energy landscape is the saddle point (or local minima) corresponding to the transition state, that is the barrier that all proteins must cross in order to fold to the native state (DOBSON, 2004). The transition state can be defined as an ensemble of partially folded conformations (also known as the Transition State Ensemble – TSE) with equal probability ($p_{fold} = 50\%$) to fold or unfold. According to this definition, a trajectory (or pathway) that passes through a transition-state conformation has the same probability to proceed to the native state or to an unfolded state (WEIKL, 2010; SNOW; RHEE; PANDE, 2006). A conformation corresponding to the transition state is indicated in the Figure. The native structure is shown at the bottom of the surface, while at the top unfolded conformations are represented. (VENDRUSCOLO et al., 2001) presents an approach for the determination of the structure of the transition state from experimental data. They suggest that the native structure is almost defined in the transition state despite the structures are disordered, and that it will almost invariably be generated during the final stages of the folding process. Their approach also incorporates experimental data to the energy function in order to transform the landscape. It creates a minimum corresponding to the state observed experimentally and, consequently, allows an efficient conformational space sampling (see Figure 4(c)).

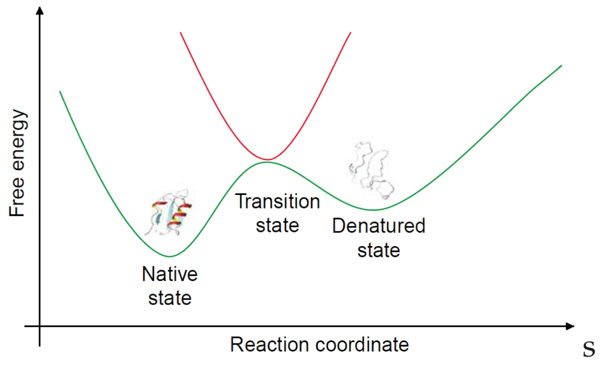
It is also known that fluctuations in the conformation of a polypeptide allow the formation of contacts between residues (DOBSON, 2004).



(a)


Representative starting structures
Transition state
Free energy
Number of native interactions
Number of residue contacts
Native structure

(b)



(c)

Figure 4: (a) An example of a folding funnel diagram, (b) A schematic energy landscape for protein folding and (c) Schematic representation of the transformation of the energy landscape used to determine the transition state. Green and red lines represent the energy landscape and the transformed landscape, respectively.

Source: (a) Adapted from (KARPLUS, 2011), (b) adapted from (DOBSON, 2004), (c) adapted from (VENDRUSCOLO et al., 2001).

These paradigms have guided the interpretation of experimental observations in the last decades. However, several important questions are still unsolved (SOSNICK; BARRICK, 2011; RUMBLEY et al., 2001):

- how is it possible for the amino acid sequence to code the final native conformation?
- what role is played by the residual structure in the unfolded state?
- how many distinct pathways are present, and how different are they from one another?
- is there a free-energy barrier for folding, and what is its magnitude?

2.3 PROTEIN DATABASES

Finding protein functions has been, since long time ago, an important topic in the Bioinformatics community. As mentioned in Section 2.1.2, the function of a protein is directly related to its structure. Due to its great importance for Medicine and Biochemistry, many research has been done about proteins (including the many genome sequencing projects) and, consequently, many information is available. There are many resources related to protein structure and function. Table 3 lists some of the main protein databases. Basically, the protein databases can be classified into two classes: sequence and structure databases.

In this work we used the Protein Data Bank (PDB) (BERNSTEIN et al., 1977; BERMAN et al., 2000) that is an international repository of three-dimensional structure of biological macromolecules. The data is, typically, obtained by X-ray crystallography (SUNDE; BLAKE, 1997; DRENTH, 1999) or NMR spectroscopy (Nuclear Magnetic Resonance) (WÜTHRICH, 1986; JARONIEC et al., 2004).

2.4 COMPUTATIONAL MODELS OF PROTEINS

Basically, there are two main classes of representation of protein structures: analytical (also known as all-atom) and coarse-grained models, with implicit and explicit representations of the solvent. Analytical models have a detailed description of the three-dimensional structure of a protein, including information about all its individual atoms. On the other hand, coarse-grained models describe protein structures in a very reduced level of details. Coarse-grained models have recently gained renewed interest due to two main factors: the simulation of all-atom models is not always computationally feasible, and coarse-grained models allow biologically relevant simulations with enhanced computational power. The main challenge in modelling coarse-grained models is to reduce complexity while maintaining biological details (MIRNY; SHAKHNOVICH, 2001; ISTRAIL; LAM, 2009). Although simulations with coarse-grained models still cannot be considered as predictive as all-atom simulations, noticeable advances have been achieved, concerning the use of more rigorous methodologies for parameterization and novel algorithms for sampling the configurational space (TOZZINI, 2005). Analytical and coarse-grained models will be presented in next sections.

Table 3: Some important protein databases.

Database	Description	Web Address
PDB	repository of protein structures	http://www.pdb.org
UniProtKB	repository of protein amino acid sequences, name/description, taxonomic data and citation information /TrEMBL	http://www.uniprot.org/
PIR	protein sequence databases	http://pir.georgetown.edu/
PROSITE	documentation entries describing sequence motif definitions, protein domains, families and functional patterns	http://www.expasy.org/prosite/
PRINTS	Fingerprints information on protein sequences	http://www.bioinf.man.ac.uk/dbbrowser/
BLOCKS	Multiple-alignment blocks	http://blocks.fhcrc.org/
eMOTIF	protein motif database, derived from PRINT and BLOCKS	http://motif.stanford.edu/distributions/
PRODOM	protein domain databases	http://prodom.prabi.fr/
InterPro	protein families and domains	http://www.ebi.ac.uk/interpro/

Source: Own work. The URLs were accessed in may, 2015.

2.4.1 ANALYTICAL MODEL

The analytical model was presented by (RICHARDSON, 1981) and (NGO; MARKS; KARPLUS, 1994). In this model, proteins are considered as a collection of atoms connected each other by bonds. Therefore, the three-dimensional structure of a protein can be specified by angles, lenghts and torsion of each bond among atoms in the structure. As a consequence, the complexity of this representation is very high, and its simulation requires significant computational efforts, despite the great advances in recent years.

In this model, the free-energy function considers bonded and non-bonded atoms contributions. Terms related to bond lenghts, angles and torsions for bonded atoms, and physical principles are used for non-bonded atoms (for example, Coulomb and Van der Waals forces) or statistical informations inferred from known structures.

Computationally, the simulation with atomistic models is *NP*-hard (NGO; MARKS; KARPLUS, 1994).

Best (2012) presents a review about the recent progress in atomistic molecular simulations of protein folding and highlights future challenges in this area.

2.4.2 COARSE-GRAINED MODELS

Consequently, researchers have been developed several coarse-grained (CG) models for representing protein structures. A possible approach to obtain simplified models is to reduce the level of details through the integration of a huge number of degrees of freedom into a few, that is, coarse graining. Models should be rather simple to allow computational feasibility and not too simple to miss important biological aspects of the folding process. In the last decades, different coarse-grained models have been proposed, using discretized and continuous space representation of the conformational space.

2.4.2.1 DISCRETE MODELS

The simplest computational models for the PFP are known as the *lattice models*. In these models, the protein structure is represented considering solely a sequence of beads, which represent the amino acids, positioned in a lattice. The connections between amino acids are restricted by the lattice structure, that usually is two-dimensional (plane) or three-dimensional (space). In a valid conformation, a given position in the lattice can be occupied by, at most, one amino acid, and adjacent amino acids can occupy adjacent positions in the lattice. Several *lattice models* have been developed and applied to the PFP. For instance, the simplest lattice model for the PFP is known as Hydrophobic-Polar (HP) model, both in two (2D-HP) and three (3D-HP) dimensions (DILL et al., 1995). Figures 5(a) and 5(b) show the 2D-HP and 3D-HP models. In the HP model, the 20 amino acids are classified into two types: Hydrophilic (or

Polar) and Hydrophobic. Therefore, a protein is represented by a string of characters defined over a binary alphabet $\{H, P\}$, and it is considered that interactions between hydrophobic amino acids represent the most important contribution to the free-energy of the protein. The more hydrophobic interactions, the small the free-energy of the protein.

Although simple, the computational approach for searching a solution for the PFP using HP models was proved to be *NP*-complete (ATKINS; HART, 1999; BERGER; LEIGHTON, 1998; CRESCENZI et al., 1998).

Other simple lattice models were employed in the modeling of protein structures. For instance, Diamond, Face-Centered-Cubic (FCC) and Body-Centered-Cubic (BCC) show a better description of the local geometry of real proteins than simple lattice models (RAGHUNATHAN; JERNIGAN, 1997). These models allow the representation of α-helices and β-sheets with a reasonable similarity to real proteins (KOLINSKI; SKOLNICK, 2004). The FCC model (see Figure 5(d)) describes the local packing in folded proteins (RAGHUNATHAN; JERNIGAN, 1997) and can produce high resolution backbone models (PARK; LEVITT, 1995). It can be considered as the best choice among the lattice models and have been used in many protein structure studies (ULLAH et al., 2009). Shakhnovich and Gutin (1993) presented the *Perturbed Homopolymer* (PH), which is a HP-based model, where interactions among residues of same type (i.e. H-H or P-P) are taken into account.

Unger and Moult (1993a) proposed the *Lattice Polymer Embedding* (LPE). In this model, protein structures are embedded in a cubic lattice, each residue interaction (i.e. between s_i and s_j amino acids) has an affinity coefficient ($c(s_i, s_j)$), and the energy function to be minimized is the sum of the product of the affinity coefficient by the distance between amino acids. The simulation using this model is also a *NP*-complete problem (UNGER; MOULT, 1993a).

Ngo, Marks and Karplus (1994) presented the Charge Graph Embedding model (CGE). In this model, a charge is assigned to each amino acid ($C(s_i) \in \{-1, 0, 1\}$) and the influence of a given amino acid on another one is not taken into account when the distance between them is lower than a *cutoff* value.

Bornberg and Bauer (1997) proposed the HPNX model in order to address the degeneracy problem (i.e. different possible conformations with the same energy) presented by HP models. (BACKOFEN; WILL; BORNBERG-BAUER, 1999) showed that the HPNX model can reduce the degeneracy problem and lead to a more effective protein folding simulation. The HPNX model classifies the amino acids into four classes: hydrophobic (H), positive (P), negative (N) and Neutral (X). This model is also based on the HP model and considers hydrophobic interactions as attractive forces, decreasing the free-energy value by -4.0, and interactions between positive (P-P) and negative (N-N) amino acids as repulsive interactions, increasing the free-energy by 1.0. Hoque, Chetty and Sattar (2009) proposed an improved version of the HPNX model, which is called hHPNX model, by extracting important features from the YhHX model (BORNBERG-BAUER, 1997). Modifications in the HP model with different codes for the amino acid sequence were proposed by (LOCKER; HERNANDEZ, 2001; KAYA; CHAN, 2002; SALI; SHAKHNOVICH; KARPLUS, 1994).

The side chain packing is probably the main factor for the native structure formation (KOLINSKI; SKOLNICK, 2004). The lattice models presented previously do not consider the side chain packing. However, several lattice models that include side chains have been proposed. For instance, Li, Klimov and Thirumalai (2002) present a more realistic model that includes the side-chain (SC) of the amino acids in the HP models, that is known as the 2D-HP-SC and 3D-HP-SC (see Figure 5(c)) model for the two and three-dimensional versions, respectively. Recall that all the standard amino acids have the same basic structure (backbone), but different side-chains define their physico-chemical properties. Therefore, a protein is modeled by a common backbone and a side-chain, either Hydrophobic (H) or Polar (P). Differently from the 2D-HP and 3D-HP, the 3D-HP-SC model is very sparsely studied in the recent literature. For instance, Benítez and Lopes (2010) describes a Hierarchical Parallel Genetic Algorithm (HPGA) applied to the PFP, using the 3D-HP-SC. Heun (2003) proposed an extended HP side chain model, extending the cubic lattice by diagonals in the plane in order to eliminate the drawback imposed by the bipartiteness of the cubic lattice. Besides the lattice models with side chains, the *side chain only* model (SICHO) (KOLINSKI; SKOLNICK, 1998; NALS; KOLINSKI; SKOLNICK, 2002) uses an extremely simple representation of strings and beads chains connecting the centers of mass of the side group of the residues in their rotational isomeric conformations (see Figure 5(f)). The side-chain centers are calculated by averaging the positions of all heavy side-chain atoms including the Cα atom (for glycine the virtual particle is positioned at the position of the Cα atom). This model allows a large number of possible virtual bonds between side groups. In addition, the position of each Cα is defined by an approximation obtained from the positions of the three consecutive side groups. Here, it is important to recall that the aproximation of the Cα position is actually more accurate than the accuracy of the side chain positions (KOLINSKI; SKOLNICK, 2004; FEIG et al., 2000).

Moderate resolution lattice models have been developed. For instance, the three-dimensional "chess-knight" (also known as the *210 lattice* model) represents the Cα trace by a chain of cyclic permutation vectors (± 2, ± 1, 0). These lattice vectors are like a "knight's walk" in chess, but in three dimensions. Figure 5(e) shows the "chess-knight" model. In this model, the number of possibly virtual Cα–Cα bond orientations is equal to 24 and the side chains are restricted to a three-dimensional lattice, separated from the backbone by a diamond lattice vector (± 1, ± 1, ± 1). This model enables the low resolution study of real proteins (SKOLNICK; KOLINSKI, 1991).

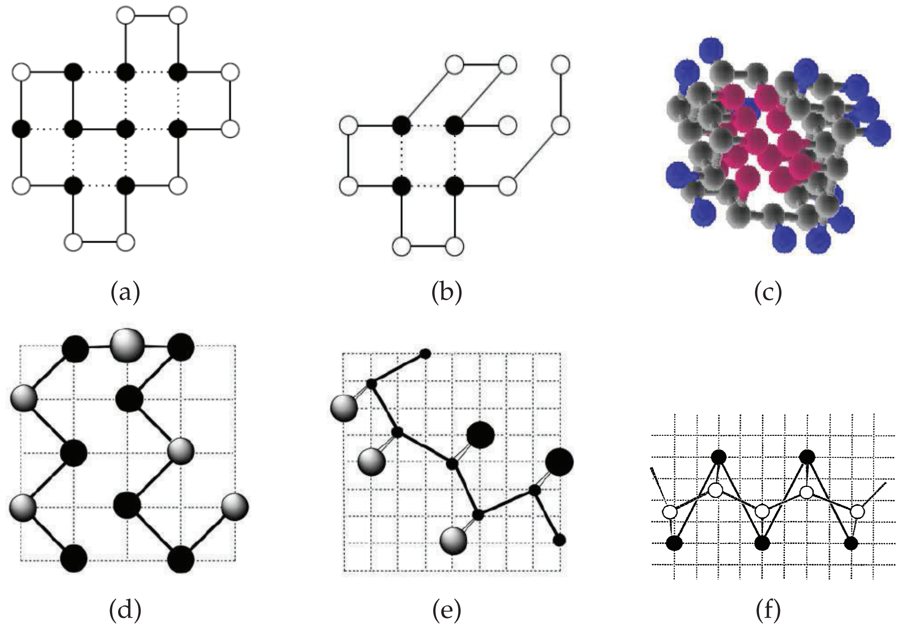


Figure 5: Example of hypothetical protein structures using lattice models: 2D-HP (a), 3D-HP (b), 3D-HP-SC (c), FCC (d), "Chess knight" (e) and SICHO (f)

Source: Adapted from (DILL et al., 1995), (LI; KLIMOV; THIRUMALAI, 2002), (KOLINSKI; SKOLNICK, 2004), (SKOLNICK; KOLINSKI, 1991) and (KOLINSKI; SKOLNICK, 1998).

2.4.2.2 CONTINUOUS MODELS

Besides lattice models, several *off-lattice* models have been proposed with various levels of generalization. *Off-lattice* models are also known as *bead models* and they are based on a *united-atom* representation of the amino acid molecule, involving one to six sites or interacting centers. In general, the smaller the number of beads representing an amino acid molecule, the more feasible the computational simulations. In other words, the computational cost of a simulation with bead models is proportional to the number of beads per amino acid. On the other hand, the parameterization of the force field that defines the energy equation is more complex for coarser models because more specific interactions must be effectively included using few parameters. The models can be classified into four main classes according to the number of sites for each amino acid, from the coarsest (*one-bead* models) and the finest (*four and six beads*) classes:

- **One-bead models**: most of the currently one-bead models (see Figure 6(a)) are an evolution of Gō models (UEDA; TAKETOMI; Gō, 1978), that includes more sophisticated potentials in the energy equation. Hills and Brooks (2009) review the many variations of the Gō model that have been used to yield insights into folding mechanisms. They discuss the ability of Gō models to capture sequence effects in folding and conformational transitions. According to (BAKER, 2000; KOGA; TAKADA, 2001), Gō models are consistent because they represent a perfectly funneled model for protein folding that correctly includes the topology of the native state. In these models, each amino acid is represented by one bead positioned at the Cα position. For instance, Brown, Fawzi and Head-Gordon (2003) present a one-bead model that requires *a priori* knowledge of a reference secondary structure for the parameterization of the angle and dihedral terms. This model also includes a sequence specificity in non-bonded terms (that is, terms that describe hydrophobicity and hydrophilicity).

  Tozzini, Rocchia and McCammon (2006) present a one-bead model for proteins that accounts for transitions between α-helix to β-sheet secondary structures. In the one and two beads coarse-grained models, the two conformational dihedrals ϕ and ψ that describe the backbone geometry are no longer present as explicit internal coordinates, thus the information contained in the Ramachandran plot cannot be used directly (TOZZINI; ROCCHIA; McCammon, 2006). Those authors also derive an analytical mapping of the all-atom internal backbone coordinates (i.e. conformational dihedrals ϕ and ψ) to the coarse-grained model internal backbone coordinates (θ and α) that allow to map the Ramachandran plot onto a new conformational density plot. They also proposed a new force field with an accurate bond angle interaction based on the backbone coordinates.

  Yap, Fawzi and Head-Gordon (2008) developed a one-bead model that uses an Cα trace to represent the backbone inspired in a previous model proposed by (HONEYCUTT; THIRUMALAI, 1992). This model includes a potential force for hydrogen bonding for secondary structures formation.

Wolff, Vendruscolo and Porto (2011) proposed a one-bead model which represents the protein chain as a tube with a finite thickness (with diameter of 3.3Å and bond length of 3.8Å between amino acids) in order to describe the self-avoidance effects of the chain. This model uses a two-term energy function, one term for effective-connectivity (EC) and other term based on Gō model potential that can be defined using a contact map of a given native conformation (see Section 2.5 for further information).

Two simple models are the HP-TSSC (Hydrophobic-Polar Tangent Spheres Side Chain Model) and the AB *off-lattice* models, which were proposed by (HART; ISTRAIL, 1997) and (STILLINGER; HEAD-GORDON, 1995), respectively. The HP-TSSC is based on HP models, but it does not embed the amino acids in a lattice. In this model, a protein is represented by a graph that is transformed in a set of tangent spheres with equal radius, that represent the backbone and side chains.

In this work, the AB *off-lattice* model is used to represent protein structures and it is detailed in Section 2.4.2.3.

- **Two-three beads models**: as shown in Figure 6(b), two-bead models represent each amino acid with two beads, where one bead is positioned at the Cα position and additional beads are used for representing the side chain). For instance, a physics-based united-residue force field (UNRES) model is presented by (LIWO; KHALILI; SCHERAGA, 2005). In this model, a polypeptide chain is represented as a sequence of Cα atoms, where the Cα atoms are linked together by means of virtual bonds (designated as dC). United side chains (SCs) are connected to the backbone by virtual bonds (designated as dX). Peptide groups (p) are positioned in the centers of the dCs and the center of mass of the side chains are at the ends of the dXs (See Figure 6(d)).

  Combe and Frenkel (2007) proposed a two-bead model, which considers interactions between side chains and hydrogen bonds. This model does not describe the solvent molecules. To account hydrogen bonds, this model presents a plane modeled as a *spin* that can rotate perpendicularly to the Cα-Cα bond. Thus, hydrogen bonds are taken into account through the interaction between spins. Side chains are represented by three different types of spheres: hydrophobic (H), hydrophilic positive (P) and hydrophilic negative (N). They used this model for studying the aggregation of proteins in prion diseases and the folding pathway through a free-energy analysis.

Bahar and Jernigan (1997) proposed a two-bead model with a parameterized force field based on a statistical analysis of a set of experimental structures (REITH; PUTZ; MULLER-PLATHE, 2003). This model is independent of a reference structure, but the energy terms are complex. An updated version of this approach with a simpler force field had been proposed by (MUKHERJEE; BAGCHI, 2004), which has an addicional term for α-helix propensity.

Gu et al. (2009) proposed an improved Cα-SC energy potential designed for protein fold recognition with three main interactions: inter-residue contacts, hydrophobicity and pseudo-dihedral potentials.

Zacharias (2003) proposed a three-bead model with one additional bead for larger side chains. They used this model for a protein-protein docking that allows side chain flexibility.

Capriles, Custódio and Dardenne (2010) proposed a two-beads model that is defined by torsion angles (dihedral angles, ϕ and ψ) based on (MAUPETIT; TUFFERY; DERREUMAUX, 2007), where each side chain is placed at the geometrical center of the side chain group. The energy function of this model is calculed using the GROMOS96 force field (SCHULER; DAURA; GUNSTEREN, 2001).

Bezkorovaynaya (2011) presents the first model for oligoalanine that consists of a linear chain with two types of spherical beads, representing the Cα and the side chain, respectively (See Figure 6(e)). To parameterize the interactions, they extended a methodology previously developed by (VILLA; PETER; VAN DER VEGT., 2009). In this model, two different types of water representations are considered for the coarse-grained system: an implicit water representation, where the effect of the water molecules on the peptide is accounted through side chain interactions, and an explicit water model, where each water molecule is represented by a coarse-grained bead (See Figure 6(f)).

- **Four-six-bead models**: four to six beads models represent explicitly the backbone atoms (i.e. Cα, N, and C) with additional "beads" for the side chain, the oxygen (O) of the carboxyl group and the hydrogen (H) of the backbone (see Figure 6(c)). These models use the conformational dihedral angles (ϕ and ψ), that are used to plot the Ramachandran plot, formed by the backbone atoms C-N-Cα-C and N-Cα-C-N respectively. As explained before, in the one and two beads models only the Cαs of the backbone, including beads for the side chain in two beads models, are represented explicitly.

Bereau and Deserno (2009) proposed a four-bead model with implicit solvent, where each amino acid is represented by three or four beads. These beads represent the amine group, Cα, the carboxyl group and a side chain. This model also uses geometric parameters taken from a previous models (DING et al., 2003). The energy function of this model incorporates bonded and non-bonded interactions, such as hydrogen bonds, hydrophobicity (side chain) and dipolar interactions in order to study thermodynamics and kinetics of a three-helix bundle. The parameters are tuned to reproduce probability distribution functions (PDF) of the dihedral angles, which are an extension of the Ramachandran plot, and experiments were done using a Monte Carlo (MC) approach.

Maupetit, Tuffery and Derreumaux (2007) proposed a six-beads model, which consists of a detailed representation of the backbone, modelled by all their atoms (i.e. N, H, Cα, C, O), and one bead for side-chains (except the proline which is represented by all atoms). This model uses the OPEP (Optimized Potential for Efficient structure Prediction) energy function, that includes 4-body hydrogen bonding terms, local potentials related to bond angle and torsional forces, and short-range and long-range non-bonded potentials. The force constants associated with the backbone atoms are taken from AMBER (Assisted Model Building with Energy Refinement) (CASE et al., 2005).

Gopal et al. (2010) proposed a coarse-grained model called PRIMO/PRIMONA (Protein Intermediate Model and Protein/nucleic-acid version). In this model, the simulation is performed with explicit water and the backbone is represented with N, Cα and a combined carbonyl site placed at the geometric center of the carbonyl C and O atoms. This is done to preserve the hydrogen bond interactions, which are essential to describe secondary structures. Non-glycine side chains are represented with one to five sites. Amino acids Ala, Cys, Pro, Ser and Val have only one bead; amino acids Ile, Leu and Thr are modeled with two beads; amino acids Asn, Asp, Gln, His, Met and Phe have three beads; amino acids Lys, Trp and Tyr have four beads; and amino acid Arg has five beads. This model uses the CHARMM22 force field (MACKERELL et al., 1998) with the CMAP correction (MACKERELL; FEIG; BROOKS, 2004).

Tian et al. (2011) developed a knowledge-based scoring function, called NCACO-score that integrates structural information to model protein structure from sequence. They consider five-beads coarse-grained model, where each amino acid is represented by four main chain atoms (N, Cα, C and O) and a pseudo side chain (except for Glycine). The pseudo side chain center of mass (SCM) was determined from backbone torsion angles (ϕ and ψ). The NCACO-score considers four energy terms: a pairwise atom-atom contact potential, a sequence-dependent local potential, a solvation potential and a β-sheet geometry propensity potential. For performance assessment, they use two quality measures: the rank of native structure relative to decoy structures based on the energy (Rn) and the Z-score, which is a quantitative measure of energy bias for the native structure against decoy structures. They demonstrate that NCACO-score can guide fragment assembly for protein structure prediction.

A different coarse-graining method is the so-called *Elastic Network Model* (ENM) in which the protein is represented by a network of beads connected by elastic springs (TIRION, 1996; ATILGAN et al., 2001; KUNDU; SORENSEN; PHILLIPS, 2004; WU et al., 2003; BURIONI et al., 2004; ZHANG; SHI; LIU, 2003; MICHELETTI; CARLONI; MARITAN, 2004). Micheletti, Carloni and Maritan (2004) proposed a two-bead ENM, that includes a bead for side-chain representation. This model presents the same computational complexity of a one-bead ENM and is proven to give a good accuracy. Williams and Toon (2010) proposed an extension of the ENM, called DNM (Damped Network Model), for describing the folding pathways, where both bonded and nondbonded interactions are modeled with a quadratic potential that extends over a finite radius of influence with the oscillations being damped by a friction term.

2.4.2.3 THE AB OFF-LATTICE MODEL

The AB *off-lattice* model was introduced by (STILLINGER; HEAD-GORDON, 1995) to represent protein structures. In this model each residue is represented by a single interaction site located at the Cα position. These sites are linked by rigid unit-length bonds ($\hat{b}_i$) to form the protein structure. The three-dimensional structure of a N-length protein is specified by the $N-1$ bond vectors $\hat{b}_i$, $N-2$ bond angles τ_i and $N-3$ torsional angles α_i, as shown in Figure 7.

In this model, the 20 proteinogenic amino acids are classified into two classes, according to their affinity to water (hydrophobicity): 'A' (hydrophobic) and 'B' (hydrophilic or polar). This model does not describe the solvent molecules. However, solvent effects such as the formation of the hydrophobic core are taken into account

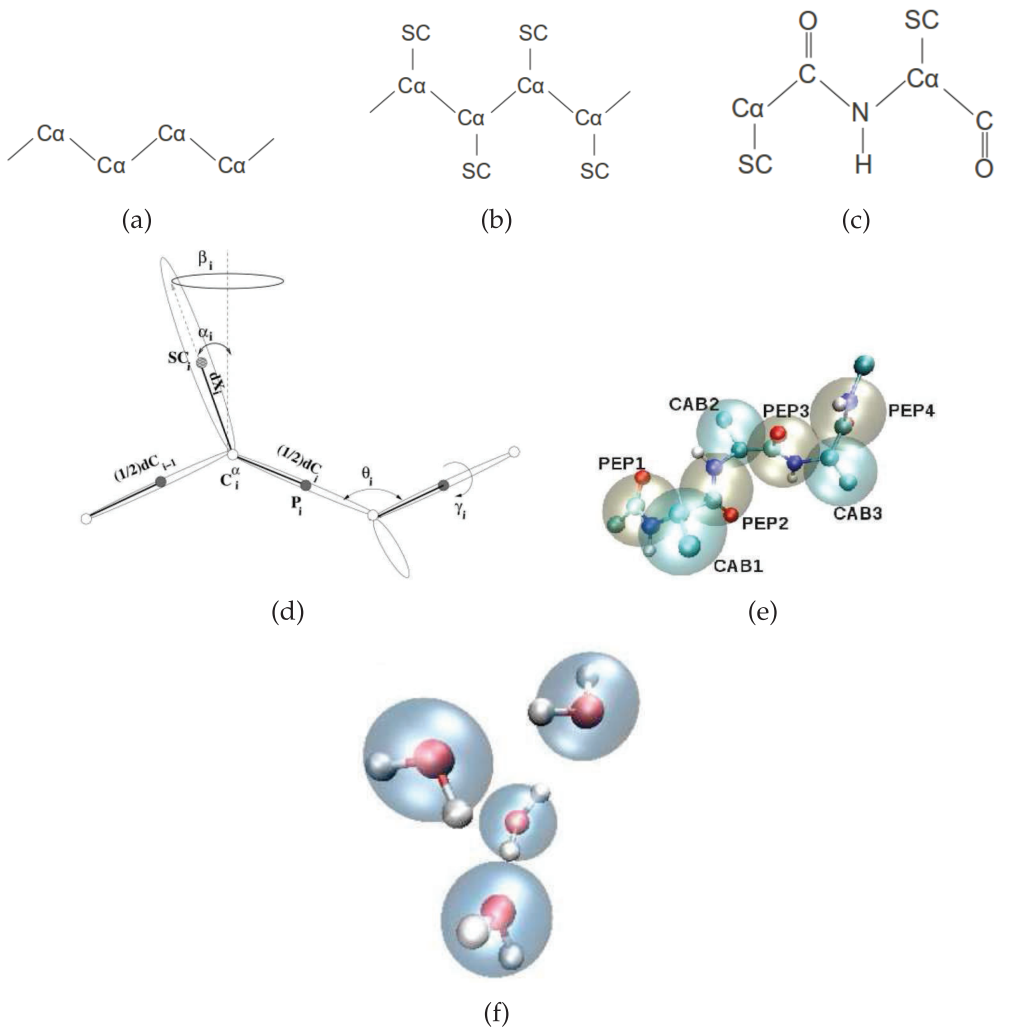


Figure 6: Examples of continuous models of protein

Source: (a, b and c) Own work. (d) Adapted from (LIWO; KHALILI; SCHERAGA, 2005). (e and f) Adapted from (BEZKOROVAYNAYA, 2011).

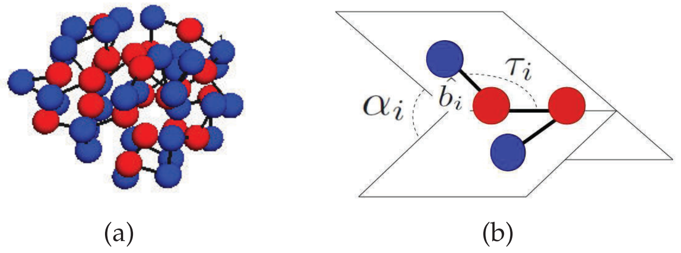


Figure 7: The AB *off-lattice* model: (a) Example of a hypothetic protein structure and (b) definition of $\hat{b}_i$, τ_i and α_i. Blue balls represent the polar residues and Red ones represent the hydrophobic residues. The backbone and the connections between elements are shown in black lines.

Source: Adapted from (IRBACK et al., 1997)

through interactions between residues, according to their hydrophobicity (species-dependent global interactions).

When a protein is folded into its native conformation, the hydrophobic amino acids tend to pack inside the protein, in such a way to get protected from the solvent by an aggregation of polar amino acids that are positioned outwards. Interactions between amino acids take place and the energy of the conformation tends to decrease. The conformation tends to converge to its native state, in accordance with the Anfinsen's thermodynamic hypothesis (ANFINSEN, 1973).

The energy function of a folding is given by (IRBACK et al., 1997):

$$E(\hat{b_i};\sigma) = E_{Angles} + E_{torsion} + E_{LJ} \tag{2}$$

where
E_{Angles} and $E_{torsion}$ are the energies from bond angles and torsional forces, respectively; and are given by Equations 3 and 4.

$$E_{Angles} = -k_1 \sum_{i=1}^{N-2} \hat{b_i} \cdot \hat{b_{i+1}} \tag{3}$$

$$E_{torsion} = -k_2 \sum_{i=1}^{N-3} \hat{b_i} \cdot \hat{b_{i+2}} \tag{4}$$

where
$\hat{b_i}$ represents the ith bond that joins the $(i-1)$th and the ith residues, and it is represented by the vector $\hat{b_i}$, as shown in Equation 5 and $k_1 = -1$; $k_2 = +1/2$ (IRBACK et al., 1997).

$$\hat{b_i} = \vec{r}_i - \vec{r}_{i-1} \tag{5}$$

The species-dependent global interactions are given by the Lennard-Jones potencial (E_{LJ}); for pairs of ith and jth residues separated by a distance of r_{ij}, as shown in Equation 6.

$$E_{LJ} = \sum_{i=1}^{N-2} \sum_{j=i+2}^{N} 4\varepsilon(\sigma_i,\sigma_j)(r_{ij}^{-12} - r_{ij}^{-6}) \tag{6}$$

where
r_{ij} represents the distance between ith and jth residues; σ = σ_0, ..., σ_N form a binary string that represents the protein sequence.
$\varepsilon(\sigma_i, \sigma_j)$ is chosen to favor the formation of the hydrophobic core ('A' residues). Thus, $\varepsilon(\sigma_i, \sigma_j)$ is 1 for AA interactions and 1/2 for BB/AB interactions, as shown in Equation 7.

$$\varepsilon(\sigma_i, \sigma_j) = \begin{cases} 1 & \text{if AA interaction} \\ \frac{1}{2} & \text{if BB or AB interaction} \end{cases} \tag{7}$$

Finally, it is important to mention that the model can be explored for different values of k_1 and k_2 as stated by (IRBACK et al., 1997).

2.4.3 RECONSTRUCTION OF PROTEIN STRUCTURES

The applicability of a coarse-grained model can be evaluated by comparing the obtained structures with real protein structures (i.e. protein structures extracted from PDB), where a structural transformation procedure has to be done. Most coarse-grained models do not describe how to return to all-atom model structure (MAUPETIT; P.DERREUMAUX; TUFFÉRY, 2009). In order to enable the transformation of a real protein structure into a coarse-grained model and *vice versa*, a representative model must be obtained. For instance, Manuch and Gaur (2006) have shown that this problem for the 3D-cubic lattice model is *NP*-complete and named it the *Protein Chain Lattice Fitting* (PCLF). Reva et al. (1995) presented an approach to solve the PCLF problem including side chains. They developed a dynamic programming approach to find an optimal solution according to an error function. Mann et al. (2012) introduce a new tool (called *LatFit*) to produce high-accuracy lattice protein models from PDB files on three commonly used lattices, previously mentioned in this section: 3D cubic, face-centered and knight's walk. To the best of our knowledge, *LatFit* is the first study of lattice quality and the only available tool that enables a high resolution fitting (available via web interface and as stand-alone tool) [3] for lattice protein models.

In addition, Rotkiewicz and Skolnick (2008) present a fast and robust method for the reconstruction of full-atom protein models starting from a reduced protein structure representation called PULCHRA. Their method is suitable as a “bridge“ between coarse-grained models and full-atom simulations. The accuracy of the method

[3]Available in: http://cpsp.informatik.uni-freiburg.de:8080/LatFit.jsp

was tested on a set of high-resolution crystallographic structures as well as on a set of protein decoys generated by the protein structure prediction algorithm called TASSER (Threading/ASSembly/Refinement) (ZHANG; ARAKAKI; SKOLNICK, 2005).

There are other web-based programs for all-atom structure reconstruction. For instance, Maupetit, Gautier and Tufféry (2006) present an approach for backbone atoms reconstruction from one-bead models, called SABBAC (Structural Alphabet based protein Backbone Builder from Alpha Carbon trace) [4], where the side chains are positioned using the SCit approach (GAUTIER; CAMPROUX; TUFFÉRY, 2004). Krivov, Shapovalov and Dunbrack (2009) proposed a method for side chain reconstruction from four-six-beads models, called SCWRL4.

2.5 CONTACT MAPS

There are alternative representations of protein structures: Contact Maps (VENDRUSCOLO; KUSSELL; DOMANY, 1997), the Ramachandran plot (NELSON; COX, 2008) and Weighted Secondary Structure element Graph (WSG) (ZAKI et al., 2004). In this work, we use Contact Maps (CM), that are minimalistic representations of protein structures. The contact map for a protein sequence with N amino acids is a $N \times N$ binary symmetrical matrix (C), which is defined as follows: each position of the matrix (ith,jth) is 1 if the amino acid pair (ith and jth amino acids) fulfills the connectivity condition. One can define a contact between two amino acids in different ways. For instance, we can consider two amino acids in contact when their Cα atoms are closer than a threshold distance (VENDRUSCOLO; KUSSELL; DOMANY, 1997).

Figure 8 presents the contact map of a given protein structure, indicating its secondary structures.

The contact map serves not only as a representation of structure, but also as an energy fingerprint of the protein conformation (MIRNY; DOMANY, 1996). Contact maps can represent secondary structures. For instance, an α-helix contains contacts between the pairs (i, $i \pm 4$)th and (i, $i \pm 3$)th amino acids. The α-helix are parallel and near to the main diagonal of the contact map. Parallel β-sheets are formed by contacts between $(i+k)$th and $(j+k)$th amino acids, where $k = 0, 1, 2, ...$. On the other hand, antiparallel β-sheets contains contacts between $(i+k)$th and $(j-k)$th amino acids. Therefore, β-sheets appear as group of contacts that are perpendicular to the main diagonal

[4] Available in http://bioserv.rpbs.univ-paris-diderot.fr/cgi-bin/SABBAC

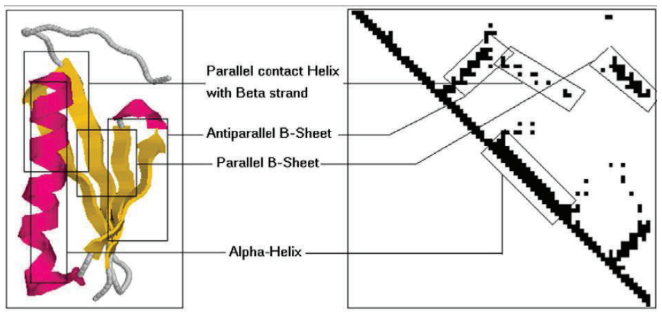


Figure 8: Contact map (right) of a given protein structure (left)
Source: Adapted from (ABU-DOLEH; AL-JARRAH; ALKHATEEB, 2012).

of the map (MIRNY; DOMANY, 1996).

Hu et al. (2002) describe how data mining can be used to extract valuable information from CMs. For instance, secondary structures can easily be discerned from CMs. For example, α-helices appear as thick bands along the main diagonal since they involve contacts between one amino acid and its four successors, while β-sheets are thin bands parallel or anti-parallel to the main diagonal, among others (HU et al., 2002).

CMs can be also considered as graphs, where the contacts are the edges and the residues are the nodes (BARTOLI et al., 2008).

CM prediction has become an alternative way to predict protein structures. Several methods have been developed for CM prediction from sequence. For instance, Neural Networks (NN) (PUNTA; ROST, 2005; G.POLLASTRI; BALDI, 2002), Hidden Markov Models (HMM) (BYSTROFF; SHAO, 2002; ZAKI; JIN; BYSTROFF, 2003) and Genetic Programming (GP) (MACCALLUM, 2004).

Pietal, Tuszynska and Bujnicki (2007) developed a tool for contact map comparison, called Protmap2D [5], that provides a sensitive measure of protein structure similarity. Vehlow et al. (2011) developed a tool called CMView [6] which integrates rich contact map analysis with 3D visualizatoin using PyMol [7]. This tool provides functions for contact map calculation from structure and visualization in contact map and 3D space and structural comparison with different built-in alignment methods.

See Section 2.10.6 for further information about contact map prediction and

[5]Protmap2D is available at http://genesilico.pl/protmap2d.htm
[6]CMView is available at http://www.bioinformatics.org/cmview
[7]PyMol is available at http://www.pymol.org

reconstruction of protein structures from contact maps.

2.6 MOLECULAR DYNAMICS

Molecular Dynamics (MD) is a computational simulation of the physical movements of particles (atoms or molecules). The theoretical basis for MD embodies many of the important results produced by the great names of analytical mechanics – Newton, Euler, Hamilton and Lagrange. The basic form of MD involves little more than Newton's second law (RAPAPORT, 2004). The idea of MD is to generate the trajectory of a system with N particles through numerically integration of the classical equations of motion.

MD is a deterministic approach, differently from Monte Carlo simulations that are stochastic (SUTMANN, 2002). Thus, a MD simulation will always generate the same trajectory from the same initial condition.

The high level of detail in MD simulations gives general physical conclusions. MD has been limited by the current computer hardware. These simulations are usually limited to biological events that occur on short timescales (typically, ηs) because the calculation of the physical forces is computationally expensive. Two solutions to overcome the computational cost are to use coarse-grained models and use faster hardware (HARDER et al., 2012) in MD simulations. For instance, (SHAW et al., 2010; LINDORFF-LARSEN et al., 2011) achieved simulations on timescales of microseconds or milliseconds using a custom built special-purpose hardware, called Anton.

Another important issue in Molecular Dynamics and Monte Carlo simulations is the design and parameterization of the force field for a given model. Ponder, Case et al. (2003) presents a review of force fields for all-atom peptide and protein modeling.

See Section 3.1 for information about the implementation of the Molecular Dynamics (DM) algorithm for the Protein Folding Problem (PFP) using the 3D-AB *off-lattice* model of proteins.

2.7 A "NEW KIND OF SCIENCE" AND CELLULAR AUTOMATA (CA)

Stephen Wolfram proposed a "New kind of Science" that is based on general types of rules that can be embodied in simple computer programs for reproducing real-world complex behaviors, instead using traditional mathematical methods (WOL-

FRAM, 2002). A particular class of such computer program are the Cellular Automata (CAs), which are simple discrete idealizations of natural systems. CAs are families of simple, finite-state machines that exhibit emergent behaviors through their interactions (LUGER, 2008).

The computational simulation of a CA system is relatively simple, where a configurational state of the CA is determined according to its predecessor state and a transition rule. However, finding a transition rule for a given dynamic behavior is a very difficult task, for which there is no efficient method (WEINERT; LOPES, 2010).

Cellular Automata (CAs) were introduced by John von Neumann in his work on *self-reproducing machines* (NEUMANN, 1966) and have been used to model several biological, physical and engineering systems. For instance, simulation of the HIV infection dynamics (MO et al., 2014), and water flow simulation (TOPA; MLOCEK, 2013).

Basically, CAs are discrete dynamic systems that are represented by a d-dimensional array, composed of identical interconnected components (cells).

The dynamic behavior of a CA is represented by its spatio-temporal diagram. Each cell has a discrete state that is updated on discrete time steps, considering its current state and the state of the neighboring cells (neighborhood relationship). All cells of the d-dimensional matrix are updated at the same time step by the application of a transition rule. Thus, sucessive applications of the transition rule will lead to a dynamic behavior, from the initial state in t_0 to successive states in subsequent time steps $(t_1, \cdots, t_n)$.

The following formal notation for CAs is presented by (MITCHELL, 1998): $\sum$ set of possible states of a cell; k number of elements of the set $\sum$; i index of a specific cell; S_i^t state of a cell in a given time t; η_i^t neighborhood of cell i; $\Phi(\eta_i^t)$ transition rule that defines the next state S_i^{t+1} for each cell i, as function of η_i^t.

The neighborhood of each cell $(c_{i,j})$ of a two-dimensional Cellular Automaton (2D-CA) is composed following a neighborhood relationship. The neighborhood relationship is determined by a predefined radius (r). The size (number of cells) of the neighborhood (m) is defined as a function of r, acoording to $m = 2r+1$. The most common types of neighborhood for 2D-CA are the von Neumann and the Moore neighborhoods. The size of the von Neumann neighborhood with $r = 1$ is $m = 5$, comprising the four orthogonally neighboring cells $(c_{i-1,j}, c_{i,j-1}, c_{i+1,j}, c_{i,j+1})$ surrounding a central cell ($c_{i,j}$). On the other hand, the Moore neighborhood is composed by the central cell and eight neighboring cells ($c_{i-1,j}$, $c_{i,j-1}$, $c_{i+1,j}$, $c_{i,j+1}$, $c_{i-1,j-1}$, $c_{i+1,j-1}$, $c_{i-1,j+1}$, $c_{i+1,j+1}$).

In addition, boundary conditions are used to allow the connection between cells that are situated at the extremities, forming a toroidal arrangement. Thus, the transition rule ($\Phi(\eta_i^t)$) is applied over all cells of the CA, without failure.

The number of transitions (possible configurations of the neighborhood) that compose the rule Φ is given by k^{2r+1} and the number of rules represented by those transitions is $k^{k^{2r+1}}$. For instance, the rule of a binary 2D-CA with von Neumann neighborhood (with $r = 1$) is composed of 32 transitions. In other words, the rule is composed of 32 bits, where each bit represents the result of a transition (i.e. $c_{i,j}^{(t)} \to c_{i,j}^{(t+1)}$), according to the concept of elementary automata proposed by Wolfram. The number of possible *transition rules* with the von Neumann *neighborhood* is $k^{k^{2r+1}} \approx 4.295 \times 10^9$. On the other hand, using the Moore *neighborhood* (with $r = 1$), there are $\approx 1.34 \times 10^{154}$ possible *transition rules*.

2.8 BIOINSPIRED COMPUTATION

The two main families of bioinspired optimization algorithms are the Evolutionary Computation (EC) and the Swarm Intelligence (SI).

Evolutionary Computation is a field of Computational Intelligence which uses strategies that mimick the natural evolution process.

The origins of Evolutionary Computation date back to the 1950's, with the works of (BOX, 1957), (FRIEDBERG; DUNHAM; NORTH, 1958) and (BREMERMANN, 1962). The fundamental approaches emerged during the 1970's with the works of (HOLLAND, 1975) (Genetic Algorithms – GA), (FOGEL, 1962) (Evolutionary Programming – EP) and (RECHENBERG, 1965) (Evolutionary Strategies – ES). However, the EC has started to receive significant attention since the 1990's and the number of publications and congresses related to this area have growth as well as novel approaches have emerged, descending from the three main approaches, previously cited: GA, EP and ES, where some of the most used are: Memetic Algorithms (MOSCATO; COTTA; MENDES, 2004), Differential Evolution (STORN; PRICE, 1997), Genetic Programming (KOZA, 1992) and Gene Expression Programming (FERREIRA, 2001).

Gerardo Beni and Jin Wang are known as the creators of the term "Swarm Intelligence" (SI). Basically, Swarm intelligence is the collective behavior of decentralized, self-organized systems, natural or artificial (BENI; WANG, 1989). The inspirations of SI algorithms are related to the behavior of social living beings (BONABEAU; DORIGO; THERAULAZ, 1999), such as fish schools, ant colonies, bee colonies, bird flocking, the hunting nechanism of gray wolves, bacterial growth, among others.

The main SI approaches are the Ant Colony Optimization (ACO) (DORIGO; STÜTZLE, 2004), Particle Swarm Optimization (PSO) (POLI, 2008) and the Artificial Bee Colony Optimization (ABC) (KARABOGA; BASTURK, 2008). Novel approaches and applications of SI are presented in (PARPINELLI; LOPES, 2012).

Many Bioinformatics problems are featured mainly to be non-linear and strongly constrained due to the lack of exact methods for solving such a class of problems, the need for robust heuristic methods arises. Along decades, Evolutionary Computation (EC) and Swarm Intelligence (SI) have provided a large range of flexible and robust optimization methods, capable of dealing successfully with complex optimization problems. Both EC and SI are population-based methods in which each individual of a population represents a tentative solution to the problem to be solved. In recent years several other SI algorithms have appeared, as presented by (PARPINELLI; LOPES, 2011). With such diversity of search strategies, it is possible to establish an analogy with the dynamics of biological ecosystems. Section 2.8.6 presents the main concepts of the ecological-inspired algorithm, named ECO.

Sections 2.8.1, 2.8.2, 2.8.3, 2.8.4, 2.8.5 and 2.8.6 present EC and SI algorithms that are related to this work.

2.8.1 GENE-EXPRESSION PROGRAMMING (GEP)

Gene-Expression Programming (GEP) is an extension of Genetic Programming (GP) that was proposed by (FERREIRA, 2001). The difference between these approaches lies in how the individuals are represented. In GP, the individuals are nonlinear entities of different sizes and shapes (concrete syntax trees). On the other hand, in GEP the individuals are encoded as fixed-size linear strings (also known as genome or chromosome), which are afterwards expressed as nonlinear entities of different sizes and shapes (i.e. expression trees or diagram representations) (FERREIRA, 2001).

The encoding of individuals in GEP is based on the biological concept of open read frame (ORF), that is the coding sequence of a gene. However, it is important to know that genes are composed of more sequences than the respective ORF. In biology, an ORF is composed of amino acid codons, beginning with a ”start“ codon and ending at a termination codon. GEP genes are composed of a head and a tail. The head has symbols that represent functions and terminals. On the other hand, the tail contains only terminals. GEP genes can have noncoding regions, that, in fact, are the essence of GEP, allowing modifications of the genome using any genetic operator without restrictions, producing valid programs without the need for editing processes(FERREIRA, 2001). In other words, the encoding region of a gene (ORF) can ”activate“ or ”deactivate“ portions of the genetic material algorithm, according to the functions and their arities (i.e. number of arguments) encoded in the head of the gene.

GEPCLASS (WEINERT; LOPES, 2006) is an implementation of GEP specially designed for finding rules for classification problems based on supervised learning, where it is aimed to find rules for modeling a given domain of known data samples, and then classify unseen sets of data. In GEPCLASS, a population of individuals evolves for a number of generations, where selected individuals are subjected to genetic operators (mutation, recombination and transposition), generating diversity and, consequently, allowing the evolutionary process to continue for more generations, increasing the chances of finding even better solutions. The head of the gene can have elements belonging only to the set of functions, such as logical and comparison operators. (i.ex.: *AND, OR, NOT*, $=$, $\neq$, $<$ and $>$). The tail, in turn, can have elements either from the set of functions or from the set of terminals, which in turn, includes the attributes that describe particular values.

The mapping between the genotype to the phenotype is carried out as follows. The chromosome is transcribed into a variable-size expression tree (ET), following the Karva language (FERREIRA, 2001), where each gene is trascribed to a separated subtree. Then, all sub-trees are joined together by a linking function (*AND* or *OR* operator), composing the ET that represents a candidate solution to a given problem. The quality of the candidate solutions is measure by a fitness function.

Algorithm 11 (see Annex B) presents the pseudo-code of the GEP algorithm, as proposed by (FERREIRA, 2001).

2.8.2 PARTICLE SWARM OPTIMIZATION (PSO)

Swarm-based algorithms are inspired by the behavior of some social living beings, such as ants, bees, birds, and fishes (PARPINELLI; LOPES, 2012a). Self-organization and decentralized control are remarkable features of swarm-based systems that, such as in nature, leads to an emergent behavior. Emergent behavior is a property that emerges through local interactions among system components that is not possible to be achieved by any of the components of the system acting alone (GARNIER; GAUTRAIS; THERAULAZ, 2007).

The Particle Swarm Optimization algorithm (PSO) was motivated by the coordinate movement of fish schools and bird flocks (KENNEDY; EBERHART, 1995). The PSO is compounded by a swarm of particles that interacts each other in a continuous search space. The position of each particle represents a potential solution to the problem being solved and it is represented as an n-dimensional vector. In PSO, particles "fly" through the hyperdimensional search space, and changes to their positions are based on the socio-cognitive tendency of particles to emulate the success achieved by other particles. Each particle of the swarm has its own life experience and is able to evaluate the quality of own experience. As social individuals, they also have knowledge about how well their neighbors have behaved. These two kinds of information corresponds to the cognitive component (individual learning) and the social component (cultural transmission), respectively. Hence, decisions of an individual takes into account both the cognitive and the social components, thus leading the population (swarm) to an emergent behavior. The PSO is shown in Algorithm 12 (see Annex B). In this Algorithm, each solution $\vec{x}_i = [x_{i1}, x_{i2}, ..., x_{id}]$ of dimension d is evaluated by a fitness function $f(\vec{x}_i)$, where $i = 1, ..., n$. As mentioned before, particles "fly" through the hyperdimensional search space according to their velocity $\vec{v}_i$. Each individual of a population has its own life experience ($\vec{p}_i$) They are social individuals, and so they also have knowledge about the quality of their neighbors ($\vec{g}$) . These two sources of information correspond to the cognitive and social components, respectively. The parameters φ_p, φ_g and w determine the relative influence of the cognitive social and inertia components, respectively. r_p and r_g are randomly generated numbers (PARPINELLI; TEODORO; LOPES, 2012). The PSO algorithm may converge to a local optimal during the search. This can be avoided or, at least, delayed through parameters tuning or mechanism, such as decimation or explosion. It is important to known that the probability of finding a better solution is very low after the stagnation of the search.

2.8.3 ARTIFICIAL BEE COLONY ALGORITHM (ABC)

The Artificial Bee Colony Algorithm (ABC) was inspired by the foraging behavior of honey bees. ABC was first proposed by (KARABOGA; BASTURK, 2008) for solving multi-dimensional and multi-modal optimization problems. The bees aim at discovering places of food sources (that is, regions in the search space) with high amount of nectar (good fitness values, meaning good solutions for the problem). There are three types of bees: scout bees that randomly fly in the search space without guidance, employed bees that exploit the neighborhood of their locations selecting a random solution to be perturbed, and onlooker bees that use the population fitness to select probabilistically a guiding solution to exploit its neighborhood. If the amount of nectar of a new source is higher than that of the previous one in their memory, they update the new position and forget the previous one (this is a greedy selection method). If a solution is not improved by a predetermined number of trials, the food source is abandoned by the corresponding employed bee and it becomes a scout bee. The ABC algorithm attempts to balance exploration and exploitation by using the employed and onlooker bees to perform local search, and the scout bees to perform global search, respectively. The ABC is shown in Algorithm 13 (see Annex B).

The number of applications with ABC has growth rapidly. Karaboga et al. (2014) present a survey of the advances and applications with ABC.

2.8.4 DIFFERENTIAL EVOLUTION (DE)

The Differential Evolution was proposed by (STORN; PRICE, 1997) and is a optimisation method from the Evolutionary Computation field.

The DE algorithm is a population-based meta-heuristic composed by n candidate solutions which, in turn, are represented as solution vectors. Each vector $\overrightarrow{x_i} = [x_{i_1}, x_{i_2}, \ldots, x_{i_d}]$ is evaluated by an objective function. The main idea of DE is to generate a new population using difference vectors to generate perturbations in the current population, where new vectors (that form a mutant population) are generated through the combination of randomly selected vectors weighted by a constant F (differential weight). This operation is analogue to the mutation operation in EC algorithms. Then, the generated vector are probabilistically recombined with a predetermined vector (or target vector) according to a crossover rate (CR), yielding to a trial vector ($\overrightarrow{y_i}$). Finally, a greedy selection is done, where the trial vector is accepted for the new population if it minimize the value of the objective function.

Algorithm 14 (see Annex B) shows the canonical DE with DE/rand/1/bin scheme as described by (STORN; PRICE, 1997), where a random solution is perturbed ("rand"), a difference vector is used in the perturbation ("1") and a binomial recombination operator is used ("bin"). It is important to recall that many other schemes for creation of candidate solutions are possible. For instance, the *best* solution could be selected to be perturbed.

2.8.5 BIOGEOGRAPHY-BASED OPTIMIZATION

The Biogegraphy-based Optimization (BBO) was proposed by (SIMON, 2008). It is a population-based algorithm, where each individual is considered as a "habitat" with a habitat suitability index (HSI), which is similar to the fitness function that is used to evaluate the quality of the individual for a given problem. Solutions with high HSI tend to share their features with low HSI solutions in order to improve the quality of the solutions.

Algorithm 15 (see Annex B) shows the pseudo-code of the BBO algorithm, where n is the number of habitats; E is the maximum possible emigration rate; I is the maximum possible immigration rate; k is the number of species of the k-th individual; S_{max} is the largest number of species of a habitat; m_{max} is a user-defined mutation weight and $elite$ is the number of best individuals that will "survive" to next generation.

A potential solution is represent by a d-dimensional vector $\overrightarrow{x_i} = [x_{i_1}, x_{i_2}, \cdots, x_{i_d}]$, where each dimension in the solution vector is considered to be a suitability index variable (SIV). Each individual $\overrightarrow{x_i}$ has an immigration rate α_i and an emigration rate μ_i. These rates are functions of the number of species in the habitat, where a suitable solutions have high μ and low α.

2.8.6 THE ECOLOGICAL-INSPIRED APPROACH

The Ecological-Inspired algorithm (ECO) represents a new perspective to develop cooperative evolutionary algorithms. This approach was proposed by (PARPINELLI; LOPES, 2012a) and is composed by populations of individuals (candidate solutions for a problem being solved), where each population evolves according to an optimization strategy. Thus, individuals of each population are modified according to the mechanisms of diversification and intensification of an optimization strategy. The ECO algorithm can be modelled in a homogeneous or heterogeneous manner. In a homogeneous model, all populations evolve according to the same optimization strategy (configured with the same parameters). On the other hand, the hybridization of different strategies characterises a heterogeneous model.

The ecological inspiration stems from the use of some ecological concepts, such as: habitats, ecological relationships and ecological successions (BEGON; TOWNSEND; HARPER, 2006)(MAY; MCLEAN, 2007). Once dispersed in the search space, populations of individuals established in the same region constitute an ecological habitat. For instance, in a multimodal hyper-surface, each peak can become a promising habitat for some populations. Populations can move around through all the environment. However, each population may belong only to one habitat at a given time t.

Two categories of ecological relationships can be defined: intra-habitats relationships that occur between populations inside each habitat, and inter-habitats relationships that occur between habitats (BEGON; TOWNSEND; HARPER, 2006)(MAY; MCLEAN, 2007).

Populations belonging to the same habitat can establish reproductive links between their individuals (i.e. intra-habitat relationship), favouring the co-evolution of the involved populations through competition for mating. On the other hand, the inter-habitats relationship are the migrations between habitats. Individuals belonging to a given habitat can migrate to other habitats aiming at identifying promising areas for survival and mating.

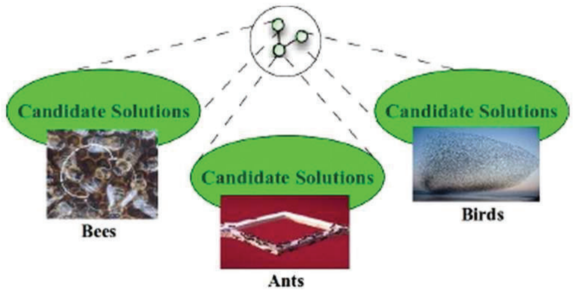
Considering the ecological context of ECO, the intra-habitats and the inter-habitats relationships are responsible for intensifying and diversifying the search, respectively. The ecological successions represent the transformational process of the system, where habitats are formed, relations between populations are established and the system stabilizes by means of self-organization.

A key concept of the ECO system is the definition of habitats. The ECO approach uses a hierarchical clustering algorithm to set-up the habitats where each cluster represents a habitat. Hence, the habitats are defined probabilistically taking into account the distance information returned by clustering algorithm. Once the habitats have been defined, the next step is the definition of the communication topology for each habitat that is probabilistically defined (PARPINELLI; LOPES, 2012b).

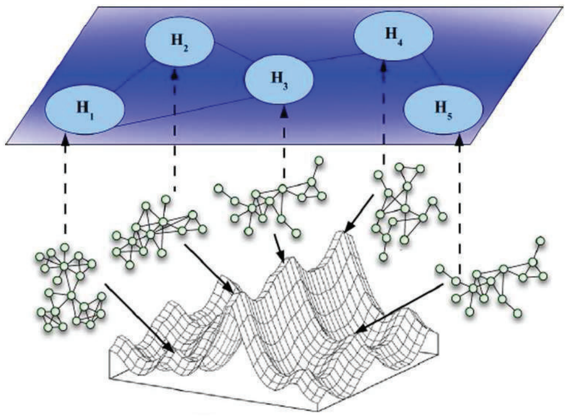
For a habitat with more than one population, intra-habitat communication occurs in such a way that each population inside the habitat chooses another population to perform communication. Here, the distance between populations influence directly the probabilistic decision. The closer two populations are from each other the higher is the chance of these two populations communicate. The opposite happens with farthest populations.

Figure 9(a) shows a possible representation of the elements of Ecological-inspired computational approach, where three populations which, in turn, represent EC algorithms with different behaviors, combining local and global search strategies. Figure 9(b) shows a general view of the Ecological-Inspired approach, where in the lower level populations search for potencial solutions for a given problem in the search space. In the intermediate level, the populations form habitats and intra-habitats communication topologies take place, where the small circles represent the populations. In the upper level, the habitats are connected through an inter-habitats communication topolopy.

This combination aims at taking advantage of the benefits of different several search strategies and the maintenance of the diversity of the populations through sporadic migrations between them.



(a)



(b)

Figure 9: General view of the Ecological-Inspired Approach: (a) Possible representation of the elements of a computational "ecosystem", (b) Hyper-surface of the search space and communication topologies.

Source: Adapted from (PARPINELLI; LOPES, 2012a).

To end this section, it is important to mention that Parpinelli and Lopes (2015) present a review of the theoretical foundations, applications and future research directions of the ECO.

2.9 PARALLEL COMPUTING

2.9.1 BEOWULF CLUSTER AND THE MESSAGE PASSING INTERFACE (MPI) LIBRARY

The concept of Beowulf clusters [8] was created by Thomas Sterling and Donald Becker at the *Center of Excellence in Space Data and Information Sciences* (CESDIS – NASA) with the main objetive of building parallel computers that incorporates mass-market PC technology to achieve the best price/performance and specific computational requirements for scientifc applications (STERLING, 2002).

A Beowulf Cluster incorporates all hardware components and software responsible for program execution, external communications, user command interface, among others. The main hardware components are the processing nodes and the interconnection network that interconnects them to form a single system. The software components are development tools for developing parallel programs and the software environment for managing the parallel resources of the cluster. The specification of a Beowulf cluster determines the costs, performance and usability of the system (EL-REWINI; ABD-EL-BARR, 2005).

Higher-level programming abstractions are used to develop parallel applications. Beowulf users converged to the Message Passing Interface (MPI) (GROPP; LUSK; THAKUR, 1999) and Parallel Virtual Machine (PVM) (GEIST et al., 1994) models.

The MPI library provides a standard library of routines that helps on the development of efficient parallel applications using the message passing paradigm (see Annex C).

A MPI application can be seen as a group of concurrent autonomous processes, in a MIMD style, that work together, exchanging data and synchronizing operations. MPI does not specify the execution model for each process. Thus, the processeses can be purely sequential or multi-threaded. The execution model can be defined by the developer. In other works, the parallelism is explicit, where the identification of

[8]The name "Beowulf" derives from an english epic poem which is possibly the oldest English long poem, dated between the 8th and 11th centuries. Sterling chose because Beowulf has "thirty men's heft of grasp in the gripe of his hand"

parallelism, creation and organization of processes are done by the programmer (EL-REWINI; ABD-EL-BARR, 2005).

2.9.2 DESIGN APPROACH OF PARALLEL ALGORITHMS

A design approach for parallel algorithm development has four phases (ROOSTA, 1999), as shown in Figure 10: partitioning, communication, grouping and mapping.

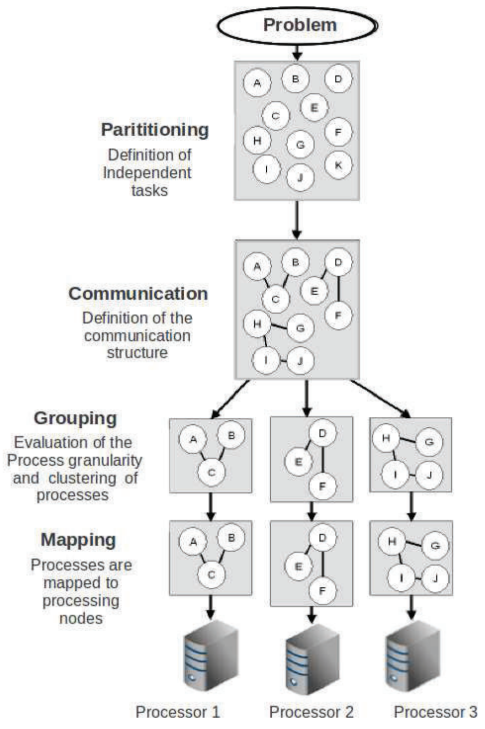

Problem
Parititioning
Definition of Independent tasks
Communication
Definition of the communication structure
Grouping
Evaluation of the Process granularity and clustering of processes
Mapping
Processes are mapped to processing nodes
Processor 1
Processor 2
Processor 3

Figure 10: Design approach of parallel algorithms

Source: Adapted from (ROOSTA, 1999).

In the first two phases, hardware-independent aspects are considered, as operation concurrency and task decomposition. On the other hand, hardware-dependent aspects are considered in the last two phases, as performance and the global communication cost.

- Partitioning: the parallelism oportunities are identified in this phase, splitting the processing into several distinct parallel routines. There are two partitioning methods:

  - Domain partitioning: focuses on data associated with the problem and determines suitable computation of the data;
  - Functional partitioning: focuses on processing decomposition into disjoint tasks.

  These methods can be applied to different components of the problem. Consequently, task or data replication must be avoided. In this phase, paradigms for parallism must also be defined.

- Communication: in this phase, the communication structure and the environment for task execution are defined.

  - Communication channel structure: related to direct or undirect link between tasks;
  - Message-passing structure: specification of messages that must be sent to or received from the communication channel.

  This phase is essential due to the communication cost of parallel computing. Generally, it is aimed to optimize the performance through the distribution of communication operations between several tasks, organizing them in order to allow parallel execution. However, thinking in terms of channel structure can help in the communication cost evaluation.

- Grouping: this phase evaluates the granularity of the tasks (or size of the process) and the communication cost between them. During this phase, the first two phases can be verified to make the right decisions, involving the available hardware and, if necessary, return to the two previous phases. The tasks with more data dependency or those that are more correlated must be grouped.

- Mapping: in this phase, each process is mapped to a processing node in order to maximize the usage of the available processing nodes and reduze the communication cost between them.

2.10 METHODS APPLIED TO THE PROTEIN FOLDING PROBLEM

The protein structure prediction (PSP) can be done experimentally or by *in silico* methods. In this section, the main approaches applied to the PSP will be presented: experimental techniques, Sequence-based comparative/homology modeling methods, Threading methods, Taxonomic methods, *Ab initio* prediction methods and protein structure reconstruction.

2.10.1 EXPERIMENTAL METHODS

Advances in experimental techniques, such as protein engineering, Nuclear Magnetic Resonance (NMR), Fluorescence Resonance Energy Transfer (FRET), have made it possible to obtain detailed information about the protein folding process (CHEN et al., 2008; DINNER et al., 2000).

NMR is an important tool for the study of protein structures, and it is the method with the highest structural resolution (NÖLTING, 2006; JARONIEC et al., 2004; WÜTHRICH, 1986). The NMR approaches can be classified into four types: the chemical shifts, coupling constants, Nuclear Overhauser Effect (NOE) and amide proton exchange rate (WÜTHRICH, 1986).

Circular dichroism (CD) is sensitive to protein conformation changes. Although its resolution is not as high as NMR, it is sensitive to secondary structure (NÖLTING, 2006; PLAXCO; DOBSON, 1996; CHEN et al., 2008).

Kinetic resolution of molecular dimensions became possible by advances in X-ray scattering (SEMISOTNOV et al., 1996; DRENTH, 1999; SUNDE; BLAKE, 1997) and dynamic light scattering (GAST et al., 1997). NMR spectrometry is also used to obtain information about local and global folding events (NÖLTING, 2006), such as residue-specific information about the structure at different stages of the folding reaction (NULAND et al., 1998). For instance, real-time NMR spectroscopy with kinetic resolution has significantly advanced into the millisecond time range (HOELTZLI; FRIEDEN, 1998; NÖLTING, 2006).

Cryo-electron Microscopy (cryo-ME) is another versatile tool for protein structure determination (FRANK, 2006; JIMENEZ et al., 2002; ORLOVA; SAIBIL, 2011).

2.10.2 HOMOLOGY MODELING METHODS

The basic premise for template-based protein structure prediction is threefold (ZAKI; BYSTROFF, 2008): (a) similar sequences adopt similar protein structures (CHOTHIA; LESK, 1986); (b) many unrelated sequences fold into similar structures (FISCHER et al., 1996); and (c) there are only a relatively small number of unique structural folds, when compared with the number of proteins in nature (ZHANG; DELISI, 1998). The first observation is the foundation of homology modeling, while the second and third assumptions are foundations of protein threading (See Section 2.10.3).

Basically, template-based homology consists of four steps (CHEN et al., 2008; PETREY; HONIG, 2005): (a) Finding known structures, considered as *templates*, related to the sequence to be modeled (target); (b) Aligning the target sequence to the template structure; (c) building structural frameworks by copying the aligned regions or by satisfying the spatial constraints from templates; (d) Constructing the unaligned loop regions and adding side-chain atoms. The first two steps can be considered as threading or fold recognition procedures. Two additional steps are usually included: model refinement and model evaluation.

The first homology model was developed by Browne *et al.* using a method called rigid body assembly (BROWNE; LESK, 1969) and, since then, several approaches have been developed. Overall, these approaches can be classified into four categories: (a) rigid body assembly, (b) segment matching, (c) spatial restraint, and (d) artificial evolution model building.

The rigid-body method has been implemented in several programs, including COMPOSER (SUTCLIFFE et al., 1987), MODELLER (ESWAR et al., 2007), Swiss-Model (PEITSCH; JONGENEEL, 1993; ARNOLD et al., 2006), PriSM (YANG; HONIG, 1999), 3D-Jigsaw (BATES et al., 2001), RigidFinder (ABYZOV et al., 2010) and (PANDURANGAN; TOPF, 2012).

Segment-matching method is developed based on the observation that most hexapeptide segments of proteins structures can be clustered into about 100 structural classes (UNGER et al., 1989).

MODELLER (ESWAR et al., 2007) is considered the most popular homology modeling program, which uses spatial constraints derived from template structures to guide the building process. DSModeller is a commercial version of MODELLER.

The artificial evolution method build structural models by simulating the natural process of structural evolution from a template structure to the target model (ZAKI; BYSTROFF, 2008; PETREY et al., 2003).

Side-chain prediction also represents a challenge in homology modeling. For instance, this problem has been formulated as a graph-theoretic problem and solved by combinatorial optimization algorithm (XU, 2005; DESMET; SPRIET; LASTERS, 2002).

Fain and Levitt (2003) proposed a method for designing a funnel surface for folding α, $\beta$and $\alpha\beta$ structures, assuming *a priori* knownledge of secondary structures. Although their method do not solve the protein folding problem and has several limitations, it is suitable for determining effective energies for homology modeling.

2.10.3 THREADING METHODS

Protein threading was introduced in the 1990s by (BOWIE; LUTHY; EINSENBERG, 1991; JONES; TAYLOR; THORNTON, 1992). It corresponds to the case where structures (templates) similar to a given target sequence exist in the PDB but are not easily identified (PETREY; HONIG, 2005). The main idea is to *thread* the amino acids of a query protein, following their sequential order and allowing for insertions and gaps into structural positions of a template structure in an optimal way, measured by a scoring function. This procedure is repeated for each template structure in a database of protein structures (See Section 2.3). Basically, threading approaches involves four issues: (a) development of energy functions for assessing the quality of a sequence-structure alignment; (b) threading algorithms for finding a sequence-structure alignment through energy function optimization; (c) statistical assessment and Folding Recognition (FR); and (d) development of a structural template library (ZAKI et al., 2004).

Threading methods use protein informations obtained from databases such as SCOP (Structural Classification of Proteins) (Lo Conte et al., 2000; ANDREEVA et al., 2008) and CATH (Class, Architecture, Topology, Homology) (PEARL et al., 2005; CUFF et al., 2011), which divide proteins into discrete families based on simple sequence relationships, superfamilies based on structural and function similarity.

To the best of our knowledge, I-TASSER (AMBRISH; XU; ZHANG, 2011) represents the threading method with the best results.

2.10.4 TAXONOMIC METHODS

The taxonomy-based method for protein fold recognition, which was proposed by (DUBCHAK et al., 1995), classifies a query protein into one of the known structures. Most implementations of the taxonomy-based method have adopted the SCOP (See Section 2.10.3 above) classification architecture (YANG; CHEN, 2011). Basically, this method has two procedures: feature extraction and a Machine Learning (ML) classifier. Features from the amino acid sequence are extracted by a feature extraction procedure. In addition, several features can be constructed through exploiting information. For instance, (CHEN; KURGAN, 2007) proposed a fold classification method, called PFRES, that uses the PSI-BLAST (ALTSCHUL, 1997) profile, which considers evolutionary information (JONES, 1999), and features generated from secondary structure predicted using the PSI-PRED (JONES, 1999). (SHEN; CHOU, 2009) proposed a novel approach that is featured by combining functional domains and the sequential evolution information through a fusion emsemble classifier. The taxonomic methods that achieve highest prediction accuracies are the TAXFOLD (YANG; CHEN, 2011), ACCFold (DONG; ZHOU; GUAN, 2009), PFRES (CHEN; KURGAN, 2007), PFP-pred (SHEN; CHOU, 2006), SVM-Fold (MELVIN et al., 2007), FUGUE (SHI; BLUNDELL; MIZUGUCHI, 2001), THREADER (JONES; TAYLOR; THORNTON, 1992), Kavousi method (KAVOUSI et al., 2011), (MOHAMMAD; NAGARAJARAM, 2011) and (KAVOUSI et al., 2012).

2.10.5 *AB INITIO* METHODS

Generally, *ab initio* (also known as *de novo*) methods perform the protein structure prediction “from first principles” without using any structural information of previously solved native structures. *Ab initio* methods are based on the principles of physics, where the candidate conformations are evaluated based on an energy function (related to a force field) or a scoring function (ZAKI et al., 2004). The main problem with *ab initio* methods is that we do not have enough knowledge about the folding of a protein. Several *ab initio* methods have been developed with different energy/scoring functions and protein structure representations (See Section 2.4).

For instance, ROSETTA (ROHL et al., 2004; SCHMITZ et al., 2012) and, the previously cited, I-TASSER (AMBRISH; XU; ZHANG, 2011) are examples of *ab initio* simulations. The ROSETTA simulation algorithm [9] uses Monte Carlo Fragment Insertion (MCFI) to predict protein structure of small proteins or protein fragments without structural templates. The MCFI is a downhill searh in a knowledge-based energy landscape (BYSTROFF; SHAO, 2004).

Arkun and Gur (2012), Liwo, Khalili and Scheraga (2005) presented Molecular Dynamics approaches for the PFP, using coarse-grained protein models, in order to analise the folding pathways.

Lindorff-Larsen et al. (2011) reports the results of atomic-level Molecular Dy-

[9]Available in www.bioinfo.rpi.edu

namics simulations executed in a specialized supercomputer, called Anton. They studied the reversible folding simulations of 12 proteins (α-helices, β-sheets and α/β that range in size from 10 to 80 amino acids.

Gin, Garrahan and Geissler (2009) presented the first systematic comparison of folding pathways within Gō-like and full models, using the Contact Appearance Order (CAO) histogram of sequences. They focused on a lattice representation of proteins and used the Metropolis Monte Carlo algorithm for evolving the chain.

Dokholyan et al. (1998) applied the Discrete Molecular Dynamics (DMD) (PROCTOR; DING; DOKHOLYAN, 2011) to the PFP, using a Gō-based model

Evolutionary Computation (EC) approaches have been applied to the *ab initio* PFP. For instance, Custódio, Barbosa and Dardenne (2010) presented a full-atom *ab initio* PFP with a Genetic Algorithm that uses a similarity-based surrogate model. Capriles, Custódio and Dardenne (2010) presented an *ab initio* PFP using Genetic Algorithms (GA) and coarse-grained model for side chains. Benítez and Lopes (2010) describes a Hierarchical Parallel Genetic Algorithm (HPGA) applied to the PFP using the 3D-HP Side-Chain model (3D-HP-SC). Shmygelska and Hoos (2005), Fidanova (2006), Chu, Till and Zomaya (2005) applied the Ant Colony Optimisation (ACO) method to the PFP using the 3D-HP model. Benítez, Parpinelli and Lopes (2012) proposed the hybridization of the Artificial Bee Colony (ABC) and a Genetic Algorithm, in a hierarchical topology, for the PFP, using the 3D-HP-SC model. More recently, García-Martínez et al. (2014) present an application of the *Universal Evolutionary Global Optimization* (UEGO) to determine the protein structure, using the 3D-HP model and conformations encoded by relative coordinates.

Local search methods such as Tabu Search (TS), Simulated Annealing and Hill-climbing have been applied as genetic operators of GAs applied to the PFP by (COX et al., 2004; JIANG et al., 2003; TANTAR et al., 2007), (LI, 2007) presented a Simulated Annealing approach applied to the PFP, using the 2D-HP model. Recently, Albrecht, Kapsokalivas and Steinhöfel (2010) presented simulations of unfolding executed by a new population-based search that uses Simulated Annealing (SA), using a cubic lattices model of proteins with two types of simplified energy functions, known as the Miyazawa-Jerningan (MJ) energy function (MIYAZAWA; JERNIGAN, 1985).

The Differential Evolution algorithm have been applied to the PFP by (BITELLO; LOPES, 2007) and (KALEGARI; LOPES, 2010), using the 2D-HP and 2D-AB *off-lattice* models, respectively.

Artificial Immune Systems (AIS) have been also applied to the PFP, using the 2D and 3D-HP (CUTELLO; NARZISI; NICOSIA, 2005). (ALMEIDA; GONÇALVES; DELGADO, 2007) proposed an hybrid system for the PFP, based on AIS, tabu search and fuzzy logic, using the 3D-HP model.

Ostrovsky et al. (2001) proposed a CA Margolus dynamics approach for the simulation of polymers, where "ON" cells represent monomers.

More recently, Santos, Villot and Diéguez (2013) presents a method inspired in the approach proposed by (KRASNOGOR et al., 2006), using the 3D-HP model of proteins. It is based on a neural-CA approach optimized with Differential Evolution.

Zaki et al. (2004) proposed an "unfolding" approach to learn the sequence of event that leads the folded protein to its unfolded state, applying graph based methods on WSGs. They demonstrate the success of their approachon proteins whose pathway is partially known.

The Critical Assessment of Structure Prediction (CASP) has been used to assess the overall prediction for protein structures by the existing prediction techniques (MOULT et al., 2007; ZAKI; BYSTROFF, 2008; MONASTYRSKYY et al., 2011).

Folding@home (FAH or F@h) is a distributed computing project applied to the protein folding simulation and computational drug design that uses idle personal computers of volunteers around the World. FAH is developed by the Pande Laboratory at the Stanford University [10], under the direction of Vijay Pande. Lane et al. (2013) present a review of protein folding achievements from Folding@home, demonstrating that it is also capable of slow-folding proteins beyond the capabilities of other systems, such as the, previously cited, special-purpose MD supercomputer, called Anton, from DESRES group [11].

2.10.6 CONTACT MAPS AND PROTEIN STRUCTURE RECONSTRUCTION

The solution of the folding problem is still lacking. Among different possibilities, the prediction of protein contact maps starting from sequence is particularly promising, since even a partial solution of it can significantly help the prediction of the protein structure (FARISELLI et al., 2001). Several approaches have been developed for contact map prediction. For instance, in (VENDRUSCOLO; KUSSELL; DOMANY,

[10]For more information, visit http://folding.stanford.edu/
[11]For more information, visit http://www.deshawresearch.com/

1997) proposed a heuristic method of growing the amino acid chain of monomers one by one, using the METROPOLIS criterion.

Abu-Doleh, Al-Jarrah and Alkhateeb (2012) introduced a new approach for contact map prediction, called JUSTcon, which consists of multiple parallel stages that are based on adaptative neuro-fuzzy inference System (ANFIS) and *K* nearest neighbors (KNN) classifier.

Cheng and Baldi (2007) proposed a new contact map preditor (called SVMcon) that uses Support Vector Machines (SVM) to predict residues contact in the proteins.

Lane et al. (2013) developed a novel machine learning approach to contact map prediction using three steps, based on two-dimensional and deep neural networks in order to refine the prediction of contacts.

Barah and Sinha (2008) used a coarse-grained network description of protein structures based on contact maps, called Protein Contact Networks (PCN), to study secondary structural elements in structures available in the PDB, without considering atomistic details.

Diaz and Tischer (2011) proposed a Genetic Algorithm approach for mining Contact Maps by using Cellular Automata models (i.e. neighborhood relationship, set of possible states of the cells and transition rule) from simulation trajectories of the protein HP35-NleNle [12] (Chicken villin subdomain, PDB 2F4K), which is formed solely by α-helices (SHERMAN, 2005). To the best of our knowledge, it represents the first application of Cellular Automata to the PFP, using Contact Maps and Genetic Algorithms. However, they do not describe their approach in a clear manner.

A reconstruction procedure of the three-dimensional structure of proteins is need after the contact map prediction. Vassura et al. (2008b) stated that this procedure is equivalent to the unit-disk-graph recognition, which has been proved to be *NP*-hard (BREU; KIRKPATRICK, 1993). For instance, (VASSURA et al., 2008b, 2008a; MEDRI, 2009; DUARTE et al., 2010; VASSURA et al., 2011) proposed approaches for protein structure reconstruction from native contact maps.

Konopka et al. (2014) present a pipeline approach applied to structure reconstruction, composed by the FT-COMAR (VASSURA et al., 2008b), the SABBAC (MAUPETIT; GAUTIER; TUFFÉRY, 2006) and the SCWRL4 (KRIVOV; SHAPOVALOV; DUNBRACK, 2009) algorithms. They obtained structures with an average RMSD of 5.27Å,

[12]Available in: https://simtk.org/home/foldvillin

using a representative set of protein structures and the KcsA ion channel.

Basically, the studies differ in terms of test sets and methodologies used for validation. Therefore, the different approaches cannot be compared. For instance, in (DUARTE et al., 2010) validations were limited to reconstruction of protein Cα traces. Taylor, Jones and Sadowski (2012) used a set of globular folds and (NUGENT; JONES, 2012) were limited to prediction of transmembrane proteins.

Typical values of threshold considered in the literature vary between 7 and 12 Å. Medri (2009) stated that, in general, higher threshold values allow a better reconstruction.

Recently, Kuo (2012) proposed a methodology for predicting distance maps from *fuzzy* CMs which, in turn, are generated from CMs at different thresholds, in order to extend the traditional CMs.

Finally, it is important to know that Protein contact site informations can significantly improve the quality of protein structure predictions. For instance, in CASP10, for the first time, contact-assisted structure predictions have been assessed (TAYLOR et al., 2014). However, the highly accurate contact map prediction and reconstruction of protein structures from contact maps are still unsolved problems, which has been proved to be NP-hard.

3 METHODOLOGY

"Divide each difficulty into as many parts as is feasible and necessary to resolve it."

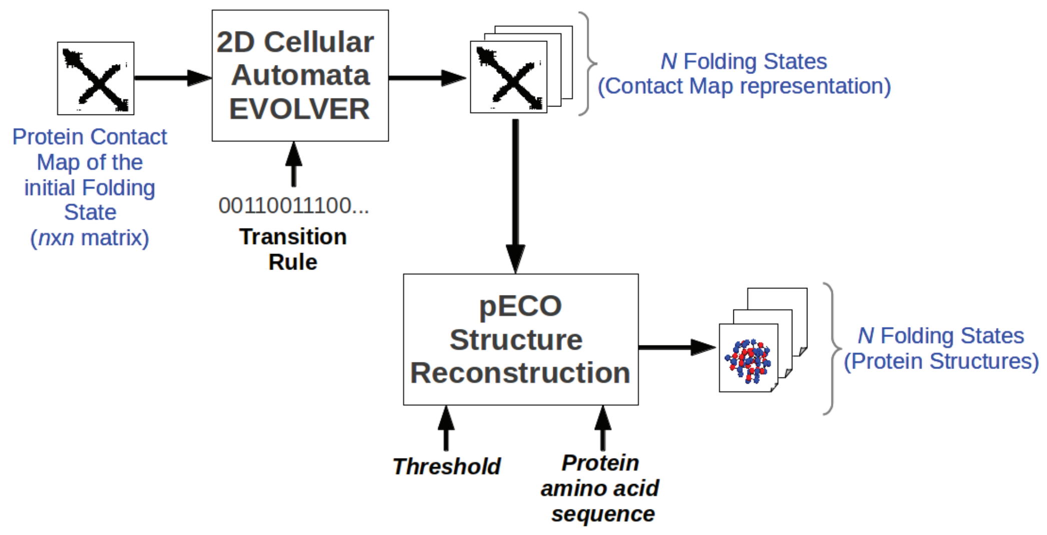
— René Descartes

This thesis proposes a novel heterogeneous computational approach based on Cellular Automata, Bioinspired Computation and Parallel Computing applied to protein folding simulation and protein structure prediction.

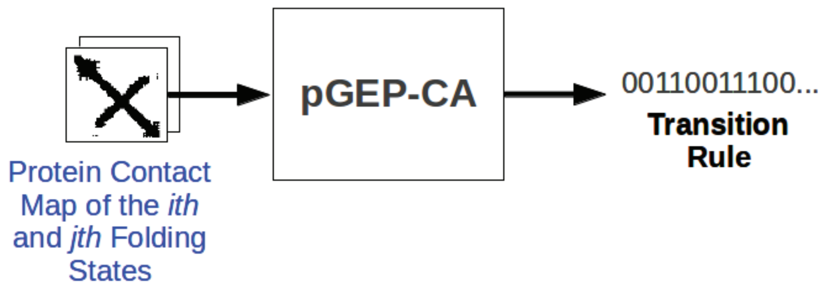
Figure 11(a) presents a simplified functional block diagram of the proposed approach. Protein folding is simulated by using a two-dimensional Cellular Automata (2D-CA) which, in turn, represent the Contact Map (CM) of the folding states, instead using the three-dimensional structure of the protein. A Cellular Automata evolver is used in order to simulate the folding process from an initial folding state that is represented by a CM. Each folding state is obtained according to its predecessor. In other words, the configurational state of the 2D-CA (that represents a folding state) is determined according to its predecessor state and a *transition rule* which, in the context of the protein folding, could be considered as a *folding rule*.

Since finding a *transition rule* for a given dynamic behavior is a very difficult task, it is proposed a novel approach for inducing *transition rules* to simulate the protein folding using CMs, called pGEP-CA, which is based on a Gene-Expression Programming (GEP). Figure 11(b) presents a simplified functional block diagram of the pGEP-CA.

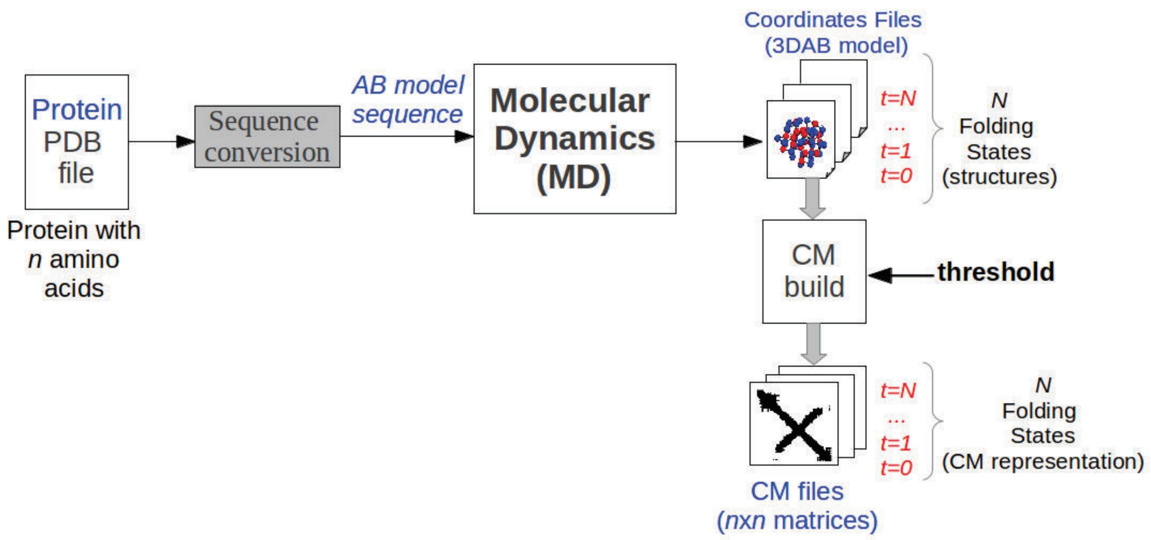
Finally, the protein structure of each folding state is reconstructed. For this purpose, a parallel Ecological-inspired Algorithm (ECO), called pECO, was implemented for protein structure reconstruction from the previously obtained Contact Maps (with a given threshold) and protein sequence.


Protein Contact Map of the initial Folding State (*nxn* matrix)
2D Cellular Automata EVOLVER
00110011100...
Transition Rule
N Folding States (Contact Map representation)
pECO Structure Reconstruction
Threshold
Protein amino acid sequence
N Folding States (Protein Structures)

(a)



(b)


Protein PDB file
Protein with *n* amino acids
Sequence conversion
AB model sequence
Molecular Dynamics (MD)
Coordinates Files (3DAB model)
t=N
...
t=1
t=0
N Folding States (structures)
CM build
threshold
t=N
...
t=1
t=0
N Folding States (CM representation)
CM files (*nxn* matrices)

(c)

Figure 11: (a) The proposed approach for simulating protein folding pathways (b) The pGEP-CA (c) Contact Maps generation procedure.

Source: Own work.

In order to evaluate the proposed approach, the 3D-AB *off-lattice* model is used to represent protein structures and Molecular Dynamics (MD) is used for generating protein structures and CMs, which to the best of our knowledge is the first application of MD to the PFP using that model. Figure 11(c) and Section 4.2 present the CM generation procedure.

In this Chapter, the next sections are organized as follows: Section 3.1 presents the MD approach. Sections 3.2 and 3.3 present the main approaches and methods proposed in this thesis. The methodologies used for validation and performance measurement are presented in Sections 3.4 and 3.5. Finally, Section 3.6 presents the protein sequences which were used in the experiments.

3.1 A MOLECULAR DYNAMICS APPROACH APPLIED TO THE FOLDING PROBLEM USING A COARSE-GRAINED MODEL

As commented in the beginning of this chapter, a Molecular Dynamics approach is proposed in this thesis and used for generating a test set of Contact Maps and three-dimensional structures that represent folding states of folding pathways, using the 3D-AB *off-lattice* model.

Algorithm 1 shows the pseudo-code of the Molecular Dynamics algorithm.

Algorithm 1 Molecular Dynamics pseudo-code

```
1:  **Start**
2:  Set the initial conditions: positions $r_i(t_0)$, velocities $v_i(t_0)$ and accelerations $a_i(t_0)$
3:  **while** $t < t_{max}$ **do**
4:      Compute forces on all particles (amino acids)
5:      Integrate equations of motion
6:      Perform ensemble control
7:      Compute geometric constraints
8:      Compute the desired physical quantities
9:      $t \leftarrow t + \delta t$
10: **end while**
11: **End**
```

- **Set the initial conditions**: in this step, initial positions, velocities and accelerations are assigned to all particles (i.e. amino acids). An initial unfolded or partially folded conformation is randomly generated. To represent the position of the amino acids, three-dimensional Cartesian coordinates are defined by a vector $\overrightarrow{r_i}$, as shown in Equation 8.

$$\overrightarrow{r_i} = (x_i, y_i, z_i) \in \Re, i = 0, ..., N-1 \tag{8}$$

Where, N is the number of amino acids; and x_i, y_i and z_i are the Cartesian coordinates.

The first amino acid of the primary structure is positioned at the origin of the Cartesian system and next ones are positioned at Cartesian coordinates relative to their immediate predecessor, and obtained from random spherical coordinates (see Figure 12), as shown in Equation 9.

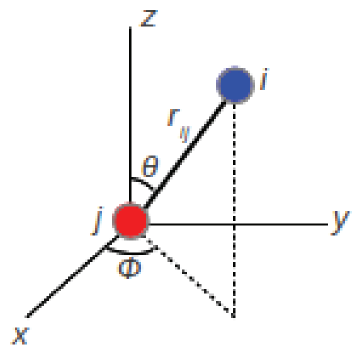


Figure 12: Example of spherical coordinates
Source: Own work.

$$\begin{aligned} x_i &= x_j + r_{ij} * \sin\theta * \cos\phi \\ y_i &= y_j + r_{ij} * \sin\phi * \sin\theta \\ z_i &= z_j + r_{ij} * \cos\theta \end{aligned} \tag{9}$$

Where $\phi \in [0, 2\pi]$ and $\theta \in [0, \pi]$. are the azimuth and inclination, respectively, r_{ij} is the radial distance, $j = i-1$ and $i > 0$ (from second amino acid).

It is important to recall that the AB model uses unity radial distances between residues, that is, unit-length bond (see Section 2.4.2.3 – page 52), as shown in Equation 10.

$$r_{ij} = |\hat{b}_i| = |\vec{r}_i - \vec{r}_{i-1}| = 1 \tag{10}$$

The initial velocities are generated in two steps. First, random directions and a fixed magnitude based on the temperature are assigned, as shown in Equation 11.

$$v_i\Big|_{t=0} = velMag * \overrightarrow{\xi}; \tag{11}$$

Where *velMag* represents the temperature-based magnitude, as shown in Equation 12; $\overrightarrow{\xi}$ is a randomly oriented vector of unit length, generated by a random number generator with uniform distribution over the interval [-1, +1] (RAPAPORT, 2004).

$$velMag = \sqrt{3.(1-\tfrac{1}{N}).T_0} \tag{12}$$

Where N and T_0 represent the number of amino acids of the protein and the initial temperature, respectively.

For generating the unit length vectors, a rejection method proposed by (MARSAGLIA, 1972) is used, where the probability distribution is related to the uniform distribution on a unit sphere, as shown in Algorithm 2.

Algorithm 2 Random Unit length vector generation algorithm

```
1:  Start
2:  $s^2 \leftarrow 2$
3:  while $s^2 > 1$ do
4:      $x \leftarrow 2 * rand() - 1$
5:      $y \leftarrow 2 * rand() - 1$
6:      $s^2 \leftarrow x^2 + y^2$
7:  end while
8:  $x \leftarrow 2 * (\sqrt{1-s^2}) * x$
9:  $y \leftarrow 2 * (\sqrt{1-s^2}) * y$
10: $z \leftarrow 1 - 2 * s^2$
11: End
```

Where, $rand()$ is a Linear Congruential Random Number generator (LCG) (KNUTH, 1981).

Next, the velocities are also adjusted to ensure that the center of mass is at rest at time zero, thereby eliminating any overall flow (RAPAPORT, 2004), as shown in Equation 13.

$$v_i\Big|_{t=0} = v_i\Big|_{t=0} - \tfrac{1}{N}\textstyle\sum_j v_j\Big|_{t=0} \tag{13}$$

All the initial accelerations initialized to zero, as shown in Equation 14.

$$a_i\Big|_{t=0} = 0\,\forall i \leq N-1 \tag{14}$$

- **Compute forces on all particles**: This section is based on (RAPAPORT, 2004). The forces f_i that act on the particles are usually derived from the potential energy $u(r)$ (see Equation 2), as shown in Equation 15.

$$f = \nabla u(r) = \nabla E(\hat{b_i};\sigma) = \nabla(E_{Angles} + E_{torsion} + E_{LJ}) \tag{15}$$

  The equations of motion are written according to Newton's second law, as shown in Equation 16.

$$f_i = m\ddot{\vec{r}}_i = \textstyle\sum_{j=1(j\neq i)}^{N} f_{ij} \tag{16}$$

  Where, N represents the number of amino acids. The Newton's third law implies that $f_{ji} = -fij$ Thus, each particle pair need to be examined only once. The AB model does not represent the mass value of residues. Thus, we used the unity dimensionless mass in this work ($m = 1$).

  According to Equation 15, the force field has three terms: bond-angle forces, bond-torsion forces and forces corresponding to the Lennard-Jones potential.

  - **Lennard-Jones potential**: The force that the jth amino acid exerts on the ith amino acid, corresponding to the Lennard-Jones potential is:

$$f_{ij} = 48 * \varepsilon(\sigma_i, \sigma_j)(r_{ij}^{-14} - \tfrac{1}{2} r_{ij}^{-8}) * \vec{r}_{ij} \tag{17}$$

  - **Bond-angle forces**: A change in the bond-angle (τ_i) produces forces on three neighbor residues $j = i-2, i-1, i$ given by:

$$-\nabla_{r_j} u(\tau_i) = -\tfrac{du(\tau)}{d(cos\tau)}\Big|_{\tau=\tau_i} f_j^{(i)} \tag{18}$$

    where $u(\tau_i)$ is the angle potential and $f_j^{(i)} = \nabla_{r_j} cos(\tau_i)$

    Since $\sum_j f_j = 0$, the bond-angle forces can be expressed by, following the notation of (RAPAPORT, 2004):

$$\begin{aligned} f_{i-2}^{(i)} &= (c_{i-1,i-1}c_{ii})^{-1/2}[\vec{b_{i-1}}(c_{i-1,i}/c_{i-1,i-1}) - \vec{b_i}] \\ f_i^{(i)} &= (c_{i-1,i-1}c_{ii})^{-1/2}[\vec{b_{i-1}} - \vec{b_i}(c_{i-1,i}/c_{ii})] \end{aligned} \tag{19}$$

    Such that $c_{i,j}$ represents the scalar product of the ith and the jth bond vectors and it is represented by the vector $c_{i,j} = \vec{b_i} \cdot \vec{b_j}$.

The potential associated with the bond angles for the AB protein model (E_{Angles}) is shown in Equation 2. This equation can be written in cosine form because the AB model uses unit-length bonds, as follows:

$$u(\tau_i) = -k_1 \hat{b_i} \cdot \hat{b_{i+1}} = -k_1 * cos(\tau_i) \tag{20}$$

and the derivative used for the forces is given by $-\frac{du(\tau)}{d(cos\tau)} = -k_1$.

– **Bond-torsion forces**: The force associated with a torsional degree of freedom is defined in terms of the relative coordinates of four consecutive residues.

The torque caused by a rotation about the ith bond generates forces on four neighbor residues ($j = i-2, ..., i+1$) and it is defined by Equation 18, but replacing the argument τ_i by α_i. Where $u(\alpha_i)$ is the angle potential and $\vec{f}_j^{(i)} = \nabla_{r_j} cos(\alpha_i)$.

Since $\sum_j f_j = 0$, the torsional forces can be expressed by, following the notation of (RAPAPORT, 2004):

$$\begin{aligned}
\vec{f}_{i-1}^{(i)} &= -(1 + c_{i-1,i}/c_{ii})\vec{f}_{i-2}^{(i)} + (c_{i,i+1}/c_{ii})\vec{f}_{i+1}^{(i)} \\
\vec{f}_{i}^{(i)} &= (c_{i-1,i}/c_{ii})\vec{f}_{i-2} - (1 + c_{i,i+1}/c_{ii})\vec{f}_{i+1}^{(i)} \\
\vec{f}_{i-2}^{(i)} &= \frac{c_{ii}}{q_i^{1/2}(c_{i-1,i-1}c_{ii} - c_{i-1,i}^2)}[w_1\vec{b}_{i-1} + w_2\vec{b}_i + w_3\vec{b}_{i+1}] \\
\vec{f}_{i+1}^{(i)} &= \frac{c_{ii}}{q_i^{1/2}(c_{ii}c_{i+1,i+1} - c_{i,i+1}^2)}[w_4\vec{b}_{i-1} + w_5\vec{b}_i + w_6\vec{b}_{i+1}]
\end{aligned} \tag{21}$$

where:

$$\begin{aligned}
w_1 &= c_{i-1,i+1}c_{ii} - c_{i-1,i}c_{i,i+1} \\
w_2 &= c_{i-1,i-1}c_{i,i+1} - c_{i-1,i}c_{i-1,i+1} \\
w_3 &= c_{i-1,i}^2 - c_{i-1,i-1}c_{ii} \\
w_4 &= c_{ii}c_{i+1,i+1} - c_{i,i+1}^2 \\
w_5 &= c_{i-1,i+1}c_{i,i+1} - c_{i-1,i}c_{i+1,i+1} \\
w_6 &= -w_1 \\
q_i &= (c_{i-1,i-1}c_{ii} - c_{i-1,i}^2)(c_{ii}c_{i+1,i+1} - c_{i,i+1}^2)
\end{aligned} \tag{22}$$

The potential associated with torsion for the AB protein model ($E_{torsion}$) is shown in Equation 2. This equation can also be written in cosine form as shown in Equation 20. The derivative used for the forces is given by $-\frac{du(\alpha)}{d(cos\alpha)} = -k_2$.

Further information about bond-angle and bond-torsion forces calculation (with an example of an alkane chain) can be found in (RAPAPORT, 2004).

- **Integrate equations of motion**:
  In this work, we use the velocity-verlet algorithm (SWOPE et al., 1982). The implementation scheme of this algorithm is:

$$\begin{aligned} \vec{r}_i(t+\delta t) &= \vec{r}_i(t) + \vec{v}_i(t)\delta t + \tfrac{1}{2}\vec{a}_i(t)\delta t^2 \\ \vec{v}_i(t+\delta t/2) &= \vec{v}_i(t) + \tfrac{1}{2}\delta t \vec{a}_i(t) \\ \vec{v}_i(t+\delta t) &= \vec{v}_i(t+\delta t/2) + \tfrac{1}{2}\vec{a}_i(t+\delta t)\delta t \end{aligned} \tag{23}$$

  Where, $\vec{r}_i(t)$,$\vec{v}_i(t)$ and $\vec{a}_i(t)$ are the position, velocity and acceleration of the *i*th residue, respectively; t and δt are the time and the timestep.

- **Perform ensemble control**:
  The MD simulation performs the canonical ensemble (also referred to as the ensemble NVT), where the number of particles (residues), the volume and the temperature are controlled at desired values. The temperature is controlled by using the method of weak coupling to a thermal bath proposed by (BERENDSEN et al., 1984). In this approach, coupling removes or adds energy to the system to maintain an approximately constant temperature. The velocities are scaled at each step using the scaling factor α, as follows:

$$\vec{v}_i(t) = \lambda * \vec{v}_i(t) \tag{24}$$

$$\lambda = \sqrt{1 + \tfrac{\delta t}{\tau_T}(\tfrac{T_{sp}}{T} - 1)} \tag{25}$$

  Where λ, τ_T, T_{sp}, T are the scaling factor, the coupling constant, the desired temperature (set-point) and the current temperature, respectively.

- **Compute geometric constraints**:
  As commented before, a protein with the AB model is subject to geometrical constraints due to the fixed unit-length bonds between amino acids ($|\vec{r}_i - \vec{r}_j|^2 = b_i^2 = 1$).

  Considering a protein with N residues, there are a total of $n_c = N - 1$ geometric constraints. In this work, we use the SHAKE algorithm (RYCKAERT; CICCOTTI; BERENDSEN, 1977) to deal with constraints.

  The SHAKE algorithm starts after the system has advanced a single timestep, while ignoring the constraints (RAPAPORT, 2004). Thus, a set of uncorrected coordinates is obtained that are represented by Equation 26.

$$\vec{r}_i'(t+\delta t/2) = 2\vec{r}_i(t) - \vec{r}_i(t-\delta t) + (\delta t/2)^2/m_i \cdot \vec{f}_i(t) \tag{26}$$

Algorithm 3 shows the SHAKE algorithm that has two parts. First, corrections along the direction of $\vec{r}_{ij}(t)$ are done. The estimated coordinates $\vec{r}_i'$ and $\vec{r}_j'$ are updated by using the correction factor γ, which is determined as shown in lines 1 and 8 of the algorithm. Next, velocities are corrected in a similar manner. Here, it is important to recall that m_i and m_j are the masses of the ith and jth amino acids, respectively. As mentioned in Section 3.1 (page 84), AB model does not represent the mass value of residues, therefore $m_i = m_j = 1$. In addition, b_i represents the bond length between the ith and jth amino acids which, as mentioned, are unit-length bonds in the AB model. The process is repeated for both directions and velocity corrections until all the constraints are satisfied.

The precision of the SHAKE algorithm is given by $|\vec{r}_0 - \vec{r}|/|\vec{r}_0| < 10^{-k}$, where 10^{-k} is the desired precision (10^{-6} in this work).

Algorithm 3 SHAKE algorithm

```
 1: **Start**
    Coordinates correction:
 2: $\gamma \leftarrow \frac{\vec{r}_{ij}^{2'} - b_i^2}{4(\delta t/2)^2(1/m_i + 1/m_j)\vec{r}_{ij}' \cdot \vec{r}_{ij}}$
 3: **while** $|\gamma| < 10^{-k} \cdot b_i^2$ **do**
 4:     $\vec{r}_i' \leftarrow \vec{r}_i - \gamma\vec{r}_{ij}$
 5:     $\vec{r}_j' \leftarrow \vec{r}_j + \gamma\vec{r}_{ij}$
 6:     $\gamma \leftarrow \frac{\vec{r}_{ij}^{2'} - b_i^2}{4(\delta t/2)^2(1/m_i + 1/m_j)\vec{r}_{ij}' \cdot \vec{r}_{ij}}$
 7: **end while**
    Velocities correction:
 8: $\gamma = \frac{\dot{\vec{r}}_{ij} \cdot \vec{r}_{ij}}{2\vec{r}_{ij}^2}$
 9: **while** $|\gamma| < 10^{-k} \cdot b_i^2$ **do**
10:     $\dot{\vec{r}}_i' \leftarrow \dot{\vec{r}}_j - \gamma\vec{r}_{ij}$
11:     $\dot{\vec{r}}_j' \leftarrow \dot{\vec{r}}_j + \gamma\vec{r}_{ij}$
12:     $\gamma = \frac{\dot{\vec{r}}_{ij} \cdot \vec{r}_{ij}}{2\vec{r}_{ij}^2}$
13: **end while**
14: **End**
```

- **Compute the desired physical quantities**:

  Besides the total energy (see Equation 2 – page 54) of the conformation, we also compute the radius of gyration (GROSBERG; KHOKHLOV, 1994), that measures the compactness of a set of points (in this case, the residues of the protein). The more compact the set of points, the smaller the radius of gyration is. The radius of gyration is computed as shown in Equation 27.

$$R_g = \sqrt{\frac{\sum_{i=0}^{N-1}[(x_i-\overline{X})^2+(y_i-\overline{Y})^2+(z_i-\overline{Z})^2]}{N}} \tag{27}$$

In this equation, x_i, y_i and z_i are the coordinates of the residues. $\overline{X}$, $\overline{Y}$ and $\overline{Z}$ are the average of all x_i, y_i and z_i; and N is the number of residues.

- General comments:
  - The simulation takes place in a cubic container, using periodic boundary conditions. The periodic boundary conditions are equivalent to considering an infinite array of identical copies of the simulation region (RAPAPORT, 2004). There are two consequences of this periodicity: particles (i.e. amino acids) that leave the simulation region through a particular bounding face immediately reenters the region through the opposite face, and particles lying within a distace of a boundary interact with particles in an adjacent copy of the system (i.e. particles near the opposite boundary). The second consequence is considered to be a wraparound effect. If a particle has moved outside the region, its coordinates are adjusted to bring it inside the simulation region, as shown in Equations 28, 29 and 30.

$$x_i = \begin{cases} x_i - L_x & \text{if } x_i \geq L_x/2 \\ x_i + L_x & \text{otherwise} \end{cases} \tag{28}$$

$$y_i = \begin{cases} y_i - L_y & \text{if } y_i \geq L_y/2 \\ y_i + L_y & \text{otherwise} \end{cases} \tag{29}$$

$$z_i = \begin{cases} z_i - L_z & \text{if } z_i \geq L_z/2 \\ z_i + L_z & \text{otherwise} \end{cases} \tag{30}$$

Where x_i, y_i and z_i represent the Cartesian coordinates of the amino acids; L_x, L_y and L_z are the region size in the x, y and z directions, respectively.

The components of the distance between amino acids are determined in a similar manner, as shown in Equations 31, 32 and 33.

$$(r_{ij})_x = \begin{cases} (r_{ij})_x - L_x & \text{if } (r_{ij})_x \geq L_x/2 \\ (r_{ij})_x + L_x & \text{otherwise} \end{cases} \tag{31}$$

$$(r_{ij})_y = \begin{cases} (r_{ij})_y - L_y & \text{if } (r_{ij})_y \geq L_y/2 \\ (r_{ij})_y + L_y & \text{otherwise} \end{cases} \tag{32}$$

$$(r_{ij})_z = \begin{cases} (r_{ij})_z - L_z & \text{if } (r_{ij})_z \geq L_z/2 \\ (r_{ij})_z + L_z & \text{otherwise} \end{cases} \tag{33}$$

Where $(r_{ij})_x$, $(r_{ij})_y$ and $(r_{ij})_z$ are the components of the distance between the ith and jth amino acids.

– We do not use real physical units because they are not defined for the AB model of proteins. Thus, the energy, temperature and length are shown in reduced (or dimensionless) units.

  It is important to note that in simulations of real molecular systems it is convenient to express physical quantities, such as temperature and pressure, in reduced units, and to use basic units in order to translate them to real units. The basic units depend on experimental data and are: length (σ), energy (ε), mass (m) and temperature (ε/K_B), where K_B is the Boltzmann constant (FRENKEL; SMIT, 2002). Moreover, the main reason for using dimensionless units in simulations with real physical units is related to scaling. Thus, properties that have been measured in dimensionless units can be scaled to the physical units for the problem of interest. From a practical point of view, the use of dimensionless units removes any risk of problems with data representation.

Finally, it is important to mention that two publications describe the MD approach presented in this section: (BENÍTEZ; LOPES, 2012) and (BENÍTEZ; LOPES, 2013).

3.2 A DISTRIBUTED GENE EXPRESSION PROGRAMMING FOR EVOLVING TWO-DIMENSIONAL CELLULAR AUTOMATA APPLIED TO THE CONTACT MAP PREDICTION (pGEP-CA)

In this thesis, the main idea consist in simulating the protein folding process in a macroscopic level, using simple mathematical idealizations, known as Cellular Automata. It evolves progressively according to deterministic rules from a particular initial state, instead using the protein structure coordinates and traditional mathematical methods. For instance, in conventional Molecular Dynamics simulations, numerical integrations of the classical equations of motion and other mathematical methods are used to generate the trajectory of a system. Here, we propose the simulation of the protein folding using Contact Maps (CM) represented by two-dimensional Cellular Automata (2D-CA).

Starting with an initial configuration (initial CM), the next folding states are generated by the application of a *transition rule* (or *folding rule*, in the protein folding context), as shown in Figure 13.

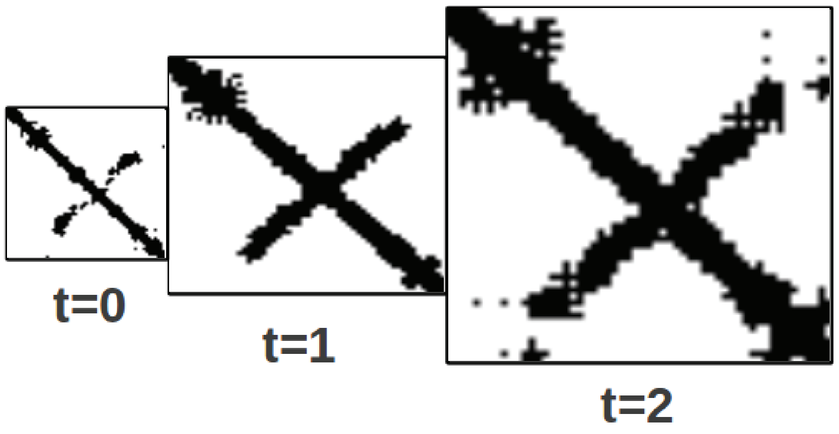


Figure 13: An example of 2D-CA evolution
Source: Own work.

The computation of the Cellular Automata is simple. However, finding *transition rules* for simulating the protein folding is very difficult. Therefore, in this thesis, we propose a novel approach for the induction of *transition rules* (or *folding rules*) of two-dimensional Cellular Automata (2D-CA) applied to the protein folding simulation.

It is important to recall that the simulation of the protein folding using 2D-CA is indendepent of the computational model used to represent protein structures, such as the 3D-AB *off-lattice* model and the models presented in Section 2.4. On the other hand, the procedure for finding *transition rules* uses two CMs previously generated from protein structures.

Below, we present a detailed description of a novel parallel approach for the induction of *transition rules* of two-dimensional Cellular Automata (2D-CA), using Gene Expression Programming, applied to the Protein Contact Maps prediction and folding pathway simulation, called pGEP-CA.

Algorithm 4 shows the pseudo-code of the pGEP-CA. A parallel master-slave architecture, described in Section 3.2.5, is employed in order to speed up processing time. In this algorithm, bold instructions are processed in parallel.

The software was developed in ANSI-C programming language, using the Message Passing Interface (MPI) for communication between processes [1] and the Mersenne Twister random number generator (MATSUMOTO; NISHIMURA, 1998).

Algorithm 4 Pseudo-code for parallel GEP-CA (pGEP-CA)

```
 1: Start
 2: Initialize population;
 3: Determine ORFs
 4: do parallel
 5:     Simulate 2D-CAs
 6:     Evaluate fitness
 7: end do parallel
 8: while stop criteria not satisfied do
 9:     Clone operator (part 1)
10:     Selection
11:     Apply genetic operators
12:     Update population
13:     Determine ORFs
14:     do parallel
15:         Simulate 2D-CAs
16:         Evaluate fitness
17:     end do parallel
18:     Clone operator (part 2)
19: end while
20: Export postprocess results: best transition rule, CM obtained, metrics
21: End
```

3.2.1 CELLULAR AUTOMATA CONFIGURATION

First of all, it is essential to know how the Two-dimensional Cellular Automata (2D-CA) are composed for understanding the proposed approach. The 2D-CA are used to represent Contact Maps (CM) which, in turn, are $N \times N$ binary symmetrical matrices representing the contacts between amino acids of a protein structure with N amino acids. Thus, the 2D-CA is a system formed by identical cells arranged in a two-dimensional cellular space.

Each cell of the 2D-CA ($c_{i,j}$) has two possible states: '0' and '1' represent *contacts* and *non-contacts* between ith and jth amino acids, respectively, as shown in Figure 14(a). Thus, following the formal notation presented by (MITCHELL, 1998) (see Section 2.7), the set of possible states of a cell with two states ($k = 2$) is $\sum = 0, 1$, representing an "elementary" CA.

[1] Available at: http://www.mcs.anl.gov/research/projects/mpich2/

The state of each cell is denoted S_i^t (where $S_i^t \in \Sigma$), and together with the state of the neighbor cells, forms a *neighborhood* (η_i^t). In this approach, it is used the von Neumann *neighborhood* with $r = 1$ and $m = 5$ that comprises the four orthogonally neighboring cells ($c_{i-1,j}$, $c_{i,j-1}$, $c_{i+1,j}$, $c_{i,j+1}$) surrounding a central cell ($c_{i,j}$), as shown in Figure 14(c).

The 2D-CA evolves deterministically in discrete time steps. At each time step (t), the states of all cells of the 2D-CA are updated synchronously according to the *transition rule* $\Phi(\eta_i^t)$ which, in turn, is a function of the *neighborhood* (η_i^t). The application of the *transition rule* is done with periodic boundary conditions (see Figure 14(b)) allowing the connection between cells that are situated at the extremities, forming a toroidal arrangement and avoiding failure. Here, it is important to recall that a real synchronous update of all cells is not possible in the von Neumann architecture. However, a real parallel update can be done using reconfigurable hardware, such as FPGA, in a *non-von Neumann style* as we proposed for one-dimensional CA in a previous work (WEINERT et al., 2007).

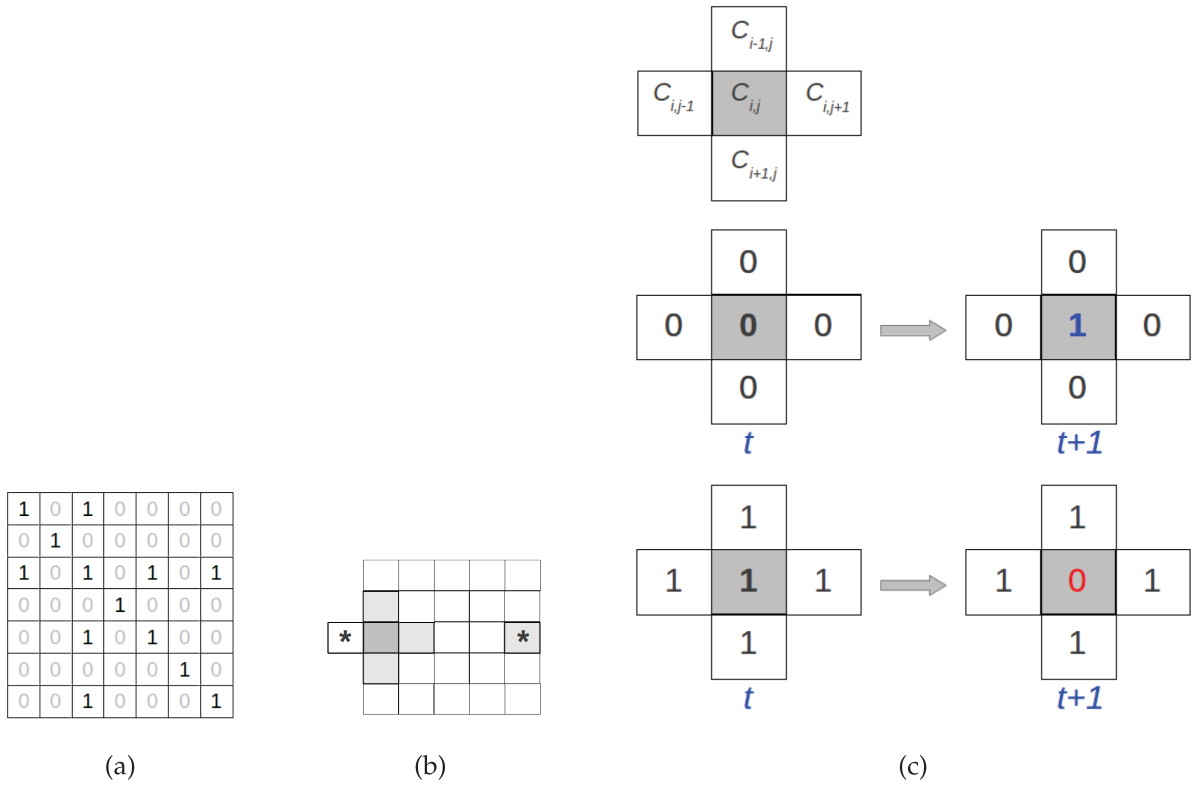


Figure 14: (a) Illustration of a two-dimensional Cellular Automaton (2D-CA), (b) An example of periodic boundary conditions, (c) Examples of transitions with the von Neumann neighborhood

Source: Own work.

According to Section 2.7, a *transition rule* is formed by concatenating all possible transitions, which, in turn, are defined by the possible combinations of the neighboring cells. Thus, there are 32 possible combinations using the von Neumann *neighborhood* with unity radius. For instance, the *transition rule* $010101110\cdots1111101100101_2$ is formed by the transitions, from right to left, ”00000_2“$\rightarrow$1; ”00001_2“$\rightarrow$0; ”00010_2“$\rightarrow$1; $\cdots$; ”11111_2“$\rightarrow$0. Figure 14(c) shows examples of the first and last possible transitions, where the cell to be updated is shaded.

3.2.2 SOLUTION ENCODING

An important issue when using GEP for a given problem is the encoding of the individual that represent a possible solution to the problem. The way variables are encoded can have a strong influence not only in the size of the search space, but also in the dynamics and efficiency of the algorithm. In this work, the encoding of the individuals is defined according to the set of terminals and their domains, following the Pittsburgh approach (FREITAS, 2002), in which each individual represents a arrangement of rules, forming the *transition rule*.

In GEP, the individuals are represented by two multigenic chromosomes, which are composed of more than one gene of equal size. Each gene is divided into a head and a tail. The size of the head (h) and the number of genes of each chromosome can be chosen *a priori*. On the other hand, the size of the tail (t) is determined according to the size of the head as proposed by (WEINERT; LOPES, 2006): $t = \text{IntegerPart}[0.5(h(n-1)+1)]$, where n represents the largest arity (number of arguments) among all the functions used.

Each gene is directly translated into an expression tree (ET). In this work, each chromosome is composed of two genes. Thus, the sub-ETs codified by the genes are linked together by a logical function (*AND* or *OR*), which can be chosen *a priori*.

The terminals are binary and represent the state of the neighboring cells (’1’=*contact*, ’0’=*non-contact*). The set of terminals (T) represent all possible combinations of the neighborhood, mapping all transitions of a given rule. Considering the von Neumann neighborhood with $r = 1$, the terminal set is composed of five terminals (labeled as a, b, c, d and e), where the central cell ($c_{i,j}$) is represented by c and the neighboring cells $c_{i-1,j}$, $c_{i,j-1}$, $c_{i+1,j}$ and $c_{i,j+1}$ are represented by a, b, e and d, respectively. The terminals have the same domain, which in turn, are defined by the possible states of the cells of the 2D-CA ($\Sigma = [0;1]$).

A possible transition encoded in an individual is written in the form *IF A THEN C* as in data classification systems, for example: *IF (a* $= 1$ *AND b* $= 0$*) THEN Rule* = *'1'; else Rule* = *'0'*.

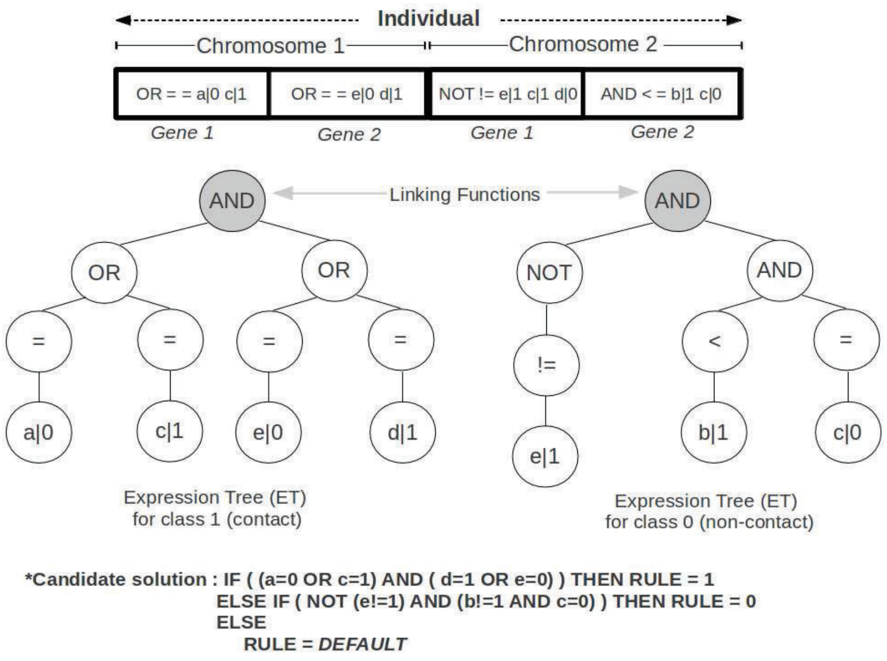

Individual
Chromosome 1
Chromosome 2
OR = = a|0 c|1
OR = = e|0 d|1
NOT != e|1 c|1 d|0
AND < = b|1 c|0
Gene 1
Gene 2
Gene 1
Gene 2
AND
Linking Functions
AND
OR
OR
NOT
AND
=
=
=
=
!=
<
=
a|0
c|1
e|0
d|1
e|1
b|1
c|0
Expression Tree (ET) for class 1 (contact)
Expression Tree (ET) for class 0 (non-contact)
*Candidate solution : IF ((a=0 OR c=1) AND (d=1 OR e=0)) THEN RULE = 1
ELSE IF (NOT (e!=1) AND (b!=1 AND c=0)) THEN RULE = 0
ELSE
RULE = DEFAULT

Figure 15: GEP Transcription process – example

Source: Own work.

Figure 15 shows a simplified example of the transcription process. In the proposed approach, a *rule* is generated from the ET represented by each individual. For instance, Table 4 shows the *rule* obtained for each transition from the ET shown in Figure 15.

It is important to recall that, for binary terminals, the relational operators $<$ and $>$ are equal to $\neq$ and $=$, respectively. The $DEFAULT$ value is user-defined ('0' or '1'). The example shown in Table 4 was done using $DEFAULT = 0$.

Finally, the complete *transition rule* is obtained from the Rule Table, shown in Table 4, concatenating the *rules* obtained. Thus, *transition rule* = $110\cdots101$.

Table 4: Rule table obtained from the ET

Combination	*Neighborhood* *a b c d e*	*Rule*
0	00000	1
2	00001	0
3	00010	1
...	...	...
29	11101	0
30	11110	1
31	11111	1

Source: Own work.

3.2.3 FITNESS FUNCTION

Contact maps (CMs) are generally sparse symmetric matrices, populated primarily with *non-contacts* (or zeros). Therefore, a similarity measure between two CMs based on the Hamming distance (PETERSON; WELDON, 1972) or Euclidean distance, that are the most common metrics, does not work well for CMs, because *contacts* (true) and *non-contacts* (false) values carry the same weight. For instance, a wrong choice would be using a fitness function based on the Hamming distance, shown in Equation 34. Figures 16(a) and 16(b) show, respectively, the expected and obtained CMs of a simulation using the fitness function based on the Hamming distance. In this example, the fitness value of the individual that represents the rule used to generate the CM (Figure 16(b)) is 0.9547, representing an accuracy of 95.47%. However, it is possible to observe that the obtained CM lacks *contacts*, when compared with the expected CM.

$$fitness_{Hamming} = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N}[1-|(C1_{i,j}-C2_{j,i})|]}{N^2} \tag{34}$$

where: N is the number of amino acids; C_1 and C_2 are the expected and obtained CMs.

Ertoz, Steinbach and Kumar (2002) stated, in their publication about a new clustering algorithm, that the presence of an attribute is a lot more important than the absence of an attribute in sparse high dimensional data sets. Thus, different measures

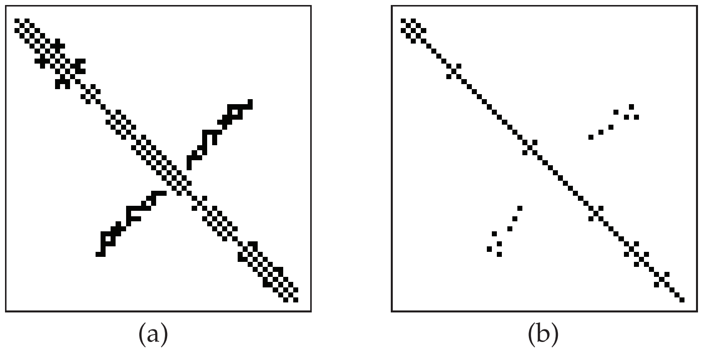


Figure 16: (a) Expected and (b) obtained CMs, using the fitness function based on the Hamming distance

Source: Own work.

could be proposed in order to give more importance to the presence of *contacts* of CMs.

Here, the challenge is to properly write a suitable fitness function for measuring the ability of a *transition rule* to generate a CA that represents a CM correctly. In this thesis, two novel fitness functions were proposed, called $fitness_1$ and $fitness_2$, that are better suited to this problem, as follows:

(a) The fitness function $fitness_1$ is based on three metrics, shown in Equation 35:

$$fitness_1 = S_C * S_{NC} * S_i^2 \tag{35}$$

where: S_C, S_{NC} are based on the sensitivity and specificity measures, respectively. Sensitivity and specificity are commonly used in classification systems. Sensitivity measures the ability of the classifier to correctly assign a data to its real class. On the other hand, specificity measures the ability to reject a given data as belonging to a class to which it does not belong. In this work, S_C and S_{NC} measure the ability of a *transition rule* to generate correct *contacts* and *non-contacts*, respectively. S_C and S_{NC} are defined following four types of results, shown in Equations 36 and 37, respectively. S_i measures the symmetry of the CM, as shown in Equation 38, where m and n are the number of rows and columns of the CM, respectively.

$$S_C = \frac{T_C}{T_C + F_{NC}} \quad (36) \qquad S_{NC} = \frac{T_{NC}}{F_C + T_{NC}} \quad (37) \qquad S_i = \sum_{i=0}^{i<m-1} \sum_{j=i+1}^{j<n} [1 - |(C_{i,j} - C_{j,i})|] \quad (38)$$

where:

- **True contacts (T_C):** number of contacts generated by the *transition rule* that, in fact, are contacts;
- **True non-contacts (T_{NC}):** number of non-contacts generated by the *transition rule* that, in fact, are non-contacts;
- **False contacts (F_C):** it counts the contacts generated by the *transition rule* that, in fact, are non-contacts;
- **False non-contacts (F_{NC}):** it counts the non-contacts generated by the *transition rule* that, in fact, are contacts;
- $C_{i,j}$ represents the value at cell location (i, j) of the CM ($m \times n$ matrix).

(b) The fitness function $fitness_2$ is based on the Jaccard similarity coefficient (JACCARD, 1908). Basically, the Jaccard similarity coefficient (J) is used to measure the similarity between two sample sets (A and B). For instance, Equation 39 shows the Jaccard similarity coefficient for binary sample sets.

$$J(A,B) = \frac{A \bigcap B}{A \bigcup B} = \frac{M_{11}}{M_{01} + M_{10} + M_{11}} \tag{39}$$

where: $0 \leq J(A,B) \leq 1$;
M_{11} represents the total number of attributes that are '1' in A and B;
M_{01} represents the total number of attributes that are '0' in A and '1' in B;
M_{10} represents the total number of attributes that are '1' in A and '0' in B;
and, clearly, $J(A,B) = 1$ when A and B are equal.

Considering C_1 and C_2 as the expected and obtained CMs, and the metrics T_C, T_{NC}, F_C and F_{NC}, the Jaccard similarity coefficient $J(C_1, C_2)$ may be rewritten as shown in Equation 40.

$$J(C_1, C_2) = \frac{T_C}{F_{NC} + F_C + T_C} \tag{40}$$

Equation 41 presents the fitness function $fitness_2$.

$$fitness_2 = J * S_i^2 \tag{41}$$

where: J is the Jaccard similarity coefficient, defined as shown in Equation 40;
S_i measures the symmetry of the CM, as shown in Equation 38.

3.2.4 RUNNING PARAMETERS AND GENETIC OPERATORS

The individuals of the initial population are created randomly, according to the functions, terminals, size of the head of each gene and number of genes of the chromosome, specified by the user. We used the Mersenne Twister pseudo-random number generator (MATSUMOTO; NISHIMURA, 1998), which is known as one of the best generators for this purpose.

In GEP, the individuals of the population are selected through a selection method, according to a pre-defined criteria. Then, the genetic operators modify them probabilistically, in order to create new individuals and introduce variation in the population.

It is important to recall that the selection method is not a genetic operator. The proposed approach uses the selection method known as stochastic tournament, that is a succesful method used in Genetic Algorithms and Swarm Intelligence algorithms. The stochastic tournament selection is not based on the competition among individuals of the whole population. This method chooses a number of individuals from the current population (the number of individuals chosen is known as *tourney size*). Then, the best suited individual of the group is selected.

The *crossover* operator is the first genetic operator to be applied, according to a probability (p_{cross}), to the individuals which were previously selected by the selection method. This operator swaps the genetic material between the heads of two parent chromosomes, that are randomly chosen. Basically, the one-point recombination is used, where the heads of the parents (A and B) are crossed over at a randomly chosen point (known as the *crossover point*). Finally, if the functions have the same arity, the *crossover* is done and the parents are cut at the *crossover point*, and exchange the material between them downstream from the *crossover point*, forming the offspring. It is important to know that the *crossover* operator does not change the syntactic structure of the chromosomes.

Basically, the *mutation* operator introduces genetic diversity to the population. After the application (or not) of the *crossover* operator, each recently created individual is subject to the mutation operator, according to a mutation probability (p_{mut}).

Usually, mutations can occur anywhere in the chromosome. However, in this approach, the mutation operator works as in GEPCLASS (WEINERT; LOPES, 2006), in three different levels. When a function is selected for mutation, it will be substituted by another function of the same type. For instance, when a logical function is selected for mutation, it will be substituted by another logical function. At the leaf level of the ET, when a terminal is selected for mutation, it will be randomly substituted by a new attribute. The mutation operator is inoperative in a special situation: when a NOT function is selected for mutation, since it is the only *unary* operator and cannot be substituted by another *n*-ary logical operator.

The transposable elements of GEP are the fragments of the genome that can be activated and jump to another place within the chromosome (FERREIRA, 2001). Both the IS (Insertion Sequence) and RIS (Root IS) transposition operators work over a single chromosome. On the one hand, the IS transposition modifies only the tail of a gene, changing the position of the terminals within the gene. The IS transposition operator can move genetic material from one gene to another or within the gene. On the other hand, in RIS transposition, the donor and the receptor sites are always in the tail of a gene and the first terminal of the same gene, respectively.

The Gene transposition operator, basically, transposes an entire gene of a chromossome. It is important to recall that, as a result of the gene transposition, a new individual may have duplicated genes. Lynch and Conery (2000) stated that the duplication of genes plays an important role in evolution. Thus, individuals with duplicated genes may appear during the process and contribute to the evolution of the population.

The cloning operator is a simple elitist mechanism, where the best individual of each generation is maintained. If enabled, the clone operator (part 1) save a copy of the best individual of the current generation. After the application of the genetic operators, the clone operator (part 2) substitute the worst individual of the next population by the copy.

3.2.5 PARALLELIZATION IN A BEOWULF CLUSTER

This section presents a detailed description of the parallelism of the pGEP-CA implemented in a Beowulf Cluster, following the methodology proposed by Roosta (1999) (see Section 2.9.2).

The software was developed in ANSI-C programming language, using the Message Passing Interface (MPI) for communication between processes.

A Beowulf cluster is composed by several processing nodes. In this work, each processing node has four processing cores, each one running a single process.

The processes are integrated virtually, using message passing mechanisms.

Each process is identified with a sequencial number, called *rank*, and the group of connected processes is known as *communicator*. Using the Message Passing Interface (MPI), a copy of the program is sent automatically to each process. In other words, the processes are independent and execute the same algorithm, simultanously. The MPI is responsible for managing the messages between processes through the interconnection network. However, the MPI does not provide a mechanism for parallel task division and coordination. Hence, the developer is responsible for specifying the function (or task) of each process as well as the communication protocol between processes.

Figure 17 shows the parallel architecture implemented, which is a synchronous master-slave. This model is a global system with a single population. The master process divides the processing load into several slave processes which, in turn, run the algorithm in different processors.

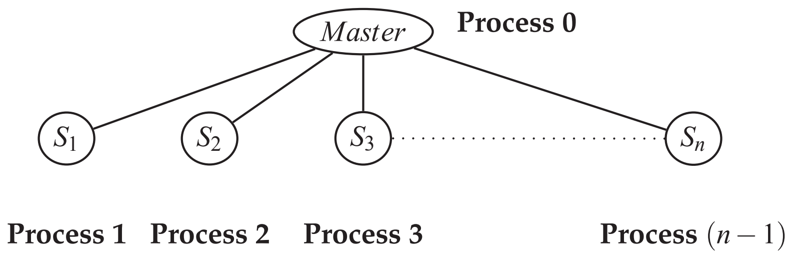


Figure 17: pGEP-CA master-slave architecture

Source: Own work.

Figures 18 and 19 present the statechart of the master and slave processes, respectively.

The master is responsible for initializing the population, determining the ORFs, performing the selection procedure, applying the operators (clone operator, mutation, crossover, IS (insertion sequence) transposition, RIS (root IS) transposition and genic transposition) and distributing individuals to the slaves.

Slaves, in turn, are responsible for reading the initial and final 2D-CAs, simulating the 2D-CAs from the initial 2D-CA, using the induced rules and computing the fitness function of each individual received, using the final (expected) 2D-CAs and the obtained 2D-CAs.

The distribution of individuals by the master processor to the slaves starts with

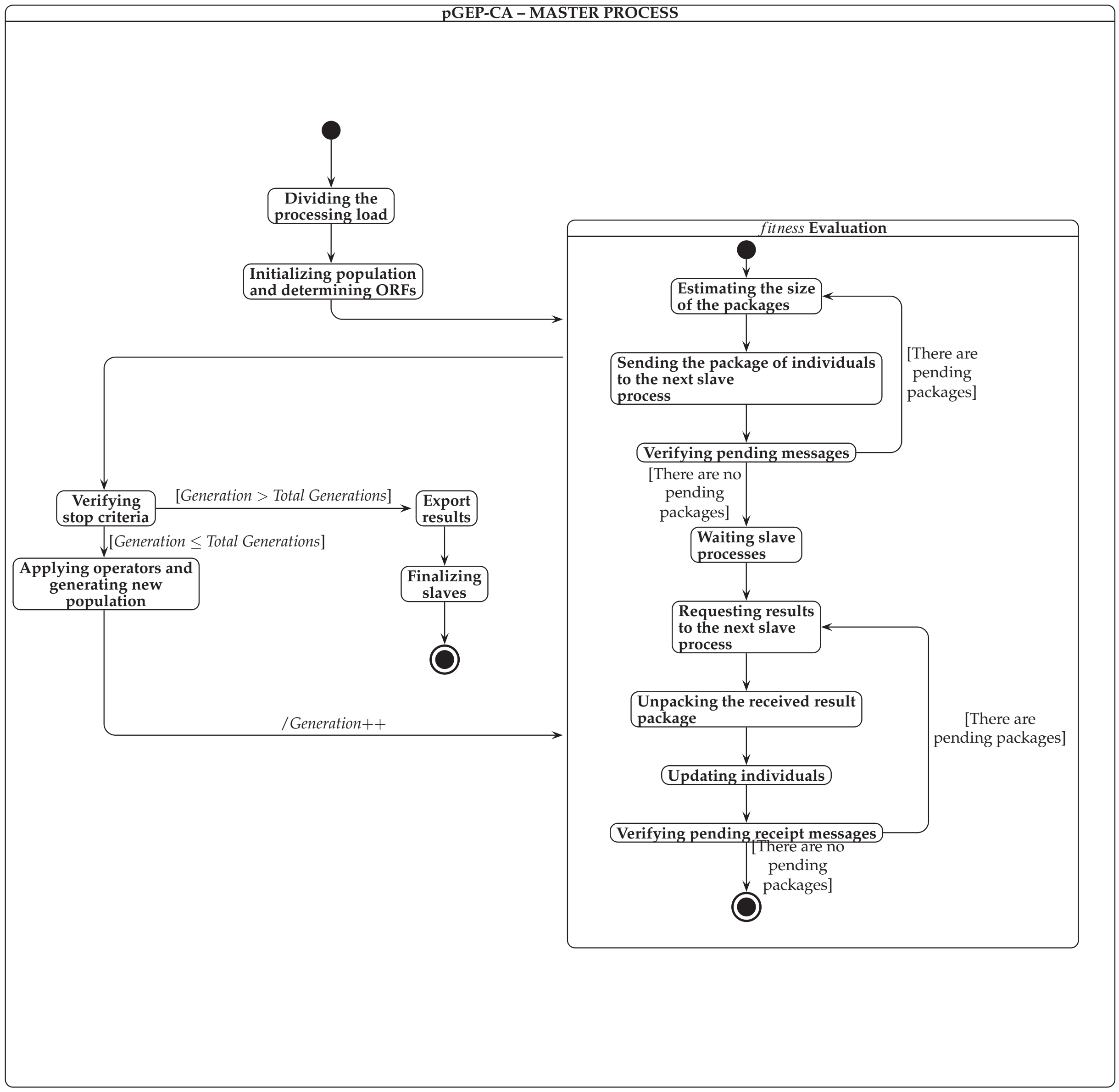

pGEP-CA – MASTER PROCESS
Dividing the processing load
Initializing population and determining ORFs
fitness Evaluation
Estimating the size of the packages
Sending the package of individuals to the next slave process
[There are pending packages]
Verifying pending messages
[There are no pending packages]
Waiting slave processes
Requesting results to the next slave process
Unpacking the received result package
Updating individuals
Verifying pending receipt messages
[There are pending packages]
[There are no pending packages]
Verifying stop criteria
[Generation > Total Generations]
Export results
Finalizing slaves
[Generation ≤ Total Generations]
Applying operators and generating new population
/Generation++

Figure 18: Statechart of the master (pGEP-CA)

Source: Own work

the load balancing, according to the number of individuals to be evaluated and the number of slaves available. Next, the size of the package of individuals to be sent to each slave is estimated. Each package is composed by a number of individuals (chromosomes) with the size of their respective ORF (i.e. the size of the coding sequence of each gene of the individual), shown in Figure 20.

The packages are sent to the slaves in the next steps, using a blocking communication (see Annex C), as follows: first, a simple *Handshaking* between the master and the slave is done. Next, the master sends the number of individuals and the size of

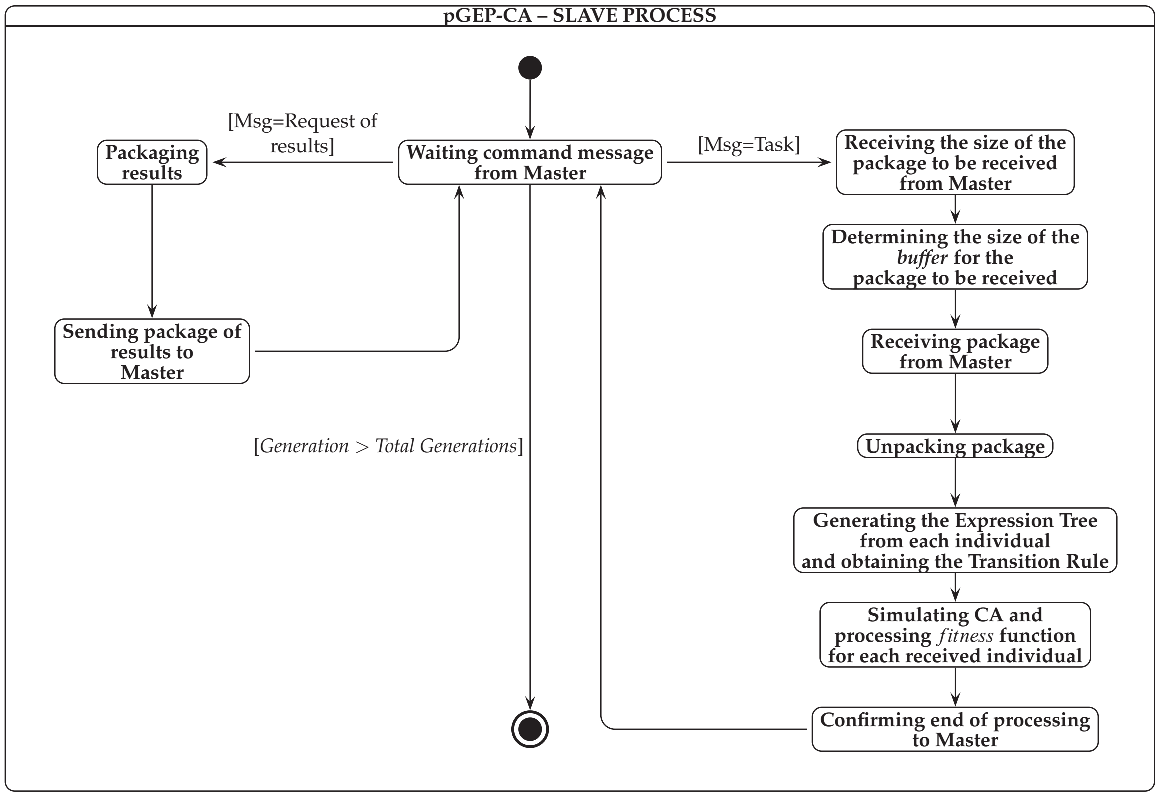

pGEP-CA – SLAVE PROCESS
[Msg=Request of results]
Packaging results
Waiting command message from Master
[Msg=Task]
Receiving the size of the package to be received from Master
Determining the size of the buffer for the package to be received
Sending package of results to Master
Receiving package from Master
[Generation > Total Generations]
Unpacking package
Generating the Expression Tree from each individual and obtaining the Transition Rule
Simulating CA and processing fitness function for each received individual
Confirming end of processing to Master

Figure 19: Statechart of the slaves (pGEP-CA)

Source: Own work.

the package to be sent. After that, the package is sent to the slave. Finally, the master waits to a confirmation message from the slave (with the number of individuals that the slave received before).

The message sent by the master is composed by two parts:

- *Message envelope*: it represents the information that is used to distinguish messages and selectively receive them. It consists of a fixed number of fields: source (master) rank, the destination (slave) rank, the label of the message (*tag*) and the communicator of the group of processes;
- *Data*: it represents the package of individuals, as shown in Figure 20.

After sending all packages to the slaves, the master requests the results (i.e. *fitness* value of each individual) from the slaves, starting with a simple *handshaking* protocol. Next, each slave determines and sends the size of the package, followed by the package with the fitness value of a number of individuals. Finally, the master sends a confirmation message (with the number of results that the slave sent before)

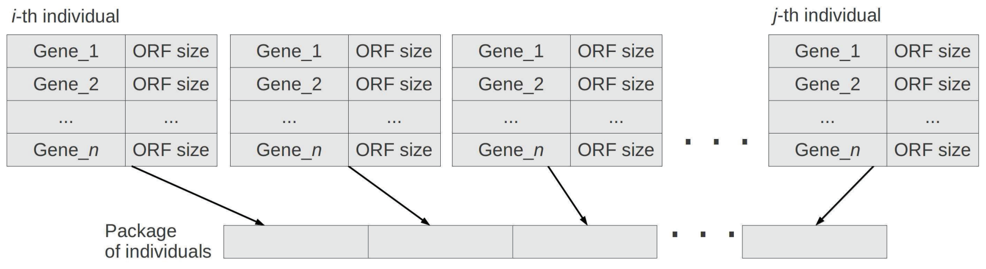


Figure 20: Package of individuals – example

Source: Own work.

and proceeds to unpack the package and update the fitness of each related individual. The packages sent by the slaves are composed by the fitness value of the received individuals, shown in Figure 21.

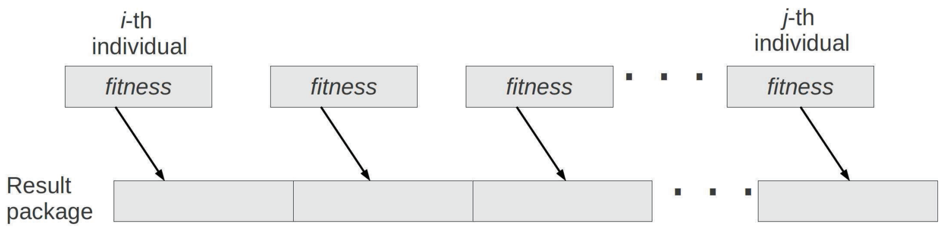


Figure 21: Package of results – example

Source: Own work.

The message sent by the slaves is composed by two parts:

- *Message envelope*: source (slave) rank, the destination (master) rank, the label of the message (*tag*) and the communicator of the group of processes;
- *Data*: it represents the package of results, shown in Figure 21.

This approach can be very efficient when the communication overload is negligible compared with the processing time of the algorithm (including the computation of the fitness function, initialization, among others).

Finally, it is important to mention that the proposed approach presented in this section was published in (BENÍTEZ; WEINERT; LOPES, 2015).

3.3 AN ECOLOGICALLY-INSPIRED APPROACH APPLIED TO THE PROTEIN STRUCTURE RECONSTRUCTION FROM CONTACT MAPS

In (PARPINELLI; LOPES, 2012a), the potentiality of some ecological concepts are illustrated presenting an ecology-inspired algorithm (ECO) for optimization. In order to explore different search strategies cooperatively in an ecologically inspired context, a novel parallel and heterogeneous application of ECO (called pECO) was developed. pECO was applied to a hard optimization problem from bioinformatics: the protein structure reconstruction from Contact Maps. Basically, the aim is to find a low energy conformation, using the Contact Map as a guide.

Different search strategies can compose a computational ecosystem. Previous works (BENÍTEZ; LOPES, 2010) and (BENÍTEZ; PARPINELLI; LOPES, 2012) showed that heterogeneous search strategies working cooperatively can perform better than using a single one. Therefore, in this work, four population-based approaches are employed cooperatively: the Artificial Bee Colony algorithm (ABC) (KARABOGA; BASTURK, 2008), Particle Swarm Optimization algorithm (PSO) (KENNEDY; EBERHART, 1995), Differential Evolution (DE) (STORN; PRICE, 1997), and a hybrid Differential Evolution / Biogeography-based Optimization algorithm (jDE-BBO) (GONG; CAI; LING, 2010).

Since the problem demands a lot of computational effort, a parallel architecture is employed to enable the application of the computational ecosystem in a reasonable computing time.

3.3.1 IMPLEMENTATION OF pECO

In the novel parallel version of ECO (named pECO), proposed in this thesis, the processing load is divided into several processors (master and slaves), under the coordination of a master processor. Each processor (master or slave) is responsible for initializing the population, and performing the evolutive period of a population independently. The master processor is also responsible for defining the communication topologies between populations and habitats. Figure 22 shows the pECO master-slave topology, where each species represents a population-based approach.

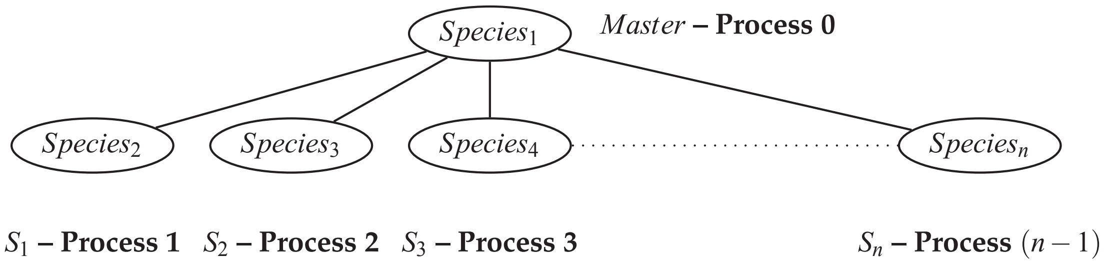


Figure 22: pECO architecture

Source: Own work.

Algorithm 5 presents the pseudo-code of the pECO algorithm, where the italic instructions are processed by each processor (master or slaves) while bold instructions are processed by the master processor. In the first step, line 3, each processor initializes a population Q_i. In line 4, the master coordinates the ecological succession loop. The first step inside this loop is the evolutive period that is carried by the processors (line 5).

As in nature, the populations can move through all the environment, interacting to each other and evolving over time. In other words, symbiosis may occur, where an individual of a population is directly affected by another individual which, in turn, might be a member of another population. In this approach, two types of ecological communication topologies are defined: the intra-habitat and the inter-habitat. On one hand, communications between populations inside each habitat take place in intra-habitat topologies. On the other hand, in inter-habitat topologies, communications between habitats take place. After the evolutive period of all populations, the region of reference of each population is determined in order to re-define the habitats of the system. The region of reference is defined by the centroid ($\vec{C_i}$), and it is calculated by Equation 42, as proposed by (PARPINELLI; LOPES, 2015).

$$\vec{C_i} = \frac{\sum_{k=1}^{POP_i} x_k}{POP_i} \tag{42}$$

where: POP_i represents the size of the ith population; x_k is the kth solution vector (as defined in next section) of the ith population.

In line 6, each processor calculates the centroids and, then, the slaves send them to the master. Next, the master creates the habitats and defines the intra-habitats communication topology, lines 7 and 8 respectively. It is based on the Euclidean distance between the centroids, according to the minimimum threshold distance ρ (PARPINELLI; LOPES, 2015). An adjacency matrix is generated, where a pair of populations are adjacent if their centroids are closer than the threshold distance ρ. After that, the habitats are defined according to the adjacency matrix. The intra-habitat communication topology is also defined from the adjacency matrix, where a group of populations can establish a relationship within the habitat. The ecosystem may have several habitats H_j (with $j = 1, \cdots, NH$), where NH represents the number of habitats of the ecosystem. Each habitat (H_j) has an intra-habitat communication topology $TC_j(t)$ that indicates the populations of the habitat that will be able to communicate to each other at time t.

Algorithm 5 Pseudo-code for parallel ECO (pECO)

```
1:  Start
2:  Let $i = 1, \ldots, NQ, j = 1, \ldots, NH$ and $t = 0$;
3:  Initialize each population $Q_i(t)$ with $POP_i$ random candidate solutions;
4:  while stop criteria not satisfied do {Ecological succession cycles}
5:      Perform evolutive period for each population $Q_i(t)$;
6:      Idendify the region of reference $\vec{C}_i$ for each population $Q_i(t)$;
7:      define_Habitats()
8:      For each habitat $H_j(t)$ define the communication topology $TC_j(t)$ between populations $Q_i^j(t)$;
9:      interactions_C()
10:     Define communication topology $TH(t)$ between $H_j(t)$ habitats;
11:     interactions_H()
12:     $t \leftarrow t + 1$;
13: end while
14: End
```

Once defined the intra-habitat communication topologies, the master coordinates the mating process, requesting the best individual of each species and sending it to an adjacent species. Next, each species replaces a randomly selected individual by a new individual, which in turn, is generated through a genetic exchange between the individual received and an individual chosen using the stochastic tournament strategy (line 9).

Once the interactions between populations are over, the master defines the inter-habitats communication topology $TH(t)$ (line 10). Basically, migrations between habitats take place, where a population of each habitat is chosen randomly. Next, the best individual of each chosen population migrates to another habitat, replacing an individual chosen at random (excluding the best individual).

The master coordinates the migration process between habitats, requesting the best individual of a randomly chosen population from each habitat and sending it to another one of an habitat also randomly chosen (line 11). Finally, the ecological succession loop restarts.

3.3.1.1 ENCODING OF CANDIDATE SOLUTIONS AND INITIAL POPULATION

An important issue when using population-based approaches for a given problem is the encoding of the candidate solutions. The encoding has a strong influence not only in the size of the search space, but also in the complexity of the problem, due to the presence of unknown epistasis between variables that form the individuals (solution vectors). In this work, a given conformation of the protein is represented as a set of bond rotation and torsion angles over a three-dimensional space (see Section 2.4.2.3).

Considering the folding of a protein with N amino acids, an individual has $(2N-5)$ variables, such that positions P_1 to P_{N-2} represent the bond rotation angles (τ_i), and P_{N-1} to P_{2N-5} represent the torsion angles (α_i), where τ and $\alpha \in [-\pi, \pi]$. Figure 23 shows an example of an individual that represents the structure of a folded protein with 13 amino acids.

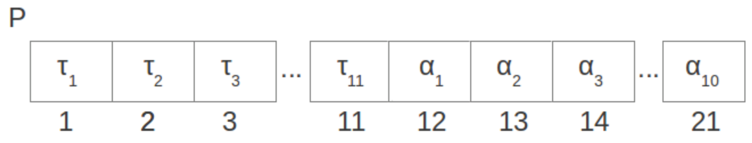


Figure 23: Example of individual

Source: Own work.

The initial population (or swarm) is randomly generated by using the Mersenne Twister algorithm (MATSUMOTO; NISHIMURA, 1998), which is known as one of the best generators for this purpose.

3.3.1.2 FITNESS FUNCTION

An individual is represented by a single vector and it is evaluated and its fitness value represents how good the solution (represented by the individual) is. The fitness function proposed here is based on the energy equation of the model of proteins and the metrics S_C and S_{NC} (previously presented in Section 3.2.3). Equation 43 shows the objective function, using the 3D-AB model (for further information, see Section 2.4.2.3).

$$fitness = E(\hat{b_i};\sigma) * S_C * S_{NC} \tag{43}$$

where:
$E(\hat{b_i};\sigma)$ represents the energy equation of the model (see Equation 2 – page 54). The energy is calculated using the coordinates of the amino acids that compose the structure of the protein which, in turn, is represented by the individual. S_C and S_{NC} are computed using the input CM and the CM of the obtained conformation. S_C measures the similarity between both CMs from the *contact* point of view (see Equation 36). On the other hand, S_{NC} measures the similarity between both CMs from the *non-contact* point of view (see Equation 37).

It is important to know that a given conformation under evaluation may have collisions between amino acids. Obviously, such conformation is physically invalid, but, anyway, the corresponding individual can carry some promising information and should not be disposed.

Algorithm 6 shows the pseudo-code of the objective function calculation procedure. Basically, this procedure has five parts: conversion of angles to Cartesian coordinates, computation of the energy, conversion of the Cartesian coordinates to the CM representation, computation of the metrics S_C and S_{NC} and computation of the $fitness$.

It is important to recall that the conversion of the Cartesian coordinates to the CM representation is done according to the same threshold value of the input CM.

Algorithm 7 shows the pseudo-code of the angles (θ_i and α_i) to Cartesian coordinates conversion procedure.

This procedure has four steps:

- *translate2origin()*: in this step, the $(i-2)^{th}$ and $(i-1)^{th}$ amino acids are translated relative to the origin, using the Equation 44 (PAUL, 1981).

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{44}$$

where,
x,y,z are the initial coordinates (before the translation);

x',y',z' are the final coordinates (after the translation);

d_x, d_y, d_z represent the translation vector, which are: $d_x = 0 - x$; $d_y = 0 - y$; $d_z = 0 - z$

- *Zaxis_rotation()*: in this step, a z-axis rotation is applied to the $(i-2)^{th}$ and $(i-1)^{th}$ amino acids in order to translate them to the xy-plan. The rotation is applied using the rotation angle ($\theta = \tau$) represented by the variable that is being evaluated, using the rotation matrix (called R) (PAUL, 1981), resulting in the point A'_{i-1}, as shown in Equation 45.

Algorithm 6 Fitness function calculation procedure

```
 1: Start
    Let $n$ be the protein size (number of amino acids)
    Let $v$ be the set of bond rotation and torsion angles $(\theta_i, \alpha_i)$
    Let $Objfunc$ be the objective function
    Let $amino$ be the structure that represents the amino acids' cartesian coordinates
    vector and types ('A' or 'B')
    Let $d$ be the Euclidean distance between amino acids
    Let $E_{Angles}$, $E_{torsion}$ and $E_{LJ}$ be the energy terms
 2: Angles2Cartesian() //angles to cartesian coordinates conversion
 3: Compute Energy:
 4: $E_{Angles} = E_{torsion} = E_{LJ} = 0$
 5: for $i = 1 \rightarrow n-1$ do
 6:     $E_{Angles} = E_{Angles} + calcBondAngleEnergy(amino[i].coord, amino[i-1].coord)$
 7: end for
 8: for $i = 1 \rightarrow n-2$ do
 9:     $E_{torsion} = E_{torsion} + calcTorsionEnergy(amino[i].coord, amino[i-1].coord)$
10: end for
11: for $i = 1 \rightarrow n-2$ do
12:     for $j = (i+2) \rightarrow n$ do
13:         if ($amino[i].type$ = 'A' AND $amino[j].type$ = 'A') then
14:             $c = 1$
15:         else
16:             $c = 0.5$
17:         end if
18:         $d = calcDistance(amino[i].coord, amino[j].coord)$
19:         $E_{LJ} = E_{LJ} + calcLJpotential(d, c)$
20:     end for
21: end for
22: $Energy = E_{Angles} + E_{torsion} + E_{LJ}$
23: Cartesian2CM() //cartesian coordinates to CM conversion
24: Compute $S_C$ and $S_{NC}$
25: $fitness = Energy * S_C * S_{NC}$
26: End
```

Algorithm 7 Polar to Cartesian coordinates conversion procedure – *Angles2Cartesian()*

```
1:  Start
    Let $n$ be the protein size (number of amino acids)
2:  $amino[0].coord \leftarrow [0,0,0]$
3:  $amino[1].coord \leftarrow [0,1,0]$
4:  for $i = 2 \rightarrow n$ do
5:      translate2origin(i)
6:      Zaxis_rotation(i)
7:      Yaxis_rotation(i)
8:      translate()
9:  end for
10: End
```

$$A'_{i-1} = R(\theta)A_{i-1} \tag{45}$$

Given a unit vector $u = (u_x, u_y, u_z)$, where $u_x^2 + u_y^2 + u_z^2 = 1$, the matrix for a rotation by an angle of θ about an axis in the direction of u is given by Equation 46.

$$R(\theta) = \begin{bmatrix} \cos\theta + u_x^2(1-\cos\theta) & u_x u_y(1-\cos\theta) - u_z\sin\theta & u_x u_z(1-\cos\theta) + u_y\sin\theta \\ u_y u_x(1-\cos\theta) + u_z\sin\theta & \cos\theta + u_y^2(1-\cos\theta) & u_y u_z(1-\cos\theta) - u_x\sin\theta \\ u_z u_x(1-\cos\theta) - u_y\sin\theta & u_z u_y(1-\cos\theta) + u_x\sin\theta & \cos\theta + u_z^2(1-\cos\theta) \end{bmatrix} \tag{46}$$

- *Yaxis_rotation()*: in this step, the torsion is applied through a y-axis rotation of the A'_{i-1}, using the torsion angle $\theta = \alpha$ and the rotation matrix MR, resulting in the point A'_i.

- *translate()*: rotations related to the origin are applied in the previous steps. Therefore, a translation of the point A'_i is required in order to obtain the coordinates of the ith amino acid. This is accomplished by adding the value of the coordinates of the $(i-1)^{th}$ amino acid to the A'_i coordinates.

Algorithm 8 shows the procedure for obtaining the CM of the conformation from Cartesian coordinates.

3.3.1.3 PARALLELIZATION IN A BEOWULF CLUSTER

The parallelism implemented in pECO is similar to the pGEP-CA (presented in Section 3.2.5), following the methodology proposed by (ROOSTA, 1999). In pGEP-

Algorithm 8 Cartesian coordinates to Contact Map conversion procedure – *Cartesian2CM()*

```
1:  Start
    Let $n$ be the protein size (number of amino acids)
2:  for $i = 1 \to n$ do
3:    for $j = 1 \to n$ do
4:      $d = calcDistance(amino[i].coord, amino[j].coord)$
5:      if $d \leq threshold$ then
6:        $CM[i][j] \leftarrow' 1'$
7:      else
8:        $CM[i][j] \leftarrow' 0'$
9:      end if
10:   end for
11: end for
12: End
```

CA, the slaves are used only for simulating the CA and processing the fitness function. On the other hand, in pECO, the slaves also perform the evolutive period of a population-based algorithm, such as the ABC algorithm. In pGEP-CA, the master performs the GEP algorithm, distributes individuals to the slaves and exports postprocess results. On the other hand, in pECO, the master also performs the evolutive period of a population-based algorithm, defines the habitats and manages the migration processes between them, in intra-habitat and inter-habitat communications. Basically, these are the main differences between pGEP-CA and pECO from the parallelism point of view.

Figure 24 presents the simplified statechart of the processes.

The message sent by the processors to the master, before the definition of the habitats, is composed by two parts:

- *Message envelope*: it represents the information that is used to distinguish messages and selectively receive them. It consists of a fixed number of fields: source (slave) rank, the destination (master) rank, the label of the message (*tag*) and the communicator of the group of processes;
- *Data*: it represents the package with the emigrant individual, as shown in Figure 25. This package also contains the fitness of the received individual (that is the best individual of the population from it was sent), the centroids of the variables and the fitness value of the centroid of the population, which are used during the definition of the habitats and in postprocess results exportation.

The message sent by the master to the processors, after the definition of the

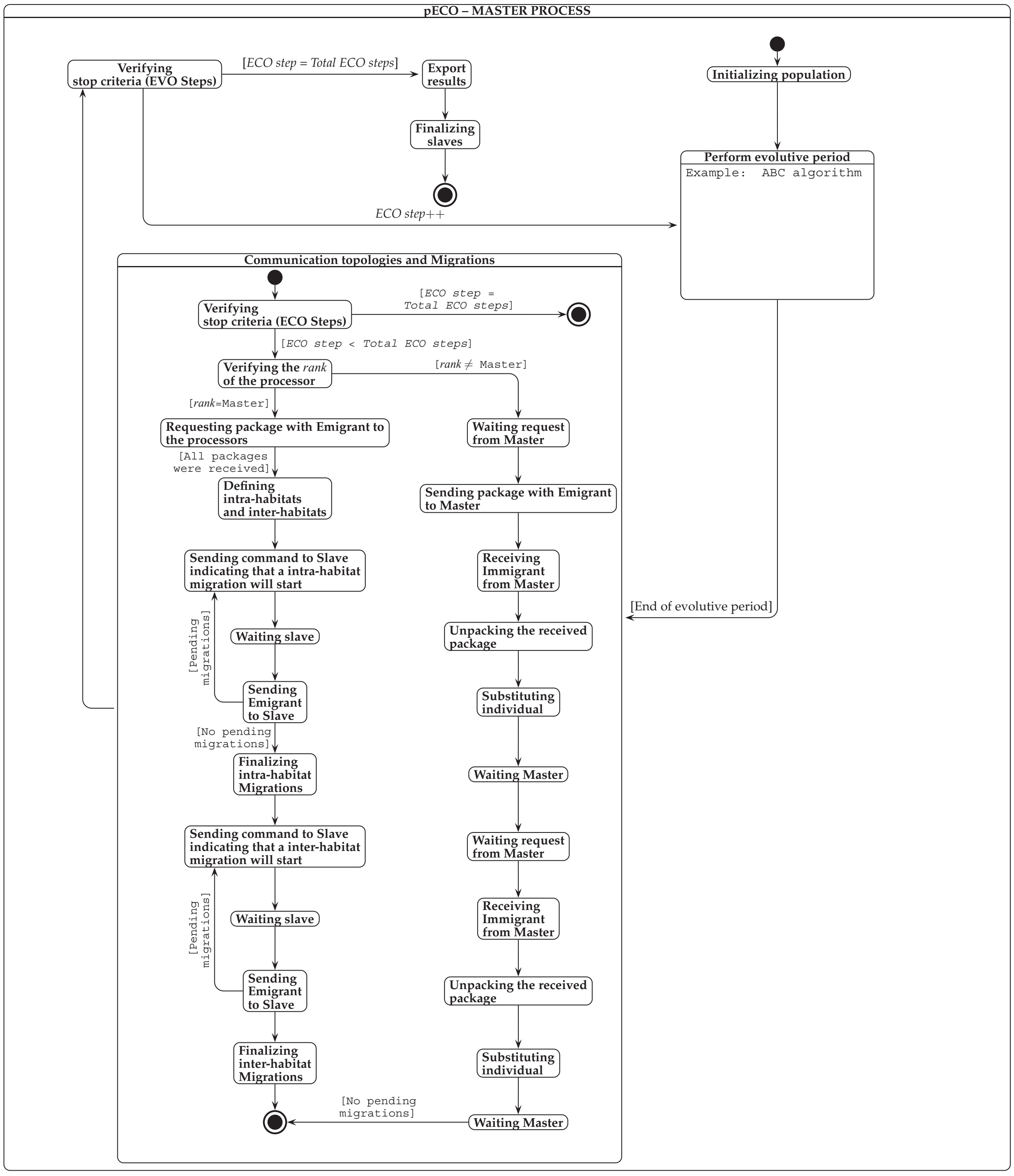

pECO – MASTER PROCESS
Verifying stop criteria (EVO Steps)
[ECO step = Total ECO steps]
Export results
Finalizing slaves
Initializing population
Perform evolutive period
Example: ABC algorithm
ECO step++
Communication topologies and Migrations
Verifying stop criteria (ECO Steps)
[ECO step = Total ECO steps]
[ECO step < Total ECO steps]
Verifying the rank of the processor
[rank ≠ Master]
[rank=Master]
Requesting package with Emigrant to the processors
[All packages were received]
Defining intra-habitats and inter-habitats
Sending command to Slave indicating that a intra-habitat migration will start
Waiting slave
[Pending migrations]
Sending Emigrant to Slave
[No pending migrations]
Finalizing intra-habitat Migrations
Sending command to Slave indicating that a inter-habitat migration will start
Waiting slave
[Pending migrations]
Sending Emigrant to Slave
Finalizing inter-habitat Migrations
Waiting request from Master
Sending package with Emigrant to Master
Receiving Immigrant from Master
Unpacking the received package
Substituting individual
Waiting Master
Waiting request from Master
Receiving Immigrant from Master
Unpacking the received package
Substituting individual
Waiting Master
[No pending migrations]
[End of evolutive period]

Figure 24: Statechart of pECO

Source: Own work.

habitats, during the migration process, is composed by two parts:

- *Message envelope*: similar to the previous, but the origin is the master and the destination is the slaves.

- *Data*: it represents the package with only the emigrant individual. In other words, the sent data is composed solely by the solution vector $\vec{x}$.

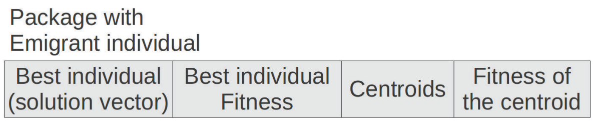


Figure 25: Package with Emigrant individual

Source: Own work.

Finally, it is important to mention that an article related to the proposed approach presented in this section is published in (BENÍTEZ; PARPINELLI; LOPES, 2013).

3.4 STRUCTURE ALIGNMENT AND VALIDATION

This work also proposes a novel method for comparing structures, using the 3D-AB model, and calculating the RMSD (Root-mean-square deviation) in [Å]. It is used for comparing the obtained structures with real structures (from PDB). Algorithm 9 presents this method.

Algorithm 9 Structure evaluation algorithm

```
1: Start
2: $AB\_like_a \leftarrow fitting(P_1)$
3: $AB\_like_b \leftarrow fitting(P_2)$
4: $RMSD \leftarrow kabsch(AB\_like_a, AB\_like_b)$
5: End
```

Basically, Algorithm 9 has three steps, where the first two steps are fitting procedures and the last one represents a quality assessment.

- **Fitting procedures**: In step 1 ($AB\_like_a \leftarrow fitting(P_1)$), the PDB file coordinates (P_1) are fitted to an off-lattice structure (called "*AB_like*"), where all bond lenghts are scaled to 3.8 Å (AB_like_a), which is the mean distance between consecutive Cα atoms (MANN et al., 2012). In step 2 ($AB\_like_b \leftarrow fitting(P_2)$), the coordinates of the obtained AB model structure (P_2) are fitted to the "*AB_like*' structure, where all unit-length bonds are also scaled to 3.8 Å (*AB_like$_b$*). Algorithm 10 shows the fitting procedure.

It is important to recall that the *atan*2 function returns a positive value for counter-clockwise angles, and a negative value for clockwise angles.

Algorithm 10 Fitting procedure – *fitting(p)*

```
1: Start
   Let $N$ be the protein size (number of amino acids)
   Let $p$ be the input coordinates (from PDB or AB)
   Let $a$ be the output "AB_like" coordinates
   Let $dx$, $dy$ and $dz$ Let $r$ be the bond lenght between $i$ and $(i+1)$ amino acids
2: for $i = 1 \rightarrow N-1$ do
3:     $dx \leftarrow p[i+1].x - p[i].x$
4:     $dy \leftarrow p[i+1].y - p[i].y$
5:     $dz \leftarrow p[i+1].z - p[i].z$
6:     $r \leftarrow \sqrt{(dx^2 + dy^2 + dz^2)}$
7:     $\theta[i] \leftarrow acos(dz/r);$
8:     $\phi[i] \leftarrow atan2(dx, dy);$
9: end for
10: $a[0].coord \leftarrow p[0]$
11: for $i = 2 \rightarrow N$ do
12:     $a[i].x \leftarrow a[i-1].x + 3.8 * \sin(\theta[i-1]) * \cos(\phi[i-1])$
13:     $a[i].y \leftarrow a[i-1].y + 3.8 * \sin(\theta[i-1]) * \sin(\phi[i-1])$
14:     $a[i].z \leftarrow a[i-1].z + 3.8 * \cos(\theta[i-1])$
15: end for
16: return $a$
17: End
```

- **Quality assesment:**

  In step 4, it is measured the similarity between the off-lattice structures obtained in steps 2 and 3. According to (PARK; LEVITT, 1995; MANN et al., 2012), RMSD is used to assess protein model quality. It measures the similarity of two structures from coordinates, as shown in Equation 47.

$$RMSD = \sqrt{\frac{\sum_{i=0}^{N-1} |P_{1_i} - P_{2_i}|}{N}} \tag{47}$$

  Where, N, P_{1_i} and P_{2_i} represent the number of amino acids, the Cartesian coordinates of the first protein structure P_1 and the Cartesian coordinates of the second protein structure P_2, respectively.

  The RMSD evaluation depends on the superpositioning of the protein structures. Since the RMSD is a rotation-dependent measure, a optimised RMSD is done using the Kabsch method (KABSCH, 1978) in order to obtain the lowest RMSD. The main idea of the Kabsch method is to calculate the rotation matrix, which is used to minimize the RMSD. Basically, the Kabsch method algorithm has three steps. First, a translation to the origin of the centroids of both structures (AB_like_a, AB_like_b) is done by using Equation 44. It results in two translated structures P and Q, that are $N \times 3$ matrices. The ith row of each matrix represents the translated coordinates of the ith amino acid. Next, the covariance matrix (A_{ij}) is calculated, as shown in Equation 48. Then, the computation of the optimal rotation matrix (U) is done using Singular Value Decomposition (SVD) (GOLUB; LOAN, 2012), and it is used to rotate P unto Q. Finally, the $RMSD$ is calculated by using the rotated P and Q, as shown in Equation 47.

$$A_{ij} = \sum_{k=1}^{N} P_{ki} Q_{kj} \tag{48}$$

where, N is the number of amino acids; P and Q are the sets of coordinates of the translated structures.

3.5 PERFORMANCE MEASURES IN PARALLEL COMPUTING

In order to evaluate the parallel algorithms proposed in this thesis, three performance measures are used: *speedup*, *efficiency* and *serial fraction*.

Probably, speedup is the most widely used performance measure in parallel computing (ALBA, 2005). This measure aims at evaluating how much a parallel algorithm is faster than the equivalent sequential version. Speedup (s_m) is defined as the time needed for running a given algorithm in one processor (T_1) divided by the running time of the same parallel algorithm, running in m processors (T_m), as in Equation 49.

$$s_m = \frac{T_1}{T_m} \tag{49}$$

From Equation (49) three types of speedup behavior can be clearly identified: sublinear speedup ($s_m < m$), linear speedup ($s_m = m$), and superlinear speedup ($s_m > m$). In (ALBA, 2005), there is an interesting taxonomy for the measurement of speedup in parallel processing systems that was useful for this work:

- Strong speedup: compares the execution time of the parallel version with the best sequential version of the algorithm (also known as best-so-far sequential algorithm).
- Weak speedup: compares the execution time of the parallel version of an algorithm developed by a given programmer with its own sequential version of the algorithm. There are two variations for this type of measure:
  - Versus panmixia: compares the parallelized algorithm with the sequential version;
  - Orthodox: compares the parallel algorithm running in one processor against the same algorithm running in m processors.

Efficiency is a measure capable of evaluating the amount of time that a given processor is actually used for processing (ROOSTA, 1999). Equation 50 shows how the Efficiency of a parallel computing system is computed: it is simply the speedup divided by the number of processors.

$$e_m = \frac{s_m}{m} \tag{50}$$

An interesting metric for measuring and observe more effects rather than using the speedup metric is the serial fraction (f_m), proposed by (KARP; FLATT, 1990) and it defined in Equation 51.

$$f_m = \frac{1/s_m - 1/m}{1 - 1/m} \tag{51}$$

To the best of our knowledge, the first attempt to use the serial fraction to analyze Parallel Evolutionary Algorithm was done by (ALBA, 2002). He showed that if super-linear speedup occurs, the serial fraction would take a negative value. They also showed that an increasing of f_m is a warning that the granularity of the parallel processes is too fine.

3.6 MATERIALS AND METHODS

3.6.1 SYNTHETIC SEQUENCES

We used synthetic protein sequences based on the Fibonacci numbers that were proposed by (STILLINGER; HEAD-GORDON, 1995). These sequences were used for the 3D AB model by several researchers. For instance, the best results with those sequences were obtained by (LIANG, 2004), (BACHMANN; ARKM; JANKE, 2005) and (KIM; LEE; LEE, 2005), who presented an *Annealing Contour Monte Carlo* (ACMC), a *Multicanonical Monte Carlo* and a *Conformational Space Annealing* approaches, respectively.

The arithmetic sequence of Fibonacci numbers $(f_0, f_1, ..., f_N)$ is generated by Equation 52.

$$f_0 = 0, f_1 = 1, f_i = f_{i-1} + f_{i-2} \, (\forall i > 1) \tag{52}$$

The magnitudes of the f_i numbers increase with index i, approaching the asymtotic limiting form (Equation 53):

$$f_i \sim a.\gamma^i \tag{53}$$

where: $a = 0.447213$ and γ represents the "golden mean" ratio, that is given by the Equation 54.

$$\gamma = (5^{1/2} + 1)/2 \tag{54}$$

A string of A's and B's is generated from the Fibonacci numbers according to Equation 55.

$$S_0 = 0, S_1 = 1, S_{i+1} = S_{i-1} k S_i \, (\forall i > 0) \tag{55}$$

Where k means the concatenation of the strings. For example, *A*, *B*, *AB*, *ABBAB*, *BABABBBAB*, ... are the first five elements of the Fibonacci sequence, that are generated following the rule, and mapping $1 \rightarrow A$ and $0 \rightarrow B$.

Some properties of the Fibonacci sequences were shown by (STILLINGER;

HEAD-GORDON, 1995):

- A residues are isolated along the backbone, flanked on both sides by B residues;
- B residues appear only isolated or in pairs;
- the number os residues in S_i is f_{i+1}

Table 5 shows the Fibonacci sequences that were used in this work and in the related works previously cited in this section.

Table 5: Benchmark Fibonacci sequences for the 3D AB off-lattice model, proposed by (STILLINGER; HEAD-GORDON, 1995) for the AB models

id	*N*	Sequence
1	13	$(ABB)^2(AB)^2BAB$
2	21	$(BA)^2B^2ABA(BBA)^2BAB^2AB$
3	34	$(ABB)^2(AB)^2(BABBA)^3B^2ABAB^2AB$
4	55	$(BA)^2B^2ABA(BBA)^2(BABBA)^2B^2ABA(BBA)^2(BABBA)^2B^2ABAB^2AB$

Source: Adapted from (STILLINGER; HEAD-GORDON, 1995).

Where N represents the protein sequence size (number of amino acids) and the s^k means k repetitions of a substring s.

Besides the Fibonacci sequences, other synthetic sequences were used, which were proposed by (YUE; DILL, 1993) for the 3DHP model, previously used by (BENÍTEZ; LOPES, 2010) for the 3DHP-SC model and have either 27, 31, 36 and 48 amino acids. These sequences are shown in Table 6.

Table 6: Benchmark sequences for the AB off-lattice model, proposed by (YUE; DILL, 1993) for the 3DHP models.

id	*N*	Sequence
5	27	$AB^4A^4B^2ABABA^3BAB^2A^2B^2A$
6	27	$ABBBAAAABABAABBBABAABABBBAB$
7	27	$AB(AABB)^2A^4(BBBA)^2A^2B^2A$
8	31	$(AAB)^2A^6(BBAAAAA)^2A^2$
9	36	$BA(BBA)^{11}B$

Source: Adapted from (YUE; DILL, 1993).

3.6.2 REAL PROTEIN SEQUENCES

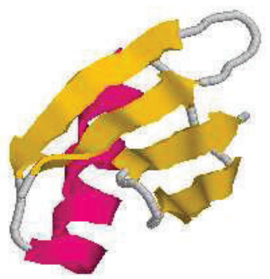
Table 7 shows the list of real protein sequences that were used in this work. In this table, the second column, third and fourth columns identify, respectively, the PDB code, name, size (N) and the structural class of the proteins. The structural class $\alpha + \beta$ indicates that the protein is formed by α-helices and antiparallel β-sheets. On the other hand, α/β means that the protein is formed by α-helices and parallel β-sheets. These sequences were chosen according to their size. Table 8 shows the amino acids sequence of each protein, using the three letter encoding.

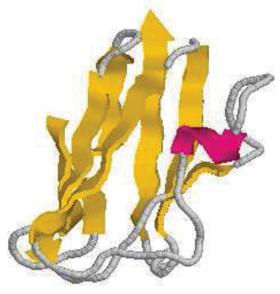
Table 7: List of real protein sequences

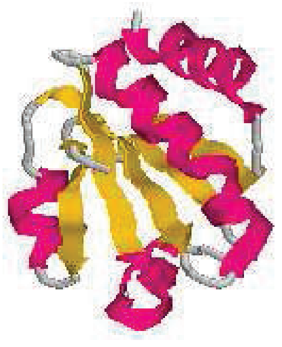
id	PDB code	Name	N	Structural class
10	2gb1	Protein G	56	$\alpha + \beta$
11	1pcy	Oxidized Poplar Plastocyanin	99	$\alpha + \beta$
12	2trx	*Thioredoxin da Escherichia coli*	108	α/β
13	3fxn	*Clostridium Beijerinckii Flavodoxin*	138	α/β

Source: Own work, based on PDB files.

Figures 26(a), 26(b), 26(c) and 26(d) show, respectively, the native conformations of the real proteins used in this work, using RasMol [2].


(a)


(b)


(c)

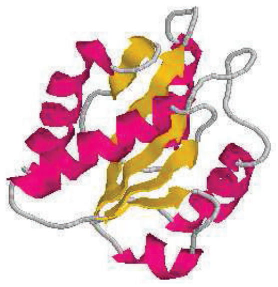

(d)

Figure 26: Native conformations of the proteins 2gb11 (a), 1pcy (b), 2trx (c) and 3fxn (d) Source: These figures were generated from PDB files, using the *Cartoon visualization* of the RasMol application.

[2]RasMol is a molecular visualization software. Available at http://www.rasmol.org

Table 8: List of real protein sequences

id	PDB code	Sequence
10	2gb1	MET THR TYR LYS LEU ILE LEU ASN GLY LYS THR LEU LYS GLY GLU THR THR THR GLU ALA VAL ASP ALA ALA THR ALA GLU LYS VAL PHE LYS GLN TYR ALA ASN ASP ASN GLY VAL ASP GLY GLU TRP THR TYR ASP ASP ALA THR LYS THR PHE THR VAL THR GLU
11	1pcy	ILE ASP VAL LEU LEU GLY ALA ASP ASP GLY SER LEU ALA PHE VAL PRO SER GLU PHE SER ILE SER PRO GLY GLU LYS ILE VAL PHE LYS ASN ASN ALA GLY PHE PRO HIS ASN ILE VAL PHE ASP GLU ASP SER ILE PRO SER GLY VAL ASP ALA SER LYS ILE SER MET SER GLU GLU ASP LEU LEU ASN ALA LYS GLY GLU THR PHE GLU VAL ALA LEU SER ASN LYS GLY GLU TYR SER PHE TYR CYS SER PRO HIS GLN GLY ALA GLY MET VAL GLY LYS VAL THR VAL ASN
12	2trx	SER ASP LYS ILE ILE HIS LEU THR ASP ASP SER PHE ASP THR ASP VAL LEU LYS ALA ASP GLY ALA ILE LEU VAL ASP PHE TRP ALA GLU TRP CYS GLY PRO CYS LYS MET ILE ALA PRO ILE LEU ASP GLU ILE ALA ASP GLU TYR GLN GLY LYS LEU THR VAL ALA LYS LEU ASN ILE ASP GLN ASN PRO GLY THR ALA PRO LYS TYR GLY ILE ARG GLY ILE PRO THR LEU LEU LEU PHE LYS ASN GLY GLU VAL ALA ALA THR LYS VAL GLY ALA LEU SER LYS GLY GLN LEU LYS GLU PHE LEU ASP ALA ASN LEU ALA
13	3fxn	MET LYS ILE VAL TYR TRP SER GLY THR GLY ASN THR GLU LYS MET ALA GLU LEU ILE ALA LYS GLY ILE ILE GLU SER GLY LYS ASP VAL ASN THR ILE ASN VAL SER ASP VAL ASN ILE ASP GLU LEU LEU ASN GLU ASP ILE LEU ILE LEU GLY CYS SER ALA MET GLY ASP GLU VAL LEU GLU GLU SER GLU PHE GLU PRO PHE ILE GLU GLU ILE SER THR LYS ILE SER GLY LYS LYS VAL ALA LEU PHE GLY SER TYR GLY TRP GLY ASP GLY LYS TRP MET ARG ASP PHE GLU GLU ARG MET ASN GLY TYR GLY CYS VAL VAL VAL GLU THR PRO LEU ILE VAL GLN ASN GLU PRO ASP GLU ALA GLU GLN ASP CYS ILE GLU PHE GLY LYS LYS ILE ALA ASN ILE

Source: Adapted from PDB files.

These proteins were extracted from PDB files [3]. The PDB format has 12 sections, where different records are listed in a specific order. Table 9 shows the sections and records of the PDB file format [4].

Table 9: Sections and records of the PDB file

Section	Description	Records
Title	Summary Description	HEADER, OBSLTE, TITLE, CAVEAT, AUTHOR, REVDAT, SPRSDE, JRNL KEYWDS, EXPDTA, COMPND, SOURCE
Remark	References, Description of the biological function	REMARKS 1,...*n*
Primary structure	sequence of residues in each chain of the macromolecule	DBREF, SEQADV, SEQRES, MODRES
Heterogen description	complete description of non-standard residues in the entry	HET, HETNAM HETSYN, FORMUL
Secondary structure	description of the secondary structures	HELIX, SHEET TURN
Connectivity annotation	chemical connectivity	SSBOND, LINK HYDBND, SLTBRG, CISPEP
Miscellaneous features	feature descriptions of the macromolecule	SITE
Crystallographic	Description of the Crystallographic Cell	CRYST1
Coordinate transformation	coordinate system transformations	ORIGXn, SCALEn, MTRIXn, TVECT
Coordinate	atomic coordinates	MODEL, ATOM, SIGATM, ANISOU, SIGUIJ, TER, HETATM, ENDMDL
Connectivity	chemical connectivity	CONECT
Bookkeping	final information about the file	MASTER, END

Source: Adapted from the PDB file format documentation.

In this work, the amino acid sequence and the coordinates of the amino acids of the protein are required. Thus, we used the SEQRES and ATOM records of the PDB file. Table 10 shows the SEQRES record format, where first, second and third columns show the columns numbers, name and definition of each field.

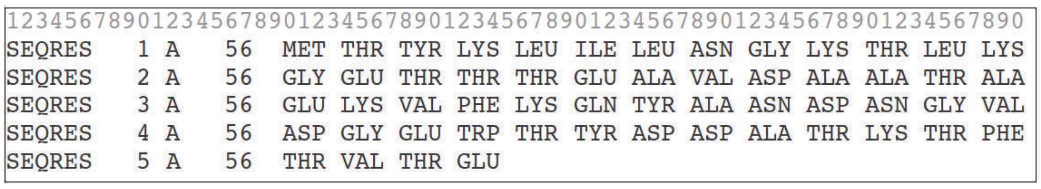
[3] Available in *http://www.pdb.org*
[4] Available at http://www.wwpdb.org/documentation/file-format

Table 10: SEQRES record format

Columns	Field	Definition
1-6	Record name	"SEQRES"
9-10	serNum	serial number
12	chainID	Chain identifier
14-17	numRes	number of residues
20-22	resName	Residue name (three letters code)
24–26		
...		
68–70		

Source: Adapted from the PDB file format documentation.

For example, Figure 27 shows the SEQRES records of the PDB file of the 2gb1 protein.



```
1234567890123456789012345678901234567890123456789012345678901234567890
SEQRES   1 A   56  MET THR TYR LYS LEU ILE LEU ASN GLY LYS THR LEU LYS
SEQRES   2 A   56  GLY GLU THR THR THR GLU ALA VAL ASP ALA ALA THR ALA
SEQRES   3 A   56  GLU LYS VAL PHE LYS GLN TYR ALA ASN ASP ASN GLY VAL
SEQRES   4 A   56  ASP GLY GLU TRP THR TYR ASP ASP ALA THR LYS THR PHE
SEQRES   5 A   56  THR VAL THR GLU
```

Figure 27: Example of SEQRES records: 2gb1 PDB file

Source: Adapted from 2gb1 PDB file.

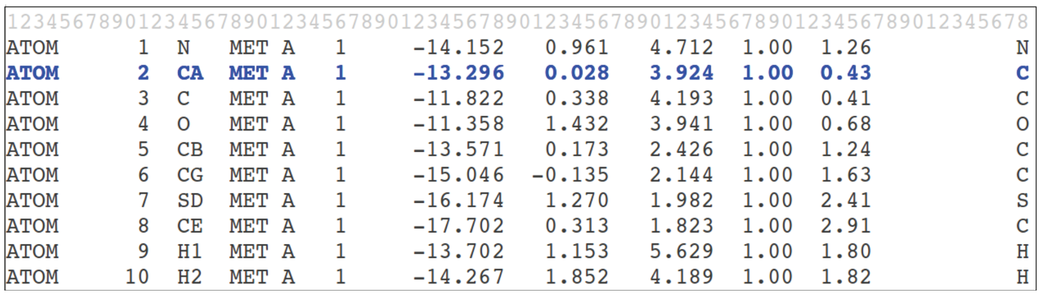
Table 11 shows the ATOM record format, where first, second and third columns show the columns numbers, name and definition of each field.

Figure 28 shows the first ten ATOM records of the PDB file of the 2gb1 protein. It is important to recall that we use the backbone of the proteins, which is formed by the Cαs of the amino acids. Hence, our approach reads only the Cα coordinates from the PDB file. In Figure 28, the bold text shows the Cα coordinates of the first amino acid of the chain, where "CA" (name field) indicates the Cα.

Table 11: ATOM record format

Columns	Field	Definition
1-6	Record name	"ATOM "
7-11	serNum	serial number
13-16	Name	atom role name
17	altLoc	atom variant
18–20	resName	amino acid code (three letters code)
22	chainId	chain ID
23–26	resSeq	residue sequence number
27	iCode	Insertion code
31–38	*x*	atom *x* coordinate
39–46	*y*	atom *y* coordinate
47–54	*z*	atom *z* coordinate
55–60	occupancy	atom occupancy
61–66	tempFactor	temperature factor
77–78	element	element symbol
79–80	charge	charge on the atom

Source: Adapted from the PDB file format documentation.



```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
ATOM      1  N   MET A   1     -14.152   0.961   4.712  1.00  1.26           N
ATOM      2  CA  MET A   1     -13.296   0.028   3.924  1.00  0.43           C
ATOM      3  C   MET A   1     -11.822   0.338   4.193  1.00  0.41           C
ATOM      4  O   MET A   1     -11.358   1.432   3.941  1.00  0.68           O
ATOM      5  CB  MET A   1     -13.571   0.173   2.426  1.00  1.24           C
ATOM      6  CG  MET A   1     -15.046  -0.135   2.144  1.00  1.63           C
ATOM      7  SD  MET A   1     -16.174   1.270   1.982  1.00  2.41           S
ATOM      8  CE  MET A   1     -17.702   0.313   1.823  1.00  2.91           C
ATOM      9  H1  MET A   1     -13.702   1.153   5.629  1.00  1.80           H
ATOM     10  H2  MET A   1     -14.267   1.852   4.189  1.00  1.82           H
```

Figure 28: Example of ATOM records: 2gb1 PDB file

Source: Adapted from 2gb1 PDB file.

4 COMPUTATIONAL EXPERIMENTS AND RESULTS

"No man's knowledge here can go beyond his experience."

— John Locke

In this chapter, a series of computational experiments that explores each part of the proposed approach is presented. First, Section 4.1 presents how the Molecular Dynamics simulations were conducted. After that, the experiments for evaluating the performance of the parallel approaches pGEP-CA and pECO are presented in Sections 4.3 and 4.4, respectively.

All experiments reported in this work were run in computers with a Intel processor Quad Core, running Arch Linux [1]. Parallel approaches were developed using the MPICH2 for the message passing interface. All algorithms were implemented in ANSI-C programming language. The figures were generated using Gnuplot [2] and a developed OpenGL[3] -based interface.

4.1 MOLECULAR DYNAMICS SIMULATIONS

The basic MD parameters used in all the experiments are: time-step: $\delta t = 0.0001$; stop criterion: $t_{max} = 300$ (this value represents 3×10^6 energy evaluations); and coupling constant (Berendsen's thermostat): $\tau_p = 0.01$. The parameters were chosen empirically.

[1] Available at: https://www.archlinux.org/
[2] Available at: http://www.gnuplot.info
[3] Available at: http://www.opengl.org/

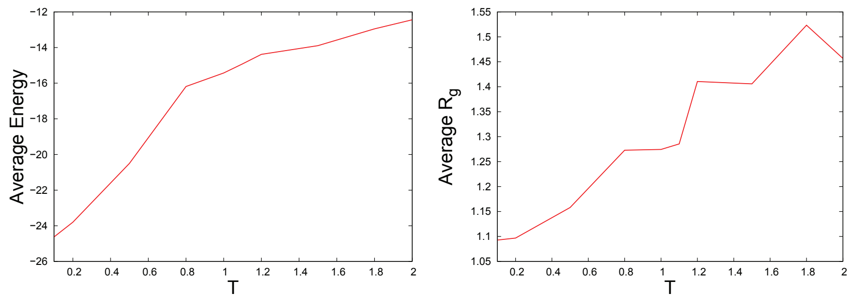
4.1.1 EXPERIMENTS USING SYNTHETIC SEQUENCES

In this section we present the experiments and results obtained through Molecular Dynamics simulations, using synthetic sequences. This Section is based on our previous conference paper (BENÍTEZ; LOPES, 2012).

Tables 5 and 6 (see Section 3.6.1) show the synthetic sequences that were used in the experiments reported in this section.

In order to study the temperature dependence of the protein folding, experiments were done under different values of the environment temperature ($T = [0.1; 0.2; 0.5; 0.8; 1.0; 1.1; 1.2; 1.5; 1.8; 2.0]$) and using the 13-amino-acid-long sequence. The energy and radius of gyration of the best conformation obtained were recorded in order to analise the thermodynamic behavior of the protein. For each temperature, 10 independent runs were done using different initial unfolded conformations, and the average results are shown in Figures 29(a) and 29(b).

The overall size of the molecule (compactness of the residues), as measured by R_g, increases substantially when T increases as shown in Figure 29(b). The total energy also increases when T increases. This indicates a denaturation process. The best results (i.e. lower energy conformations) were obtained at $T = 0.1$. Therefore, this value was fixed for the remaining experiments.



(a) (b)

Figure 29: Thermodynamic properties: average E (a) and R_g (b) (in dimensionless MD units). Source: Own work.

Table 12 presents the results obtained through Molecular Dynamics simulations, using synthetic sequences. In this table, the first column identifies the size of the sequences. For comparison purpose, the second column shows the best results obtained for the 3D AB model of the literature. The third column shows the best results found. Next, the fourth column shows the percent difference between them and our results. The fifth column presents the average (± standard deviation) of the energy values obtained. Finally, the average processing time is shown in last column.

Here, it is important to known that (LIANG, 2004) and (BACHMANN; ARKM; JANKE, 2005) obtained the best result for the 13 and 21-amino-acid-long sequences. In addition, (KIM; LEE; LEE, 2005) obtained the best results for the 34 and 55-amino-acid-long sequences. The other five sequences were only used for simple HP models. Therefore, there are no comparison values for these sequences.

Analyzing the percent difference, it is observed that our results are slightly worse than the best results found in the literature. Probably, even better results would be found by increasing the stop criterion parameter (t_{max}) or starting from other initial unfolded conformations.

Since the MD simulation is deterministic, the average of the energy values obtained indicates that different folding trajectories were generated. Thus, different initial conditions led to different structures.

Overall, the processing time is a function of the length of the sequence, growing as the number of amino acids of the sequence increases.

Table 12: Results for the 3D AB off-lattice model. N/A = not available

N	*Energy*				$t_p(s)$
	Best (literature)	Best	diff (%)	Avg±stdev	
13	-26.507	-26.4661	0.15	-24.18±1.83	131.34
21	-52.917	-51.7720	2.18	-48.07±1.91	234.90
34	-97.7321	-91.3662	6.73	-87.11±3.81	326.50
55	-173.9803	-160.9863	7.75	-153.45±4.45	504.39
27	N/A	-75.8225	–	-71.44±3.38	302.56
27	N/A	-73.0161	–	-67.96±3.52	264.34
27	N/A	-74.3461	–	-68.63±3.56	325.72
31	N/A	-103.4963	–	-99.36±3.08	247.09
36	N/A	-94.0439	–	-89.92±2.59	271.53

Source: Own work.

An example of the folding pathway of the 13-amino-acid-long is presented in Figure 30. An OpenGL [4] interface was developed for generating this figure. In this figure, six folding states obtained in the simulation are shown. The captions below each protein structure shows the energy (E) and radius of gyration (R_g) at different times (t). The folding process starts with a denatured conformation with high energy (in this example: $E = -6.626$). The protein folds through a series of intermediate states, where

[4]The OpenGL library is available at: http://www.opengl.org/

fragments start to pack and the protein leaves the misfolded (or partially-structured) intermediate states and forms a native-like structure. In order to reinforce our observation about the folding pathway, Figures 31(a) and 31(b) show the time dependence of the total energy and the radius of gyration.

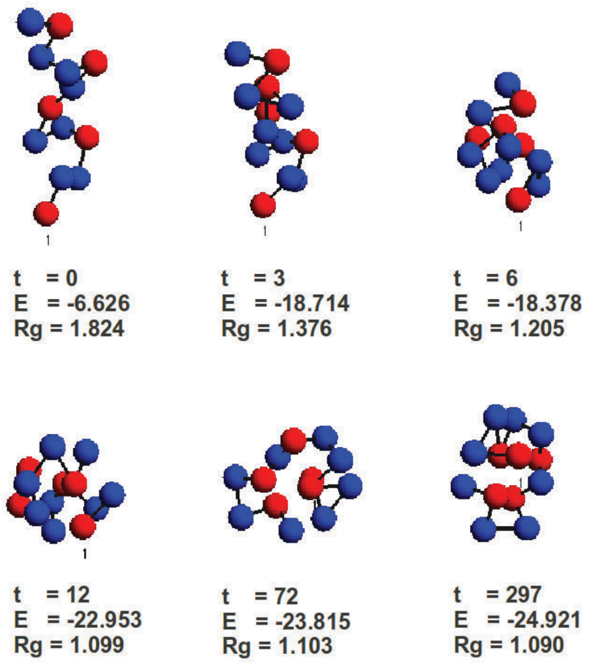
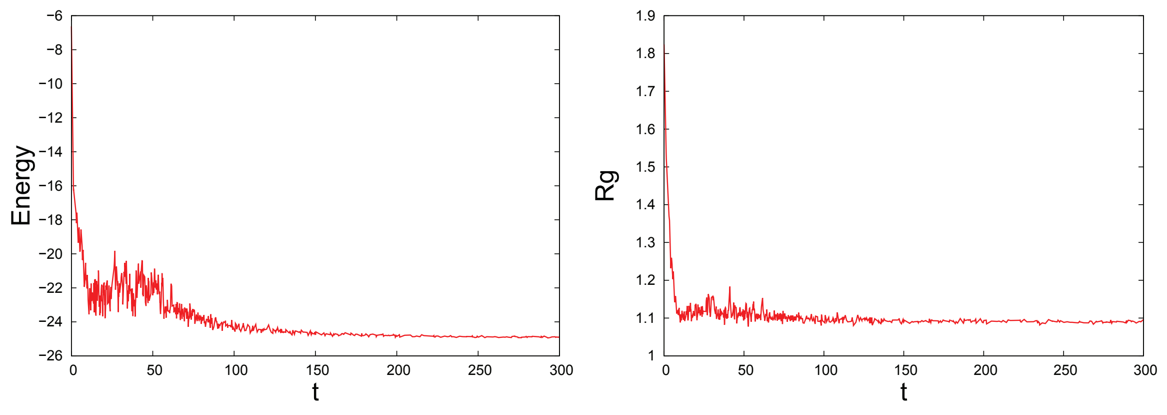


Figure 30: Sample snapshot from a folding pathway. The number 1 denotes the N terminus of the chain.

Source: Own work.



(a) (b)

Figure 31: *energy*(t) **(a) and** $r_g(t)$**(b) (in dimensionless MD units).**

Source: Own work.

4.1.2 EXPERIMENTS USING REAL SEQUENCES

4.1.2.1 STRUCTURE CONVERSION

Real coordinates, extracted from PDB files, were converted into coordinates of an *off-lattice* structure (called "*AB_like*"). According to Section 3.4, all bond lengths are scaled to 3.8 Å (AB_like_a), which is the mean distance between consecutive Cα atoms in protein structures (MANN et al., 2012). It is important to note that the "*AB_like*" are considered here to be the native structures, for comparison purposes.

Table 7 (see Section 3.6.2) shows the real sequences that were used in the experiments reported in this thesis. Figures 32(a), 32(b), 32(c) and 32(d) show the obtained "*AB_like*" structures for proteins 2gb1, 1pcy, 2trx and 3fxn, respectively. An OpenGL based [5] interface was developed for generating these figures.

Table 13 shows the RMSD between *AB_like* structures obtained from real proteins (obtained from PDB files) and AB structures (with all bond lengths equal to 3.8 Å). The RMSD was computed using the Kabsch algorithm (See Section 3.4 for further information). In this table, first, second and third columns show the identification, name and size of the protein sequences (N). The fourth column shows the RMSD between the protein structures, and for further information, fifth column shows the average bond length between Cαs of the real protein structures for 10 independent runs.

Table 13: RMSD between real proteins (from PDB) and *AB_like* structures

	Protein		RMSD (Å)	Average
id	Name	*N*		Bond length (PDB)
10	2gb1	56	0.040776	3.809915
11	1pcy	99	1.605116	3.672578
12	2trx	108	0.828630	3.764471
13	3fxn	138	0.559579	3.798874

Source: Own work.

In Table 13, we can observe that the RMSD is not correlated with the size of the protein sequence (N), and the lower the average bond length between Cαs of the real structures, the larger the RMSD. In other words, the further the average from 3.8 Å, larger the error due to the conversion procedure to the "*AB_like*" structures and, consequently, the larger the RMSD.

[5]The OpenGL library is available at: http://www.opengl.org/

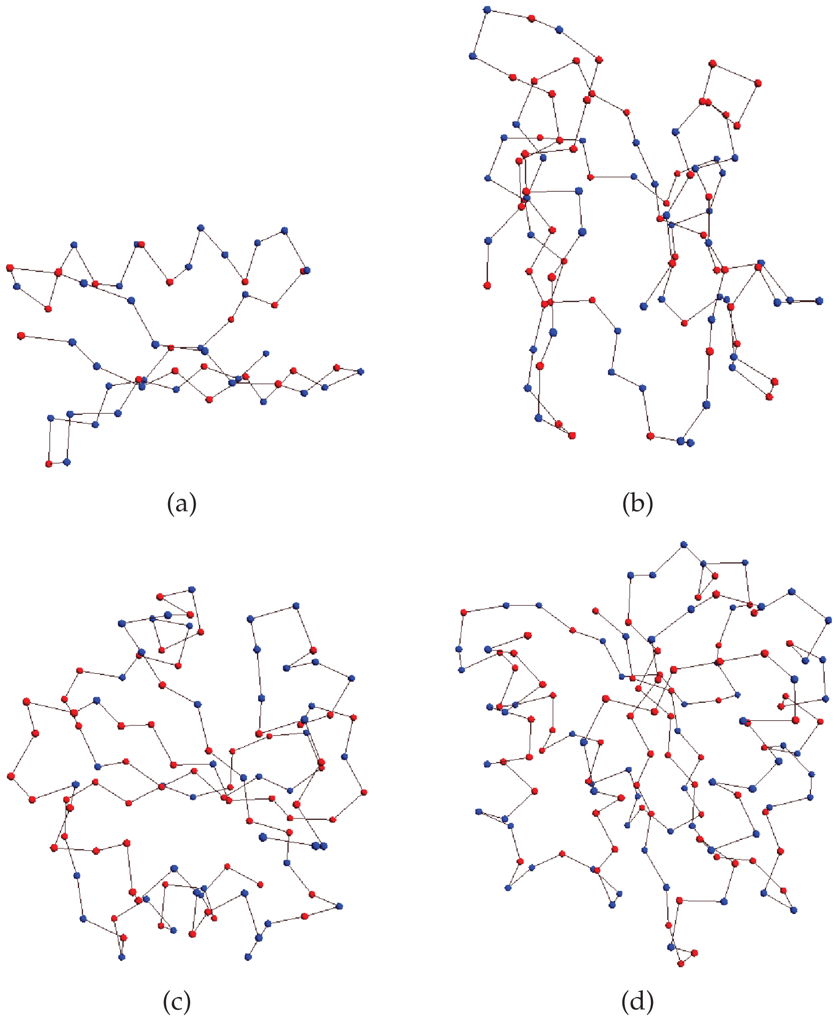


Figure 32: *AB_like* **conformations of proteins 2gb1 (a), 1pcy (b), 2trx (c) and 3fxn (d), where all bond lengths are equal to 3.8 Å. Blue and red balls represent the polar and the hydrophobic residues, respectively. The backbone and the connections between elements are shown in black lines. In this example, we used the amino acid conversion following the classification by (ALBERTS et al., 2002), as shown in Table 2.**

Source: Own work.

4.1.3 HYDROPHOBICITY SCALES AND AMINO ACID CONVERSION

In order to convert the protein sequences of the PDB into the AB model alphabet (i.e.: 'A' and 'B' for hydrophobic and hydrophilic residues, respectively) we need to use an amino acid conversion table. In this work, we used the amino acid type classification shown in (ALBERTS et al., 2002) and derived an amino acid type classification from the hydrophobicity scales presented in Section 2.1.1.

Many experiments were done to verify the adequacy of the amino acid conversion tables which, in turn, are obtained from the hydrophobicity scales presented in Section 2.1.1, for the PFP using the AB model of proteins. The evaluation of these experiments takes into account not only the quality of the solutions obtained, which is measured through the RMSD value between the folded structures and *AB_like* structures from PDB files, but also, the correlation between the estimated energy and RMSD values.

Table 14 shows the equivalent AB sequences of the proteins 2gb1, 1pcy, 2trx and 3fxn for each hydrophobicity scale (or amino acid classification, according to its hydrophobicity). In this table, the second and third columns identify the name of the amino acid conversion table and the equivalent sequence written in the AB model alphabet.

Table 15 shows the results obtained through Molecular Dynamics simulations, using the equivalent AB sequences of the real sequences 2gb1, 1pcy, 2trx and 3fxn. In this table, the first column identifies the name of the protein. The second column identifies the name of the amino acid conversion table. The third and fourth columns identify the best results and average ($\pm$ standard deviation) of the energy values obtained. The fifth column represents the RMSD values obtained between *AB_like* structures (see Section 4.1.2.1). The eighth column identifies the average radius of gyration (R_g) of the AB structures obtained. The last column shows the average processing time (T_p). The correlation coefficient r (also referred to as the *Pearson product moment correlation coefficient*) (RODGERS; NICEWANDER, 1988) is also presented in the seventh column. It measures the linear relationship between the energy of the structures obtained and RMSD.

In Table 15, it is observed that there is no linear correlation between the energy and RMSD values for these sequences, but the correlation obtained using the "Hessa" conversion for sequence 1pcy ($r = -0.77$) almost represents a correlation between the energy and RMSD. Moreover, it is possible to observe that the best result for sequence 2gb1, regarding the RMSD measure, was achieved using the conversion table following the classification by (ALBERTS et al., 2002).

In order to compare the conversion tables, regarding the RMSD values, Figures 33(a) – 33(d) show the comparative boxplots obtained from Table 15. The boxplots represent an informative graphical display that visually highlights the location and spread of sets of data and it is often highly suggestive of the need for inferential comparisons of means and standard deviations. The mean is indicated by a line segment drawn within the box parallel to the edges, and the variability of the data is represen-

Table 14: Equivalent AB sequences of the proteins

Protein	Classification	Sequence
2gb1	Alberts	$AB^3A^3BAB^2ABAB^4B(AAB)^2AB^2A^2(BBBA)^3A(BA)^2B(BBBA)^2BAB^2$
	Kyte	$AB^3A^3B^4AB^6(BAA)^2BAB^2A^2B^3AB^4AB^8AB^3(AB)^2B$
	Wimley	$(AB)^2A^2(ABBBB)^2B^{13}(ABB)^2B^7(AB)^2B^5AB^4$
	Hessa	$AB^3A^2(ABBBB)^4B^4AB^7A(ABBBBBBBB)^2B^4(AB)^2B$
	Eisenberg	$(AB)^2A^3BAB(BA)^2B^4(BAA)^2BA(BBAA)^2B^3A^2(BA)^3B^2AB^3(AB)^2B$
	Janin	$AB^3A^3BAB(BA)^2B^4(BAA)^2BAB^2A^2(BBBA)^3(AB)^3(BBBA)^2BAB^2$
	Engelman	$A^2B^2A^3(BA)^2(AB)^2A(AAB)^2A^2(AABB)^2BAB^3A^2(BA)^2AB^3A^2BA^5B$
1pcy	Alberts	$A(BAAAAABBA)^2(BA)^2AB^2A^3B^3A^4B^2A^3B^4(AAB)^2AB(BA)^2B^4A^2$ $(BA)^2B(BA)^2A^2(BBBA)^2(BA)^2BBAAAAAABABAB$
	Kyte	$ABA^3BAB^4A^4B^3(AB)^2B^4A^3B^2(BA)^2B^3A^3B(BBBA)^2BAB(BA)^2B^4$ $A^2BAB^3(BA)^2A^2B^7(AB)^2B^3(BA)^2AB(BA)^2B$
	Wimley	$AB^2A^2B^6(AB)^2B^2(BA)^2B^5(AB)^2B(BBBA)^2BAB^4AB^8(AB)^2B^3A^2$ $B^6(ABBB)^2B(BA)^2A^2B^7AB^7$
	Hessa	$ABA^3B^6ABA^2B^2(BA)^2B^5A^3B^5AB^3A^3B(BBBA)^2AB^3(BA)^2B^4A^2$ $B^6(AB)^3B^6(AB)^2B^6A^2B(BA)^2B$
	Eisenberg	$ABA^5B^2ABA^5B(BA)^3AB^2A^3B^3A(AAABB)^2B(BAA)^2BAB(BA)^2B^4A^2$ $(BA)^2B^2ABA^3B^2(BA)^3A(AB)^2BA^6(BA)^2B$
	Janin	$ABA^5B(BA)^2A^3B^2(BA)^2B^2AB^2(AAABBB)^3BAB^2A^2BAB(BA)^2B^4A$ $(AB)^3(BA)^2A^2(BBBA)^2BAB^4A^5(AB)^3$
	Engelman	$ABA^5B^2A^6(BA)^2A^2(AB)^2B(AAABBB)^3A^2BA^3(BAA)^2A^2B^3A(AB)^3$ $A^2BA^3(ABB)^2(AAB)^2B^2A^6BA^3B$
2trx	Alberts	$B^3A(AB)^2(BBBA)^2(AB)^2A^2(AAAB)^2A^5BA^6B^2A^2B^4(AB)^2A^2(BA)^2$ $B^2(BAA)^2B^2A^2(BAAA)^2AB(BA)^2A^2B^2A^4B(BA)^2B^2A^2(BA)^2A$
	Kyte	$B^3A^2(BABBB)^2A^2BAB^2A^4(BA)^2(BBA)^2BA^3B(AABB)^2B^3(BA)^2$ $(AB)^3B(BBBBA)^2(BBA)^2A^3B^4A^2AB(BA)^2AB^2(BBA)^2A(BA)^2A$
	Wimley	$B^3AABA(BBBBA)^2B^3(BBAA)^3B^2ABA^2B^2A^2B^2(ABBB)^3(BA)^2B^8$ $(BA)^2(BBA)^2A^3B^8(BBBBA)^2B^2A^2B^3AB$
	Hessa	$B^3A^2BAB(BBBA)^2AB^5A^2(AB)^2B(BBA)^2B(AABB)^2AB^7(AB)^2(BA)^2$ $B^9(BBA)^3A^3(BBBBA)^2B^2AB^4AB^2A^2B^3AB$
	Eisenberg	$B^3A^2BAB^4AB^3A^2(BA)^2A^4(BAAA)^2A^2BA^4(AABB)^2(AB)^3A(AB)^3$ $B(BAA)^3A^2(BAAA)^2AB(BA)^2A^2B^2A^4B(BA)^2B^2A(AB)^2A^2$
	Janin	$B^3A(AB)^2(BBBA)^2A(BA)^2A(AAAB)^3ABA(AAB)^2BA^2B^4(AB)^2A^2$ $(BA)^2B^4(AB)^2B(BAA)^2B^2A^4B(BA)^2A^2B^2A^4B(BA)^2B^2A^2(BA)^2A$
	Engelman	$AB^2(AAB)^2BA^2(BA)^2(AB)^2A^2(AAAB)^3ABA(AAB)^2BA^2B^4ABA^4(BA)^2$ $B^4A^3B^3(AAB)^2A^5B(BA)^2A^3BA^5(BA)^2B^2A^2(BA)^2A$
3fxn	Alberts	$ABA^2(BA)^3B^4A^2(BAAA)^2(BBA)^3(BABBA)^2AB^3A^6BA$ $(AABB)^2B(BA)^2A(ABB)^2B(AB)^2BA^5B^2A^3(BA)^2AB^2AB^3$ $(AB)^2A(AAAABB)^2BAB^2AB^3(AAB)^2BA^2BA$
	Kyte	$ABA^2B^{10}A^2BA^3B^2A^2B^5AB(BA)^2B(BA)^2B^2A^2B^3$ $A^4(BA)^2AB^3A^2B^4AB^2A^2B^2AB^3AB^4A^4B^8(BBA)^2B^3$ $AB^4A^4B^3A^3B^6AB^3A^2BAB^3A^2BA$
	Wimley	$(AB)^2A^2B^8AB^2A^2B^2A^2B^8AB^6AB^2A^2B^3A^4B(ABB)^2$ $B^2AB^4AB^2A^2B^2AB^3AB^6A^2B(BA)^2B^4A^2B^2AB(BBA)^2$ $BAB^6A^2B^{11}A^2BAB(BBA)^2$
	Hessa	$ABA^2B^{10}AB^2A^2B^3AAB^3(BBA)^2(BABBA)^2AB^3A^4BAB^2$ $AB^3A^2B^4AB^2A^2B^2AB^3AB^3(BA)^2AB^{10}AB^2AB^3AB^4A^4$ $B^3A^3B^{10}A^2BAB(BBA)^2$
	Eisenberg	$ABA^4(BA)^2B^3(BAA)^2ABA^3(BBA)^3BAB(BA)^2B^2A^2B^3A^6BA^3$ $B^2A^2B^4ABA^3B^2AB^2(BA)^2B^2A^5BA^4(BA)^2AB^2AB^3ABA^7B^2$ $A^4B(BBA)^2B^2(BAA)^2B^2A^2BA$
	Janin	$ABA(AB)^4B^2(BAA)^2ABA^3(BBA)^3(BABBA)^2AB^3A^6BA^3B^2A^2B^4$ $AB^2A^2B^2AB^2(BA)^2B^2A^5B^2A^3(BA)^2AB^2AB^2(BA)^3A^2$ $(AAABBB)^2(BBBA)^2ABA^2B^2A^2BA$
	Engelman	$A(BAA)^2A^2(AB)^2(BAA)^2(ABAA)^2B^2A(BAA)^2(BA)^2B^2A^2B^3$ $A^8(AABB)^2(AB)^2BA^2B^2(AAAB)^2BA^3(AAAB)^2ABA^2B^2AB^2(BA)^3A^3$ $(AB)^2A^3B^3(BBBA)^2ABA^2B^2A^2BA$

Source: Own work

Table 15: Results for the equivalent AB sequences of the real sequences 2gb1, 1pcy, 2trx and 3fxn.

Protein	Class	Energy		RMSD		r	Avg(R_g)	Avg(T_p) (s)
		Best	Average	Best	Average			
2gb1	Alberts	-166.96	-159.44 ± 3.46	7.35	9.52 ± 0.79	-0.02	1.85	2621.50
	Kyte	-157.31	-151.19 ± 4.17	9.46	9.96 ± 0.32	0.06	1.85	2615.70
	Wimley	-147.74	-144.14 ± 2.76	8.90	9.91 ± 0.50	0.46	1.84	2610.20
	Hessa	-148.88	-145.86 ± 2.51	7.73	9.33 ± 0.84	0.30	1.82	2609.50
	Eisenberg	-171.44	-165.46 ± 3.81	8.94	9.53 ± 0.56	0.04	1.83	2624.60
	Janin	-165.84	-159.84 ± 4.72	8.45	9.61 ± 0.72	0.42	1.85	2623.60
	Engelman	-184.84	-179.83 ± 2.61	8.29	9.39 ± 0.71	-0.47	1.84	2622.30
1pcy	Alberts	-350.97	-339.41 ± 6.76	10.5	12.34 ± 1.11	0.20	2.46	8377.56
	Kyte	-318.75	-303.09 ± 7.31	10.17	11.65 ± 0.94	0.03	2.27	8309.89
	Wimley	-288.86	-283.38 ± 4.78	10.05	11.62 ± 0.74	-0.34	2.28	8341.89
	Hessa	-292.38	-289.48 ± 2.35	11.96	12.32 ± 0.32	-0.77	2.26	8308.67
	Eisenberg	-346.53	-339.59 ± 6.85	11.21	12.15 ± 0.58	-0.10	2.33	8331.89
	Janin	-331.43	-325.43 ± 5.04	11.24	12.27 ± 0.71	-0.13	2.29	8349.22
	Engelman	-360.31	-351.51 ± 7.15	11.40	12.22 ± 0.45	0.42	2.38	8407.56
2trx	Alberts	-393.17	-379.57 ± 5.94	11.06	11.94 ± 0.75	-0.51	2.36	12954.22
	Kyte	-351.73	-342.92 ± 4.82	10.26	11.61 ± 0.93	0.17	2.35	13074.56
	Wimley	-332.86	-318.80 ± 6.42	10.75	11.94 ± 0.84	0.54	2.35	12920.22
	Hessa	-328.38	-319.79 ± 6.43	10.44	12.16 ± 0.94	-0.53	2.32	12994.33
	Eisenberg	-399.10	-380.91 ± 10.46	11.91	12.55 ± 0.50	-0.10	2.36	13013.56
	Janin	-378.33	-367.26 ± 6.95	10.53	11.87 ± 0.79	0.09	2.39	12934.89
	Engelman	-392.30	-385.42 ± 7.39	11.37	12.34 ± 0.59	-0.10	2.40	12947.56
3fxn	Alberts	-490.08	-474.30 ± 7.09	11.25	12.54 ± 0.87	-0.19	2.61	21301.80
	Kyte	-441.65	-431.24 ± 8.38	11.26	12.70 ± 0.88	0.26	2.66	21431.10
	Wimley	-419.11	-414.39 ± 6.44	12.42	13.33 ± 0.59	-0.13	2.64	21170.00
	Hessa	-431.96	-422.38 ± 8.31	11.53	12.89 ± 0.95	0.17	2.63	21244.50
	Eisenberg	-500.49	-483.85 ± 11.16	11.01	13.06 ± 1.03	0.45	2.69	21351.30
	Janin	-484.32	-473.74 ± 7.03	12.23	13.45 ± 0.67	0.11	2.60	21206.00
	Engelman	-515.18	-500.94 ± 7.51	11.11	12.67 ± 0.73	0.21	2.62	21246.40

Source: Own work.

ted by the lengths of vertical lines drawn from the edges of the boxes to upper and lower adjacent values (MASON; GUNST; HESS, 2003).

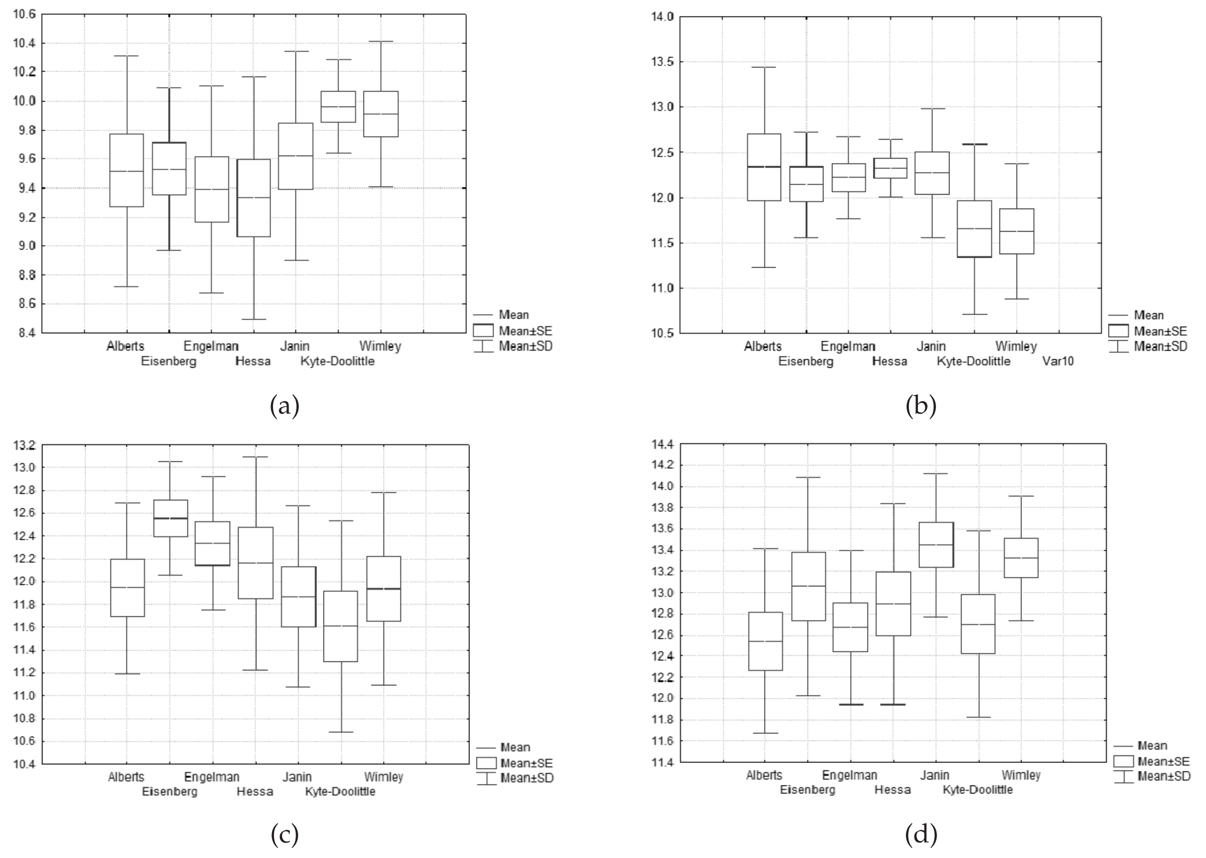


Figure 33: Comparative boxplots – RMSD for conversion tables: (a) 2gb1, (b) 1pcy, (c) 2trx and (d) 3fxn

Source: Own work.

The informations displayed in the boxplots suggest that the conversion table obtained from the hydrophobicity scale introduced by (HESSA et al., 2005), (WIMLEY; WHITE, 1996), (KYTE; DOOLITTLE, 1982) and (ALBERTS et al., 2002) lead to the best results for sequences 2gb1, 1pcy, 2trx and 3fxn, respectively, regarding the average RMSD measure. However, there is no strong evidence to suggest that these are the most suitable conversion table for the PFP using the AB model. Thus, it is necessary to perform a statistical test in order to verify if there is a statistically significant difference between the results obtained using the conversion tables and, consequently, to identify the conversion table that best fits to the PFP using the AB model. Hence, we performed a single-factor analysis of variance (ANOVA) (MONTGOMERY, 2001) of the results obtained. The ANOVA is the common statistical method for testing the differences between more than two samples, representing the appropriate procedure for testing the equality of several means (MONTGOMERY, 2001; FISHER, 1959). The null hyphotesis for ANOVA test states equality of the means between the data. That is $H_0 : \mu_1 = \mu_2 = \mu_3 = ... = \mu_k$, where H_0 is the null hyphotesis and μ_i represent the means. On the other hand, the alternative hyphotesis is defined as the negation of the null hypothesis, that is, $H_1 : \mu_i \neq \mu_j$ for at least one pair (i, j).

Table 16 reports the results obtained for the ANOVA test. In this table, second, third, fourth and last columns represent the F test statistic, the $P-value$, the F critical value and the significance level (α). F represents the test statistic for the hypothesis of no differences in treatment means (MONTGOMERY, 2001). We should reject the null hypothesis (H_0) and conclude that there are differences between the means if $F > F_{critical}$. We could also use the $P-value$ approach for decision making, where the null hypothesis is rejected if this probability is less than or equal to the significance level (i.e. $P-value \leq \alpha$). In Table 16, it is observed that $F < F_{critical}$ and $P-value > \alpha$. Thus, we do not reject H_0 and conclude that the means do not differ; that is, there is no significant difference between the results obtained using the conversion tables. Therefore, the conversion tables are equally suitable for the PFP using the AB model for sequences 2gb1, 1pcy, 2trx and 3fxn. Hence, the classification by (ALBERTS et al., 2002) was chosen to be used in the remaining experiments.

Table 16: Results obtained of the ANOVA. Source of variations: Between groups

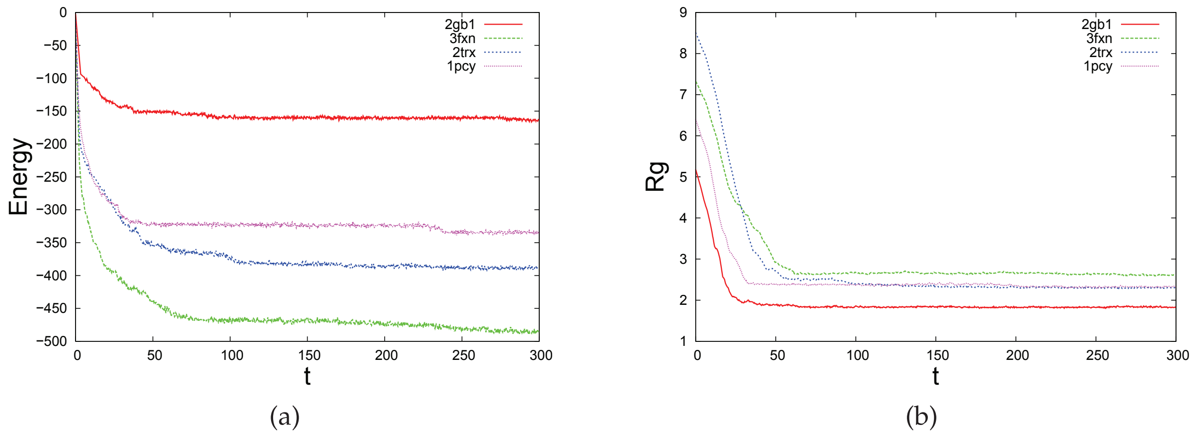
Protein	F	$P-value$	$F_{critical}$	α
2gb1	1.360	0.245	2.246	0.05
1pcy	1.595	0.166	2.266	0.05
2trx	1.483	0.201	2.266	0.05
3fxn	1.722	0.130	2.246	0.05

Source: Own work.

4.1.4 TIME DEPENDENCE OF PHYSICAL QUANTITIES

Figures 34(a), 34(b) and 34(c) show the time dependence of the total energy of the best conformation of each sequence, radius of gyration of the best conformation of each sequence and the radius of gyration of the hydrophobic (Rg_H) and hydrophilic (Rg_P) residues of protein 2gb1, using the conversion table following the classification by (ALBERTS et al., 2002). Such plots confirm the Anfinsen's thermodynamic hypothesis, where a denatured conformation has high energy and folding to the native state, the free energy of the protein decreases significantly.

Fluctuations in radius of gyration (R_g) provide a sensitive way to characterize such conformational motions of proteins (PICKOVER, 2001). Notice that, in Figure 34(c), it is possible to observe the formation of a compact hydrophobic core, surrounded by polar residues, during folding because the radius of gyration of the hydrophobic residues is much lower than that of the polar residues (that is, $Rg_H < Rg_P$). Also, in the maximized plot of this figure, it is observed an evidence of conformational fluctuations of the protein structure (large groups of atoms in the protein move in unison). These fluctuations are known as breathing motion (PICKOVER, 2001; RAMANATHAN; AGARWAL, 2009).



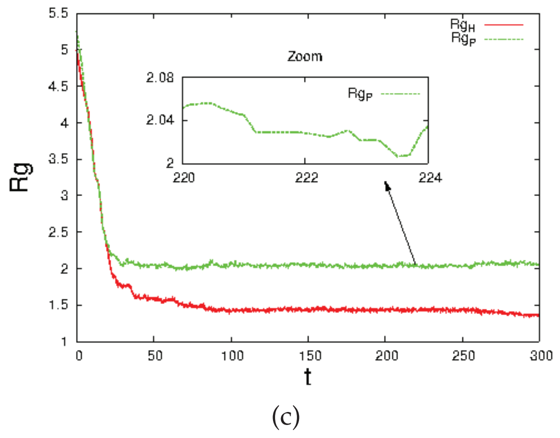


Figure 34: Properties (in dimensionless MD units): (a) Energy of the best conformation of each sequence, (b) radius of gyration of the best conformation of each sequence and (c) represent the radius of gyration of the hydrophobic (Rg_H) and hydrophilic (Rg_P) residues of sequence 2gb1.

Source: Own work.

4.1.5 PROCESSING TIME

Figure 35 shows the average processing time of the MD simulations for different protein sizes. In this figure, it is observed that the processing time would grow quadratically with larger proteins. It strongly suggests the need for high performance approaches for dealing with this problem. With the advantage of parallel processing, it will be possible to simulate several folding pathways, which could allow the exploration of the energy landscape of the AB model.

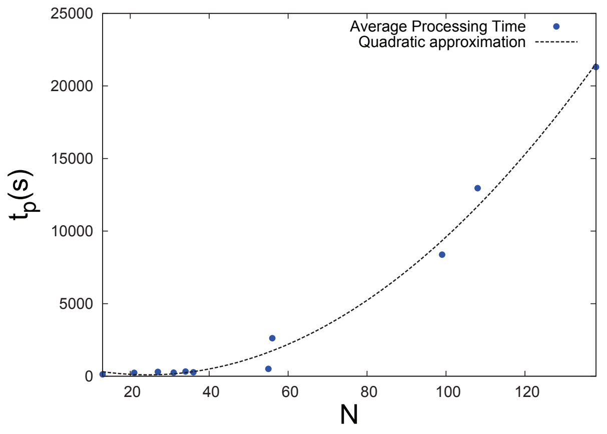


Figure 35: Average processing time t_p(s) for different protein sizes N

Source: Own work.

4.2 CONTACT MAPS GENERATION

The performance of the pGEP-CA and pECO, described in Sections 3.2 and 3.3, respectively, was evaluated using Contact Maps (CMs) which, in turn, were generated from 3D protein structures obtained by Molecular Dynamics (MD) simulations.

Figure 11(c) (page 80) shows the overall CM generation procedure. As shown in Section 4.1.4, the 3D structures were generated for four protein sequences with different lengths (N), previously presented in Table 7, 2gb1 ($N = 56$), 1pcy ($N = 99$), 2trx ($N = 108$) and 3fxn ($N = 138$) using the the amino acid type classification proposed by (ALBERTS et al., 2002), by MD simulations with time-step: $\delta t = 0.0001$ and stop criterion: $t_{max} = 300$, leading to 3×10^6 folding states for each protein sequence.

From the structures obtained by MD simulations at equal intervals of time between 0 and t_{max}, 100 CMs were generated for the following threshold values: 6.65, 7, 8, 9, 10, 11 and 12Å, as shown in Algorithm 8). The first value was obtained from the dimensionless value defined by (IRBACK et al., 1997). They stated that two monomers i and j are taken to be in contact if $r_{ij}^2 < 1.75$. Considering that the unity dimensionless distance is 3.8Å, 1.75 is equal to 6.65Å. The other ones are typical values of threshold considered in the literature. Thus, 700 CMs were generated for each protein sequence, and a total of 2800 CMs for the four sequences.

For instance, for the protein 2gb1, composed by 56 amino acids, each CM is a 56x56 matrix and represents a folding state of the folding process.

As commented before, the CMs generated following the procedure presented in this section were used in the computational experiments for the pGEP-CA and pECO which, in turn, will be presented later in Sections 4.3 and 4.4, respectively.

4.3 pGEP-CA – COMPUTATIONAL EXPERIMENTS

The contact maps (CMs) used in the experiments to validate the proposed approach – the pGEP-CA – were generated as described in Section 4.2. In the pGEP-CA, the CMs are represented by 2D-CAs, which in turn, are simulated using evolved (or induced) transition rules by GEP simulations. In order to evaluate the proposed approach, experiments were done using consecutive ith and jth CMs, which in turn, represent the initial 2D-CA configuration and the expected final 2D-CA configuration, respectively. The fitness of GEP individuals is computed using the expected CM (obtained by MD simulations) and the 2D-CA achieved using the rules obtained.

Due to the stochastic nature of the algorithm, 30 independent runs were done with different initial random seeds for each pair of CMs, totaling 103,950 experiments.

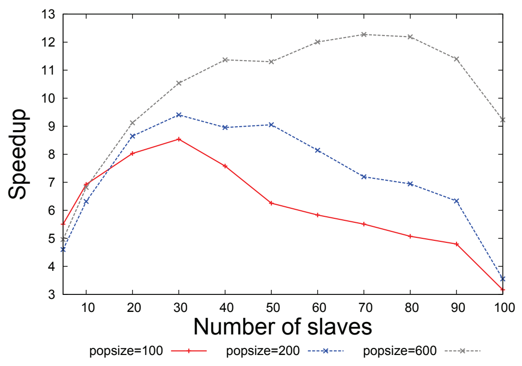
Section 4.3.1 presents a brief analysis of the load balancing and performance measures in the pGEP-CA. Next, Section 4.3.2 shows the running parameters used in the experiments with the CMs of the protein sequences. Then, the results obtained using the protein sequences 2gb1, 1pcy, 2trx and 3fxn are presented in Sections 4.3.2 and 4.3.3.

4.3.1 LOAD BALANCING AND PERFORMANCE MEASURES IN pGEP-CA

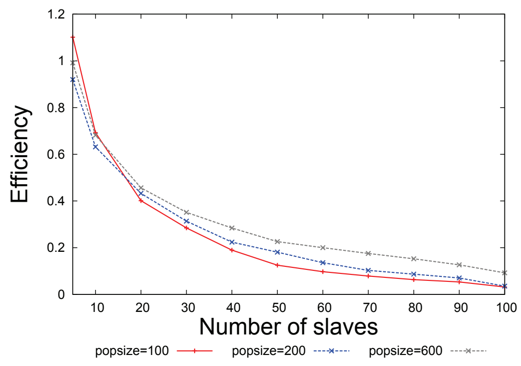
Since the best sequential version of the GEP-CA algorithm is not known, the "Strong speedup" measure cannot be used. Thus, we shall use the "Versus panmixia" approach to evaluate the parallel implementation (the pGEP-CA), using the sequential version of the algorithm as a reference.

Figures 36(a) – 36(c) were generated using three different values of GEP population size ($popsize = [100; 200; 600]$), one 2D-CA interaction ($CAiter = 1$), number of slave processors varying from 5 to 100 ($s = 5; 10; 20; \cdots; 100$) and CMs of the protein sequence 2gb1 (with 56 amino acids). The basic parameters of GEP and CA algorithms are presented in Section 4.3.2.

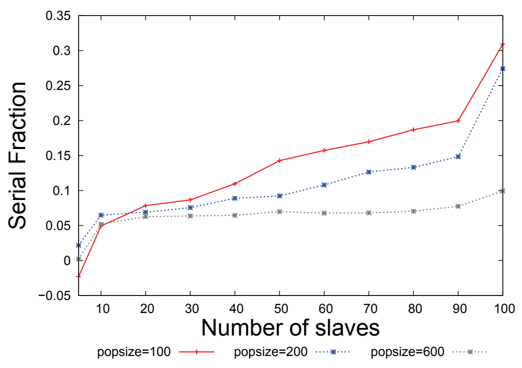
Figure 36(a) shows the speedup curve of the pGEP-CA. In this figure, it is observed that the speedup is always sublinear and tends to get apart from linear speedup as the number of slave processors increase.



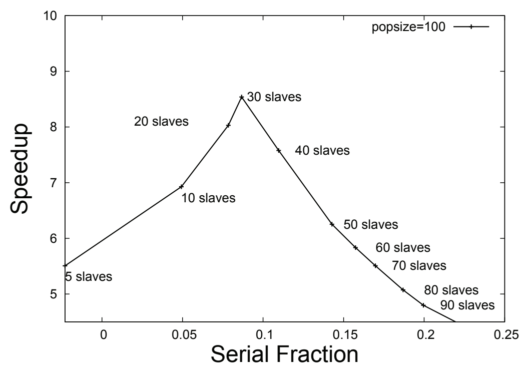
(a)



(b)



(c)



(d)

Figure 36: pGEP-CA performance measures using Versus Panmixia approach: (a) Speedup (b) Efficiency (c) Serial Fraction and (d) Speedup vs. Serial Fraction with $popsize = 100$

Source: Own work.

Figure 36(b) shows the efficiency curve of the pGEP-CA. Ideally, efficiency should be close to the unity. However, in practice, this is not always possible, since processors are not used 100% of time for processing, but also for communication, memory allocation and other tasks of the underlying operating system (ROOSTA, 1999). It is observed a loss of efficiency as the number of slave processors increase. Basically, this behavior is expected, due to the communication overload caused in the master processor. Also, a perfect load balancing between processors is not possible in some situations, since the number of GEP individuals to be processed divided by the number of slave processors is not an exact number. This may cause a loss of efficiency since, in a given instant of time, there might be some slaves working and others waiting.

It is also important to recall that both the speedup and efficiency may increase with the number of individuals of the GEP population, the size of the CMs, the number of 2D-CA interactions and the size of the neighboorhood (e.g. the Moore neighboorhood), because the processing load due to the fitness computation of the overall population will increase, as shown in Figures 36(a) and 36(b).

Figure 36(c) shows the serial fraction (f_m) curve of the pGEP-CA. The increase of the serial fraction with the number of slaves indicates that the granularity of the parallel processes decreases. In other words, the higher the number of slaves, the finer the granularity is. It is also possible to observe that a super-linear speedup occurs, when the pGEP-CA has 100 individuals and 5 slaves due to the negative value of the f_m.

Theoretically, the less the value of f_m, the better the parallelization is. However, Figure 36(d) shows that smaller values of serial fraction are better if they result in higher speedup. It is also possible to observe that the increase of the number of processors may lead to higher values of serial fraction, resulting in larger load imbalance.

4.3.2 NUMERICAL RESULTS

The running parameters for the pGEP-CA were defined empirically and they are shown in Table 17. A better understanding of the relationship between the parameters and the performance of the algorithm is quoted as an important future work to address.

Tables 20 – 23 show the results obtained using the CMs of the protein sequences 2gb1 ($N = 56$), 1pcy ($N = 99$), 2trx ($N = 108$) and 3fxn ($N = 138$). In order to facilitate the reading, these tables are shown in Appendix A. In these experiments, the fitness function $fitness_1$ (see Equation 35) was used. In these tables, the first column shows the metrics of the best individuals: T_C, T_{NC}, F_C and F_{NC} represent the number true contacts, true non-contacts, false contacts and false non-contacts, respectively. S_C, S_{NC} measure the ability of a *transition rule* to generate correct *contacts* and *non-contacts*, respectively. S_i measures the symmetry of the CM. Next columns show their values for each threshold value. We can observe, from the $fitness$, S_C and S_{N_C} values, that the induced transition are able to generate 2D-CAs with correct contacts and non-contacts, despite

Table 17: pGEP-CA parameters

	Parameter	Description	Value
GEP parameters	*popsize*	Population size	100
	Maxgen	maximum number of generations	350
	link	linking function	AND
	F	function set	[AND, OR, NOT]
	T	terminal set	$[a, b, c, d, e]$
	N	Number of classes	2 ('0' and '1')
	n_g	number of genes per chromosome	2
	h	size of the head	10
	sel	selection method	stochastic tournament
	Tourney size	tourney size	10% of *popsize*
	pcross	recombination probability	0.8
	pmut	mutation probability	0.1
	ptIS	IS transposition probability	0.7
	ptRIS	RIS transposition probability	0.7
	pgt	genic transposition probability	0.7
CA parameters	Σ	set of possible states of a cell	[0,1]
	η_i^t	neighboorhood	2D-CA von Neumann neighboorhood with $r = 1, m = 5$
	l	length of the transition rule $\Phi(\eta_i^t)$	32
	CAiter	number of 2D-CA interactions	1 (one transition)

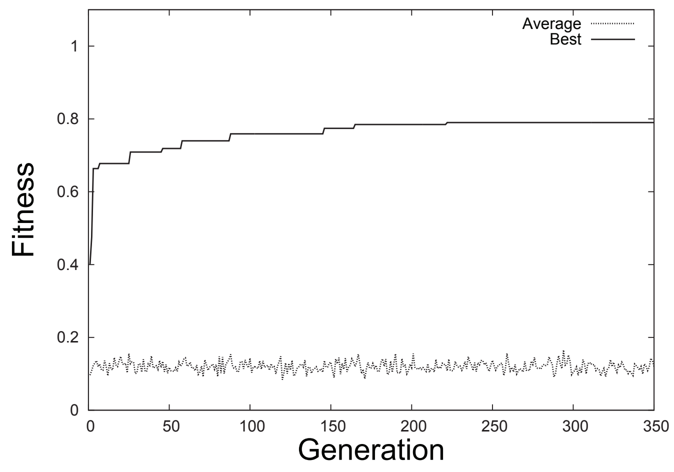
Source: Own work.

the F_C and F_{NC}. From the minimum values of S_C obtained, it is also important to recall that the ability of the obtained *transition rules* to generate correct *contacts* may decrease with the size of the CMs. For instance, it is possible to observe by comparing the minimum values of S_C obtained for the CMs of 2gb1 and 3fxn, specially with the threshold values 6.65Å, 7Å and 10Å.

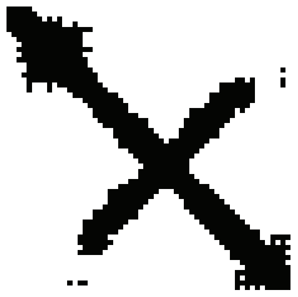
Table 24 (see Appendix A) shows the results obtained using the CMs of the protein sequence 2gb1 and the second proposed fitness function ($fitness_2$) which, in turn, is based on the Jaccard similarity coefficient (see Equation 41). Comparing the results obtained using both proposed fitness functions ($fitness_1$ and $fitness_2$) which are shown in Tables 20 and 24, it is possible to observe that both the proposed fitness functions are suitable to the problem and the fitness function $fitness_1$ is slightly better for CMs with lower threshold values (6.65 and 7Å).

4.3.3 GRAPHICAL RESULTS

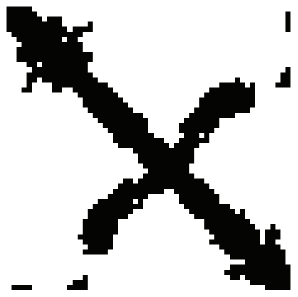
Figure 37(a) shows a plot of the fitness (best and average) obtained in a simulation. In this figure, it is observed that the genetic diversity is preserved along the run, since the distance from average to best is maintained throughout GEP generations. Figures 37(b) and (c) show a expected (final) CM and the CM generated using the induced rule ("$01111111011111110111111101111111_2$", with $fitness = 0.85$). The CM obtained suggests that the proposed fitness function is adequate to induce transition rules for evolving 2D-CAs, which in turn, represent CMs. Best results can be obtained, using a knowledge-based strategy to correct the drawbacks of the obtained CMs or using a specialized procedure for generating the initial population. For instance, a specialized decimation and diversification strategy may be used in order to improve the quality and diversity of the population, in the transition rule space (transition rules obtained from individuals), similar to the method called "Decimation-and-Hot-Boot" (SCAPIN; LOPES, 2007)(BENÍTEZ; LOPES, 2010).



(a)



(b)



(c)

Figure 37: (a) Example of performance of the pGEP-CA and Example of an obtained CM: (b) Expected CM (c) Obtained CM, where cells in states '0' and '1' are represented by white and black squares (or dots), respectively.

Source: Own work.

It may be possible to improve the obtained CMs using larger neighborhoods, such as the Moore or the extended Neumann neighborhoods as well as using a method for self-adjustment of parameters, such as REVAC (NANNEN; EIBEN, 2007). It is quoted as an important future work to address.

4.4 pECO COMPUTATIONAL EXPERIMENTS

The Contact Maps (CMs) used in the experiments to validate the proposed approach applied to the reconstruction of protein structures from CMs (pECO) were generated as described in Section 4.2. The CMs obtained by pGEP-CA simulations were also used in pECO computational experiments.

Sections 4.4.1 and 4.4.2 present the pECO control parameters and the parameters of each algorithm, respectively. As commented in Section 4.3.2, it is important to recall that the parameter adjustment is not the focus of the present work and it is quoted as an important future work to address.

4.4.1 pECO CONTROL PARAMETERS

The parameters used for the pECO algorithm are: number of populations (*NQ*) that will be co-evolved, the initial population size (*POP*), number of cycles for ecological successions (*ECO-STEP*), the size of the evolutive period (*EVO-STEP*) that represents the number of function evaluations in each *ECO-STEP*, the minimum threshold distance (ρ), and the tournament size (*T-SIZE*) used to choose solutions to perform intra and inter-habitat communications. The values for these parameters were defined empirically with: *NQ* = 40, *POP* = 50, *ECO-STEP = 2,000*, *EVO-STEP = 100* , $\rho = 0.5$ and *T-SIZE = 5*. The heterogeneous model of the pECO approach, combines all four algorithms (ABC-PSO-DE-jDE/BBO) in which 1/4 of the populations behaves according to one of these strategies. In this work, the number of processors (m) is equal to the number of populations ($m = NQ$).

4.4.2 PARAMETERS OF THE ALGORITHMS

Default parameters recommended in the literature were used in the algorithms employed. *POP* is a common parameter between all algorithms and is adjusted as mentioned in Section 4.4.1. For ABC algorithm, there is only one control parameter, *limit = 100* (KARABOGA; AKAY, 2009). For PSO algorithm, besides *POP*, the parameters were set to standard values[6]: inertia weight $W = 0.721$; cognitive and social components $\varphi_p = \varphi_g = 1.193$, respectively. For DE algorithm, the parameters are $F = 0.9$ (F controls the amplification of the differential variation) and $CR = 1.0$ (crossover constant) with DE/rand/1/bin. And for jDE/BBO the parameters used are $I = E = 1.0$ (maximum possible immigration and emigration rates), $CR = 0.9$, $F = 0.5$, and $S_{max} = POP$ (GONG; CAI; LING, 2010).

4.4.3 LOAD BALANCING AND PERFORMANCE MEASURES IN pECO

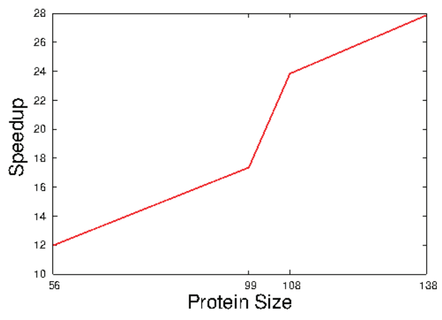
Figures 38(a), 38(b) and 38(c) show the performance measures of the pECO approach. The "Versus panmixia" approach is used to evaluate the parallel implementation (the pECO), using the sequential version of the algorithm as a reference. In Figure 38(a), a sublinear speedup ($s_m < m$, where $m = NQ = 40$) behavior can be clearly identified. Recall that a speedup higher than one suggests that the parallelization of the algorithm decreases the overall computational cost. Ideally, the speedup should be linear, but this is not possible in practice, since processors are not used only for processing, but also for other tasks such as for *message-passing* communication between them. It is also possible to observe that the speedup increases with the protein size. This is due to the relatively high time needed to transmit data between processes for small proteins, when compared with the processing load. Therefore, it is necessary to establish a load balance between the processing and communication loads between processes. Better speedups can be achieved for larger proteins.

Figure 38(b) presents the computed efficiency. These values suggest that the processors are not fully used all the time. In fact, speedup and efficiency are a direct consequence of the balance between the processing load of the slaves and the communication load between master and slaves. The serial fraction is shown in Figure 38(c) and indicates that the granularity of the parallel approach decreases when increasing the protein length. Thus, the approach is more efficient for larger protein sequences.

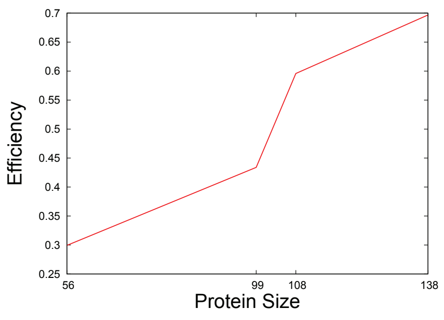
4.4.4 NUMERICAL RESULTS

Table 18 presents the results obtained through pECO simulations, using CMs of the 2gb1 ($N = 56$) sequence, which were generated by MD and pGEP-CA simulations. The first column shows the metrics of the best individuals. Next columns show

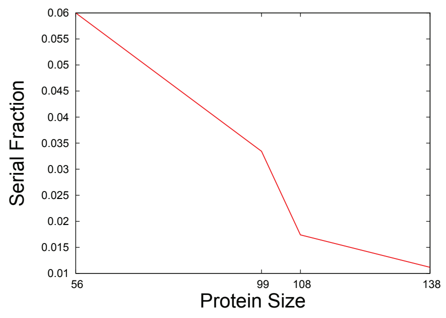
[6]Standard PSO (SPSO-07): http://www.particleswarm.info/Programs.html



(a)



(b)



(c)

Figure 38: Plots for the Versus Panmixia performance measures according to the protein size Source: Own work.

their values for each threshold value. It is important to recall that the S_C, S_{NC} metrics and the RMSD are computed using the input CM and the CM of the obtained conformation. Overall, from the S_C and RMSD values, it is possible to observe that better results are obtained for larger threshold values, since the S_C increases and the RMSD decreases when increasing the threshold. Comparing the obtained results shown in Table 18 from the RMSD point of view, it is observed that the results are slightly worse using the CMs obtained by pGEP-CA simulations due to the error in the CMs.

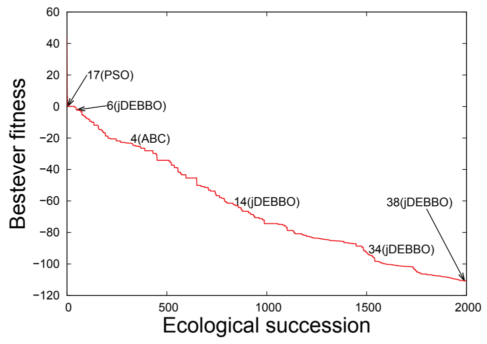
In addition, experiments were done using CMs of the 1pcy ($N = 99$), 2trx ($N = 108$) and 3fxn ($N = 138$) sequences obtained by MD simulations (with threshold of 7Å). The obtained results are shown in Table 19. As expected, from the S_C, RMSD and processing time values, it is possible to observe that the performance of the approach decreases with the protein length, since the search space complexity grows exponentially.

4.4.5 GRAPHICAL RESULTS

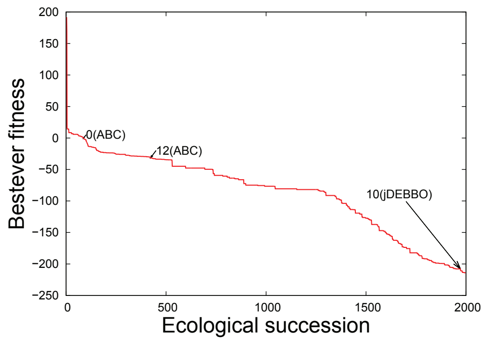
Figures 39(a), 39(b), 39(c) and 39(d) show four examples of the convergence plot for the CMs of the sequences 2gb1 ($N = 56$), 1pcy ($N = 99$), 2trx ($N = 108$) and

3fxn ($N = 138$), respectively. In these figures, each label indicates the number of the species and its algorithm. In these figures, the x-*axis* shows the number of Ecological sucessions and the y-*axis* represents the best-ever fitness value. Analyzing these plots it is observed that for the 56 and 99 amino-acids-long sequences the convergence is not accentuated in the direction of a stagnation point during the ecological successions. Thus, best solutions may be achieved increasing the number of ecological successions. For the 108 and 138 amino-acids-long sequences the convergence seems to be slow and attracted to a basin region of local minima. It is possible to verify that small improvements are achieved from half of the ecological successions forwards. These convergence plots indicate that, in order to improve the results, strategies for maintaining diversity inside populations are required as well as a method to detect and escape from the attraction basin regions of local minima. Also, these figures show some labels indicating which algorithm achieved the best solution at each ecological succession. Once a different algorithm updates the best solution, a new label is added. For example, for the 56 amino acids-long sequence a population with the PSO algorithm achieved the best solution until around succession 20, where the 17^{th} species found the best solution. From successions 21 to around 49 a population with the jDE/BBO algorithm (the 6^{th} species) achieved the best solution; from successions 319 to 345 a population with the ABC algorithm (the 14^{th} species) achieved the best solution, and from successions 370 to 2,000 different populations with the jDE/BBO algorithm achieved the best solution. Analysing these labels, it is possible to notice the coevolution between the different search strategies (ABC/PSO/DE/jDE-BBO) because they alternate in finding the best solutions. Possibly this is due to the peculiarity of each method in searching the space of solutions.

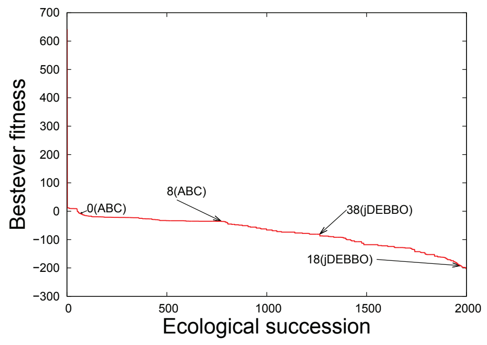
Figures 40 (a), 40 (b), 40 (c) and 40 (d) show four examples of the evolution of the number of habitats for each ecological succession step for the CMs of the sequences 2gb1 ($N = 56$), 1pcy ($N = 99$), 2trx ($N = 108$) and 3fxn ($N = 138$), respectively. It is observed that, at the beginning of the optimization process, with the populations widely dispersed in the search space, there is a large number of habitats. As the optimization process moves through the ecological successions, the populations tend to move through the search space converging to specific regions. As shown in these figures, the number of habitats decreases with the ecological succession cycles, indicating that the populations tend to converge to points close to each other. Overall, due to the high complexity of the problem, the populations are dispersed through the search space during all successions. This indicates that more ecological successions the pECO



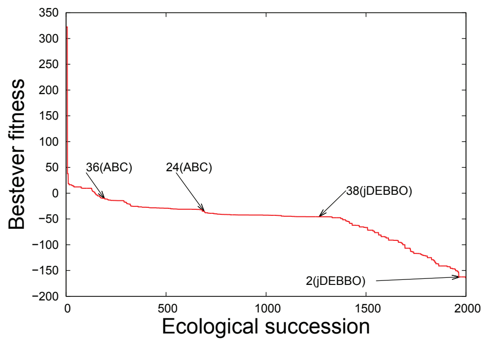
(a) Convergence for the 56 amino-acids-long sequence



(b) Convergence for the 99 amino-acids-long sequence



(c) Convergence for the 108 amino-acids-long sequence



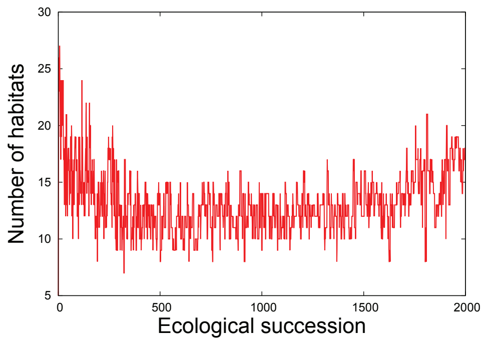
(d) Convergence for the 138 amino-acids-long sequence

Figure 39: Plots for the pECO convergence.

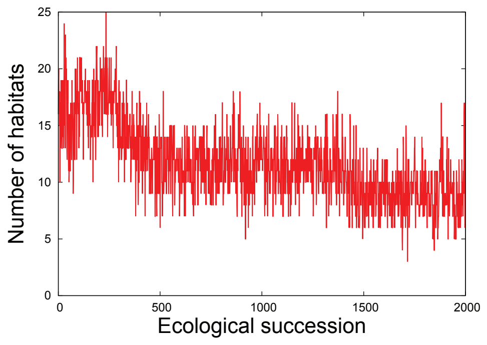
Source: Own work.

approach could lead to even better results.

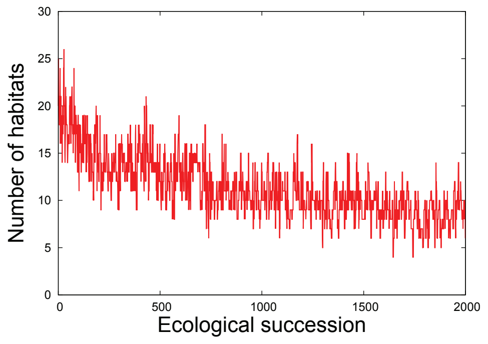
Figures 41(a), 41(b), and 41(c) show the best values of the the metrics S_C, S_{NC} and RMSD obtained for each CM of the 2gb1 sequence with different threshold values (for the sake of simplification, in these plots, only the results for three different threshold values are shown). Figure 41(a) shows that higher values of S_C are obtained using CMs with higher threshold values. Basically, it indicates that the approach obtained better structures using CMs with more *contacts*. On the other hand, Figure 41(b) shows that better values of S_{NC} are obtained for CMs with lower threshold. In Figure 41(c), it is possible to observe that lower RMSD values were obtained for higher threshold values. This indicates that the *contacts* of the structures obtained are more important than the *non-contacts* in the process of structure reconstruction. Overall, better results are obtained for CMs with higher threshold values.



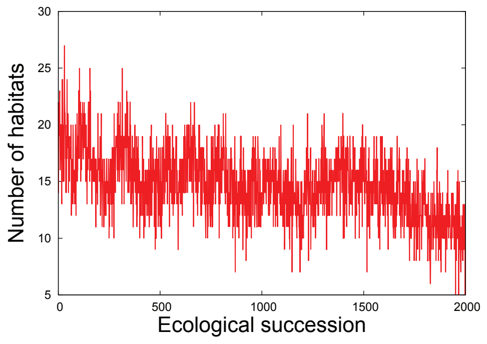
(a) Convergence for the 56 amino-acids-long sequence



(b) Convergence for the 99 amino-acids-long sequence



(c) Convergence for the 108 amino-acids-long sequence



(d) Convergence for the 138 amino-acids-long sequence

Figure 40: Plots for the pECO convergence: evolution of the number of habitats.

Source: Own work.

Table 18: Numerical results obtained using CMs with different threshold values (generated by MD and pGEP-CA simulations) – sequence 2gb1

Metric	Results using CMs generated by MD simulations			
	CM threshold [Å]			
Avg(*Min*/*Max*)	6.65	7	8	9
Best *fitness*	-88.68 (-120.59/-82.27)	-84.47 (-114.80/-77.56)	-83.51 (-108.90/-77.52)	-83.87 (-106.16/-76.22)
T_C	268.68 (138/296)	506.22 (296/562)	781.06 (396/856)	999.00 (456/1108)
F_C	112.94 (68/242)	172 (102/292)	194.94 (116/352)	191.70 (124/346)
T_{NC}	2600.8 (2562/2742)	2206.3 (2120/2554)	1813.88 (1694/2378)	1561.38 (1436/2320)
F_{NC}	153.58 (14/180)	251.48 (10/310)	346.62 (10/422)	383.92 (14/482)
S_C	0.64 (0.59/0.907)	0.67 (0.60/0.967)	0.697 (0.64/0.975)	0.73 (0.66/0.976)
S_{NC}	0.96 (0.91/0.975)	0.93 (0.89/0.96)	0.90 (0.86/0.94)	0.89 (0.84/0.94)
Energy	-144.74 (-152.86/-128.79)	-135.52 (-146.08/-124.15)	-132.78 (-143.76/-123.53)	-129.65 (-137.65/-115.72)
Kabsch RMSD [Å]	7.24 (5.22/11.96)	6.15 (3.31/11.91)	5.82 (3.41/11.04)	5.508 (3.12/7.43)
Metric	Results using CMs generated by pGEP-CA simulations			
	CM threshold [Å]			
Avg(*Min*/*Max*)	6.65	7	8	9
Best *fitness*	-83.32 (-105.66/-75.85)	-78.06 (-98.16/-69.06)	-76.83 (-96.56/-67.35)	-76.85 (-93.01/-64.32)
T_C	260.83 (234/288)	484.46 (394/548)	748.59 (516/826)	955.11 (614/1042)
F_C	114.89 (84/152)	182.74 (124/256)	202.79 (104/320)	208.32 (132/338)
T_{NC}	2593.03 (2527/2675)	2189.57 (2120/2365)	1798.18 (1670/2277)	1537.59 (1368/2200)
F_{NC}	167.25 (84/219)	279.22 (121/344)	386.42 (87/492)	434.97 (92/542)
S_C	0.61 (0.54/0.75)	0.64 (0.55/0.79)	0.66 (0.58/0.86)	0.69 (0.61/0.87)
S_{NC}	0.96 (0.94/0.97)	0.92 (0.89/0.95)	0.89 (0.83/0.95)	0.88 (0.80/0.92)
Energy	-142.60 (-153.04/-132.91)	-133.02 (-142.26/-120.08)	-129.21 (-142.84/-118.56)	-126.59 (-136.45/-111.33)
Kabsch RMSD [Å]	7.43 (5.72/10.67)	6.62 (4.83/10.82)	6.14 (4.03/9.62)	6.01 (4.08/7.86)
Avg t_p(s)	304	295.49	330.996	330.66

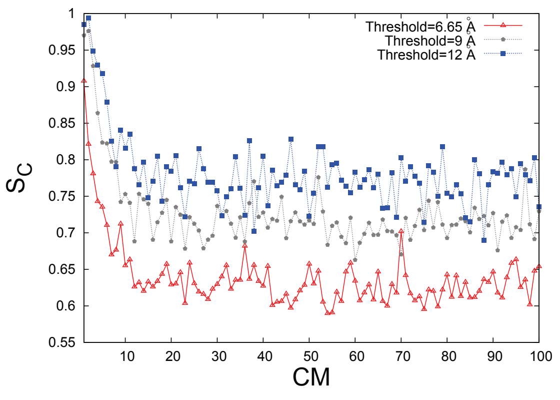
Metric	Results using CMs generated by MD simulations		
	CM threshold [Å]		
Avg(*Min*/*Max*)	10	11	12
Best *fitness*	-82.33 (-104.48/-74.09)	-80.59 (-103.29/-70.60)	-79.93 (-102.86/-64.15)
T_C	1210.34 (554/1310)	1506.16 (670/1634)	1734.00 (776/1918)
F_C	185.88 (106/306)	151.06 (82/328)	115.62 (66/380)
T_{NC}	1325.48 (1172/2260)	1010.18 (860/2130)	791.66 (664/1980)
F_{NC}	414.30 (16/552)	468.60 (8/622)	494.72 (6/712)
S_C	0.75 (0.67/0.97)	0.76 (0.70/0.99)	0.78 (0.69/0.99)
S_{NC}	0.88 (0.80/0.94)	0.87 (0.80/0.93)	0.87 (0.80/0.92)
Energy	-125.46 (-137.60/-116.01)	-120.98 (-130.76/-109.92)	-117.13 (-128.19/-105.53)
Kabsch RMSD [Å]	5.61 (3.41/7.31)	5.55 (3.64/7.27)	5.75 (4.14/7.18)
Metric	Results using CMs generated by pGEP-CA simulations		
	CM threshold [Å]		
Avg(*Min*/*Max*)	10	11	12
Best *fitness*	-73.47 (-93.56/-50.21)	-67.92 (-95.42/-43.86)	-60.34 (-98.24/-36.40)
T_C	1146.04 (730/1306)	1379.79 (834/1602)	1526.26 (956/1866)
F_C	192.97 (100/282)	143.31 (58/306)	105.37 (38/312)
T_{NC}	1315.02	1018.55 (840/1917)	806.75 (672/1803)
F_{NC}	481.97 (86/704)	594.34 (79/934)	
S_C	0.708 (0.59/0.91)	0.703 (0.54/0.91)	0.692 (0.54/0.97)
S_{NC}	0.87 (0.80/0.93)	0.88 (0.79/0.95)	0.89 (0.79/0.96)
Energy	-119.09 (-130.99/-94.00)	-109.78 (-128.04/-85.11)	-97.67 (-120.23/-73.92)
Kabsch RMSD [Å]	5.98 (3.56/8.21)	6.26 (4.29/9.01)	6.38 (4.22/9.64)
Avg t_p(s)	324.70	340.39	294.70

Source: Own work.

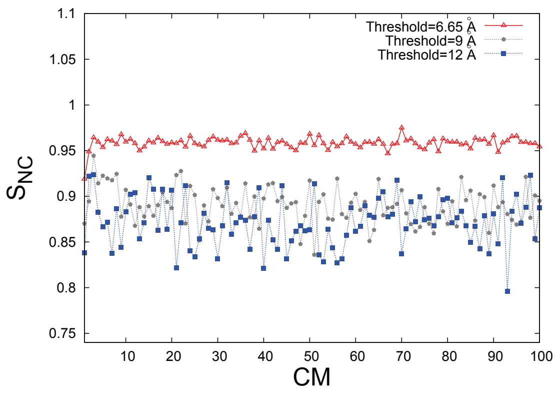
Table 19: Numerical results obtained using CMs with threshold = 7Å– sequences 1pcy, 2trx and 3fxn

Metric	Sequence		
Avg(*Min*/*Max*)	1pcy	2trx	3fxn
Best *fitness*	-114.72 ± 19.26 (-221.59/-90.79)	-99.72 ± 22.34 (-240.53/-67.75)	-81.55 ± 20.64 (-207.33/-56.72)
T_C	765.78 ± 43.95 (450/840)	741.64 ± 41.41 (556/828)	824.4 ± 52.82 (592/948)
F_C	377.84 ± 69.68 (230/692)	351.50 ± 100.35 (138/690)	248.5 ± 84.59 (94/752)
T_{NC}	7919.02 ± 155.93 (7709/8629)	9651.2 ± 178.71 (9382/10374)	16628.88 ± 231.40 (16346/17588)
F_{NC}	738.36 ± 144.13 (30/852)	919.66 ± 170.84 (44/1094)	1342.22 ± 243.04 (148/1564)
S_C	0.52 ± 0.08 (0.46/0.94)	0.45 ± 0.08 (0.38/0.93)	0.39 ± 0.07 (0.32/0.8)
S_{NC}	0.95 ± 0.008 (0.93/0.97)	0.96 ± 0.01 (0.94/0.98)	0.98 ± 0.005 (0.96/0.99)
Energy	-232.13 ± 12.47 (-259.61/-195.51)	-226.66 ± 21.44 (-276.83/-179.11)	-211.72 ± 20.79 (-270.26/-168.96)
Kabsch RMSD [Å]	10.22 ± 2.19 (6.92/23.51)	12.56 ± 2.96 (8.39/27.44)	20.16 ± 4.27 (12.14/32.39)
Avg t_p(s)	496.01	551.08	771.41

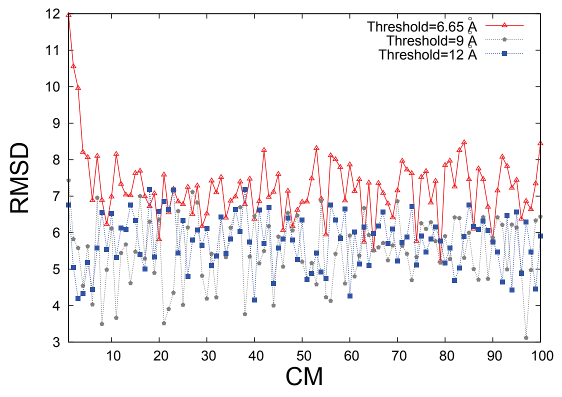
Source: Own work.



(a)



(b)



(c)

Figure 41: Best values of the metrics S_C (a), S_{NC} (b) and RMSD (c) for CMs with different threshold values

Source: Own work.

5 CONCLUSIONS AND FUTURE WORKS

"Art is never finished, only abandoned."
— Leonardo da Vinci

5.1 CONCLUSIONS

The PFP is still an open problem in Bioinformatics for which there is no closed computational solution. Even using the simplest model, the computational approach for searching a solution for the PFP was proved to be *NP*-complete.

In this thesis, we proposed computational approaches based on Cellular Automata, Bioinspired Computation and Parallel Computing applied to the protein folding pathways simulation and the protein structure prediction. The main idea is to simulate the folding process using a minimalistic representation of protein structures, known as Contact Maps (CM), represented by two-dimensional Cellular Automata (2D-CA), instead using traditional mathematical methods, such as the Molecular Dynamics (MD) simulations, and three-dimensional protein structures. The computation of the 2D-CA is simple, when compared to the MD. However, finding *transition rules* for simulating the protein folding is very difficult. Therefore, a novel approach based on Gene Expression Programming (GEP) for inducting *transition rules* of 2D-CA for simulating the protein folding is proposed in this thesis.

A procedure for reconstructing the three-dimensional structure of proteins is need after the CM prediction. Therefore, a novel ecological-inspired approach is also proposed for this issue.

To the best of our knowledge, this work presents the first implementation of MD for the 3D-AB *off-lattice* model. This approach was used to simulate the folding process and generate the structures and CMs that were used for evaluating the performance of the proposed approaches. It is believed that the use of MD for the PFP using a coarse-grained model is promising for this area of research. Therefore, an important contribution of this thesis are the results regarding this issue.

Since the 3D-AB *off-lattice* model is subject to geometrical constraints, we used the Shake algorithm to deal with them. The results indicates that this method is suitable for the problem.

The 3D-AB *off-lattice* has been sparsely approached, even more using Evolutionary Computation. As a consequence, there are few benchmarks for this model. Therefore, this work also offered new reference values for benchmark sequences that can be used in the future by other researchers for testing computational approaches applied to this problem.

Synthetic and real protein sequences with different sizes were used for evaluating the performance of the MD approach, where the real ones were extracted from PDB according to their size. In this work, only the sequence and the coordinates of the amino acids of the protein were required. Thus, a procedure for reading the sequence and the real coordinates of the central carbon atoms from the PDB files was implemented. In order to convert the protein sequences into the AB model alphabet, we used seven amino acid type classifications as conversion tables. Many experiments were done in order to verify their adequacy, taking into account the quality of the solutions obtained as well as the correlation between the estimated energy and RMSD values. Overall, the results showed that the conversion tables are equally suitable for the PFP using the AB model for the selected sequences (see Table 16).

By using a method of weak coupling to a thermal bath, the temperature dependence of the protein folding under different conditions was analysed. A denaturation process was observed when the temperature increases. Moreover, the observation of the time dependence of physical quantities confirmed the Anfinsen's thermodynamic hypothesis and it showed some biological properties of real proteins during folding. For instance, it was observed that a hydrophobic core is formed, partially surrounded by polar amino acids, during the folding process because the radius of gyration of the hydrophobic residues is much lower than that of the polar residues. Also, an evidence of conformational fluctuations was verified. Therefore, it is possible to conclude that the proposed approach is suitable for the problem.

Finally, it is important to recall that the three-dimensional structures obtained through MD simulations were converted into Contact Maps for evaluating the performance of the other approaches.

The process of 2D-CA transition rules induction for simulating dynamic behaviors is still an open research problem. Therefore, the approach presented in this thesis represents an important contribution regarding this issue. This thesis also contributes significantly to Bioinformatics, presenting the first implementation of a parallel computational approach based on GEP and CA applied to the prediction of Contact Maps.

There are two basic issues for applying GEP for a given optimization problem. The first issue is the representation of the problem, and the latter, the evaluation problem. It is known that the way variables are encoded has a significant influence in the dynamics and efficiency of GEP. Therefore, we used a parsimonious encoding, where the size of the individual can be chosen *a priori* and does not depend on the size of the transition rule. Moreover, it is important to mention that the encoding does not depend on the size of the proteins. Basically, the encoding of the individuals is defined according to the set of terminals and their domains which, in turn, are defined by the possible states of the cells of the 2D-CA. Then, the individuals are directly translated into expression trees for generating the resulting transition rules.

Since the Contact Maps are sparse symmetric matrices, specialized fitness functions were proposed. Experiments were done ir order to evaluate their adequacy and it was observed that they are suitable for the problem. Overall, this work showed that GEP can be an efficient way to deal with the problem. For instance, it was observed, from the S_C and S_{N_C} values (see Appendix A), that the induced *transition rules* were able to generate CMs with correct contacts and non-contacts. Also, by comparing the minimum values of S_C obtained for the CMs of 2gb1 and 3fxn sequences, specially with the threshold values 6.65Å, 7Å and 10Å, it was observed that the ability of the obtained *transition rules* to generate correct *contacts* may decrease with the size of the CMs. Although the results obtained cannot be considered optimal, they are coherent with the model. Certainly, the results can be improved hybridizing the GEP algorithm with specialized strategies in order to keep high genetic diversity and explore the search space efficiently leading to better individuals, such as local search methods or coevolution with other Evolutionary Computation algorithms.

The reconstruction of protein structures from CMs is still an unsolved problem, which has been proved to be *NP*-hard. In this thesis, the performance of a parallel ecologically-inspired optimization algorithm (pECO) was analysed, under the task of reconstructing the structure of proteins from CMs, featuring the 3D-AB *off-lattice* model. Four population-based algorithms (ABC, PSO, DE, and jDE/BBO) were employed in an ecological heterogeneous model. It is possible to conclude that, in this problem case, these strategies are quite complementary, even during few successions. The convergence plots indicate that ABC and PSO algorithms are best suited for global search (initial ecological successions), whilst the DE and jDE/BBO algorithms are best suited for local search (final ecological successions). According to the obtained results, it was observed that a smooth convergence is achieved throughout the pECO simulations, avoiding stagnation of the search. Overall, from the S_C and RMSD values (see Table 18), it is possible to observe that better results (i.e. conformations with lower RMSD values) are obtained for CMs with larger threshold values, since the S_C increases and the RMSD decreases when increasing the threshold. Furthermore, due to the high complexity of the problem, it was observed that the populations were dispersed in the search space during ecological successions. This indicates that even better results would be found by increasing the number of successions or through the diversification of evolutive behaviors of the computational ecosystem, by inserting other algorithms. For instance, local-search strategies could be used to improve the quality of the obtained solutions.

An important drawback is regarding the processing time for the simulations. It is clear that there is an increase of processing time as the length of the protein grows in all the approaches presented in this thesis. This fact, by itself, strongly suggests that parallel processing is essential to allow us to obtain results in a reasonable processing time. Therefore, two parallel approaches were implemented in a Beowulf cluster to speed up processing time: the pGEP-CA and pECO. Also, a brief analysis of the load balancing of the parallel approaches is presented, based on the performance measures *speedup*, *efficiency* and *serial fraction*, that are a direct consequence of the balance between the processing load and the communication load between master and slaves processors. Ideally, the speedup should be linear, but it is not possible in practice, since processors are not used only for processing, but also for other tasks. In fact, it was observed that the obtained speedup is sublinear. For instance, the speedups achieved by the pECO approach were 11.98, 17.36, 23.84 and 27.87 for the 56, 99, 108 and 138 amino acid-long sequences, respectively, as shown in Figure 38(a). Although the pECO approach have shown good speedup, the processing time would increase for larger proteins, since the search space complexity grows. It is important to recall that the speedup can be improved including more processing cores. For instance, slave processors can be added for processing the fitness function of the individuals of each population, leading to a Hierarchical Parallel ECO algorithm (HpECO) which has three levels, based on our previous work for GAs (BENÍTEZ; LOPES, 2010). Basically, in the lower level single-population master-slaves search for potencial solutions for a given problem in the search space, where the processing load (fitness function computation) is divided into several slaves, under the coordination of a master which, in turn, represents a population. The next two levels are defined as in the pECO approach, where in the intermediate level the populations form habitats, and intra-habitats communication topologies take place and in the upper level, the habitats are connected through an inter-habitats communication topolopy. A master processor is also responsible for defining the communication topologies between populations and habitats. This combination aims at taking advantage of the benefits of both approaches. The implementation and analysis of the HpECO is a future work to address.

Another drawback is regarding the high number of user-defined parameters for the approaches. It is important to recall that there is no specific procedure for adjusting running parameters of Evolutionary Algorithms for a given problem (LOBO; LIMA; MICHALEWICZ, 2007) and that it represents one of the grand challenges of the Evolutionary Computation (EC) field. A strategy frequently used in the literature is setting a range for all important parameters of the algorithm and testing all possible combinations. This procedure is known as factorial experiment (BOX; HUNTER; HUNTER, 2005). Furthermore, Eiben and Smit (2011) present a conceptual framework for parameter tuning and provide a survey of tuning methods. Although self-adjustment of parameters tends to be more efficient than trial-and-error design and factorial experiments, this was not the focus of the present work. Although not optimal for any instances, the parameters used in the approaches presented in this thesis could be an initial reference to other researchers and they will be an important issue for future works.

It is also important to mention that a novel method for comparing the structures obtained is also proposed, using the 3D-AB *off-lattice* model and the Kabsch method in order to obtain the lowest RMSD.

In a broader sense, it is believed that the computational approaches proposed in this thesis are very promising for the research areas related to Cellular Automata, Evolutionary Computation, Molecular Dynamics and the Protein Folding Problem. Although there are interesting research directions that suggest the continuity of this work (see Section 5.2), the initial objectives were achieved satisfactorily. The contributions of this thesis were theoretical and methodological. Moreover, the investigated subjects have generated scientific publications that lead to some relevant contributions. These publications, presented in Section 1.4, also provides theoretical and technical insights for future developments.

5.2 FUTURE WORKS

There are research directions for future works. For instance:

- Future work will include simulations and analysis of folding pathways using structures built by MD simulations from real protein structures extracted from the PDB, using a more realistic coarse-grained model for proteins.
- MD simulations with different thermodynamic ensembles will be done in future works, such as the isothermal-isobaric (NPT – moles, pressure, temperature) ensemble.
- Regarding the high processing time of the MD with larger protein sequences, future work will also investigate other parallel versions for the MD approach, such as GPGPU (General Purpose Graphics Processing Units) and hardware-based accelerators (BENÍTEZ et al., 2011).
- Non-uniform Cellular Automata (NCA) (CATTANEO et al., 2009) with neighborhood definition of the cells changing in space and time could be investigated. Since it is likely that the state of some cells of the CM (represented by the 2D-CA) do not depend on other, the NCA can be suitable to the problem.
- In future works, an analysis of the space of transition rules will be done in order to develop strategies for improving the quality and diversity of the population, in the transition rule space (transition rules obtained from individuals).
- Knowledge-based strategies to correct the faults of the CMs obtained by CA simulations will be also addressed.
- The hybridization of the GEP with specialized strategies, such as local search methods and coevolution with other EC algorithms will be focused in future works.
- Considering the intrinsic parallelism of 2D-CAs, the application of Reconfigurable Hardware Computing for simulating their dynamic behavior, such as the

approach proposed in a our previous work (WEINERT et al., 2007), will be studied in future works.

- Different fitness functions based on binary similarity/dissimilarity measures, such as the Tanimoto and Hamann similarity coefficients, as well as knowledge-based operators can be implemented and analysed.

- Since the EC approaches involve a large number of parameter settings and design choices, efficient self-tuning or automated algorithm configuration mechanism will be studied. It could not only release the user for higher level tasks, but also offer the possibility of using suitable parameters for each step of the search process.

REFERENCES

ABU-DOLEH, A.; AL-JARRAH, O.; ALKHATEEB, A. Protein contact map prediction using multi-stage hybrid intelligence inference systems. **Journal of Biomedical Informatics**, v. 45, n. 1, p. 173–183, 2012.

ABYZOV, A. et al. Rigidfinder: a fast and sensitive method to detect rigid blocks in large macromolecular complexes. **Proteins**, v. 78, n. 2, p. 309–324, 2010.

ALBA, E. Parallel evolutionary algorithms can achieve super-linear performance. **Information Processing Letters**, v. 82, n. 1, p. 7–13, 2002.

ALBA, H. **Parallel Metaheuristics: A New Class of Algorithms**. New York, USA: Wiley-Interscience, 2005.

ALBERTS, B. et al. **Molecular Biology of The Cell**. New York, USA: Garland Science, 2002.

ALBRECHT, A. A.; KAPSOKALIVAS, L.; STEINHÖFEL, K. Uphill unfolding of native protein conformations in cubic lattices. **Journal of Computational Science**, v. 1, p. 6–12, 2010.

ALMEIDA, C.; GONÇALVES, R.; DELGADO, M. Hybrid immune-based system for the protein folding problem. In: **Proceedings of the 7th European conference on Evolutionary computation in combinatorial optimization**. Berlin, Heidelberg: Springer-Verlag, 2007. (EvoCOP'07), p. 13–24.

ALTSCHUL, S. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. **Nucleic Acids Research**, v. 25, n. 17, p. 3389–3402, 1997.

AMBRISH, R.; XU, D.; ZHANG, Y. A protocol for computer-based structure and function prediction. **Journal of Visualized Experiments**, v. 57, n. 1, 2011.

ANDREEVA, A. et al. Data growth and its impact on the scop database: new developments. **Nucleic Acids Research**, v. 36, n. 1, p. D419–D425, 2008.

ANFINSEN, C. Principles that govern the folding of protein chains. **Science**, v. 181, n. 96, p. 223–230, 1973.

ARKUN, Y.; GUR, M. Combining optimal control theory and molecular dynamics for protein folding. **PLoS ONE**, v. 7, n. 1, p. e29628, 2012.

ARNOLD, K. et al. The swiss-model workspace: a web-based environment for protein structure homology modelling. **Bioinformatics**, v. 22, n. 2, p. 195–201, 2006.

ATILGAN, A. et al. Anisotropy of fluctuations dynamics of proteins with an elastic network model. **Biophysics Journal**, v. 80, n. 1, p. 505–515, 2001.

ATKINS, J.; HART, W. On the intractability of protein folding with a finite alphabet. **Algorithmica**, v. 25, n. 2, p. 279–294, 1999.

BACHMANN, M.; ARKM, H.; JANKE, W. Multicanonical study of coarse-grained off-lattice models for folding heteropolymers. **Physical Review E**, v. 71, n. 3, p. 1–11, 2005.

BACKOFEN, R.; WILL, S.; BORNBERG-BAUER, E. Application of constraint programming techniques for structure prediction of lattice proteins with extended alphabets. **Bioinformatics**, v. 15, n. 3, p. 234–242, 1999.

BAHAR, I.; JERNIGAN, R. Inter-residue potentials in globular proteins and the dominance of highly specific hydrophilic interactions at close separation. **Journal of Molecular Biology**, v. 266, n. 1, p. 195–214, 1997.

BAKER, D. A suprising simplicity to protein folding. **Nature**, v. 405, p. 39–42, 2000.

BARAH, P.; SINHA, S. Analysis of protein folds using protein contact networks. **PRAMANA–Journal of Physics**, v. 71, p. 369–378, 2008.

BARTOLI, L. et al. The pros and cons of predicting protein contact maps. In: ZAKI, M.; BYSTROFF, C. (Ed.). **Protein Structure Prediction**. Totowa, NJ, USA: Humana Press, 2008. cap. 8, p. 199–217.

BATES, P. et al. Enhancement of protein modeling by human intervention in applying the automatic programs 3D-JIGSAW and 3D-PSSM. **Proteins**, v. 45, n. 5, p. 39–46, 2001.

BEGON, M.; TOWNSEND, C. R.; HARPER, J. L. **Ecology: from individuals to ecosystems**. 4th. ed. Oxford, UK: Blackwell Publishing, 2006.

BENI, G.; WANG, J. Swarm intelligence in cellular robotic systems. In: **NATO Advanced Workshop on Robots and Biological Systems**. Tuscany, Italy: Springer, 1989. p. 26–30.

BENÍTEZ, C.; LOPES, H. Hierarchical parallel genetic algorithm applied to the three-dimensional HP side-chain protein folding problem. In: **Proceedings of the IEEE International Conference on Systems, Man and Cybernetics**. Piscataway, USA: IEEE Press, 2010.

BENÍTEZ, C.; LOPES, H. Molecular dynamics for simulating the protein folding process using the 3D-AB off-lattice model. In: **Advances in Bioinformatics and Computational Biology**. Heidelberg: Springer, 2012, (Lecture Notes in Bioinformatics, 1). p. 61–72.

BENÍTEZ, C.; LOPES, H. Ab-initio protein folding using molecular dynamics and a simplified off-lattice model. **Journal of Bionanoscience**, v. 7, p. 391–402, 2013.

BENÍTEZ, C. et al. Reconfigurable hardware computing for accelerating protein folding simulations using the harmony search algorithm and the 3D-HP-side chain model. In: **International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP)**. Heidelberg: Springer, 2011, (Lecture Notes in Computer Science). p. 363–374.

BENÍTEZ, C.; PARPINELLI, R. S.; LOPES, H. A heterogeneous parallel ecologically-inspired approach applied to the 3d-ab off-lattice protein structure prediction problem. In: IEEE. **Computational Intelligence and 11th Brazilian Congress on Computational Intelligence (BRICS-CCI CBIC), 2013 BRICS Congress on**. Piscataway, USA, 2013. p. 592–597.

BENÍTEZ, C.; WEINERT, W.; LOPES, H. Gene expression programming for evolving two-dimensional cellular automata in a distributed environment. In: CAMACHO, D. et al. (Ed.). **Intelligent Distributed Computing VIII**. Heidelberg: Springer International Publishing, 2015, (Studies in Computational Intelligence, v. 570). p. 107–117.

BENÍTEZ, C. M. V.; PARPINELLI, R. S.; LOPES, H. S. Parallelism, hybridism and coevolution in a multi-level ABC-GA approach for the protein structure prediction problem. **Concurrency and Computation: Practice and Experience**, v. 24, n. 6, p. 635–646, 2012.

BEREAU, T.; DESERNO, M. Generic coarse-grained model for protein folding and aggregation. **The Journal of Chemical Physics**, v. 130, p. 235106, 2009.

BERENDSEN, H. et al. Molecular dynamics with coupling to an external bath. **Journal of Chemical Physics**, v. 81, n. 8, p. 3684, 1984.

BERGER, B.; LEIGHTON, F. Protein folding in the hydrophobic-hydrophilic HP model is NP-complete. **Journal of Computational Biology**, v. 5, n. 1, p. 27–40, 1998.

BERMAN, H. et al. UniProt archive. **Nucleic Acids Research**, v. 28, n. 1, p. 235–242, 2000.

BERNSTEIN, F. et al. The protein databank: A computer-based archival file for macromolecular structures. **Journal of Molecular Biology**, v. 112, n. 3, p. 535–542, 1977.

BEST, R. Atomistic molecular simulations of protein folding. **Current Opinion in Structural Biology**, v. 22, p. 52–61, 2012.

BEZKOROVAYNAYA, O. **Coarse-grained peptide models: conformational sampling, peptide association and dynamical properties for peptides**. Thesis (PhD Thesis) — Max-Planck-Institut für Polymerforschung. Johannes Gutenberg-Universität, 2011.

BITELLO, R.; LOPES, H. A differential evolution approach for protein folding. **Journal of Computer Science and Technology**, v. 22, n. 6, p. 904–908, 2007.

BOERIO-GOATES, B. O. J. **Chemical Thermodynamics – Principles and Applications**. 5^{th}. ed. San Diego, CA, USA: Academic Press, 2000.

BONABEAU, E.; DORIGO, M.; THERAULAZ, G. **Swarm Intelligence: From Natural to Artificial Systems**. New York, NY, USA: Oxford University Press, Inc., 1999.

BORNBERG-BAUER, E. Chain growth algorithms for HP-type lattice proteins. **Proceedings of the first annual international conference on Computational molecular biology (RECOMB '97)**, p. 47–55, 1997.

BORNBERG, S.; BAUER, E. Chain growth algorithms for HP-type lattice proteins. In: **Proceedings of the 1st Annual International Conference on Computational Molecular Biology**. New York, USA: ACM, 1997. p. 47–55.

BOWIE, J.; LUTHY, R.; EINSENBERG, D. A method to identifity protein sequences that fold into a known three-dimensional structure. **Science**, v. 253, n. 5016, p. 164–170, 1991.

BOX, G. Evolutionary operation: A method for increasing industrial productivity. **Applied Statistics**, v. 6, n. 2, p. 81–101, 1957.

BOX, G.; HUNTER, W.; HUNTER, J. **Statistics for Experimenters: Design, Innovation, and Discovery**. 2^{nd}. ed. New York, USA: J. Wiley & Sons, 2005.

BRANDEN, C.; TOOZE, J. **Introduction to Protein Structure**. New York, USA: Garland Publishing, 1999.

BREMERMANN, H. Optimization through evolution and recombination. In: **Self-Organizing Systems**. Washington, DC, USA: Spartan Books, 1962. p. 93–106.

BREU, H.; KIRKPATRICK, D. Unit disk graph recognition is NP-hard. **Computational Geometry. Theory and Applications**, v. 9, n. 1-2, 1993.

BROADLEY, S.; HARTL, F. U. The role of molecular chaperones in human misfolding diseases. **Federation of European Biochemical Societies Letters**, v. 583, n. 16, p. 2647–2653, 2009.

BROGLIA, R.; TIANA, G. Physical models for protein folding and drug design. In: **Proceedings Idea-Finding Symposium**. Frankfurt, Germany: Frankfurt Institute for Advanced Studies, 2003. p. 23–33.

BROWN, S.; FAWZI, N.; HEAD-GORDON, T. Coarse grained sequences for protein folding and design. **Proceedings of the National Academy of Sciences**, v. 100, n. 19, p. 10712–10717, 2003.

BROWNE, W.; LESK, A. The relation between the divergence of sequence and structure in proteins. **Journal of Molecular Biology**, v. 42, n. 1, p. 65–86, 1969.

BURIONI, F. et al. Topological thermal instability and length of proteins. **Proteins**, v. 55, n. 3, p. 529–535, 2004.

BYSTROFF, C.; SHAO, Y. Fully automated ab initio protein structure prediction using I-SITES, HMMSTR and ROSETTA. **Bioinformatics**, v. 18, n. 1, p. S54–S61, 2002.

BYSTROFF, C.; SHAO, Y. Modeling protein folding pathways. In: BUJNICKI, J. (Ed.). **Practical Bioinformatics**. Heidelberg: Springer-Verlag, 2004. p. 97–122.

CAPRILES, P.; CUSTóDIO, F. L.; DARDENNE, L. Ab initio protein structure prediction via genetic algorithms using a coarse-grained model for side chains. In: **Poster Proceedings of the Brazilian Symposium on Bioinformatics (BSB2010)**. Rio de Janeiro, Brazil: Brazilian Computer Society, 2010. p. 23–27.

CASE, D. et al. The amber biomolecular simulation programs. **Journal of Computational Chemistry**, v. 26, n. 16, p. 1668–1688, 2005.

CATTANEO, G. et al. Non-uniform cellular automata. In: **Language and Automata Theory and Applications**. Heidelberg: Springer, 2009. p. 302–313.

CHEN, K.; KURGAN, L. PFRES: protein fold classification using evolutionary information and predicted secondary structure. **Bioinformatics**, v. 23, n. 21, p. 2843–2850, 2007.

CHEN, Y. et al. Protein folding: Then and now. **Archives of Biochemistry and Biophysics**, v. 469, n. 1, p. 4–19, 2008.

CHENG, J.; BALDI, P. Improved residue contact prediction using support vector machines and a large feature set. **BMC Bioinformatics**, v. 8, n. 1, p. 113, 2007.

CHIDAMBARAM, C. et al. A comparative study of machine learning and evolutionary computation approaches for protein secondary structure classification. In: LOPES, H.; CRUZ, L. (Ed.). **Computational Biology and Applied Bioinformatics**. Rijeka: Intech, 2011. cap. 12, p. 239–258.

CHOTHIA, C.; LESK, A. The relation between the divergence of sequence and structure in proteins. **EMBO Journal**, v. 5, p. 823–826, 1986.

CHU, D.; TILL, M.; ZOMAYA, A. Parallel ant colony optimization for 3D protein structure prediction using the HP lattice model. In: **Proc.** 19th **IEEE Int. Parallel and Distributed Processing Symp.** Washington, DC, USA: IEEE Computer Society, 2005. p. 193–199.

COMBE, N.; FRENKEL, D. Simple off-lattice model to study the folding and aggregation of peptides. **Molecular Physics**, v. 105, n. 4, p. 375–385, 2007.

COOPER, G. **The Cell: A Molecular Approach**. Sunderland, UK: Sinauer Associates, 2000.

COX, G. et al. Development and optimisation of a novel genetic algorithm for studying model protein folding. **Theoretical Chemistry Accounts: Theory, Computation, and Modeling (Theoretica Chimica Acta)**, v. 112, n. 3, p. 163–178, 2004.

CREIGHTON, T. **Proteins – Structures and Molecular Properties**. New York, USA: W.H. Freeman and Company, 1993.

CRESCENZI, P. et al. On the complexity of protein folding. **Journal of Computational Biolology**, v. 5, p. 423–446, 1998.

CUFF, A. et al. Extending CATH: increasing coverage of the protein sctructure universe and linking structure with function. **Nucleic Acids Research**, v. 39, n. 1, p. D420–D426, 2011.

CUSTÓDIO, F.; BARBOSA, H.; DARDENNE, L. Full-atom ab initio protein structure prediction with a genetic algorithm using a similarity-based surrogate model. In: **IEEE Evolutionary Computation (CEC)**. Piscataway, USA: IEEE Press, 2010. p. 1–8.

CUTELLO, V.; NARZISI, G.; NICOSIA, G. A class of pareto archived evolution strategy algorithms using immune inspired operators for ab-initio protein structure prediction. In: **Applications on Evolutionary Computing**. Heidelberg, Germany: Springer-Verlag, 2005. v. 3449, p. 54–63.

DAY, R.; DAGGETT, V. All-atom simulations of protein folding and unfolding. **Advances in Protein Chemistry**, v. 66, n. 1, p. 373–403, 2003.

DESMET, J.; SPRIET, J.; LASTERS, I. Fast and accurate side-chain topology and energy refinement (FASTER) as a new method for protein structure optimization. **Proteins**, v. 48, p. 31–43, 2002.

DIAZ, N.; TISCHER, I. Mining cellular automata models on protein folding trajectories. In: **6th Colombian Computing Congress (CCC)**. Piscataway, USA: IEEE Press, 2011. p. 1–6.

DILL, K. Dominant forces in protein folding. **Biochemistry**, v. 29, n. 31, p. 7133–7155, 1990.

DILL, K. Polymer principles and protein folding. **Protein Science**, v. 8, n. 6, p. 1166–1180, 1999.

DILL, K. et al. Principles of protein folding - a perspective from simple exact models. **Protein Science**, v. 4, n. 4, p. 561–602, 1995.

DILL, K. et al. The protein folding problem. **Annual Review of Biophysics**, v. 37, p. 289–316, 2008.

DING, F. et al. A mechanism for the alpha-helix to beta-hairpin transition. **Proteins: Structure, Function and Genetics**, v. 53, n. 2, p. 220–228, 2003.

DINNER, A. et al. Understanding protein folding via free-energy surfaces from theory and experiment. **Trends in Biochemical Sciences**, v. 25, n. 7, p. 331–339, 2000.

DOBSON, C. Experimental investigation of protein folding and misfolding. **Methods**, v. 34, n. 1, p. 4–14, 2004.

DOKHOLYAN, N. V. et al. Discrete molecular dynamics studies of the folding of a protein-like model. **Folding & Design**, v. 3, n. 6, p. 577–587, 1998.

DONG, Q.; ZHOU, S.; GUAN, J. A new taxonomy-based protein fold recognition approach based on autocross-covariance transformation. **Bioinformatics**, v. 25, n. 20, p. 2655–2662, 2009.

DORIGO, M.; STÜTZLE, T. **Ant Colony Optimization**. Cambridge, MA, USA: MIT Press, 2004.

DRENTH, J. **Principles of Protein X-Ray Crystallography**. New York, USA: Springer-Verlag, 1999.

DUARTE, J. et al. Optimal contact definition for reconstruction of contact maps. **BMC Bioinformatics**, v. 11, n. 1, p. 283, 2010.

DUBCHAK, I. et al. Prediction of protein folding class using global description of amino acid sequence. **Proceedings of the National Academy of Sciences**, v. 92, n. 19, p. 8700–8704, 1995.

EIBEN, A. E.; SMIT, S. K. Parameter tuning for configuring and analyzing evolutionary algorithms. **Swarm and Evolutionary Computation**, v. 1, n. 1, p. 19–31, 2011.

EISENBERG, D. et al. Analysis of membrane and surface protein sequences with the hydrophobic moment plot. **Journal of Molecular Biology**, v. 179, n. 1, p. 125–142, 1984.

EL-REWINI, H.; ABD-EL-BARR, M. **Advanced Computer Architecture and Parallel Processing**. Hoboken, NJ, USA: Wiley, 2005.

ENGELMAN, D.; STEITZ, T.; GOLDMAN, A. Identifying nonpolar transbilayer helices in amino acid sequences of membrane proteins. **Annual Review of Biophysics and Biophysical Chemistry**, v. 15, n. 1, p. 321–353, 1986.

ENGLANDER, S. W. Protein folding intermedites and pathways studied by hydrogen exchange. **Annual Review of Biophysics and Biomolecular Structure**, v. 29, n. 1, p. 213–238, 2000.

ERTOZ, L.; STEINBACH, M.; KUMAR, V. A new shared nearest neighbor clustering algorithm and its applications. In: SIAM. **Workshop on Clustering High Dimensional Data and its Applications at 2nd SIAM International Conference on Data Mining**. Chicago, 2002. p. 105–115.

ESWAR, N. et al. Comparative protein structure modeling using MODELLER. **Current Protocols in Protein Science**, v. 50, n. 1, p. 1–31, 2007.

FAIN, B.; LEVITT, M. Funnel sculpting for in silico assembly of secondary structure elements of proteins. **Proceedings of the National Academy of Sciences**, v. 100, n. 19, p. 10700–10705, 2003.

FARISELLI, P. et al. Progress in predicting inter-residue contacts of proteins with neural networks and correlated mutations. **Proteins**, v. 45, n. 1, p. 157–162, 2001.

FEIG, M. et al. Accurate reconstruction of all-atom protein representations from side-chain-based low-resolution models. **Proteins: Structure, Function and Genetics**, v. 41, n. 1, p. 86–97, 2000.

FERREIRA, C. Gene expression programming: a new adaptive algorithm for solving problems. **Complex Systems**, v. 13, n. 2, p. 87–129, 2001.

FERSHT, A. **Struture and Mechanism in Protein Science: A guide to Enzyme Catalysis and Protein Folding**. New York, USA: WH Freeman, 1999.

FIDANOVA, S. 3D HP protein folding problem using ant algorithm. In: BioPS. **Proceedings of BioPS'06 International Conference**. Sofia, Bulgaria, 2006. p. III.19–III.26.

FISCHER, D. et al. Assigning amino acid sequences to 3-dimensional protein folds. **FASEB Journal**, v. 10, n. 1, p. 126–136, 1996.

FISHER, R. **Statistical Methods and Scientific Inference**. New York, USA: Hafner Publishing Co, 1959.

FOGEL, L. Autonomous automata. **Industrial Research**, v. 4, n. 2, p. 14–19, 1962.

FRANK, J. **Three-Dimensional Electron Microscopy of Macromolecular Assemblies**. New York, USA: Oxford University Press, 2006.

FREITAS, A. **Data Mining and Knowledge Discovery with Evolutionary Algorithms**. Heidelberg: Springer-Verlag, 2002.

FRENKEL, D.; SMIT, B. **Understanding Molecular Simulation. From algorithms to applications**. Florida, USA: Academic Press, 2002.

FRIEDBERG, R.; DUNHAM, B.; NORTH, J. A learning machine: Part I. **IBM Journal**, v. 2, n. 1, p. 2–13, 1958.

FRIGORI, R. B.; RIZZI, L. G.; ALVES, N. A. Microcanonical thermostatistics of coarse-grained proteins with amyloidogenic propensity. **Journal of Chemical Physics**, v. 138, p. 015102, 2013.

GARCÍA-MARTÍNEZ, J. et al. An efficient approach for solving the hp protein folding problem based on UEGO. **Journal of Mathematical Chemistry**, v. 53, n. 3, p. 1–13, 2014. ISSN 0259-9791.

GARNIER, S.; GAUTRAIS, J.; THERAULAZ, G. The biological principles of swarm intelligence. **Swarm Intelligence**, v. 1, n. 1, p. 3–31, 2007.

GAST, K. et al. Stopped-flow dynamic light scattering as a method to monitor compaction during protein folding. **European Biophysics Journal**, v. 25, n. 3, p. 211–219, 1997.

GAUTIER, R.; CAMPROUX, A.-C.; TUFFÉRY, P. SCit: web tools for protein side chain conformation analysis. **Nucleic Acids Research**, v. 32, n. 1, p. W508–W511, 2004.

GEIST, A. et al. **PVM: Parallel Virtual Machine. A Users' Guide and Tutorial for Network Parallel Computing**. Cambridge, MA, USA: MIT Press, 1994.

GIN, B.; GARRAHAN, J.; GEISSLER, P. The limited role of non-native contacts in folding pathways of a lattice protein. **Journal of Molecular Biology**, v. 392, p. 1303–1314, 2009.

GOLUB, G. H.; LOAN, C. F. V. **Matrix computations**. MD: Johns Hopkins University Press, 2012. 70–73 p.

GONG, W.; CAI, Z.; LING, C. X. DE/BBO: a hybrid differential evolution with biogeography-based optimization for global numerical optimization. **Soft Computing**, v. 15, n. 4, p. 645–665, 2010.

GOPAL, S. et al. PRIMO/PRIMONA: A coarse-grained model for proteins and nucleic acids that preserves near-atomistic accuracy. **Proteins**, v. 78, n. 5, p. 1266–1281, 2010.

G.POLLASTRI; BALDI, P. Prediction of contact maps by GIOHMMs and recurrent neural networks using lateral propagation from all four cardinal corners. **Bioinformatics**, v. 18, n. 1, p. S62–S70, 2002.

GRIFFITHS, A. et al. **An Introduction to Genetic Analysis**. 7^{th}. ed. New York, USA: W.H. Freeman, 2000.

GROPP, W.; LUSK, E.; THAKUR, R. **Using MPI-2: Advanced Features of the Message-Passing Interface**. Cambridge, MA, USA: MIT Press, 1999.

GROSBERG, A.; KHOKHLOV, A. **Statistical Physics of Macromolecules**. New York, USA: AIP Press, 1994.

GRUEBELE, M. Protein folding: the free energy surface. **Current Opinion in Structural Biology**, v. 12, n. 1, p. 161–168, 2002.

GU, J. et al. A simple Cα-SC potential with higher accuracy for protein fold recognition. **Biochemical and Biophysical Research Communications**, v. 379, n. 2, p. 610–615, 2009.

HARDER, T. et al. An efficient null model for conformational fluctuations in proteins. **Structure**, v. 20, n. 6, p. 1028–1039, 2012.

HARPER, E.; ROSE, G. Helix stop signals in proteins and peptides: The capping box. **Biochemistry**, v. 32, n. 30, p. 7605–7609, 1993.

HART, W.; ISTRAIL, S. Lattice and off-lattice side chain models of protein folding. **Journal of Computational Biology**, v. 4, n. 3, p. 241–259, 1997.

HESSA, T. et al. Recognition of transmembrane helices by the endoplasmic reticulum translocon. **Nature**, v. 433, n. 7024, p. 377–381, 2005.

HEUN, V. Approximate protein folding in the HP side chain model on extended cubic lattices. **Discrete Applied Mathematics**, v. 127, p. 163–177, 2003.

HILLS, R.; BROOKS, C. Insights from coarse-grained gō models for protein folding and dynamics. **International Journal of Molecular Sciences**, v. 10, n. 3, p. 889–905, 2009.

HOELTZLI, S.; FRIEDEN, C. Refolding of [6-19f] tryptophan-labeled *Escherichia coli* dihydrofolate reductase in the presence of ligand: A stopped-flow NMR spectroscopy study. **Biochemistry**, v. 37, n. 1, p. 387–398, 1998.

HOLLAND, J. H. **Adaptation in Natural and Artificial Systems**. Cambridge, MA, USA: MIT Press, 1975.

HONEYCUTT, J.; THIRUMALAI, D. The nature of the folded state of globular proteins. **Biopolymers**, v. 32, n. 6, p. 695–709, 1992.

HOQUE, T.; CHETTY, M.; SATTAR, A. Extended HP model for protein structure prediction. **Journal of Computational Biology**, v. 16, n. 1, p. 85–103, 2009.

HU, J. et al. Mining protein contact maps. In: **Proceedings of BIOKDD02: Workshop on Data Mining in Bioinformatics**. California, USA: Chapman & Hall/CRC Press, 2002.

IRBACK, A. et al. Local interactions and protein folding: A three-dimensional off-lattice approach. **Journal of Chemical Physics**, v. 107, n. 1, p. 273–282, 1997.

ISTRAIL, S.; LAM, F. Combinatorial algorithms for protein folding in lattice models: a survey of mathematical results. **Communications in Information and Systems**, v. 9, n. 4, p. 303–346, 2009.

JACCARD, P. Nouvelles recherches sur la distribution florale. **Bulletin de la Sociéte Vaudoise des Sciences Naturelles**, v. 44, p. 223–270, 1908.

JANIN, J. Surface and inside volumes in globular proteins. **Nature**, v. 277, n. 5696, p. 491–492, 1979.

JARONIEC, C. et al. High resolution molecular structure of a peptide in an amyloid fibril determined by magic angle spinning NMR spectroscopy. **Proceedings of the National Academy of Sciences**, USA, v. 101, n. 3, p. 711–716, 2004.

JIANG, T. et al. Protein folding simulations of the hydrophobic-hydrophilic model by combining tabu search with genetic algorithms. **Journal of Chemical Physics**, v. 119, n. 8, p. 4592–4596, 2003.

JIMENEZ, J. L. et al. The protofilament structure of insulin amyloid fibrils. **Proceedings of the National Academy of Sciences**, USA, v. 50, n. 14, p. 9196–9201, 2002.

JONES, D. Protein secondary structure prediction based on position-specific scoring matrices. **Journal of Molecular Biology**, v. 292, p. 195–202, 1999.

JONES, D.; TAYLOR, W.; THORNTON, J. A new approach to protein fold recognition. **Nature**, v. 358, p. 86–89, 1992.

KABSCH, W. A discussion of the solution of the best rotation to relate two sets of vectors. **Acta Crystallographica**, A34, p. 827–828, 1978.

KALEGARI, D.; LOPES, H. A differential evolution approach for protein structure optimisation using a 2D off-lattice model. **International Journal of Bio-Inspired Computation**, v. 2, n. 3/4, p. 242–250, 2010.

KARABOGA, D.; AKAY, B. A comparative study of artificial bee colony algorithm. **Applied Mathematics and Computation**, v. 214, n. 1, p. 108–132, 2009.

KARABOGA, D.; BASTURK, B. On the performance of the artificial bee colony (ABC) algorithm. **Applied Soft Computing**, v. 8, n. 1, p. 687–697, 2008.

KARABOGA, D. et al. A comprehensive survey: Artificial bee colony (ABC) algorithm and applications. **Artificial Intelligence Review**, Springer Netherlands, v. 42, n. 1, p. 21–57, 2014. ISSN 0269-2821.

KARP, A. H.; FLATT, H. P. Measuring parallel processor performance. **Commun. ACM**, ACM, New York, NY, USA, v. 33, n. 5, p. 539–543, maio 1990. ISSN 0001-0782.

KARPLUS, M. The Levinthal paradox: yesterday and today. **Folding & Design**, v. 2, n. 4, p. S69–S75, 1997.

KARPLUS, M. Behind the folding funnel diagram. **Nature Chemical Biology**, v. 7, p. 401–404, 2011.

KAUZMANN, W. Some factors in the interpretation of protein denaturation. **Advances in Protein Chemistry**, v. 97, p. 1–63, 1959.

KAVOUSI, K. et al. A protein fold classifier formed by fusing different modes of pseudo amino acid composition via PSSM. **Computational Biology and Chemistry**, v. 35, p. 1–9, 2011.

KAVOUSI, K. et al. Evidence theoretic protein fold classification based on the concept of hyperfold. **Mathematical Biosciences**, v. 240, n. 2, p. 148–160, 2012.

KAYA, H.; CHAN, H. Towards a consistent modeling of protein thermodynamic and kinetic cooperativity: how applicable is the transition state picture to folding and unfolding? **Journal of Molecular Biology**, v. 315, p. 899–909, 2002.

KENNEDY, J.; EBERHART, R. Particle Swarm Optimization. In: **Proceedings of the IEEE International Conference on Neural Networks**. Los Alamitos, USA: IEEE Computer Society, 1995. v. 4, p. 1942–1948.

KIM, S.; LEE, S.; LEE, J. Structure optimization by conformational space annealing in an off-lattice protein model. **Physical Review E**, v. 72, p. 1–6, 2005.

KNUTH, D. **The Art of Computer Programming**. USA: Addison-Wesley, 1981.

KOGA, N.; TAKADA, S. Roles of native topology and chain-length scaling in protein folding: a simulation study with a gō-like model. **Journal of Molecular Biology**, v. 313, p. 171–180, 2001.

KOLINSKI, A.; SKOLNICK, J. Assembly of protein structure from sparse experimental data: an efficient monte carlo model. **Proteins**, v. 32, p. 475–494, 1998.

KOLINSKI, A.; SKOLNICK, J. Reduced models of proteins and their applications. **Polymer**, v. 45, p. 511–524, 2004.

KONOPKA, B. M. et al. Automated procedure for contact-map-based protein structure reconstruction. **Journal of Membrane Biology**, v. 247, p. 409–420, 2014.

KOZA, J. **Genetic Programming: on the programming of computers by means of natural selection**. Cambridge, MA, USA: MIT Press, 1992.

KRASNOGOR, N. et al. A critical view of the evolutionary design of self-assembling systems. In: TALBI, E.-G. et al. (Ed.). **Artificial Evolution**. Heidelberg: Springer Berlin Heidelberg, 2006, (Lecture Notes in Computer Science, v. 3871). p. 179–188. ISBN 978-3-540-33589-4.

KRIVOV, G.; SHAPOVALOV, M.; DUNBRACK, R. Improved prediction of protein side-chain conformation with SCWRL4. **Proteins: Structure, Function and Bioinformatics**, v. 77, n. 4, p. 778–795, 2009.

KUNDU, S.; SORENSEN, D.; PHILLIPS, G. Automatic domain decomposition of proteins by gaussian network model. **Proteins**, v. 57, p. 725–733, 2004.

KUO, T. **A Computational Approach to Predicting Distance Maps from Contact Maps**. Thesis (Doctorate) — Queen's University, Kingston, Ontario, Canada, 2012.

KYTE, J.; DOOLITTLE, R. A simple method for displaying the hydropathic character of a protein. **Journal of Molecular Biology**, v. 157, n. 1, p. 105–132, 1982.

LANE, T. J. et al. To milliseconds and beyond: challenges in the simulation of protein folding. **Current opinion in structural biology**, Elsevier, v. 23, n. 1, p. 58–65, 2013.

LEOPOLD, P.; MONTAL, M.; ONUCHIC, J. Protein folding funnels: a kinetic approach to the sequence-structure relantionship. **Proceedings of the National Academy of Science**, v. 89, p. 8721–8725, 1992.

LEVINTHAL, C. Are there pathways for protein folding? **Journal de Chimie Physique**, v. 65, p. 44–45, 1968.

LEWIS, P.; MOMANY, F.; SCHERAGA, H. Chain reversals in proteins. **Biochimica et Biophysica Acta**, v. 303, n. 2, p. 211–229, 1973.

LI, M.; KLIMOV, D.; THIRUMALAI, D. Folding in lattice models with side chains. **Computer Physics Communications**, v. 147, n. 1, p. 625–628, 2002.

LI, X. Protein folding based on simulated annealing algorithm. In: **Proceedings of International Conference on Natural Computation**. Los Alamitos, CA, USA: IEEE Computer Society, 2007. v. 4, p. 256–259.

LIANG, F. Annealing contour Monte Carlo algorithm for structure optimization in an off-lattice protein model. **Chemical Physics**, v. 120, n. 14, p. 6756–6763, 2004.

LINDORFF-LARSEN, K. et al. How fast-folding proteins fold. **Science**, v. 334, n. 6055, p. 517–520, 2011.

LIWO, A.; KHALILI, M.; SCHERAGA, H. A. Ab initio simulations of protein-folding pathways by molecular dynamics with the united-residue model of polypeptide chains. **Proceedings of the National Academy of Sciences**, v. 102, n. 7, p. 2362–2367, 2005.

Lo Conte, L. et al. SCOP: a structural classification of proteins database. **Nucleic Acids Research**, v. 28, n. 1, p. 257–259, 2000.

LOBO, F.; LIMA, C.; MICHALEWICZ, Z. **Parameter Setting in Evolutionary Algorithms**. New York, USA: Springer, 2007.

LOCKER, C.; HERNANDEZ, R. A minimalist model protein with multiple folding funnels. **Proceedings of the National Academy of Sciences**, v. 98, n. 16, p. 9074–9079, 2001.

LODISH, H. et al. **Molecular Cell Biology**. 4^{th}. ed. New York, USA: Freeman, 2000.

LOPES, H. Evolutionary algorithms for the protein folding problem: A review and current trends. In: **Computational Intelligence in Biomedicine and Bioinformatics**. Heidelberg: Springer-Verlag, 2008. I, p. 297–315.

LUGER, G. **Artificial Intelligence: Structures and Strategies for Complex Problem Solving**. 6^{th}. ed. USA: Addison-Wesley Publishing Company, 2008.

LUHESHI, L.; DOBSON, C. Bridging the gap: From protein misfolding to protein misfolding diseases. **Federation of European Biochemical Societies Letters**, v. 583, n. 16, p. 2581–2586, 2009.

LYNCH, M.; CONERY, J. S. The evolutionary fate and consequences of duplicated genes. **Science**, v. 290, n. 5494, p. 1151–1155, 2000.

MACCALLUM, R. Striped sheets and protein contact prediction. **Bioinformatics**, v. 20, n. 1, p. I224–I231, 2004.

MACKERELL, A. et al. All-atom empirical potential for molecular modeling and dynamics studies of proteins. **Journal of Physical Chemistry B**, v. 102, n. 18, p. 3586–3616, 1998.

MACKERELL, A.; FEIG, M.; BROOKS, C. Improved treatment of the protein backbone in empirical force fields. **Journal of the American Chemical Society**, v. 126, n. 3, p. 698–699, 2004.

MANN, M. et al. Producing high-accuracy lattice models from protein atomistic coordinates including side chains. **Advances in Bioinformatics**, v. 2012, n. 1, p. 148045, 2012.

MANUCH, J.; GAUR, D. Fitting protein chains to cubic lattice is NP-complete. **Journal of Bioinformatics and Computational Biology**, v. 6, p. 93–106, 2006.

MARSAGLIA, G. Choosing a point from the surface of a sphere. In: **The Annals of Mathematical Statistics**. USA: The Institute of Mathematical Statistics, 1972. v. 43, n. 2, p. 645–646.

MASON, R.; GUNST, R.; HESS, J. **Statistical Design and Analysis of Experiments**. New York, USA: Wiley & Sons Inc., 2003.

MATSUMOTO, M.; NISHIMURA, T. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. **ACM Transactions on Modeling and Computer Simulation**, v. 8, n. 1, p. 3–30, 1998.

MAUPETIT, J.; GAUTIER, R.; TUFFÉRY, P. Sabbac: online structural alphabet-based protein backbone reconstruction from alpha-carbon trace. **Nucleic Acids Research**, v. 34, n. 2, p. W147–W151, 2006.

MAUPETIT, J.; P.DERREUMAUX; TUFFÉRY, P. Pep-fold: an online resource for de novo peptide structure prediction. **Nucleic Acids Research**, v. 37, p. W498–503, 2009.

MAUPETIT, J.; TUFFERY, P.; DERREUMAUX, P. A coarse-grained protein force field for folding and structure prediction. **Proteins: Structure, Function and Bioinformatics**, v. 69, n. 2, p. 394–408, 2007.

MAY, R. M. C.; MCLEAN, A. R. **Theoretical Ecology: Principles and Applications**. Oxford, UK: Oxford University Press, 2007.

MEDRI, F. **3-Dimensional Protein Reconstruction from Contacts Maps: Complexity and Experimental Results**. Bologna, Italy, 2009.

MELVIN, I. et al. SVM-fold: a tool for discriminative multi-class protein fold and superfamily recognition. **BMC Bioinformatics**, v. 8, n. 4, p. S2, 2007.

MICHELETTI, C.; CARLONI, P.; MARITAN, A. Accurate description of protein vibrational dynamics: comparing molecular dynamics and Gaussian models. **Proteins**, v. 55, n. 3, p. 635–645, 2004.

MIRNY, L.; DOMANY, E. Protein fold recognition and dynamics in the space of contact maps. **PROTEINS: Structure, Function and Genetics**, v. 26, n. 4, p. 391, 1996.

MIRNY, L.; SHAKHNOVICH, E. Protein foding theory: From lattice to all-atom models. **Annual Review of Biophysics and Biomolecular Structure**, v. 30, n. 1, p. 361–396, 2001.

MITCHELL, M. Computation in cellular automata: a select review. In: **Nonstandard Computation**. 1. ed. Weinheim: VHC Verlagsgesellschaft, 1998. p. 95–140.

MIYAZAWA, S.; JERNIGAN, R. Esimation of effective interresidue contact energies from protein crystal structures: quasi-chemical approximation. **Macromolecules**, v. 18, n. 3, p. 534–552, 1985.

MO, Y. et al. The 2-dimensional cellular automata for HIV infection. **Physica A**, v. 399, n. 1, p. 31–39, 2014.

MOHAMMAD, T.; NAGARAJARAM, H. A hierarchical approach to protein fold prediction. **Journal of Integrative Bioinformatics**, v. 8, n. 1, p. 185, 2011.

MONASTYRSKYY, B. et al. Evaluation of residue–residue contact predictions in CASP9. **Proteins: Structure, Function, and Bioinformatics**, v. 79, n. S10, p. 119–125, 2011.

MONTGOMERY, D. **Design and Analysis of Experiments**. New York, USA: Wiley & Sons Inc., 2001.

MOSCATO, P.; COTTA, C.; MENDES, A. Memetic algorithms. In: **New optimization techniques in engineering**. Heidelberg: Springer, 2004. p. 53–85.

MOULT, J. et al. Critical assessment of methods of protein structure – round VII. **Proteins**, v. 69, n. 8, p. 3–9, 2007.

MUKHERJEE, A.; BAGCHI, B. Contact pair dynamics during folding of two small proteins: chicken villin head piece and the Alzheimer protein β-amyloid. **Journal of Chemical Physics**, v. 120, n. 3, p. 1602–1612, 2004.

NALS, J. V.; KOLINSKI, A.; SKOLNICK, J. Numerical study of the entropy loss of dimerization and the folding thermodynamics of the GCN4 leucine zipper. **Biophysical Journal**, v. 83, n. 5, p. 2801–2811, 2002.

NANNEN, V.; EIBEN, A. E. Relevance estimation and value calibration of evolutionary algorithm parameters. In: VELOSO, M. (Ed.). **Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)**. California, USA: Morgan Kaufmann, 2007. v. 7, n. 1, p. 975–980.

NELSON, D.; COX, M. **Lehninger Principles of Biochemistry**. 5^{th}. ed. Boston, USA: W.H. Freeman, 2008.

NEUMANN, J. V. **Theory of self-reproducing automata**. Urbana-Champaign: University of Illinois Press, 1966.

NGO, J.; MARKS, J.; KARPLUS, M. Computational complexity, protein structure prediction, and the Levinthal paradox. In: MERZ, K.; LEGRAND, S. (Ed.). **The Protein Folding Problem and Terciary Structure Prediction**. Boston, USA: Birkhäuser, 1994. p. 433–506.

NICOSIA, G.; STRACQUADANIO, G. Generalized pattern search algorithm for peptide structure pediction. **Biophysical Journal**, v. 95, n. 10, p. 4988–4999, 2008.

NÖLTING, B. **Protein Folding Kinetics**. 2^{nd}. ed. Berlin, Germany: Springer, 2006.

NUGENT, T.; JONES, D. T. Accurate de novo structure prediction of large transmembrane protein domains using fragment-assembly and correlated mutation analysis. **Proceedings of the National Academy of Sciences**, v. 109, n. 24, p. E1540–E1547, 2012.

NULAND, N. V. et al. Real-time NMR studies of protein folding. **Accounts of Chemical Research**, v. 31, n. 11, p. 773–780, 1998.

ORLOVA, E.; SAIBIL, H. Structural analysis of macromolecular assemblies by electron microscopy. **Chemical Reviews**, v. 111, n. 12, p. 7710–7748, 2011.

OSTROVSKY, B. et al. Cellular automata for polymer simulation with application to polymer melts and polymer collapse including implications for protein folding. **Parallel Computing**, v. 27, n. 5, p. 613–641, 2001.

OZKAN, S. B.; DILL, K.; BAHAR, I. Fast-folding protein kinetics, hidden intermediates and the sequential stabilization model. **Protein Science**, v. 11, n. 8, p. 1958–1970, 2002.

PANDE, V. et al. Pathways for protein folding: is a "new view"needed? **Current Opinion in Structural Biology**, v. 8, n. 1, p. 68–79, 1998.

PANDURANGAN, A.; TOPF, M. RIBFIND: a web server for identifying rigid bodies in protein structures and to aid flexible fitting into cryo EM maps. **Bioinformatics**, v. 28, n. 18, p. 2391–2393, 2012.

PARK, B.; LEVITT, M. The complexity and accuracy of discrete state models of protein structure. **Journal of Molecular Biology**, v. 249, n. 7, p. 493–507, 1995.

PARPINELLI, R. et al. Performance analysis of swarm intelligence algorithms for the 3D-AB off-lattice protein folding problem. **Journal of Multiple-Valued Logic and Soft Computing**, v. 22, n. 1, p. 267–286, 2014.

PARPINELLI, R.; LOPES, H. **Theory and New Applications of Swarm Intelligence**. 1^{st}. ed. Rijeka, Croatia: Intech, 2012.

PARPINELLI, R.; TEODORO, F.; LOPES, H. A comparison of swarm intelligence algorithms for structural engineering optimization. **International Journal for Numerical Methods in Engineering**, v. 91, n. 6, p. 666–684, 2012.

PARPINELLI, R. S.; LOPES, H. S. New inspirations in swarm intelligence: a survey. **International Journal of Bio-Inspired Computation**, v. 3, n. 1, 2011.

PARPINELLI, R. S.; LOPES, H. S. Biological plausibility in optimization: An ecosystemic view. **International Journal of Bio-Inspired Computation**, v. 4, n. 6, p. 345–358, 2012.

PARPINELLI, R. S.; LOPES, H. S. A hierarchical clustering strategy to improve the biological plausibility of an ecology-based evolutionary algorithm. In: PAVÓN, J. et al. (Ed.). **Ibero-American Conference on Artificial Intelligence (IBERAMIA)**. Berlin, Heidelberg: Springer-Verlag, 2012. (Lecture Notes in Computer Science/Lecture Notes in Artificial Intelligence, 1), p. 310–319.

PARPINELLI, R. S.; LOPES, H. S. A computational ecosystem for optimization: review and perspectives for future research. **Memetic Computing**, Springer, v. 7, n. 1, p. 29–41, 2015.

PAUL, R. P. **Robot manipulators: mathematics, programming, and control: the computer control of robot manipulators**. Cambridge, MA: MIT Press, 1981.

PAULING, L.; COREY, R.; BRANSON, H. Configurations of polypeptide chains with favored orientations of the polypeptide around single bonds: two pleated sheets. **Proceedings of the National Academy of Science of the USA**, v. 37, n. 11, p. 729–740, 1951.

PAULING, L.; COREY, R.; BRANSON, H. The structure of proteins: two hydrogen-bonded helicel configurations of the polypeptide chain. **Proceedings of the National Academy of Science of the USA**, v. 37, n. 4, p. 205–211, 1951.

PEARL, F. et al. The CATH domain structure database and related resources gene3d and DHS provide comprehensive domain family information for genome analysis. **Nucleic Acids Research**, v. 33, n. 1, p. D247–d251, 2005.

PEDERSEN, C. **Algorithms in Computational Biology**. Thesis (PhD Thesis) — Department of Computer Science, University of Aarhus, Denmark, 2000.

PEITSCH, M.; JONGENEEL, C. A 3D model for the CD40 ligand predicts that it is a compact trimer similar to the tumor necrosis factors. **International Immunology**, v. 5, p. 223–228, 1993.

PETERSON, W.; WELDON, E. **Error-correcting Codes**. Cambridge, MA: MIT Press, 1972.

PETREY, D.; HONIG, B. Protein structure prediction: Inroads to biology. **Molecular Cell**, v. 20, n. 6, p. 811–819, 2005.

PETREY, D. et al. Using multiple structure alignments, fast model building, and energetic analysis in fold recognition and homology modeling. **Proteins**, v. 53, n. 6, p. 430–435, 2003.

PICKOVER, C. A. **Computers, Pattern, Chaos, and Beauty**. Toronto, Canada: Dover, 2001.

PIETAL, M.; TUSZYNSKA, I.; BUJNICKI, J. PROTMAP2D: visualization, comparison and analysis of 2D maps of protein structure. **Bioinformatics**, v. 23, n. 11, p. 1429–1430, 2007.

PLAXCO, K.; DOBSON, C. Time-resolved biophysical methods in the study of protein folding. **Current Opinion in Structural Biology**, v. 6, n. 5, p. 630–636, 1996.

POLI, R. Analysis of the publications on the applications of particle swarm optimisation. **Journal of Artificial Evolution and Applications**, Hindawi Publishing Corporation, New York, NY, United States, v. 2008, n. 1, p. 4:1–4:10, 2008.

PONDER, J. W.; CASE, D. A. et al. Force fields for protein simulations. **Advances in Protein Chemistry**, v. 66, n. 1, p. 27–86, 2003.

PROCTOR, E.; DING, F.; DOKHOLYAN, N. Discrete molecular dynamics. **Computational Molecular Science**, v. 1, n. 1, p. 80–92, 2011.

PUNTA, M.; ROST, B. PROFcon: novel prediction of long-range contacts. **Bioinformatics**, v. 21, n. 13, p. 2960–2968, 2005.

RAGHUNATHAN, G.; JERNIGAN, R. Ideal architecture of residue packing and its observation in protein structures. **Protein Science**, v. 6, n. 10, p. 2072–2083, 1997.

RAMANATHAN, A.; AGARWAL, P. Computational identification of slow conformational fluctuations in proteins. **Journal of Physical Chemistry B**, v. 113, n. 52, p. 16669–16680, 2009.

RAPAPORT, D. **The Art of Molecular Dynamics Simulation**. Cambridge, UK: Cambridge University Press, 2004.

RECHENBERG, I. **Cybernetic solution path of an experimental problem**. Farnborough, UK, 1965. Technical Report.

REITH, D.; PUTZ, M.; MULLER-PLATHE, F. Deriving effective mesoscale potentials from atomistic simulations. **Journal of Computational Chemistry**, v. 24, n. 13, p. 1624–1636, 2003.

REVA, B. et al. Constructing lattice models of protein chains with side groups. **Journal of Computational Biology**, v. 2, n. 4, p. 527–535, 1995.

RICHARDSON, J. The anatomy and taxonomy of protein structure and advances. **Protein Chemistry**, v. 34, p. 167–339, 1981.

RODGERS, J.; NICEWANDER, W. Thirteen ways to look at the correlation coefficient. **The American Statistician**, v. 42, n. 1, p. 59–66, 1988.

ROHL, C. et al. Modeling structurally variable regions in homologous proteins with rosetta. **Proteins**, v. 55, n. 3, p. 656–677, 2004.

ROOSTA, S. **Parallel Processing and Parallel Algorithms: Theory and Computation**. New York, USA: Springer-Verlag, 1999.

RÖTHLISBERGER, D. et al. Kemp elimination catalysts by computational enzyme design. **Nature**, v. 453, n. 7192, p. 190–195, 2008.

ROTKIEWICZ, P.; SKOLNICK, J. Fast procedure for reconstruction of full-atom protein models from reduced representations. **Journal of Computational Chemistry**, v. 29, n. 9, p. 1460–1465, 2008.

RUMBLEY, J. et al. An amino acid code for protein folding. **Proceedings of the National Academy of Sciences**, v. 98, n. 1, p. 105–112, 2001.

RYCKAERT, J.; CICCOTTI, G.; BERENDSEN, H. Numerical integration of the cartesian equations of motion of a system with constraints: Molecular dynamics of n-alkanes. **Journal of Computational Physics**, v. 23, n. 3, p. 327–341, 1977.

SALI, A.; SHAKHNOVICH, E.; KARPLUS, M. Kinetics of protein folding: a lattice model study of the requirements for folding to the native state. **Journal of Molecular Biology**, v. 235, n. 5, p. 1614–1636, 1994.

SANTOS, J.; VILLOT, P.; DIÉGUEZ, M. Protein folding with cellular automata in the 3d hp model. In: **Proceedings of the 15th Annual Conference Companion on Genetic and Evolutionary Computation**. New York, NY, USA: ACM, 2013. (GECCO '13 Companion), p. 1595–1602. ISBN 978-1-4503-1964-5.

SCALABRIN, M. et al. Population-based harmony search using GPU applied to protein structure prediction. **International Journal of Computational Science and Engineering**, v. 9, n. 1/2, p. 106–118, 2014.

SCAPIN, M.; LOPES, H. A hybrid genetic algorithm for the protein folding problem using the 2D-HP lattice model. In: YANG, A.; SHAN, Y.; BUI, L. (Ed.). **Success in Evolutionary Computation**. Heidelberg, Germany: Springer, 2007. p. 205–224.

SCHMITZ, C. et al. Protein structure determination from pseudocontact shifts using ROSETTA. **Journal of Molecular Biology**, v. 416, n. 5, p. 668–677, 2012.

SCHULER, L.; DAURA, X.; GUNSTEREN, W. V. An improved GROMOS96 force field for aliphatic hydrocarbons in the condensed phase. **Journal of Computational Chemistry**, v. 22, n. 11, p. 1205–1218, 2001.

SEMISOTNOV, G. et al. Protein globularization during folding: a study by synchrotron small-angle x-ray scattering. **Journal of Molecular Biology**, v. 262, n. 4, p. 559–574, 1996.

SHAKHNOVICH, E. Protein folding thermodynamics and dynamics: Where physics, chemistry and biology meet. **Chemical Reviews**, v. 106, n. 5, p. 1559–1588, 2006.

SHAKHNOVICH, E.; GUTIN, A. Engineering of stable and fast-folding sequences of model proteins. **Proceedings of the National Academy of Science**, v. 90, n. 15, p. 7195–7199, 1993.

SHAW, D. et al. Atomic-level characterization of the structural dynamics of proteins. **Science**, v. 330, p. 341–346, 2010.

SHEN, H.; CHOU, K. Ensemble classifier for protein fold patter recognition. **Bioinformatics**, v. 22, n. 14, p. 1717–1722, 2006.

SHEN, H.; CHOU, K. Predicting protein fold pattern with functional domain and sequential evolution information. **Journal of Theoretical Biology**, v. 256, n. 3, p. 441–446, 2009.

SHERMAN, M. The SimTK and its users. **The SimTK Engineering Journal**, v. 5, n. 1, p. 1–10, 2005.

SHI, J.; BLUNDELL, T.; MIZUGUCHI, K. FUGUE: sequence-structure homology recognition. **Journal of Molecular Biology**, v. 310, n. 1, p. 243–257, 2001.

SHMYGELSKA, A.; HOOS, H. An ant colony optimisation algorithm for the 2D and 3D hydrophobic polar protein folding problem. **Bioinformatics**, v. 6, n. 1, p. 30, 2005.

SIMON, D. Biogeography-based optimization. **IEEE transactions on Evolutionary Computation**, v. 12, n. 6, p. 702–713, 2008.

SKOLNICK, J.; KOLINSKI, A. Dynamic monte carlo simulations of a new lattice model of globular protein folding, structure and dynamics. **Journal of Molecular Biology**, v. 221, n. 2, p. 499–531, 1991.

SNAPP, E. et al. The organization of engaged and quiescent translocos in the endoplasmic reticulum of mammalian cells. **Journal of Cell Biology**, v. 164, n. 7, p. 997–1007, 2004.

SNOW, C.; RHEE, Y.; PANDE, V. Kinetic definition of protein folding transition state ensembles and reaction coordinates. **Biophysical Journal**, v. 91, n. 1, p. 14–24, 2006.

SOSNICK, T.; BARRICK, D. The folding of single domain proteins – have we reached a consensus. **Current Opinion in Structural Biology**, v. 21, n. 1, p. 12–24, 2011.

STERLING, T. **Beowulf Cluster Computing with Linux**. Cambridge, MA, USA: MIT Press, 2002.

STILLINGER, F.; HEAD-GORDON, T. Collective aspects of protein folding illustrated by a toy model. **Physical Review E**, v. 52, n. 3, p. 2872–2877, 1995.

STORN, R.; PRICE, K. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. **Journal of Global Optimization**, v. 11, n. 4, p. 341–359, 1997.

SUNDE, M.; BLAKE, C. The structure of amyloid fibrils by electron microscopy and x-ray diffraction. **Advances in Protein Chemistry**, v. 50, 1997.

SUTCLIFFE, M. et al. Knowledge based modelling of homologous proteins. part I: Three-dimensional frameworks derived from the simultaneous superposition of multiple structures. **Protein Engineering**, v. 1, n. 5, p. 377–384, 1987.

SUTMANN, G. Classical Molecular Dynamics. In: GROTENDORST, J.; MARX, D.; MURAMATSU, A. (Ed.). **Quantum Simulations of Complex Many-Body Systems: from Theory to Algorithms**. Jülich, Germany: John von Neumann-Institut für Computing, 2002, (NIC Notes, v. 10). p. 211–254.

SWOPE, W. et al. A computer simulation method for the calculation of equilibrium constants for the formation of physical clusters of molecules: Application to small water clusters. **The Journal of Chemical Physics**, v. 76, n. 1, p. 637, 1982.

TANTAR, A. et al. A parallel hybrid genetic algorithm for protein structure prediction on the computational grid. **Future Generation Computer Systems**, v. 23, n. 3, p. 398–409, 2007.

TAYLOR, T. J. et al. Assessment of casp10 contact-assisted predictions. **Proteins: Structure, Function, and Bioinformatics**, v. 82, n. S2, p. 84–97, 2014.

TAYLOR, W. R.; JONES, D. T.; SADOWSKI, M. I. Protein topology from predicted residue contacts. **Protein Science**, v. 21, n. 2, p. 299–305, 2012.

THORNTON, J. et al. Protein folds: Towards understanding folding from inspection of native structures. **Philosophical Transactions of the Royal Society B: : Biological Sciences**, v. 348, n. 1, p. 71–79, 1995.

TIAN, L. et al. NCACO-score: An effective main-chain dependent scoring function for structure modeling. **BMC Bioinformatics**, v. 12, n. 1, p. 208, 2011.

TIRION, M. Large amplitude elastic motions in proteins from a single-parameter, atomic analysis. **Physical Review Letters**, v. 77, p. 1905–1908, 1996.

TOPA, P.; MLOCEK, P. Using shared memory as a cache in cellular automata water flow simulations on GPUs. **Computer Science**, v. 14, n. 1, p. 385–401, 2013.

TOZZINI, V. Coarse-grained models for proteins. **Current Opinion in Structural Biology**, v. 15, n. 2, p. 144–150, 2005.

TOZZINI, V.; ROCCHIA, W.; McCammon, J. Mapping all-atom models onto one-bead coarse grained models: general properties and applications to a minimal polypeptide model. **Journal of Chemical Theory and Computation**, v. 2, n. 3, p. 667–673, 2006.

UEDA, Y.; TAKETOMI, H.; Gō, N. Studies on protein folding, unfolding and fluctuations by computer simulation. a three-dimensional lattice model of lysozyme. **Biophysical Journal**, v. 17, n. 6, p. 1531–1548, 1978.

ULLAH, A. et al. Protein folding simulation by two-stage optimization. In: **Proceedings of the International Symposium on Intelligence Computation and Applications (ISICA)**. Wuhan, China: Springer, 2009. p. 23–25.

UNGER, R. et al. A 3D building blocks approach to analyzing and predicting structure of proteins. **Proteins**, v. 5, n. 4, p. 355–373, 1989.

UNGER, R.; MOULT, J. Finding the lowest free energy conformation of a protein is a NP-hard problem: proof and implications. **Bulletin of Mathematical Biology**, v. 55, n. 6, p. 1183–1198, 1993a.

VASSURA, M. et al. Blurring contact maps of thousands of proteins: what we can learn by reconstructing 3D structure. **BioData Mining**, v. 4, p. 1–15, 2011.

VASSURA, M. et al. FT-COMAR: fault tolerant three-dimensional structure reconstruction from protein contact maps. **Bioinformatics**, v. 24, n. 10, p. 1313–1315, 2008.

VASSURA, M. et al. Reconstruction of 3D structures from protein contact maps. **IEEE/ACM Transactions on Computational Biology and Bioinformatics**, v. 5, n. 3, p. 357–367, 2008.

VEHLOW, C. et al. CMView: Interactive contact map visualization and analysis. **Bioinformatics**, v. 27, n. 11, p. 1573–1574, 2011.

VENDRUSCOLO, M.; KUSSELL, E.; DOMANY, E. Recovery of protein structure from contact maps. **Folding & Design**, v. 2, n. 5, p. 295–306, 1997.

VENDRUSCOLO, M. et al. Three key residues form a critical contact network in a transition state for protein folding. **Nature**, v. 409, p. 641–646, 2001.

VILLA, A.; PETER, C.; VAN DER VEGT., N. Self-assembling dipeptides: conformational sampling in solvent-free coarse-grained simulations. **Physical Chemistry Chemical Physics**, v. 11, n. 12, p. 2077–2086, 2009.

WEIKL, T. Transition states in protein folding. **Communications in Computational Physics**, v. 7, n. 1, p. 283–300, 2010.

WEINERT, W. et al. Simulation of the dynamic behavior of one-dimensional cellular automata using reconfigurable computing. In: **Reconfigurable Computing: Architectures, Tools and Applications**. Heidelberg: Springer, 2007. p. 385–390.

WEINERT, W.; LOPES, H. Evaluation of dynamic behavior forecasting parameters in the process of transition rule induction of unidimensional cellular automata. **BioSystems**, v. 99, n. 1, p. 6–16, 2010.

WEINERT, W. R.; LOPES, H. S. GEPCLASS: a classification rule discovery tool using gene expression programming. In: **Advanced Data Mining and Applications**. Heidelberg: Springer, 2006, (Lecture Notes in Computer Science). p. 871–880.

WILLIAMS, G.; TOON, A. Protein folding pathways and state transitions described by classical equations of motion of an elastic network model. **Protein Science**, v. 19, n. 12, p. 2451–2461, 2010.

WIMLEY, W.; WHITE, S. Experimentally determined hydrophobicity scale for proteins at membrane interfaces. **Nature Structural & Molecular Biology**, v. 3, n. 10, p. 842–848, 1996.

WOLFF, K.; VENDRUSCOLO, M.; PORTO, M. Coarse-grained model for protein folding based on structural profiles. **Physical Review E**, v. 84, p. 041934, 2011.

WOLFRAM, S. **A New Kind of Science**. Boston, USA: Champaign: Wolfram Media, 2002.

WOLYNES, P.; ONUCHIC, J.; THIRUMALAI, D. Navigating the folding routes. **Science**, v. 267, n. 5204, p. 1619–1620, 1995.

WU, Y. et al. Universal behavior of localizatoin of residue fluctuations in globular proteins. **Physical Review E**, v. 64, n. 4, p. 041909, 2003.

WÜTHRICH, K. **NMR of Proteins and Nucleic Acids**. New York, USA: John Wiley & Sons, 1986.

XU, J. Rapid protein side-chain packing via tree decomposition. **Proceedings of the Conference on Research in Computational Molecular Biology (RECOMB)**, p. 423–39, 2005.

YANG, A.; HONIG, B. Sequence to structure alignment in comparative modeling using prISM. **Proteins**, v. 3, n. 1, p. 66–72, 1999.

YANG, J.-Y.; CHEN, X. Improving taxonomy-based protein fold recognition by using global and local features. **Proteins**, v. 79, n. 7, p. 2053–2064, 2011.

YAP, E.-H.; FAWZI, N.; HEAD-GORDON, T. A coarse-grained α-carbon protein model with anisotropic hydrogen-bonding. **Proteins: Structure, Function and Bioinformatics**, v. 70, p. 626–637, 2008.

YUE, K.; DILL, K. Sequence-structure relationships in proteins and copolymers. **Physical Review E**, v. 48, n. 3, p. 2267–2278, 1993.

ZACHARIAS, M. Protein docking with a reduced protein model accounting for side-chain flexibility. **Protein Science**, v. 12, p. 1271–1282, 2003.

ZAKI, M.; BYSTROFF, C. **Protein Structure Prediction**. 2^{nd} edition. ed. Totowa, NJ, USA: Humana Press/Springer, 2008.

ZAKI, M.; JIN, S.; BYSTROFF, C. Mining residue contacts in proteins using local structure predictions. In: **IEEE Transactions on Systems, Man and Cybernetics**. Piscataway, USA: IEEE Press, 2003. p. 168.

ZAKI, M. et al. Predicting protein folding pathways. **Bioinformatics**, v. 20, n. 1, p. 386–393, 2004.

ZHANG, C.; DELISI, C. Estimating the number of protein folds:. **Journal of Molecular Biology**, v. 284, n. 5, p. 1301–1305, 1998.

ZHANG, Y.; ARAKAKI, A.; SKOLNICK, J. TASSER: An automated method for the prediction of protein tertiary structures in CASP6. **Proteins: Structure, Function and Bioinformatics**, v. 61, n. S7, p. 91–98, 2005.

ZHANG, Z.; SHI, Y.; LIU, H. Molecular dynamics simulations of peptides and proteins with amplified collective motions. **Biophysical Journal**, v. 84, p. 3583–3593, 2003.

APPENDIX A – NUMERICAL RESULTS OF pGEP-CA

Tables 20 – 24 show the results obtained using the CMs of the protein sequences 2gb1 ($N = 56$), 1pcy ($N = 99$), 2trx ($N = 108$) and 3fxn ($N = 138$), where T_C, T_{NC}, F_C and F_{NC} represent the number true contacts, true non-contacts, false contacts and false non-contacts, respectively. S_C, S_{NC} measure the ability of a *transition rule* to generate correct *contacts* and *non-contacts*, respectively. S_i measures the symmetry of the CM.

Table 20: Numerical results obtained using CMs with different threshold values and the fitness function $fitness_1$ – sequence 2gb1 ($N = 56$)

Metric	CM threshold [Å]						
$Avg(Min/Max)$	6.65	7	8	9	10	11	12
Best $fitness$	0.91 (0.73/0.97)	0.86 (0.75/0.91)	0.88 (0.74/0.93)	0.92 (0.76/0.97)	0.897 (0.76/0.93)	0.88 (0.74/0.93)	0.89 (0.74/0.95)
T_C	393.3 (216/428)	678.9 (334/722)	1047.2 (478/1100)	1323.4 (556/1392)	1548.6 (650/1644)	1888.4 (656/1992)	2160.4 (770/2280)
F_C	34.1 (12/277)	85.1 (46/348)	85.2 (22/336)	63.1 (12/211)	78.62 (20/202)	83.7 (4/203)	61.12 (2/182)
T_{NC}	2677.2 (2493/2789)	2290.1 (2162/2625)	1916.8 (1850/2491)	1682.6 (1608/2408)	1424.4 (1328/2307)	1068.2 (964/2240)	834.9 (730/2130)
F_{NC}	31.4 (12/277)	81.9 (32/146)	86.8 (40/194)	66.9 (28/206)	84.4 (32/218)	95.8 (46/308)	79.64 (26/338)
S_C	0.92 (0.76/0.98)	0.89 (0.78/0.94)	0.92 (0.74/0.96)	0.95 (0.76/0.98)	0.95 (0.78/0.98)	0.95 (0.74/0.98)	0.96 (0.74/0.988)
S_{NC}	0.98 (0.9/0.996)	0.96 (0.86/0.98)	0.96 (0.85/0.99)	0.96 (0.9/0.99)	0.95 (0.90/0.99)	0.93 (0.88/0.998)	0.93 (0.87/0.999)
S_i	0.9998	0.9999	0.9999	0.9999	0.99997	0.99998	0.99999
Avg t_p(s)	12.68	12.70	12.69	12.69	12.70	12.69	12.69

Source: Own work.

Table 21: Numerical results obtained using CMs with different threshold values and the fitness function $fitness_1$ – sequence 1pcy ($N = 99$)

Metric	CM threshold [Å]						
Avg(*Min*/*Max*)	6.65	7	8	9	10	11	12
Best *fitness*	0.93 (0.66/0.97)	0.83 (0.65/0.92)	0.87 (0.69/0.94)	0.91 (0.72/0.97)	0.90 (0.70/0.95)	0.91 (0.74/0.95)	0.94 (0.75/0.97)
T_C	757.15 (346/818)	904.208 (286/1462)	1226.55 (464/2278)	1559.69 (536/2916)	1833.31 (548/3478)	2410.73 (700/4474)	2619.75 (770/5220)
F_C	60.85 (24/530)	112.06 (12/879)	97.39 (22/684)	71.94 (12/534)	96.65 (8/487)	108.34 (4/671)	80.28 (14/485)
T_{NC}	8932.42 (8603/9258)	4233.93 (2230/8843)	2821.35 (1850/8572)	2535.93 (1608/8415)	2215.15 (1328/8267)	2410.73 (700/4474)	1453.77 (730/7835)
F_{NC}	50.59 (20/156)	107.47 (46/246)	101.54 (40/300)	79.28 (28/310)	103.72 (32/392)	120.46 (46/456)	93.03 (26/524)
S_C	0.94 (0.69/0.98)	0.891 (0.68/0.955)	0.92 (0.74/0.965)	0.95 (0.76/0.98)	0.95 (0.72/0.98)	0.95 (0.74/0.98)	0.96 (0.78/0.988)
S_{NC}	0.99 (0.94/0.997)	0.97 (0.897/0.996)	0.96 (0.92/0.99)	0.97 (0.93/0.995)	0.95 (0.91/0.996)	0.94 (0.87/0.998)	0.93 (0.87/0.998)
S_i	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
Avg t_p(s)	14.68	13.36	13.04	13.03	13.05	13.19	17.15

Source: Own work.

Table 22: Numerical results obtained using CMs with different threshold values and the fitness function $fitness_1$ – sequence 2trx ($N = 108$)

Metric	CM threshold [Å]						
Avg(*Min*/*Max*)	6.65	7	8	9	10	11	12
Best *fitness*	0.93 (0.66/0.97)	0.83 (0.65/0.92)	0.87 (0.69/0.94)	0.91 (0.72/0.97)	0.91 (0.70/0.94)	0.91 (0.74/0.95)	0.94 (0.75/0.97)
T_C	770.78 (346/892)	930.02 (286/1618)	1272.87 (464/2584)	1619.50 (536/3318)	1904.54 (548/3960)	2538.78 (700/5100)	2739.67 (770/6020)
F_C	62.64 (22/905)	115.29 (12/1159)	101.42 (22/759)	72.83 (12/475)	98.03 (8/733)	114.46 (4/625)	83.24 (18/442)
T_{NC}	9226.89 (8603/10966)	4514.28 (2230/10575)	3078.49 (1850/10185)	2784.27 (1608/9959)	2447.97 (1328/9748)	2294.99 (964/9443)	1636.91 (730/9186)
F_{NC}	51.19 (20/156)	108.58 (46/294)	104.55 (40/364)	80.72 (28/360)	106.79 (32/466)	125.94 (46/524)	97.52 (26/578)
S_C	0.94 (0.69/0.98)	0.89 (0.68/0.95)	0.92 (0.74/0.96)	0.95 (0.76/0.98)	0.95 (0.72/0.98)	0.95 (0.74/0.98)	0.96 (0.78/0.989)
S_{NC}	0.99 (0.92/0.997)	0.97 (0.88/0.99)	0.96 (0.92/0.99)	0.97 (0.93/0.996)	0.95 (0.72/0.98)	0.94 (0.74/0.98)	0.93 (0.87/0.99)
S_i	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
Avg t_p(s)	14.79	13.47	13.15	13.15	13.16	13.34	17.26

Source: Own work.

Table 23: Numerical results obtained using CMs with different threshold values and the fitness function $fitness_1$ – sequence 3fxn ($N = 138$)

Metric	CM threshold [Å]						
$Avg(Min/Max)$	6.65	7	8	9	10	11	12
Best $fitness$	0.93 (0.66/0.98)	0.83 (0.65/0.94)	0.87 (0.69/0.94)	0.91 (0.72/0.97)	0.91 (0.70/0.95)	0.91 (0.73/0.95)	0.94 (0.75/0.97)
T_C	813.48 (346/1180)	1008.23 (286/2188)	1397.79 (464/3464)	1785.12 (536/4464)	2111.24 (548/5400)	2905.18 (700/7044)	3073.71 (770/8462)
F_C	68.31 (24/1165)	135.23 (12/1647)	112.85 (22/1563)	80.38 (12/724)	110.19 (8/985)	133.83 (4/895)	97.49 (18/590)
T_{NC}	10406.52 (8603/18131)	5639.51 (2230/17475)	4162.98 (1850/17045)	3832.96 (1608/16679)	3446.70 (1328/16353)	3564.36 (964/15939)	2501.78 (730/15583)
F_{NC}	68.31 (24/1165)	115.19 (46/386)	113.71 (40/430)	88.87 (28/474)	119.19 (32/566)	148.099 (46/674)	114.36 (26/666)
S_C	0.94 (0.69/0.98)	0.89 (0.67/0.97)	0.92 (0.74/0.97)	0.95 (0.76/0.98)	0.95 (0.72/0.98)	0.95 (0.74/0.977)	0.96 (0.78/0.988)
S_{NC}	0.99 (0.93/0.998)	0.97 (0.90/0.995)	0.96 (0.90/0.99)	0.97 (0.93/0.99)	0.95 (0.91/0.996)	0.94 (0.87/0.997)	0.94 (0.87/0.996)
S_i	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
Avg t_p(s)	15.23	13.91	13.59	13.59	13.60	13.95	17.71

Source: Own work.

Table 24: Numerical results obtained using CMs with different threshold values and the fitness function $fitness_2$ – sequence 2gb1 ($N = 56$)

Metric	CM threshold [Å]						
$Avg(Min/Max)$	6.65	7	8	9	10	11	12
Best $fitness$	0.86 (0.50/0.93)	0.80 (0.65/0.87)	0.86 (0.69/0.92)	0.91 (0.72/0.96)	0.90 (0.70/0.94)	0.91 (0.74/0.95)	0.94 (0.75/0.97)
T_C	392.45 (144/428)	676.93 (286/722)	1046.35 (464/1100)	1323.11 (536/1392)	1549.01 (548/1644)	1892.93 (700/1992)	2168.46 (770/2280)
F_C	30.80 (8/70)	80.36 (12/128)	82.92 (22/174)	61.41 (12/166)	76.62 (8/202)	83.37 (4/146)	63.43 (18/182)
T_{NC}	2680.21 (2644/2848)	2293.38 (2230/2702)	1918.11 (1850/2492)	1682.45 (1608/2425)	1424.39 (1328/2358)	1064.77 (964/2221)	829.29 (730/2130)
F_{NC}	32.54 (8/136)	85.33 (46/166)	88.62 (40/194)	69.03 (28/206)	85.98 (32/232)	94.93 (46/308)	74.81 (26/266)
S_C	0.922 (0.51/0.98)	0.89 (0.68/0.94)	0.92 (0.74/0.97)	0.95 (0.76/0.98)	0.95 (0.72/0.98)	0.95 (0.74/0.93)	0.97 (0.78/0.99)
S_{NC}	0.99 (0.98/0.997)	0.97 (0.95/0.995)	0.96 (0.93/0.991)	0.97 (0.96/0.995)	0.95 (0.91/0.996)	0.93 (0.87/0.998)	0.93 (0.86/0.992)
S_i	0.9999	0.9999	0.99995	0.99993	0.99996	0.99995	0.99997
Avg t_p(s)	13.82	12.69	12.70	12.70	12.71	12.74	17.64

Source: Own work.

ANNEX A – AMINO ACIDS

Table 25: List of Amino acids and AB classification according to (ALBERTS et al., 2002)

Amino acid		Type		
name	symbol	Class (ALBERTS et al., 2002)	AB model	Frequency in proteins (%)(CREIGHTON, 1993)
Aspartic acid	ASP	Hydrophilic	B	5
Glutamic acid	GLU	Hydrophilic	B	6
Arginine	ARG	Hydrophilic	B	6
Lysine	LYS	Hydrophilic	B	6
Histidine	HIS	Hydrophilic	B	2
Asparagine	ASN	Hydrophilic	B	4
Glutamine	GLN	Hydrophilic	B	4
Serine	SER	Hydrophilic	B	7
Threonine	THR	Hydrophilic	B	6
Tyrosine	TYR	Hydrophilic	B	3
Alanine	ALA	Hydrophobic	A	8
Glycine	GLY	Hydrophobic	A	7
Valine	VAL	Hydrophobic	A	7
Leucine	LEU	Hydrophobic	A	9
Isoleucine	ILE	Hydrophobic	A	5
Proline	PRO	Hydrophobic	A	5
Phenylalanine	PHE	Hydrophobic	A	4
Methionine	MET	Hydrophobic	A	2
Tryptophan	TRP	Hydrophobic	A	1
Cysteine	CYS	Hydrophobic	A	2

Source: Adapted from (ALBERTS et al., 2002) and (CREIGHTON, 1993).

Table 26: Preferences for different types of secondary structure

Amino acid	Conformational preference parameter (THORNTON et al., 1995)		
Name	α-helix	β-sheet	Turn
Aspartic acid	0.9 – 1.1	0.5 – 0.7	1.4 – 1.5
Glutamic acid	1.4	0.5 – 0.8	0.7 – 1.0
Arginine	0.9 – 1.4	0.7 – 1.0	0.9 – 1.0
Lysine	1.1 – 1.2	0.7 – 0.9	1.0
Histidine	1.0 – 1.2	0.8 – 1.1	0.7 – 1.0
Asparagine	0.8 – 0.9	0.6 – 0.7	1.3 – 1.6
Glutamine	1.1 – 1.4	0.8 – 1.0	1.0
Serine	0.7 – 0.8	0.9 – 1.0	1.3 – 1.4
Threonine	0.7 – 0.8	1.2 – 1.3	1.0
Tyrosine	0.7 – 0.9	1.2 – 1.5	1.1
Alanine	1.3 – 1.5	0.8 – 0.9	0.7
Glycine	0.4 – 0.6	0.6 – 0.9	1.6
Valine	0.9 – 1.0	1.5 – 1.7	0.5
Leucine	1.3	1.0 – 1.2	0.6
Isoleucine	1.0 – 1.1	1.5 – 1.8	0.5
Proline	0.5 – 0.6	0.4 – 0.6	1.5 – 1.9
Phenylalanine	1.0 – 1.1	1.2 – 1.4	0.6
Methionine	1.3 – 1.4	1.0 – 1.3	0.4 – 0.6
Tryptophan	1.0	1.2	0.8 – 1.0
Cysteine	0.9 – 1.0	0.8 – 1.2	0.9 – 1.2

Source: Adapted from (THORNTON et al., 1995).

ANNEX B – BIO-INSPIRED COMPUTATION ALGORITHMS

This section presents the EC and SI algorithms that are related to this work.

Algorithm 11 Pseudo-code for GEP

```
1: Start
2: Initialize population;
3: Express Chromosomes
4: Execute each program
5: Evaluate fitness
6: while stop criteria not satisfied do
7:     Keep Best Program
8:     Select Programs
9:     Reproduction:
10:    Replication
11:    Mutation
12:    IS transposition
13:    RIS transposition
14:    Gene transposition
15:    1-Point Recombination
16:    2-Point Recombination
17:    Gene Recombination
18:    Prepare New Program of Next Generation
19: end while
20: Postprocess results and visualization
21: End
```

Algorithm 12 Canonical PSO

```
 1: Set parameters: $n,\ \varphi_p,\ \varphi_g, w$
 2: Initialize $f(\vec{g})$
 3: for $i = 1$ to $n$ do
 4:     Initialize the positions $\vec{x}_i$ and velocities $\vec{v}_i$ randomly
 5:     Evaluate fitness $f(\vec{x}_i)$
 6:     Initialize the particle's best known position to its initial position: $\vec{p}_i = \vec{x}_i$
 7:     if $f(\vec{p}_i)$ is better than $f(\vec{g})$ then
 8:         Update the swarm's best known position: $\vec{g} = \vec{p}_i$
 9:     end if
10: end for
11: while stop condition not met do
12:     for $i = 1$ to $n$ do
13:         Update particles' velocity: $\vec{v}_i = \vec{v}_i + \varphi_p * r_p * (\vec{p}_i - \vec{x}_i) + \varphi_g * r_g * (\vec{g} - \vec{x}_i)$
14:         Update particles' position: $\vec{x}_i = \vec{x}_i + \vec{v}_i$
15:         if $f(\vec{x}_i)$ is better than $f(\vec{p}_i)$ then
16:             $\vec{p}_i = \vec{x}_i$
17:             if $f(\vec{p}_i)$ is better than $f(\vec{g})$ then
18:                 $\vec{g} = \vec{p}_i$
19:             end if
20:         end if
21:     end for
22: end while
23: Postprocess results and visualization
```

Algorithm 13 Canonical ABC

```
1: Set parameters: $n$, $limit$
2: Initialize the food sources $\vec{x}_i$ randomly
3: Evaluate fitness $f(\vec{x}_i)$ of the population
4: $count_i \leftarrow 0$
5: **while** stop condition not met **do**
6:    **for** $i = 1$ **to** $n/2$ **do** {*Employed phase*}
7:       Select $k$, $j$ and $r$ at random such that $k \in \{1,2,...,n\}$, $j \in \{1,2,...,d\}$,
8:       $r \in [0,1]$
9:       $\vec{v} \leftarrow x_{ij} + r \cdot (x_{ij} - x_{kj})$
10:      Evaluate solutions $\vec{v}$ and $\vec{x}_i$
11:      **if** $f(\vec{v})$ is better than $f(\vec{x})$ **then**
12:         Greedy selection
13:      **else**
14:         $count_i \leftarrow count_i + 1$
15:      **end if**
16:   **end for**
17:   **for** $i = n/2+1$ **to** $n$ **do** {*Onlooker phase*}
18:      Calculate selection probability
19:         $P(\vec{x}_k) \leftarrow \frac{f(\vec{x}_k)}{\sum_{k=i}^{n} f(\vec{x}_k)}$
20:      Select a bee using the selection probability
21:      Produce a new solution $\vec{v}$ from the selected bee
22:      Evaluate solutions $\vec{v}$ and $\vec{x}_i$
23:      **if** $f(\vec{v})$ is better than $f(\vec{x})$ **then**
24:         Greedy selection
25:      **else**
26:         $count_i \leftarrow count_i + 1$
27:      **end if**
28:   **end for**
29:   **for** $i = 1$ **to** $n$ **do** {*Scout phase*}
30:      **if** $count_i > limit$ **then**
31:         $\vec{x}_i \leftarrow$ rand()
32:         $count_i \leftarrow 0$
33:      **end if**
34:   **end for**
35:   Memorize the best solution achieved so far
36: **end while**
37: Postprocess results and visualization
```

Algorithm 14 Canonical DE with DE/rand/1/bin scheme

```
1: Set parameters: $n$, $F$, $CR$
2: **for** $i = 1$ **to** $n$ **do**
3:     Initialize population $POP$ (solution vector $\overrightarrow{x_i}$) randomly
4:     Evaluate objetive function $f(\overrightarrow{x_i})$
5: **end for**
6: Find the best solution vector $S_{best}$ in $POP$
7: **while** stop condition not met **do**
8:     **for** $i = 1$ **to** $n$ **do**
9:         Select random indices $r_1, r_2, r_3 \in n$ where $r_1 \neq r_2 \neq r_3 \neq i$
10:        Select a dimension $p \in d$ randomly
11:        **for** $j = 1$ **to** $d$ **do**
12:            **if** $((j = p) \vee$ (rand() $\leq CR$)) where rand() $\sim$ U[0,1] **then**
13:                $y_j \leftarrow x_{r_3,j} + F \times (x_{r_1,j} - x_{r_2,j})$
14:            **else**
15:                $y_j \leftarrow x_{ij}$
16:            **end if**
17:        **end for**
18:        Evaluate $f(\overrightarrow{y})$
19:        **if** $f(\overrightarrow{y}) < f(\overrightarrow{x_i})$ **then**
20:            $NewPop \leftarrow \overrightarrow{y}$
21:        **else**
22:            $NewPop \leftarrow \overrightarrow{x_i}$
23:        **end if**
24:    **end for**
25:    $POP \leftarrow NewPop$
26:    **for** $i = 1$ **to** $n$ **do**
27:        Evaluate $f(\overrightarrow{x_i})$
28:    **end for**
29:    Find the best solution vector $S_{best}$ in $POP$
30: **end while**
31: Postprocess results and visualization
```

Algorithm 15 Canonical BBO algorithm

```
1: Set parameters: $n,\ E,\ S_{max}, m_{max}, elite$
2: for $i = 1$ to $n$ do
3:     Initialize habitat $\overrightarrow{x_i}$) randomly
4:     Evaluate objetive function $f(\overrightarrow{x_i})$
5:     Initialize the species count probability of each habitat $P_i = 1/n$
6: end for
7: while stop condition not met do
8:     Identify $elite$ habitats
9:     Map HSI to the number of species by sorting the population from best to worst
10:    for $i = 1$ to $n$ do
11:        Calculate the immigration rate $\alpha_i = I(1 - k/n)$
12:        Calculate the emigration rate $\mu_i = Ek/n$
13:    end for
14:    for $i = 1$ to $n$ (Habitat recombination) do
15:        if rand() $\leq \alpha_i$ where rand() $\sim$ U[0,1] then
16:            for $j = 1$ to $n$ do
17:                if rand() $\leq \mu_i$ where rand() $\sim$ U[0,1] then
18:                    Randomly select an SIV $\sigma$ from $\overrightarrow{x_j}$
19:                    Replace a random SIV in $\overrightarrow{x_j}$ with $\sigma$
20:                end if
21:            end for
22:        end if
23:    end for
24:    for $i = 1$ to $n$ do
25:        Compute the time derivative $P_i$ for each habitat
26:        Compute mutation rate for each habitat: $m_i = m_{max}(1 - P_i)/P_{max}$
           where $P_{max} = argmax(P_i)$
27:    end for
28:    for $i = 1$ to $n$ (Habitat mutation) do
29:        for $j = 1$ to $d$ do
30:            if rand() $\leq \mu_i$ where rand() $\sim$ U[0,1] then
31:                Replace $\overrightarrow{x_{ij}}$ with a randomly generated SIV
32:            end if
33:        end for
34:        Evaluate objective function $f(\overrightarrow{x_i})$
35:    end for
36:    Keep $elite$ habitats
37: end while
38: Postprocess results and visualization
```

ANNEX C – BASIC MPI FUNCTIONS

Table 27: Basic MPI functions

Function	Description
MPI_Init	It initializes the MPI environment.
MPI_Comm_size	It determines the size of a group that is associated with a communicator.
MPI_Comm_rank	This function determines the rank of the calling process in the communicator.
MPI_Send / MPI_Bsend	These functions perform blocking send. MPI_Bsend performs basic send with user-defined buffering
MPI_Recv	Blocking receive for a message.
MPI_Finalize	It terminates MPI execution.
MPI_Pack	This function packs a datatype into a contiguous memory.
MPI_Unpack	It unpacks a buffer according to a datatype into a contiguous memory.
MPI_Cart_Create	It creates a cartesian topology with a new communicator

Source: Adapted from the MPI documentation.