UNIVERSITY OF GRANADA

A Parallel Multi-objective Optimization Procedure for Protein Structure Prediction

by

Jose Carlos Calvo Tudela

A thesis submitted for the degree of Doctor Internacional en Ingeniería Informática

at the

CITIC-UGR Department of Computer Architecture and Computer Technology June 2012

Editor: Editorial de la Universidad de Granada
Autor: José Carlos Calvo Tudela
D.L.: GR 502-2013
ISBN: 978-84-9028-385-1

Declaración de Autoría

D. Julio Ortega Lopera y Dña. Mancia Anguita López, Catedrático y Profesora Titular de Universidad respectivamente del Departamento de Arquitectura y Tecnología de los Computadores

CERTIFICAN

Que la memoria titulada: "A Parallel Multi-objective Optimization Algorithm to Protein Structure Prediction" ha sido realizada por D. Jose Carlos Calvo Tudela bajo su dirección en el Departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada para optar al grado de Doctor Europeo en Ingeniería Informática.

Granada, a 1 de junio de 2012

Fdo. Julio Ortega Lopera
Director de la Tesis

Fdo. Mancia Anguita Lopez
Directora de la Tesis

"Research is to see what everybody else has seen, and to think what nobody else has though"

"Investigar es ver lo que todo el mundo ve, y pensar lo que nadie ha pensado"

Albert Szent-Gyorgyi

"What is a scientist after all? It is a curious man looking through a keyhole, the keyhole of nature, trying to know what's going on."

"¿Qué es un cientifico después de todo? Un curioso mirando a través de un agujerito, un agujero hacia la naturaleza, intentando saber que está pasando ahí fuera"

Jacques Yves Cousteau

"El investigador sufre las decepciones, los largos meses pasados en una dirección equivocada, los fracasos. Pero los fracasos son también útiles, porque, bien analizados, pueden conducir al éxito."

Alexander Fleming

Dedicada a todas las personas que durante tanto tiempo me han dicho una y otra vez cosas como "¿Cuándo terminas la tesis? dale un empujón que ya la tienes casi hecha", ya que esa frase me dice que les importo, que significo algo para ellos y que quieren lo mejor para mí.

UNIVERSITY OF GRANADA

Abstract

CITIC-UGR Department of Computer Architecture and Computer Technology Doctor Internacional en Ingeniería Informática

by Jose Carlos Calvo Tudela

Proteins are chains of amino acids whose sequence determines its 3D structure after a folding process. As the 3D structure of a protein exclusively determines its functionality (transport and transduction of biological signals, the possible enzymatic activity of some proteins, etc.), there is a high interest in the determination of the structure of any given proteins. Experimental methods such as X-ray crystallography and nuclear magnetic resonance (NMR) allow the determination of the 3D structure of a protein although they are complex and expensive. Thus, only about the 2% of the known proteins has known structures currently. The so called, protein structure prediction (PSP) problem is the approach to find the 3D structures of proteins by using computers.

This work proposes an approach to the protein structure prediction (PSP) problem: *PITAGORAS-PSP* (Parallel Implemented procedure with Template information, Ab initio Global Optimization, and Rotamer Analysis and Statistics for Protein Structure Prediction). This way, taking into

account its name, our procedure represents a hybrid approach that takes advantage of previous knowledge about the known protein structures to improve the effectiveness of an *ab initio* procedure for the PSP problem. Moreover, the procedure benefits from a parallel and distributed implementation of a multi-objective evolutionary approach that allows faster and wider exploration of the conformation space. The experimental results obtained from the present implementation of our procedure show improvements with respect to previously proposed procedures in the proteins selected as benchmarks from the CASP set (up to 27% of RMSD improvement with respect to one of the best procedures known at this moment in some proteins). We also present a new method to extract better torsion angles from protein structures, it can be used to build an improved data base for torsion angles that aids in the knowledge extraction from the known structures.

Our hybrid approach can be used as an efficient method to predict protein structures, but it can be also used to refine predictions of other methods, due to its capabilities to take advantage of results from prior knowledge.

Acknowledgements

I started to work in this topic five years ago. During this journey some people have supported me day after day and I would like to thank every and each one. Also, I have to thank the *Ministerio de Educación y Ciencia* for bringing me the opportunity to research under the FPU program.

I would like to express my deepest appreciation to Julio Ortega, from the University of Granada, for guiding me in this journey. I have learnt a lot from him due to his predisposition, hard work, guidance and commitment along the way. We have opened a lot of doors together, and in every moment I did not see the way to pass throw a problem he was there to give me hope and new ideas.

Also, I would like to thank Mancia Anguita, for her advice, support and collaboration during the last years of this way.

I would also like to thank Joshua Knowles and Julia Handl, from the University of Manchester, for receiving me in their University and help me in the development of a new parallel PAES approach.

Also, a very special thank to Albert Zomaya and Javid Taheri, from the University of Sydney, for receiving me with open arms, and spend time to create a new approach to extract optimized torsion angles from a protein.

Moreover, I would like to thank the Department of Computer Architecture and Computer Technology and its staff, specially José Luis Bernier, Alberto

Prieto, Ignacio Rojas, Manuel Rodriguez, Hector Pomares and Encarnación Redondo.

Finally, I would like to express a very special gratitude to my whole family, for their support, patience, love and for making this journey as pleasant as it can be.

Contents

List of Figures

List of Tables

Abbreviations

PSP	**P**rotein **S**tructure **P**rediction
PITAGORAS-PSP	**P**arallel **I**mplemented procedure with **T**emplate information, **A**b initio **G**lobal **O**ptimization, and **R**otamer **A**nalysis and **S**tatistics for **P**rotein **S**tructure **P**rediction
PF	**P**rotein **F**olding
CHARMM	**C**hemistry at **HAR**vard **M**acromolecular **M**echanics
AMBER	**A**ssisted at **M**odel **B**uilding with **E**nergy **R**efinement
NMR	**N**uclear **M**agnetic **R**esonance
MOGA	**M**ulti **O**bjective **G**enetic **A**lgorithm
NSGA	**N**on-dominated **S**orting **G**enetic **A**lgorithm
PAES	**P**areto **A**chieved **E**volution **S**trategy
CMAES	**C**ovariance **M**atrix **A**daptation **E**volution **S**trategy
SPEA	**S**trength **P**areto **E**volutionary **A**pproach

PDB	**P**rotein **D**ata **B**ank
RMSD	**R**oot **M**ean **S**quare **D**eviation
EA	**E**volutionary **A**lgorithm
GDT-TS	**G**lobal **D**istance **T**est - **T**otal **S**core
CASP	**C**ritical **A**ssessment of Techniques for Protein **S**tructure **P**rediction
TASSER	**T**hreading **ASSE**mbly **R**efinement
I-TASSER	**I**terative **T**hreading **ASSE**mbly **R**efinement
BOINC	**B**erkeley **O**pen **I**nfrastructure for **N**etwork **C**omputing

Prefacio

A través del trabajo de investigación que se refleja en esta memoria de tesis se ha buscado poner de manifiesto los beneficios que las arquitecturas de computadores de altas prestaciones, la disponibilidad de información a través de Internet, y los algoritmos evolutivos multiobjetivo, pueden aportar a la resolución de problemas complejos que aparecen en ciencia e ingeniería y que acarrean un interés práctico considerable. Aquí nos hemos centrado en la predicción de la estructura 3D de las proteínas. Se trata de un problema que se considera no resuelto, para el que existe abundante información y técnicas accesibles a través de Internet. La integración del trabajo previo realizado dentro de un procedimiento de optimización multiobjetivo que explore eficientemente el espacio de soluciones, y el aprovechamiento de las posibilidades que ofrece la tecnología de computadores a través del procesamiento paralelo, constituyen el objetivo primordial de esta tesis.

Las líneas de investigación con las que este trabajo tiene relación son:

1. Algoritmos evolutivos de optimización multi-objetivo.

2. Procesamiento paralelo.

3. Aplicación de las técnicas anteriores en un problema bioinformático como es la Predicción de Estructuras de Proteínas.

Durante toda esta memoria se abordará cada uno de estos temas y como se hibridan para generar una aproximación eficiente al problema bioinformático seleccionado. Debido a las características de este tipo de problemas, la carga de conceptos biológicos es alta, pero ello no hace que esta memoria sea más compleja de leer para alguien que no sea un biólogo experto, sino que sea más interesante si cabe. En cualquier caso, tiene sentido comenzar esta lectura con una visión general del problema biológico para avanzar posteriormente en los objetivos técnicos del trabajo que se presenta.

Las proteínas rigen el organismo realizando la mayoría de las funciones necesarias, como por ejemplo procesos encimáticos, anticuerpos y actividad celular [Lesk, 2002, 2000]. Puede decirse que la situación actual de nuestro organismo viene definida por el estado proteico del mismo. Desde este punto de vista, tener un conocimiento pleno del conjunto de proteínas que actúan en cada momento, nos daría la información necesaria para predecir cualquier comportamiento interno en nuestro organismo a corto o medio plazo. Esta información es particularmente útil en la lucha contra enfermedades como el cáncer, la demencia senil de tipo Alzheimer, y otras que siguen constituyendo importantes retos para la medicina actual. Hoy en día podríamos decir que una persona detecta que se ve aquejada por una enfermedad por dos vías principales: por los propios síntomas o por una prueba preventiva. En cualquier caso, y antes de que ningún síntoma

se dé a conocer, el organismo se ve afectado por un problema y ya está actuando de una u otra forma. Conociendo ese estado proteico y la función de cada proteína, podríamos detectar de forma muy temprana cualquier inicio de enfermedad y actuar en consecuencia para ayudar al organismo a deshacerse de ella.

Parece ser que el organismo está continuamente generando proteínas para actuar sobre si mismo. El estudio de las proteínas, y las acciones que realiza cada una de ellas, es un campo de investigación muy prometedor para disponer de una herramienta médica que ayude a detectar enfermedades lo antes posible. La farmacología también está implicada en este objetivo, ya que si bien el organismo genera proteínas para atacar a una enfermedad, es posible que no sea capaz de luchar contra algunas enfermedades, o que no genere suficientes proteínas en un momento dado. De esta forma el estudio de las proteínas también debería servir para entender su funcionamiento y poder crear proteínas optimizadas que luchen de forma más eficiente y en cantidades idóneas contra una enfermedad dada. Por tanto, no sólo se trata de entender cómo funcionan las proteínas actuales, sino de ser capaces de crear nuevas proteínas no conocidas hasta el momento, o de aumentar la proporción de proteínas que unas personas son capaces de generar y otras no.

Para todos estos avances parece que la clave está en entender el funcionamiento de una proteína, ya sea nativa del cuerpo humano o sintetizada en laboratorios. Y aún puede complicarse más el problema si hacemos de este estudio algo extensivo a todos los seres vivos, ya que las proteínas no

son algo único en la raza humana, sino que está presente en cualquier tipo de vida.

Según los expertos, el funcionamiento de una proteína radica en su estructura tridimensional más que en su secuencia molecular. Por tanto, en el estudio de las funciones de las proteínas, parece que el primer paso es conocer la estructura tridimensional de cada una. Obtener la estructura tridimensional de una proteína con suficiente precisión no es una tarea sencilla. Requiere meses de trabajo de un equipo de expertos, con laboratorios y maquinaria específica, lo que hace que obtener una sola estructura sea algo costoso tanto en dinero como en tiempo. Además hay proteínas que no pueden ser sometidas a los procesos que se realizan en laboratorio, de forma que no podríamos conocer experimentalmente su estructura tridimensional de forma precisa. Debido a la lentitud y coste del proceso de obtención experimental de estructuras 3D, y a la velocidad que se descubren nuevas proteínas, la diferencia entre secuencias de proteínas conocidas y estructuras tridimensionales calculadas está en continuo crecimiento. Tenemos un gran desconocimiento de la funcionalidad de las proteínas dado el hecho de que el porcentaje de secuencias en UniProt [UniProt, 2008] (es decir, el número de proteínas conocidas) con estructura tridimensional conocida en la base de datos online del Protein Data Bank [RCSB, 2009], era del 2.0% en 2004, 1.2% en 2007 y a finales de 2009 se situó en el 0.6%. Cada poco tiempo salen a la luz nuevas proteínas, lo que hace que el abismo de desconocimiento crezca cada día.

La bioinfomática es una ciencia que trata de resolver problemas biológicos

por medios informáticos, en definitiva trata de poner los recursos computacionales al servicio de los biólogos, médicos, químicos y demás expertos que estudian procesos relacionados con la vida. Estos problemas biológicos suelen ser muy complejos y lo que se intenta es obtener soluciones que a pesar de no ser exactas lleguen mucho más lejos de lo que un experto por los medios usuales podría llegar. Esto se debe a que estos problemas manejan un número muy alto de factores internos y externos, y en cambio tenemos pocos experimentos o poca información sobre el entorno, lo que hace que la extracción de conocimiento sea compleja y a veces ambigua. El problema de obtener la estructura 3D de una proteína cumple con creces estas dificultades, siendo un problema aun abierto en el que estamos lejos de encontrar una solución fiable. Este problema es conocido como el problema de la predicción de estructuras de proteínas, definido por tanto como el cálculo de la estructura tridimensional de una proteína dada su secuencia de amino ácidos.

Hoy día no se conoce con nivel de detalle suficiente cómo una secuencia de amino ácidos converge hacia una estructura [Anfinsen, 1973; Levinthal, 1968], y por ello es complejo abordar este problema, ya que no hay un modelo teórico que nos respalde. Desde que la célula crea la secuencia de amino ácidos hasta que ésta converge en su estructura tridimensional estable pasan del orden de milisegundos, en cambio los procesos que hoy día tenemos para calcular estas estructuras duran meses y son aproximaciones ya que no conocemos el proceso que guía esa evolución desde la secuencia hasta la estructura. Por todo ello, esta línea de investigación está abierta y lejos de un final, aún se debe trabajar mucho para llegar a soluciones

aceptables de precisión. Los grupos de investigación lo han entendido y no son pocos los que invierten sus esfuerzos en esta línea de investigación, generando múltiples vías de avance.

En este trabajo abordamos el problema de la predicción de estructuras de proteínas a través de un procedimiento que utiliza mecanismos híbridos en el proceso de predicción, plataformas paralelas para conseguir mayor capacidad de trabajo, algoritmos evolutivos que intentan satisfacer varios objetivos deseables en una estructura tridimensional de una proteína y que aprovecha conocimiento previo que existe sobre las estructuras calculadas hasta el momento.

Los métodos bioinformáticos que se aplican a este problema tienen cada vez mejor capacidad de predicción y aunque la solución sea solo una aproximación a la realidad, puede usarse en pasos previos dentro del ciclo de investigación para la creación de nuevas proteínas sintetizadas, ya que aportan información útil que puede conseguir que el proceso de análisis de proteínas sea más eficiente. Como se ha dicho, uno de los principales demandantes de este tipo de soluciones es la industria farmacéutica, ya que para generar nuevos medicamentos que produzcan proteínas eficientes contra una enfermedad, es necesario probar muchas opciones. Los expertos conocen aproximadamente qué están buscando. Por tanto, disponer de una herramienta, que en pocas horas o días les dé una aproximación de la proteína que sintentizarán con una composición dada, puede ayudarles a descartar experimentos que se alejan mucho de la solución perseguida y pueden centrarse en un conjunto reducido de aproximaciones que cumplen

con los requisitos buscados. Posteriormente se pueden usar los laboratorios para obtener estructuras precisas. Dado el tiempo y costo que tiene un laboratorio, conseguir que una aproximación bioinformática reduzca el número de opciones, hace que el proceso en su conjunto sea más rápido y económico. Si en un futuro el problema de la predicción de estructuras de proteínas resuelto mediante modelos computacionales tuviese niveles de fiabilidad similares a los obtenidos mediante laboratorios, entraríamos en una nueva era en la lucha contra las enfermedades, ya que la fase de creación y experimentación se reduciría notablemente tanto en tiempo como en coste.

Por lo expuesto anteriormente puede apreciarse que este campo de investigación es importante y prometedor, pero a la vez complejo. Hoy día, la meta se ve aún muy lejos, pero cada paso que se da ayuda a que el proceso de investigación en la creación de nuevos fármacos sea más eficiente y a que las grandes enfermedades que la humanidad padece hoy en día puedan curarse.

Los objetivos clave en esta tesis serán:

1. Entender claramente cual es el problema de la predicción de estructuras de proteínas y como podemos afrontarlo desde la bioinformática y el procesamiento de altas prestaciones.

2. Aplicar varios algoritmos multi-objetivo a este problema, teniendo en cuenta las peculiaridades del mismo.

3. Desarrollar las heurísticas necesarias que hagan de la aplicación de algoritmos evolutivos a este problema una opción viable.

4. Generar y aplicar esquemas paralelos que nos permitan reducir el tiempo de procesamiento y aumentar la calidad de las soluciones generadas aprovechando máquinas paralelas disponibles.

Podemos decir que el resultado buscado en este trabajo tiene dos facetas clave que son contradictorios, lo que hace que el propio proceso de investigación haya sido en sí un proceso multi-objetivo:

1. Conseguir estructuras de proteínas de la mejor calidad posible.
2. Reducir el tiempo de procesamiento al mínimo posible.

El trabajo se ha estructurado de la siguiente forma, para conseguir un hilo de explicación que invite a leer el documento y profundizar el problema abordado y la solución propuesta:

Capítulo 1. Introduction pretende situarnos en un punto de inicio para ser conscientes de la magnitud del problema y las herramientas y conocimiento de los que hay que partir para poder abordarlo.

Capítulo 2. Improving in the Pre-processing Phase using Prior Knowledge nos muestra los avances realizados hasta el momento en herramientas, heurísticas y conocimiento que serán aplicados en una fase de preprocesamiento que proponemos para este problema, creando así una base de información inicial y un punto de partida para el algoritmo de predicción, de forma que éste tenga una mejor guía durante su proceso de optimización.

Capítulo 3. Proposed Evolutionary Optimization Procedure for PSP presenta la aplicación de optimización multi-objetivo que hemos creado para este problema, así como las técnicas desarrolladas para aumentar la calidad de las predicciones.

Capítulo 4. A Speculative Parallel PAES and other Parallel Implementations nos muestra los esquemas paralelos desarrollados para hacer que los algoritmos evolutivos se ejecuten de forma rápida en una arquitectura de cómputo paralela.

Capítulo 5. Experiments and Results define los experimentos que hemos realizado para probar el trabajo realizado y muestra el resultado de cada una de las fases, incluyendo la calidad de la solución y la ganancia.

Capítulos 6 y 7. Conclusions and Future Work como último capítulo, resume todas las aportaciones realizadas durante esta etapa investigadora y propone el trabajo futuro que hay en esta línea de investigación, como puede ser la mejora de esquemas de predicción paralelos para el problema PSP, o la elaboración de una base de datos online con información de ángulos de torsión optimizados, que es inexistente hoy día, ya que en este trabajo se aporta un nuevo método para extraer esta información de las proteínas que consigue mejoras de más del 90% respecto al método anterior.

Preface

The current trends towards parallel high performance computers and the accessibility to software and hardware resources through Internet, along with the availability of population- based procedures such as evolutionary optimization algorithms, offer new possibilities to solve challenge problems in science and engineering whose resolution makes it possible many useful applications. In this PhD dissertation we have considered the protein 3D structure prediction problem. It is still considered as an open problem that has received a lot of attention since years. Thus, the main goal of this thesis is the integration of the work previously done and available through Internet inside an evolutionary multi-objective optimization procedure. It would make it possible an efficient solution space exploration by taking advantage of computer technology improvements through parallel processing.

The main objectives of this research process are the followings:

1. Evolutionary and multi-objective optimization algorithms.
2. Parallel processing.

3. Application of the above mentioned techniques in a bioinformatics problem, such as the Proteins Structure Prediction.

All these subjects will be tackled in this thesis, as well as how they hybridize to generate an approximation to the selected bioinformatics problem. Due to the typology of these problems, in this document there are a lot of biological concepts. This fact does not necessarily makes it more complex to read, and it provides complex applications where many concepts of parallel processing and evolutionary computation can be analyzed. Therefore, it makes sense to start this reading with a general view of the biological problem in order to advance later on the technical objectives of this work.

Organisms are controlled by proteins that perform most of the necessary functions such as the enzymatic processes, antibodies and cellular activity [Lesk, 2002, 2000]. It can be said that the current situation of our organism is defined by the configuration of its set of proteins. From this point of view, having a full knowledge of the collection of proteins that act in each moment will give us the necessary information to predict any internal performance of our organism in the short or medium term. This information is especially useful in the fight against the diseases of this century: cancer, Alzheimer's disease and other ones. Nowadays, diseases are detected in two ways: (1) the patient who detects some symptoms or (2) by a preventive test that detects the disease in an early stage. In any case, and before any symptom is shown, the organism has detected this problem and it is acting someway. From any knowledge about the state of these proteins and about the function of each protein, we could detect the beginning of a disease in

an early stage and we could act accordingly in order to help organism to get rid of that disease.

As can be seen, it seems that the organism is continuously generating proteins to perform actions inside our body. The study of proteins, and the actions that each of them performs, is a very promising research field in order to have a medical tool which would help us to detect diseases as soon as possible. Pharmacology is also present in these research lines, because although the organism generates proteins to attack a disease, it could not be able to fight against some diseases, or it may not generate enough proteins at a given moment. In this sense, the study of proteins could be useful to understand how they act in order to create suitable quantities of optimized proteins that would fight against a disease in an efficient way. Therefore, it is not enough to know how the current proteins act, but also to create new proteins that are not known until this moment, or copy proteins that some people are able to generate and other people are not.

Taking into account all these possibilities, it seems that the key goal is to understand how a protein acts, wether it is native to human body or it is synthesized in laboratories. And it can be even more complex if we extend it to all the living beings, since proteins are not only found in human beings but also in any type of life. In any case, we have a profound ignorance of about protein functionality owing to the fact that the percentage of protein sequences in UniProt [UniProt, 2008], with a solved protein structure in the PDB library [RCSB, 2009], was 2.0% in 2004, 1.2% in 2007 and by the end of 2009 it was 0.6%. Every so often, new proteins come out and this fact makes bigger the gap of ignorance every day.

According to experts, functionality of proteins depends on its 3D structure, and not from its molecular sequence data. Thus, the first step is to know the 3D structure of each protein. Nevertheless, to obtain the 3D structure of a protein with enough precision is not an easy task. It requires months of work of expert groups, with laboratories and specific instrumentation, and consequently a lot of money and time is required to obtain the 3D structure of only one protein. Moreover, there are some proteins which structures that can not be unveiled by the current experimental techniques and it is not possible to have a precisely knowledge about their 3D structure. Due to the slowness and the cost of the experimental process to obtain 3D structures, and rate of new proteins discovery; the gap between known molecular sequences and the calculated three-dimensional structures is growing very fast.

Bioinformatics is a science that aims to solve biological problems with computing resources. All in all, the goal is to put the computing resources at the disposal of biologist, doctors, chemists and other experts that study the processes related to life. These biological problems are usually very complex so we try to obtain solutions that, despite not being accurate, are better than these that even an expert with usual resources would be able to achieve. This is due to that besides these problems deal with a large number of internal and external factors, few experiments and information about the environment are available, thus making the extraction of knowledge more difficult and ambiguous. The problem of obtaining the 3D structure of a protein presents all these difficulties, it is still an open problem and we are far from finding a completely reliable solution. This problem is known as

the protein structure prediction problem.

Nowadays, it is not known how an amino acid sequence converges to a given 3D structure [Anfinsen, 1973; Levinthal, 1968], and as there is not a theoretical model that would support us, tackle this problem is a complex task. At most, some milliseconds pass since a cell creates the amino acid sequence until it converges to its stable three-dimensional structure.

In this dissertation, we provide an approach to the protein structure prediction problem, giving a general view of it and proposing a computing system which uses:

1. An evolutionary algorithm which tries to satisfy some desirable objectives in a three-dimensional structure of a protein and makes use of previous knowledge that exists about calculated structures up to now.

2. A hybrid system that includes information about secondary structures, homology predictions, and statistic libraries in the initial population and in the variables range,

3. A parallel implementation of the procedure to achieve a higher speed and/or efficiency by taking advantage of high performance architectures.

The bioinformatics approaches to this problem have improved their prediction performance. Although the solutions that can be reached today are only approximation to the 3D structure of the proteins, they can be used in

first stages of the research cycle, because they provides useful information for the protein analysis process. The pharmaceutical industry is one of the main demanders of this kind of approaches, as the design of new medicines requires to test a lot of options. Experts know more or less what kind of protein they are looking for. In this sense, having a tool that could give them an approximation of the required protein in a reduced amount of time (from hours to days), can help them to avoid experiments that are far from the pursued solutions. With this tool, experts are able to focus on a small number of approximations which fulfil the requirements they are looking for. If in the future the protein structure prediction problem solved by computing models would be more reliable than the one obtained in laboratories, we would be in a new era in the fight against diseases. It is because the drugs creation and experimentation would come down outstandingly both in time and costs.

We can say that the result we are looking for has two key objectives which could be contradictory. This makes the research process to be a multi-objective process:

1. Obtaining protein structures of the best possible quality.
2. Minimizing the processing time.

This work has been structured as follows in order to obtain an explanation thread which invites to read the document and go into the problem and the proposed solution in depth:

Chapter 1. Introduction. It tries to place us in a starting point in order to make us aware of the magnitude of the problem and the tools and knowledge that we have to use to approach it.

Chapter 2. Improving in the Pre-processing Phase using Prior Knowledge. It shows the advances carried out up to now about tools, heuristics and knowledge that can be applied in the pre-processing stages and will guide the prediction algorithm.

Chapter 3. Proposed Evolutionary Optimization Procedure for PSP. It presents the multi-objective optimization procedure that we have created for the PSP problem problem, as well as the techniques developed to increase the predictions quality.

Chapter 4. A Speculative Parallel PAES and other Parallel Implementations. It shows the parallel procedures developed to execute the evolutionary algorithms in a parallel calculation structures in a fast and clever way.

Chapter 5. Experiments and Results. It describes the experiments that we have carried out to test the work and provides the result of each stage, including quality of the final solution found and parallel performance.

Chapters 6 and 7. Conclusions and Future Work. As they are the last chapters, they summarize all the contributions carried out during this research stage and introduce the future work of this research line.

Chapter 1

Introduction

In this Chapter we analyze the Protein Structure Prediction (PSP) problem. In order to do that, we introduce some characteristics of proteins, why they are important for us, why we need to know their structure, how to get that structure by traditional methods and how Bioinformatic tries to solve this problem. We also present the state-of-the-art of the methods proposed to approach the PSP problem.

1.1 Protein Structure Prediction Problem

The first step in this way is to present the Protein Structure Prediction problem. An introduction to protein functionality is given in this section, and, by taking into account that this structure defines its functionality,

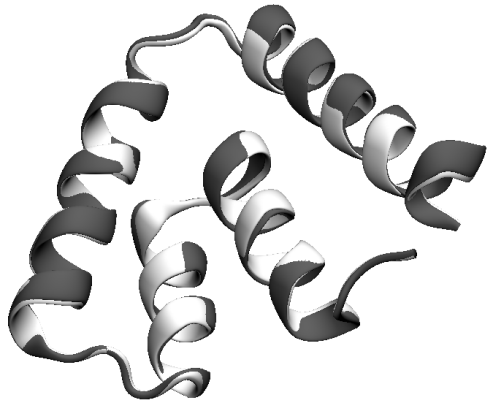


FIGURE 1.1: Structure of the protein 1UTG.

we also explain the traditional methods used to obtain the 3D structure of a protein. Finally, we present how Bioinformatic techniques have been applied to this problem, which is the main proposal of our work.

1.1.1 Proteins

Proteins (Figure 1.1) have important biological functions such as the enzymatic activity of the cell, attacking diseases, transport and biological signal transduction, among others [Lesk, 2002, 2000]. There is a high interest in the determination of the functionality of each protein because proteins manage the behavior of our body in a wide sense. Therefore, to understand how to attack diseases, we have to understand how the proteins work. From a DNA analysis we are able to predict possible behaviors of our body among our life like body changes, diseases and other problems. The interest in proteins comes with the fact that if we have a protein status of our body

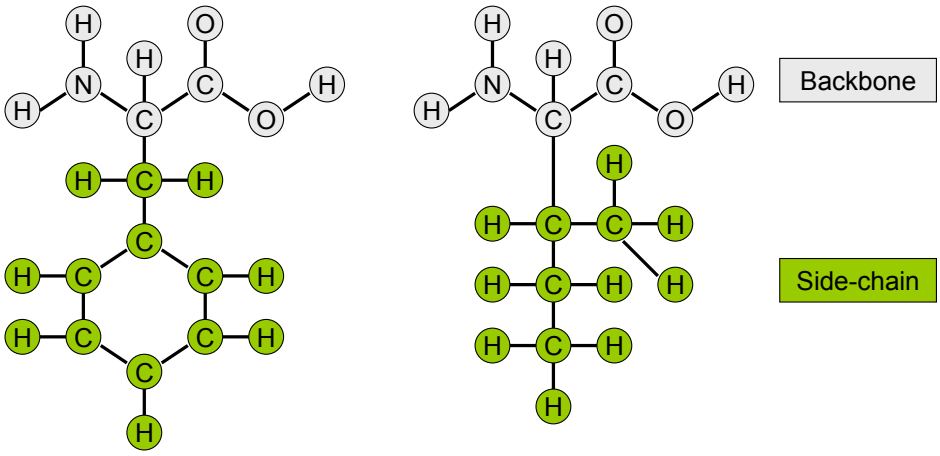


FIGURE 1.2: (left) Amino acid structure. (right) Two different amino acids, the backbone is the same in both of them, but the side-chain is different.

in this moment, knowing the functionality of each protein we could be able
Aminoácido
to detect the problems in a short time after they start, because the body reacts some time before we realize the problem. With that idea in mind, we can say that the DNA analysis helps us to prevent diseases, but protein analysis helps us to attack diseases once they appear. We also can use protein analysis to synthesize drugs that aid in the build of proteins that our body is not able to synthesize with the required levels.

Proteins are chains of amino acids selected from a set of twenty elements (Table 1.1). Each amino acid (Figure 1.2) can be considered as composed of two parts: the backbone and the side-chain. Every amino acid has the same backbone structure, but the side-chain is different for each of the twenty amino acids.

TABLE 1.1: The twenty different Amino acids in the human body. The table shows the name of each amino acid, and its representation in one and three letters.

Amino acid	One letter	Three letters
Alanine	A	Ala
Arginine	R	Arg
Asparagine	N	Asn
Aspartic acid	D	Asp
Cysteine	C	Cys
Glutamic acid	E	Glu
Glutamine	Q	Gln
Glycine	G	Gly
Histidine	H	His
Isoleucine	I	Ile
Leucine	L	Leu
Lysine	K	Lys
Methionine	M	Met
Phenylalanine	F	Phe
Proline	P	Pro
Serine	S	Ser
Threonine	T	Thr
Tryptophan	W	Trp
Tyrosine	Y	Tyr
Valine	V	Val

When a protein is being synthesized, the cell has to build each amino acid of the protein. Step by step, the cell make one amino acid that has to be connected to the rest of the protein. In the joining process, the last amino acid of the protein and the new amino acid have to be connected by their backbones. In the Figure 1.3 it is shown how two amino acids fit together

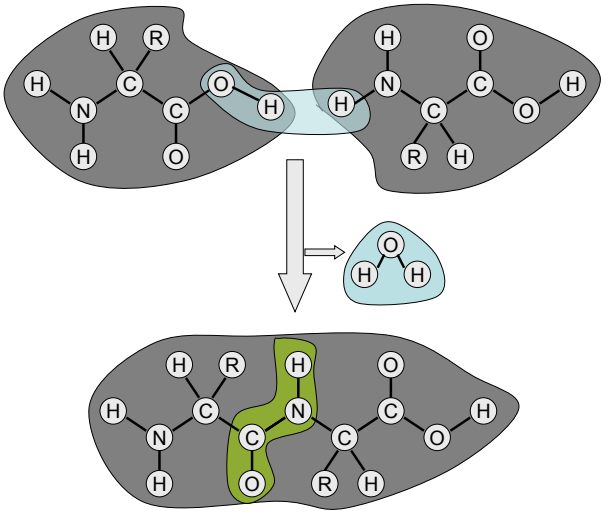


FIGURE 1.2: Two amino acids are joined and a water molecule is liberated.

and a water molecule gets free in the process. This is a peptide bond, a covalent chemical bond formed between two molecules when the carboxyl group (-COOH) of one molecule reacts with the amine group of the other molecule releasing a water molecule (H_2O).

Whenever an amino acid chain is synthesized, it folds together and determines its 3D structure [Anfinsen, 1973; Levinthal, 1968]. Moreover, although the amino acid sequence of a protein provides interesting information, the functionality of a protein is exclusively determined by its 3D structure [Lesk, 2002, 2000].

The protein structure can be divided into four levels: the primary structure, the secondary and super-secondary structure, the tertiary structure and the

quaternary structure (Fig. 1.4):

1. *The primary structure* defines the composition and the order of amino acids in the protein. The primary structure is held together by peptide bonds (Figure 1.2).

2. *The secondary structure* is a set of contiguous amino acids joined by some hydrogen bonds and presents a characteristic 3D structure that can be an α-helix or a β-strand. Then, the super-secondary structure is the combination of two secondary structures by a short connecting peptide.

3. *The tertiary structure* is a three-dimensional structure of a single sequence of a protein. All force-field atoms take part in this conformation and its determination is the goal of the PSP problem.

4. Finally, *the quaternary structure* refers to a protein formed by two or more amino acid sequences. This structure defines the relations between the different sequences of the protein.

Therefore, we face the problem to get the tertiary structure of a protein from its primary structure.

If we focus on a given disease, our body can create antibodies, but some problems could happen in this process:

1. The body could not create enough antibodies to fight against the actual episode of the disease.

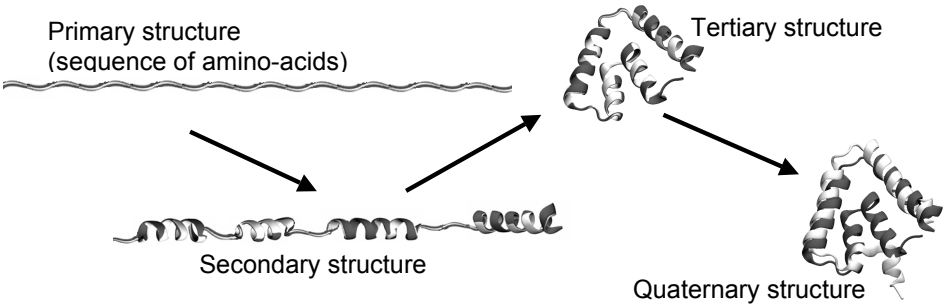


FIGURE 1.4: Protein structures. From the primary structure to the quaternary structure.

2. The body response has delayed too much time and the illness gets bigger than the capabilities of the antibodies.

3. The body is not able to create the required kind of antibodies.

4. No structured

Protein structures

4. The body is not able to create the required kind of antibodies.

Analyzing the proteins present in our bodies we could synthesize drugs to help people in the three first cases, because we could know how to fight against the disease and the only problem is the required quantity of antibodies. The last case is the hardest one, because even if we know the function of every protein, we will not aid in that case, because our body is not prepared to fight against this disease. This kind of disease makes the study of proteins even more interesting, as we need to create new proteins to fight the new diseases. This is the case of diseases like Cancer or Alzheimer. Experts may have some ideas about the structure of a protein to attack a disease, based on their experience, or proteins that although do not work

fine against the disease, they are in the way to be a good antibody. The problem here is that the experts need to create a lot of proteins to evaluate their structures in order to find the protein they are looking for. In that case, the developing of tools to get insight into the structure of a protein is a must.

This way, as the interest in the determination of the protein 3D structure has been here since years, there are a significant amount of bioinformatic approaches that have been proposed up to now. In the next two subsections we explain the most relevant procedures to cope with the PSP problem.

1.1.2 Traditional Methods

It is possible to reach the 3D structure of a protein experimentally by using methods such as *X-ray crystallographic* and *nuclear magnetic resonance* (NMR). These methods can give us a 3D structure of a protein with a noise around 2 $\mathring{A}$ RCSB [2009]:

1. *X-Ray Crystallography.* Consist in crystalize the protein, then if X-Ray radiation is applied to the molecule, the atoms diffract the radiation, thus is is possible to get something similar to a shadow of the molecule. Applying that process around the molecule we can measure the diffraction from many points of view, and it is possible to know approximately where the atoms are RCSB [2009].

2. *Nuclear Magnetic Resonance (NMR).* This method uses a nuclear magnetic resonance spectroscopy to obtain information about the

protein. The atoms in a protein are distributed conforming different contexts depending on the neighbor atoms of each one. NMR can analyze the differences in the magnetic moments for each context to get the distribution and positions of the atoms. This method can be applied to little proteins RCSB [2009].

Nevertheless, these processes are quite complex and costly as they would require months of expert work and laboratory resources. This situation comes clear if considering that less than a 2% of the protein structures have been solved [Lesk, 2002]. Also, a percentage of the known proteins can not be analyzed with these methods due that they can not be crystallized.

1.1.3 Bioinformatic Methods

An alternative approach to the determination of the 3D structure of a protein is to use high performance computing. Whenever a protein is synthesized it folds very fast. In the literature we can found that this process can take milliseconds or seconds, in any case it is something that need a very short time. Bioinformatic tries to solve biological problems using computational resources. Taking the problems of the traditional methods into account, using bioinformatic can aid with the important need of knowledge about protein structures Lesk [2002, 2000]; Handl, Kell, and Knowles [2007]. There are two main fields of research in this area: Protein Folding (PF) and Protein Structure Prediction (PSP).

Protein Folding Lesk [2002, 2000]; Handl et al. [2007] tries to simulate the whole process that controls the protein folding. In that case, this kind of approaches does not need any information about previous proteins structures. It tries to get information about how the protein folding works in a real protein. As far as we do not know how this process works in the natural way, Protein Folding is a hard and open problem. The apparent advantage of this method is that it can work from scratch, without any previous information. Therefore it could be the best solution to this problem. Nevertheless, up to now, this method is far to obtain feasible solutions to the problem.

In the other hand, we have the Protein Structure Prediction (PSP) Lesk [2002, 2000]; Handl et al. [2007], this problem does not care about the folding process, it only focus on the final structure and how we can translate a protein sequence into a protein structure. In order to do that, it is important to get information about previous knowledge to extract some information. It is also important to know some properties about protein structures like typical conformations, free energy or similar proteins. In this work we deal with the Protein Structure Prediction problem.

Nevertheless, the computational analysis of each conformation requires a significant time and this is a Grand Challenge Problem that still remains unsolved Lesk [2000]; Handl et al. [2007]; RCSB [2009]. Recently, efforts in protein structure prediction such as Rosetta@Home [Bradley, Misura, and Baker, 2005] and Predictor@Home [Taufer, An, Kerstens, and Brooks, 2006] have been developed by using grid or global computing. These proposal try

to improve previous methods and algorithms by orders of magnitude more computing power to improve the prediction quality [Bradley et al., 2005].

There are two main research lines in the area of PSP: *ab initio* and homology-based procedures:

Ab initio. They are also called *from scratch* procedures. They try to predict the tertiary structure of a protein only from its sequence of amino acids and no other information. These methods have to find the way to know whether a protein structure is feasible. If we achieve good results with this method we may be sure that the procedure is going to work with other proteins, and does not matter the kind of protein, because we would have a tool that knows whether or not a protein satisfies a feasible structure. Nowadays, this kind of methods frequently use some extra information or previous knowledge about the known structures like statistics of conformations or secondary structure predictions. The main issue of this approach is the evaluation of the conformation in order to determine its quality and take a decision about either finishing the process or following the explorations of the structures space.

In order to evaluate a protein conformation, the Quantum Mechanics could get us an accurate measure of the free energy of the molecule. Nevertheless, the problem with Quantum Mechanics is that the computational resources and time required for this approach out of the present resources and computing capabilities. Hence, Classical Mechanics is frequently used to solve the PSP problem. Classical Mechanics is not as accurate as Quantum Mechanics, the energy function obtained by this method is only an

approximation, and more information has to be included to get sufficient accurate predictions.

Homology-based procedures. They are also known as *template-based* modeling. They try to find an amino acid sequence similar to the target one inside the data base of known protein 3D structures. If a protein with similar sequence is found, probably the structure is going to be similar as well. These methods even check parts of the amino acid sequence of the target. In that sense, this procedure not only tries to find a very similar protein, but also some similar parts of a set of proteins. Then, these methods can assemble these structures to get the final conformation. In that cases, they also apply an *ab initio* method to join the parts and determine what conformation could be the best one.

In this work we present an *ab initio* protein structure prediction that can be used not only to predict a protein structure from scratch, but also to refine the results obtained by taking advantage of homology-based methods.

1.1.4 State-of-the-art

Nowadays the best algorithms in Protein Structure Prediction take part in the CASP competition [CASP, 2012]. Three of the best procedures are I-TASSER, ROSETTA@HOME, and PREDICTOR@HOME.

1.1.4.1 The CASP Competition

CASP (Critical Assessment of Techniques for Protein Structure Prediction) [CASP, 2012] is a bianual competition for the PSP approaches. This competition shows the current state-of-the-art in this field. One of its last conclusions was that we know very well how to copy, but we are far from predict a protein structure without previous information. That means that the homologies in little structures work properly, but in free modeling or big sequences, the current approaches get lost.

CASP deserves special recognition in any consideration of the role of computational methods for biology, since the process has transformed the level of recognition coming from experimentalists. CASP has become a model for all computational biology communities and an exemplar for evaluating techniques or methods beyond software of scientific computing [Wooley and Ye, 2007].

In their web page (http://predictioncenter.org/), all the information about the metrics, benchmarks, groups, methods and classifications can be found. We will use some protein structures described in the CASP web to compare our procedure.

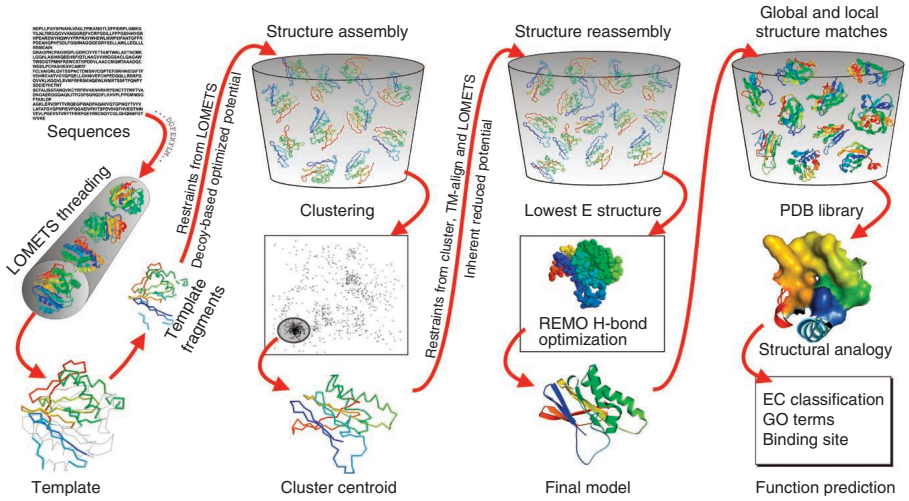


Figure 1 | A schematic representation of the I-TASSER protocol for protein structure and function prediction. The protein chain is colored from blue to red, from the N- to C-terminus.

Fig. 1.5: I-TASSER scheme from [Zhang, 2009; Roy et al., 2010; Wu et al., 2007]

1.1.4.2 I-TASSER

The iterative threading assembly refinement [Zhang, 2009; Roy et al., 2010; Wu et al., 2007] (I-TASSER) is an integrated platform for automated protein structure and function prediction based on the sequence-to-structure-to-function paradigm. Starting from an amino acid sequence, I-TASSER first generates three-dimensional (3D) atomic models from multiple threading alignments and iterative structural assembly simulations. The function of the protein is then inferred by structurally matching the 3D models with other known proteins.

When users submit an amino acid sequence, the procedure first tries to retrieve template proteins of similar folds (or super-secondary structures) from protein sequence. In the first stage of I-TASSER, the query sequence is matched against a nonredundant sequence database by position-specific iterated BLAST (PSI-BLAST)[5], to identify evolutionary relatives. A sequence profile is then created based on multiple alignment of the sequence homologs, which is also used to predict the secondary structure using the program PSIPRED[58]. Assisted by the sequence profile and the predicted secondary structure, the query sequence is then threaded through a representative PDB structure library using LOMETS, a locally installed meta-threading server combining seven state-of-the-art threading programs (FUGUE[52], HHSEARCH[44], MUSTER[53], PROSPECT[54], PPA[10], SP3 (ref. 55) and SPARKS[56]). In the individual threading programs, the templates are ranked by a variety of sequence-based and structure-based scores. The top template hits from each threading program are then selected for further consideration. The quality of the template alignments (and therefore the difficulty of modeling the targets) is judged based on the statistical significance of the best threading alignment, i.e. the *Z*-score, which is defined as the energy score in standard deviation units relative to the statistical mean of all alignments.

Stage 2: structural assembly. In the second stage, continuous fragments in threading alignments are excised from the template structures, and are used to assemble structural conformations of the sections that aligned well, with the unaligned regions (mainly loops/tails) built by *ab initio* modeling[10,12]. To improve the efficiency of conformational search, I-TASSER adopts a reduced model to represent the protein chain, with each residue described by its Cα atom and side-chain center of mass. Because the regions not aligned during the threading process usually have a lower modeling

from the PDB library by a meta-threading approach.

In the second step, the continuous fragments excised from the PDB templates are reassembled into full-length models by replica-exchange Monte Carlo simulations with the threading unaligned regions (mainly loops) built by an *ab initio* procedure. In cases where no appropriate template is identified, I-TASSER will build the whole structures by an *ab initio* procedure. The low free-energy states are identified by SPICKER [TINKER, 2004] through clustering the simulation decoys.

In the third step, the fragment assembly simulation is performed again starting from the SPICKER cluster centroids, where the spatial restrains collected from both the templates and the PDB structures are used to guide the simulations. The purpose of the second iteration is to remove the steric clash as well as to refine the global topology of the cluster centroids. The decoys generated in the second simulations are then clustered and the lowest energy structures are selected. The final full-atomic models are obtained building the atomic details from the selected I-TASSER decoys through the optimization of the hydrogen-bonding network (see Figure 1.5).

1.1.4.3 ROSETTA@HOME

ROSETTA@HOME [Bradley et al., 2005; Raman, Vernon, Thompson, Tyka, Sadreyev, Pei, Kim, Kellogg, DiMaio, Lange, Kinch, Sheffler, Kim, Das, Grishin, and Baker, 2009] is based on homologies. Moreover, ROSETTA@HOME uses BOINC [Anderson, 2004] to distribute the tasks among thousand of volunteers. This approach can be divided in three main steps:

1. Template detection, sequence alignment construction and ranking.
2. All-atom energy-based selection of templates/alignments.
3. Model generation.

In the first step, ROSETTA searches the best sequence alignments in the Protein Data Bank and ranks all the results in order to select the best model to start.

The second step is only used if there are two or more alignments with comparable scores. In this step the system makes an all-atom energy-based selection.

Finally, once the procedure has the best homology, it executes one method or another depending on the found homology:

1. *High sequence similarity template.* In that cases, the approach does not change any backbone amino acid in the aligned sequence. It only modifies regions with insertions or deletions, and regions with relatively low sequence conservation. Once the optimization process is done, ROSETTA executes a minimization of the side-chain to the whole protein. It executes only one loop.
2. *Medium sequence similarity template.* The optimization process can modify every amino acid in the sequence. Several loops of the algorithm have to be run in order to get accurate structures.

3. *Low sequence similarity template.* The same technique as in the *Medium sequence similarity template* alternative is used. Nevertheless, it is more aggressive, as it allows changes in the secondary structure, and big changes in every amino acid.

1.1.4.4 PREDICTOR@HOME

PREDICTOR@HOME [Taufer et al., 2006] also uses BOINC [Anderson, 2004] to distribute the tasks among thousand of volunteers. This approach is quite similar to ROSETTA@HOME in its architecture, but uses other techniques at low level. Its main steps are:

1. Homologies detection
2. Model generation
3. Refinement

The model generation uses a Monte Carlo conformation search to fill the gaps in the homology. Once the protein is in a low free energy, a refinement phase is executed. This refinement step is applied to every amino acid, and it executes a simulated annealing phase with CHARMM as the objective function to optimize the whole protein prior to return the solution.

1.1.4.5 Overview

The previously presented procedures are strongly based on homologies, therefore their effectiveness from *ab initio* decreases considerably. In the other hand, they use to have a refinement phase using optimization processes, but these evolutionary algorithms are quite basics, because they are local searcher, as is can be seen in their articles.

In that way, there is a field of researching in the optimization phase or in the *ab initio* methods.

1.2 Optimization Approaches to the PSP

There is no commonly accepted theory about how the protein fold into its tertiary structure. It is important to find the exact conformation of the protein, but a close enough structure could be also valid. As it has been said, the Protein Structure Prediction problem can be defined as an optimization problem that could be solved by an evolutionary algorithm that can start with an initial conformation and refines it to get a better structure generation by generation.

1.2.1 Main Concepts

The first issue that has to be considered to solve an optimization problem by using an evolutionary algorithm is the representation of the solutions

and a way to measure its fitness. With these two things, the algorithm evolves by changing the parameters that define the representation of the solutions in order to get a better solution.

1.2.1.1 Protein representation

This way, first question to deal with is to represent a protein. The main representation of the chain of amino acids that defines a protein is the 3D representation of its atoms, where each atom is represented by its three coordinates. That representation is very accurate but as it requires a lot of variables, it can make the computational methods too complex. In *ab initio* methods for PSP, we need a representation that can be modeled with as less variables as possible. The less is the number of variables used, the lower is the accuracy. Thus we need a trade off between accuracy and complexity. The main representations used (in the literature) are the followings [Cutello, Narcisi, and Nicosia, 2006]:

1. *3D atom representation.* It needs 3 variables for each atom. This means around 50 variables per amino-acid. Taking in account that each protein can have more than fifty amino-acids, this representation could be difficult to manage. Moreover, not all the configurations correspond to feasible proteins. For instance, we could represent bounded atoms far away each other, and that is not feasible.

2. *Partial 3D atom representation* This representation is similar to the previous one, but it only represents the main atoms of each amino-acid, for instance the Carbons and Nitrogen. This representation is simpler than the all atom representation, but still presents the same problem.

3. *Backbone 3D atom representation and side-chain centroids.* This representation is simpler because, for each amino-acid, we have nine atoms in the backbone, requiring 27 coordinates, plus the coordinates of the side-chain centroid, which means three coordinates more. Anyway we have only reduced the number of coordinates and, in any case, the number of variables can be also very high for proteins with a high number of amino acids.

4. *Backbone and side-chain torsion angles.* This representation can make always feasible bonds, and it needs, in each amino-acid, three torsion angles for the backbone (ϕ, ψ and ω) and zero to four torsion angles for the side-chain (χ_i). Table 1.2 shows the number of χ angles for each amino-acid and Figure 1.6 provides the torsion angles of one amino-acid.

In this work we have selected the torsion angles representation because it is the simplest, it always produces feasible bond lengths and it is one of the most used representation. Moreover as it is common to set the ω torsion angle to its ideal value of 180º, therefore the representation can require from two to six variables per amino-acid.

TABLE 1.2: χ angles per each amino acid.

residue	angles χ
GLY, ALA, PRO	only backbone
SER, CYS, THR, VAL	χ_1
ILE, LEU, ASP, ASN, HIS, PHE, TYR, TRP	χ_1, χ_2
MET, GLU, GLN	χ_1, χ_2, χ_3
LYS, ARG	χ_1, χ_2, χ_3, χ_4

With that representation, it is possible to build an all-atom representation to evaluate the whole protein. In TINKER Library [TINKER, 2004] there are some procedures to transform a torsion angles representation to a 3D all-atom representation, and to transform a 3D all-atom representation to a Protein Data Bank [RCSB, 2009] representation, which is one of the most common file formats for protein structures. Using these tools it is possible to use a torsion angles representation, which is easier to manage, inside the optimization procedure and generates a representation in the Protein Data Bank format to return a solution.

1.2.1.2 Free Energy Evaluation

As it is said before, we are going to move ourselves in the realm of Classical Physics. There are several methods to compute the free energy of a molecule using Classical Physics. The most frequently used are CHARMM (Chemistry at HARvard Macromolecular Mechanics) Cutello et al. [2006] or AMBER (Assisted Model Building with Energy Refinement) [Cornell,

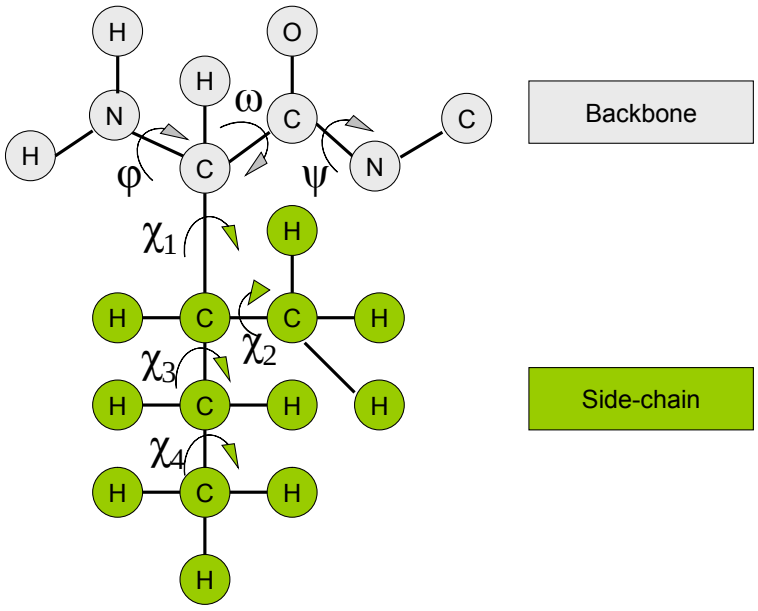


FIGURE 1.6: Each protein has 3 torsion angles in the backbone and up to 4 torsion angles in the side-chain per amino acid.

Cieplak, Bayly, Gould, Jr, Ferguson, Spellmeyer, T, Caldwell, and Kollman, 1995; Wang, Cieplak, and Kollman, 2000].

The CHARMM energy function is

$$\begin{aligned} E_{charmm} &= \underbrace{\sum_{bonds} K_b(b-b_0)^2}_{E_1} + \underbrace{\sum_{UB} k_{UB}(S-S_0)^2}_{E_2} + \underbrace{\sum_{angles} k_0(\theta-\theta_0)^2}_{E_3} \\ &+ \underbrace{\sum_{torsions} k_\chi[1+cos(n\chi-\delta)]}_{E_4} + \underbrace{\sum_{impropers} K_{imp}(\phi-\phi_0)^2}_{E_5} \\ &+ \underbrace{\sum_{non-bond} \epsilon_{ij}\left[\left(\frac{Rmin_{ij}}{\tau_{ij}}\right)^{12} - \left(\frac{Rmin_{ij}}{\tau_{ij}}\right)^{6}\right]}_{E_6} + \underbrace{\frac{q_i q_j}{e\tau_{ij}}}_{E_7} \end{aligned} \tag{1.1}$$

Whilst the AMBER energy function has the form

$$\begin{aligned} E_{amber} &= \underbrace{\sum_{bonds} \frac{1}{2}K_b(b-b_0)^2}_{E_1} + \underbrace{\sum_{angles} \frac{1}{2}k_0(\theta-\theta_0)^2}_{E_2} + \underbrace{\sum_{torsions} k_\chi[1+cos(n\chi-\delta)]}_{E_3} \\ &+ \underbrace{\sum_{j=1}^{N-1}\sum_{i=j+1}^{N}\left\{\epsilon_{i,j}\left[\left(\frac{Rmin_{ij}}{\tau_{ij}}\right)^{12} - 2\left(\frac{Rmin_{ij}}{\tau_{ij}}\right)^{6}\right] + \frac{q_i q_j}{e4\pi\tau_{ij}}\right\}}_{E_4} \end{aligned} \tag{1.2}$$

where [Cornell et al., 1995; Wang et al., 2000; Cutello et al., 2006]:

1. b is the bond length, b_0 is the bond equilibrium distance and k_0 is the bond force constant.

2. S is the distance between two atoms separated by two covalent bonds, S_0 is the equilibrium distance and K_{UB} is the Urey Bradley force constant.

3. θ is the valence angle, θ_0 is the equilibrium angle and k_0 is the valence angle force constant.

4. χ is the dihedral or torsion angle, k_χ is the dihedral force force constant, n is the multiplicity and δ is the phase angle.

5. ϕ is the improper angle, ϕ_0 is the equilibrium improper angle and k_{imp} is the improper force constant.

6. ϵ_{ij} is Leonnard Jones well depth, τ_{ij} is the distance betweens angles i and j, $Rmin_{ij}$ is the minimum interaction radius, q_i is the partial atomic charges and e is the dielectric constant.

In the CHARMM energy function, terms E_1 to E_5 model bond energies, and E_6 and E_7 represent non-bond energies. In the AMBER energy function, terms E_1 to E_3 compute the bond energies, and the last term calculate the interactions between non-bonded atoms. Bond energies are related to energies between bonded atoms, and non-bonded energies are related to energies between atoms that are close to each other in the 3D space, but far in the sequence of amino acids.

Prior to compute the free energy with AMBER or CHARMM, it could be interesting to introduce the molecule in a solvation to calculate the free energy of the protein in a real environment. In the TINKER Library [TINKER, 2004] there are some procedures to evaluate these two energy functions (CHARMM and AMBER) in a protein and there are some methods to consider the protein in a solvation (for a complete description, please see [TINKER, 2004]).

AMBER can be configured by using several alternatives, in [Wang et al., 2000] it is argued that the AMBER99 force-field (a specific configuration of AMBER) is better than the CHARMM one to work with molecules. In that sense, although we have used both force-fields, in the final version of our procedure PITAGORAS-PSP we have prefered to use AMBER99 instead of CHARMM because AMBER99 allows to specify the solvation.

1.2.2 The steps of the process

As it has been said, in a PSP problem, the input is the sequence of amino acids. After knowing the sequence of amino acids and by using the Table 1.2, it is possible to define an array of torsion angles. The optimization process has to improve this array. Whenever the optimization process needs to evaluate the fitness of a given array, the whole protein can be built by using the TINKER procedures, and the CHARMM or AMBER functions can be used to get its free energy. The lower is the free energy of the array, the better is the corresponding structure. The main steps of this process are shown in Figure 1.7. In this figure the *manager* module has to control the actual solution (or the population of solutions depending on the type of algorithm). In a mono-objective algorithm, the process can follow the scheme presented in that figure 1.7, but in a multi-objective algorithm, a decision phase is needed. In a mono-objective algorithm there is no decision phase because the best known solution is the one obtained by this algorithm. Nevertheless, in a multi-objective algorithm a set of non-dominated solutions is obtained. In that case a decision phase is required

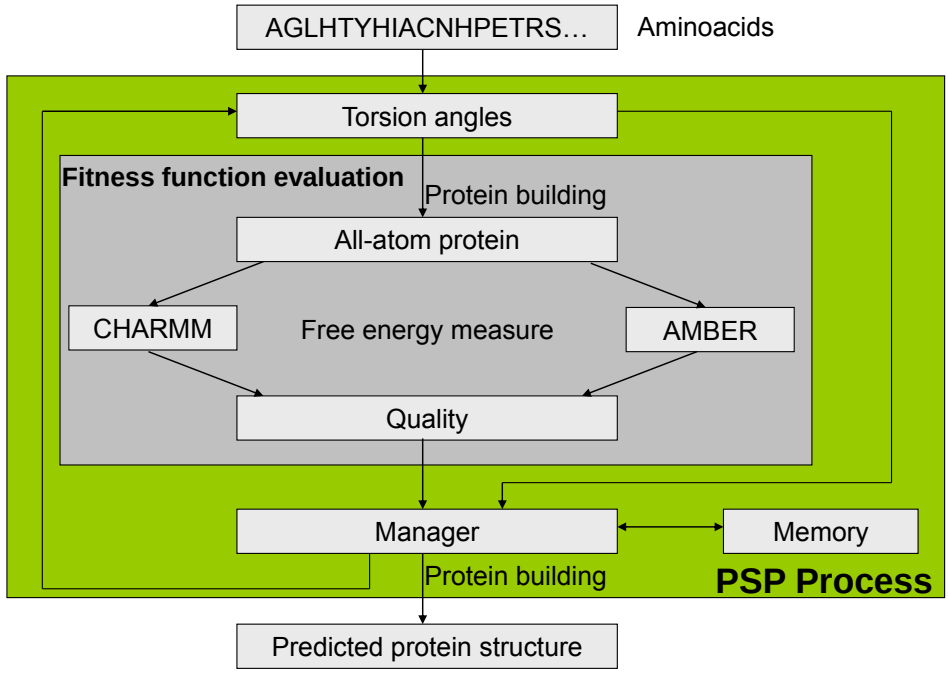


FIGURE 1.7: General scheme of an optimization process for PSP Problem.

in order to select one of this solutions as the representant of the Pareto Front.

1.3 Examples of Multi-objective Approaches to the PSP

Some methods using multi-objective optimization or state-of-the-art Evolutionary Algorithms applied to the PSP problem could be found in [Cutello et al., 2006; Cutello, Nicosia, Pavone, and Timmis, 2007; Handl et al., 2007; Day, Zydallis, and Lamont, 2002]. These methods tend to use good EAs, but they do not insert external information of the problem, or specific operators to the PSP.

Our work is focused in the creation of a hybrid approach that uses as much information as possible, including homologies, and the best multi-objective approaches with specialized operators and heuristics. To do so, we are going to take into account the previous works done up to now.

1.4 Outline

Up to now, we have present the PSP Problem and useful information to tackle this problem with evolutionary optimization. In the next chapters we will introduce our solution to this problem, and we will propose some techniques and methods to improve the quality of the predicted structures.

Chapter 2. Improving in the Pre-processing Phase using Prior Knowledge It describes some methods and the knowledge that can be applied in a pre-processing phase to introduce the optimization process as much capabilities as possible.

Chapter 3. Proposed Evolutionary Optimization Procedure for PSP exposes our evolutionary algorithms to solve the PSP problem and some heuristics and methods to improve the quality of the optimization.

Chapter 4. A Speculative Parallel PAES and other Parallel Implementations Some parallel schemes are proposed in this chapter to make the optimization process faster and better.

Chapter 5. Experiments and Results shows the results obtained with our proposal for both objectives: protein structure prediction quality and parallel performance.

Chapter 6 and 7. Conclusions and Future Work exposes the conclusion of our work, our contribution to the Protein Structure Prediction Problem and to the parallelization of evolutionary multi-objective parallel processing algorithm, and the future work in this line is presented.

Chapter 2

Improving in the Pre-processing Phase using Prior Knowledge

In this Chapter we analyze the available knowledge and information that can be taken from Internet in order to include it into our optimization process and improve the structure prediction quality. Our proposal is to take into account information about secondary and super-secondary structure prediction, the rotamer libraries and an initialization of the population based on homology predictions to improve the PSP performance.

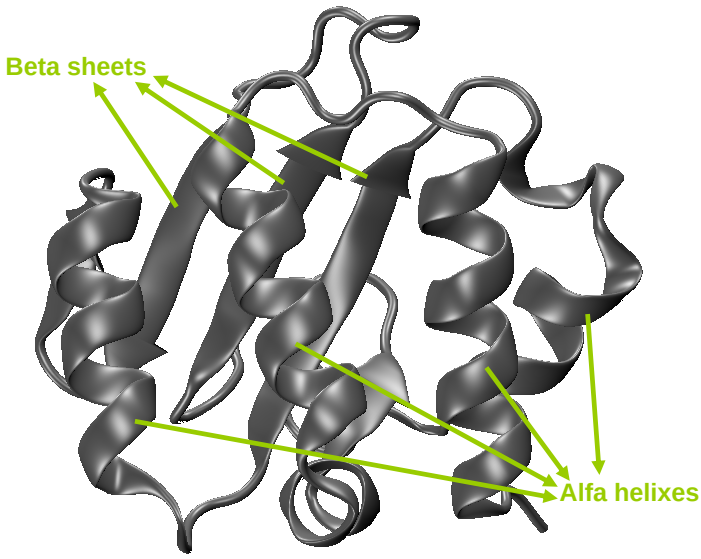


FIGURE 2.1: Examples of secondary structures (just sheets and helixes).

2.1 Secondary Structure Prediction

Some chains of amino acids have links between amino acids separated in the sequence by few other amino acids. When these links appear inside a chain of amino acids, the chain will present a regular structure such as the alpha helix or beta sheets, shown in Figure 2.1. The torsion angles of the amino acids included in one of these regular structures (alpha helixes or beta sheets) have very restrictive constraints. These structures that can be found in the protein structure are called secondary structures, and the problem of identifying them by computational methods is known as

secondary structure prediction [Singh, 2001; Goldman, Thorne, and Jones, 1996].

The previously referred constraints present in the secondary structures can be used to reduce the search space in an *ab initio* approach to the PSP problem. The Table 2.1 shows those constraints in the angles ϕ and ψ found in the α helixes and β sheets.

TABLE 2.1: Search space for each angle ϕ and ψ depending on the amino acid position in the secondary structure.

Super-secondary structure	ϕ	ψ
H (α helix)	[-75°, -55°]	[-50°, -30°]
E (β strand)	[-130°, -110°]	[110°, 130°]
undefined	[-180°, 0°]	[-180°, 180°]

2.2 Super-Secondary Structure Prediction

In order to get the super-secondary structure of a protein given its secondary structure, we have to analyze the conformation of the residues in the peptide bond between two secondary structures. They are classified into five types, namely, a, b, e, l or t [Sun and Jiang, 1996]. In this way, as it happens with the Secondary Structure prediction, a reduction in the search space of the PSP problem is obtained from the Super-Secondary Structure prediction (as shown in Table 2.2). The Figure 2.2 shows graphically where the short connecting peptide are. Sun and Jiang developed a method to predict the eleven most frequently occurring super-secondary structures:

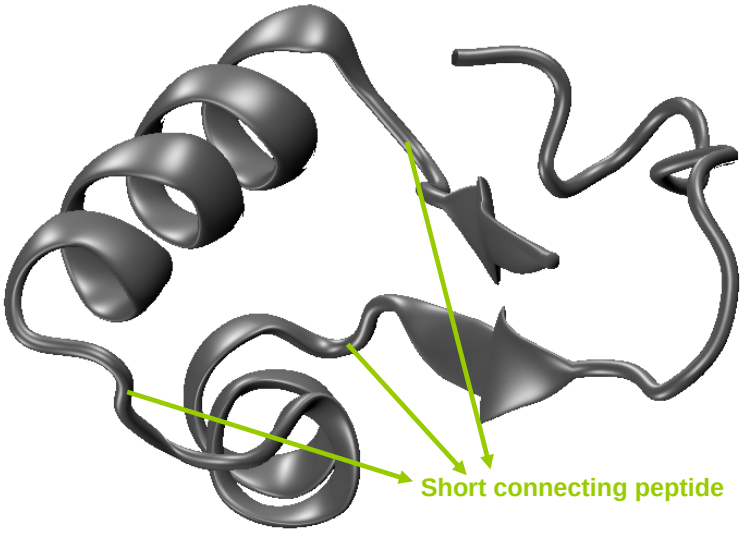


FIGURE 2.2: Short connecting peptide in the Super-Secondary structure.

H-b-H, H-t-H, H-bb-H, H-ll-E, E-aa-E, H-lbb-H, H-lba-E, E-aal-E, E-aaal-E and H-l-E where H and E are α helix and β strand, respectively (Table 2.1).

TABLE 2.2: Constraints for each angle ϕ and ψ depending on the amino acid position inside the corresponding super-secondary structure.

Super-secondary structure	ϕ	ψ
a	[-150°, -30°]	[-100°, 50°]
b	[-230°, -30°]	[100°, 200°]
e	[30°, 130°]	[130°, 260°]
l	[30°, 150°]	[-60°, 90°]
t	[-160°, -50°]	[50°, 100°]
undefined	[-180°, 0°]	[-180°, 180°]

2.3 Libraries for Torsion Angles Statistic

Given an amino acid, a rotamer library includes constraints for its side-chain torsion angles. That library provides different conformations, each one with its probability, angles and standard deviations for each side-chain torsion angle. Dumbrack and Cohen [Dunbrack and Cohen, 1997] have built many rotamers libraries that help us to identify constraints about these torsion angles. One of these libraries is the backbone-independent rotamer library. There are dependencies between side-chain torsion angles in the same amino acid, but these dependencies are not related with the backbone torsion angles. It is difficult to find a good representation for these constraints because their mutual dependency. The backbone-dependent rotamer library is more complex than the backbone-independent rotamer library, because the former includes the dependency between side-chain torsion angles and backbone torsion angles. Therefore, the size of the library increases significantly (more than 450.000 rows of information) and, although the information provided by this library is interesting, the time required to include this information in the optimization procedure also increases. Therefore, there are dependencies among backbone torsion angles and side-chain torsion angles, and there are also dependencies inside torsion angles of the side-chain.

An evolutionary optimization process has an independent range of movement for each variable. Thus, if we consider the backbone torsion angles and the side-chain torsion angles as independent variables in the set of variables, we ignore the cross-information between torsion angles. Therefore,

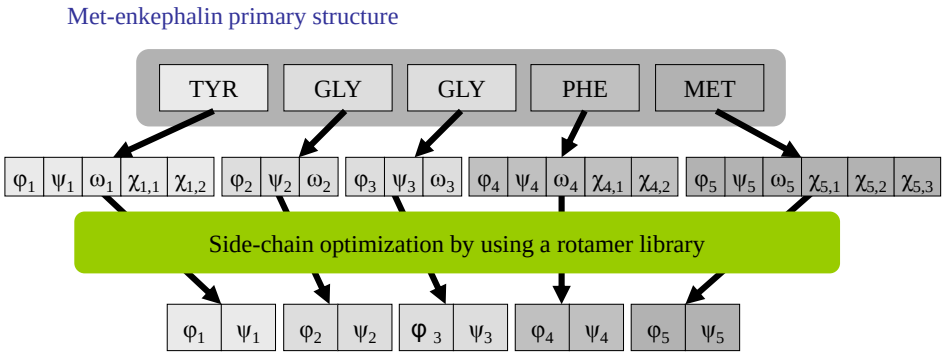


FIGURE 2.3: Search space reduction by using the backbone-dependent rotamer libraries.

a suitable way to approach the problem could be to manage the backbone torsion angles in an optimization process and, inside it, to include another mechanism to manage the side-chain torsion angles by taking into account the constraints included in the rotamer libraries and the values of the backbone angles.

Thus, in our approach we propose a new method to manage torsion angles by using the backbone-dependent rotamer library. As shown in Figure 2.3, we separate backbone torsion angles and side-chain torsion angles. The backbone torsion angles have their own constraints given by the Secondary and Super-Secondary Structure Prediction. The side-chain torsion angles have constraints depending on the values of the backbone torsion angles. Therefore, side-chain torsion angles have a range of movement that depends on the value of the backbone torsion angles of the same amino acid.

In Figure 2.4 there is a scheme for a possible algorithm to manage this information. The PSP approach manages only the backbone angles. In each

iteration of the evolutionary optimization, a new set of values are generated for the backbone angles. Then, the process gets into a side-chain optimization phase, this phase requires the backbone angles and the rotamers library. Finally, the PSP approach receives the backbone values generated before and the side-chain angles generated by the side-chain optimizator. The PSP problem executes a fixed set of interations as it is configured. Once the PSP process gets to the end, the backbone and side-chain angles are returned.

In Figure 2.4 it is shown how to manage both types of torsion angles. Whenever a mutation in the backbone torsion angles determines a change in the backbone, the side-chain torsion angles are set to the best conformation inside the new range. As this process is computationally expensive there are other options to reduce the required computing time:

1. To optimize the side-chain torsion angles only when the backbone torsion angles move to a new range of values.

2. To set the side-chain torsion angles to their most probable value depending on the backbone torsion angles value.

3. To include the side-chain torsion angles into the mutation process by using the constraints given by the backbone torsion angles, and set them to the most probable value only when the backbone torsion angles move to a new range of values.

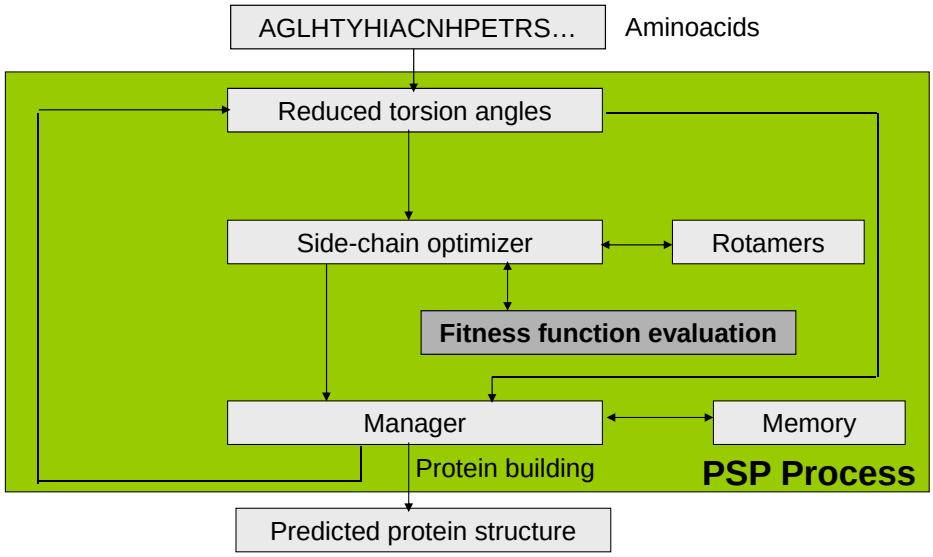


FIGURE 2.4: A possible procedure for managing torsion angles in an evolutionary algorithm by using rotamer libraries.

Algoritmo evolutivo para PSP

2.4 Optimization of the Torsion Angles Extraction Method

The Protein Data Bank includes all the known protein structures. All these structures have been obtained by using traditional procedures such X-Ray and NMR. These methods get a protein structure with a RMSD (Root Mean Square Deviation) around 2 $\mathring{A}$ depending on the size of the protein. Nevertheless, these PDB files have some noise in the 3D coordinates. Although The noise in the PDB file affects to all the atoms, globally the shape of the protein structure is very similar to the real one. Nowadays,

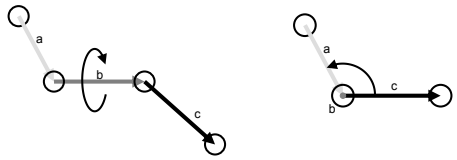


FIGURE 2.5: Representation of a torsion angle in the bond b from two points of view.

the calculation of the torsion angles is just the application of a mathematical formula. In Figure 2.5 it is represented a torsion angle between three atom bonds, a, b, and c, and equation 2.1 (where a, b and c are vectors in $\Re^3$, $\times$ is the vectorial product and $\cdot$ is the dot product) shows how to calculate a torsion angle given the three atom bonds:

$$\phi = atan2(|b|a \cdot [b \times c], [a \times b] \cdot [b \times c]) \tag{2.1}$$

As it is described before, it seems to be easy to go from a protein to a representation in terms of torsion angles and to get the same PDB file from its torsion angles, but in fact this is not true. An amino acid can have more than 20 atoms and, to represent these atoms in the 3D space, we need 60 real variables (three coordinates per atom). Thus, if we use 5 torsion angles for one amino acid we have to take into account a lot of information. Therefore, building a protein from the torsion angles involves to use not only the torsion angles but also the known bond lengths and the known angles. The problem is that the noise affects all the angles, although most of them are angles with fixed known angles. Therefore, if we use the torsion angles obtained mathematically and we mix them with the known

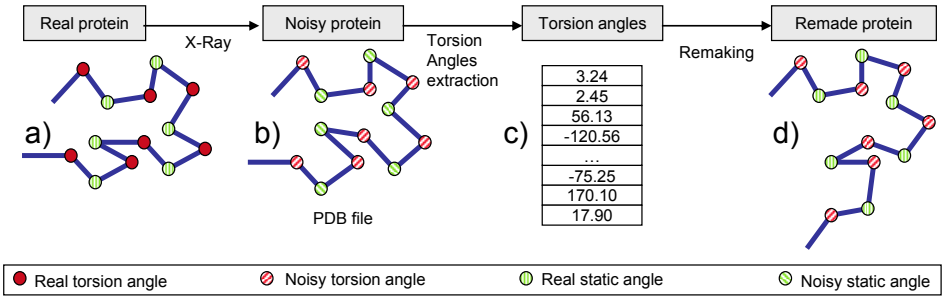
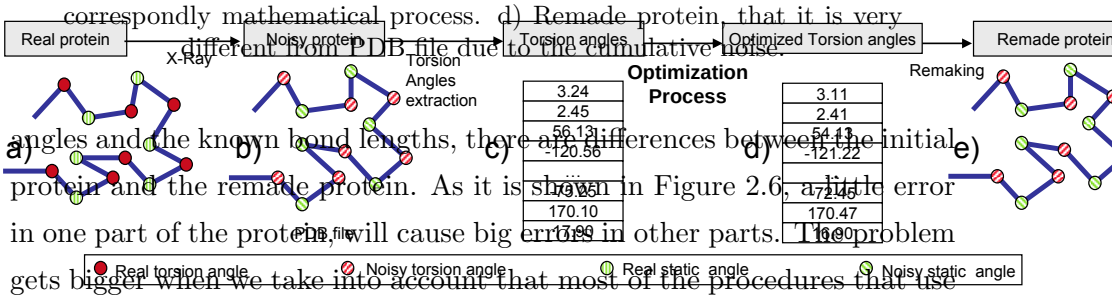


FIGURE 2.6: a) Example of a real protein structure. b) PDB file protein structure, the structure is similar to the real one, but there is noise in the atom positions. c) Torsion angles extracted from the PDB file by the correspondly mathematical process. d) Remade protein, that it is very different from PDB file due to the cumulative noise.



angles and the known bond lengths, there are differences between the initial protein and the remade protein. As it is shown in Figure 2.6, a little error in one part of the protein, will cause big errors in other parts. The problem gets bigger when we take into account that most of the procedures that use torsion angles use to set the omega torsion angle to its ideal value of 180°. In that case, the cumulative noise increases as it is shown in Figure 2.7.

Having torsion angles that represent real proteins could be very important to extract statistical information about proteins like rotamer libraries. Thus, a set of torsion angles can be seen as a summary of a protein, if we are not able to remake the protein from its torsion angles, then the information is not correct, therefore, we could create a rotamer library from noisy information. This work presents a method to minimize the difference between the initial PDB file and the remade protein by optimizing torsion

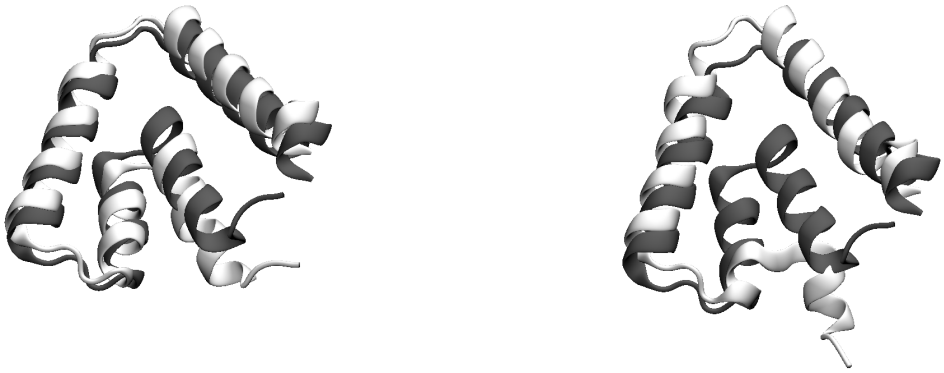
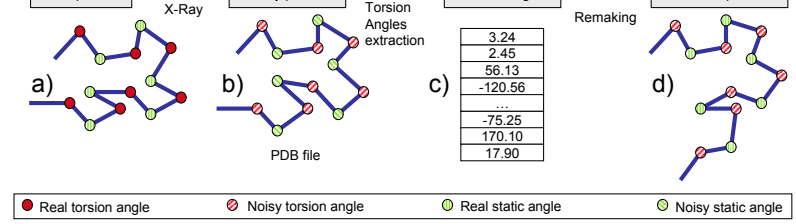


FIGURE 2.7: Real protein versus: [left] Remade protein using the mathematical torsion angles and [right] Remade protein using the mathematical torsion angles ignoring omega torsion angle

angles, and making that torsion angles absorb the noise in the known angles and the known lengths to get a remade structure with a similar shape. Therefore, these optimized torsion angles will be used to extract useful information about protein structures that aids in the future algorithms to solve the PSP problem.

Although the PDB file shape could be significantly different to the remade protein given its torsion angles, this difference is the addition of a lot of small errors. Therefore, each variable in our initial torsion angles must be very close to the optimal value we need to absorb the noise. Taken that information into account, the best strategy to refine the torsion angles is a local search. To do that, we have analyzed traditional local search algorithms like the so called gradient descent, and CMA-ES (Covariance Matrix Adaptation Evolution Strategy) [Kern, Müller, Hansen, Büche, Ocenasek,



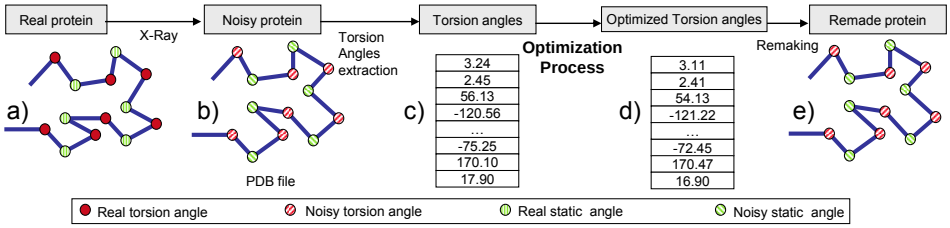


FIGURE 2.8: a), b) and c) correspond to a), b) and c) in Figure 2.6. d) Optimized torsion angles that absorb the noise in the rest of angles and bond lengths. e) Remade protein using optimized torsion angles, it is very similar to the original PDB file.

and Koumoutsakos, 2004; Hansen, 2006] one of the best local search algorithms proposed up to now.

In Figure 2.8 it is shown the effect of the optimization of the torsion angles to absorb the noise and to obtain a remade protein that is more similar to the original PDB file than the one obtained by using the mathematical torsion angles (Figure 2.6). Figure 2.9 shows the main structure followed in our procedure to refine the torsion angles.

The gradient descent process is based on the observation that if the real-valued function $f(X)$ is defined and differentiable in a neighborhood of a point X_0, then $f(X)$ decreases fastest if one goes from X_0 in the direction of the negative gradient of f at X_0, $-\nabla f(X_0)$. It follows that, if $X_{n+1} = X_n - \gamma \nabla f(X_n)$, for $\gamma > 0$ and a small enough number, and $\nabla f = \left(\frac{\delta f}{\delta x_1}, \frac{\delta f}{\delta x_2}, \cdots, \frac{\delta f}{\delta x_m}\right)$, then $f(X_n) \geq f(X_{n+1})$. With this observation in mind, if one starts with a guess solution X_0 for a local minimum of f, and considers the sequence X_0, X_1, X_2, ..., so hopefully this sequence converges to the desired local minimum. The first modification we have

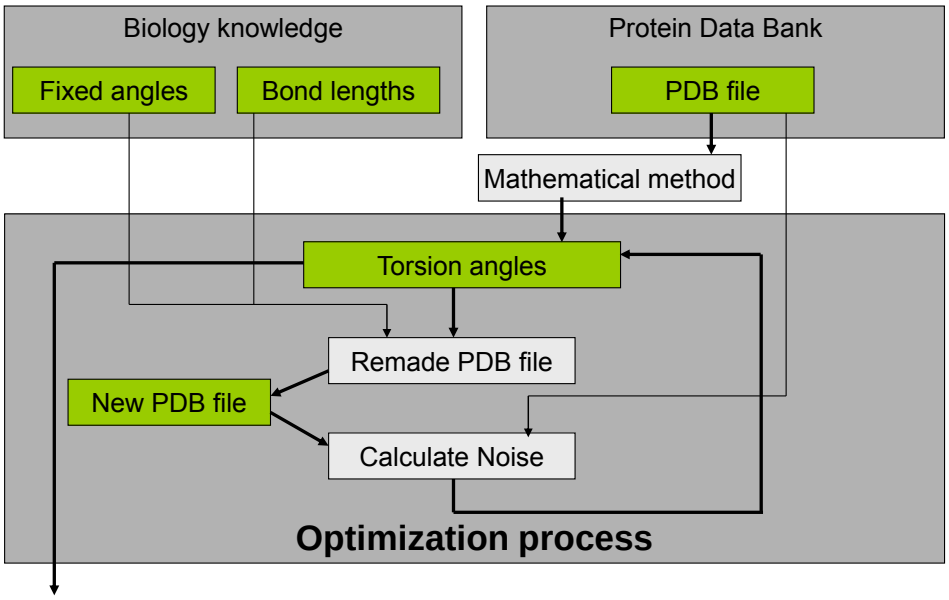


FIGURE 2.9: Generic scheme for the torsion angles optimization process based on local search. The starting point is the torsion angles set obtained by the mathematical process and these torsion angles are modified to absorb the noise in the remade protein. The optimized torsion angles obtained by this scheme can generate better remade proteins than those obtained from the mathematical torsion angles.

applied to that formulation is to check wether it is better to go in the direction of the gradient or in the opposite way, because in several problems it uses to work better due to local minimum. Thus our initial formulation is $X_{n+1} = X_n \pm \gamma \nabla f(X_n)$. Several changes in the initial formulation of gradient descent has been proposed in this work:

1. *Method 1.* To keep the initial formulation. Thus, it changes all the variables at the same time. $X_{n+1} = X_n \pm \gamma \nabla f(X_n)$

2. *Method 2.* In the initial formulation, all the variables change at the same time. This method proposes to change one variable at each time: $X_{n+1} = X_n \pm \gamma \nabla f_i(X_n); i = 0, 1, ...m; f_i(X_n) = \left(\frac{\delta f}{\delta x_i}\right)$

3. *Method 3.* This method is similar to *Method 2*, but instead of changing the variables in order, each time it changes a randomly selected variable. When it can not change any variable, it uses the *Method 2* to be sure that all variables have been selected. Thus, $X_{n+1} = X_n \pm \gamma \nabla f_i(X_n); i = rand(m)$

4. *Method 4.* Is a mix between *Method 2* and *Method 3.* This approach checks all the variables, but randomly. Hence, in each iteration it checks all the variables. It does it in a random way and does not check one variable twice in the same iteration. Thus, $X_{n+1} = X_n \pm \gamma \nabla f_i(X_n); i = rand\_without\_repetition(m)$

5. *Method 1**, *Method 2**, *Method 3** and *Method 4*.* They are the same methods explained before but they use a variable learning rate γ_n. This parameter decreases with the time, and hence it changes the optimization behavior, allowing big changes at the beginning, but being conservative at the end: $X_{n+1} = X_n \pm \gamma_n \nabla f(X_n); \gamma_{n+1} = 0.9 \times \gamma_n$

In our problem, X is the set of torsion angles, and $f(X)$ is the RMSD (Formula 3.5) between the initial and the remade PDB file. To use the RMSD formula, it is necessary to build the protein from its torsion angles, to fit both structures, and then the formula computes the error between the 3D position of each pair of corresponding atoms in both structures. This

process can not be expressed as a differentiable function, hence the gradient in each point is calculated by checking its neighborhood as seen in Formula 2.2.

$$\frac{\delta f}{\delta x_y} = \frac{f\left(\left[x_1, \cdots, x_y + \frac{\varepsilon}{2}, \cdots, x_n\right]\right) - f\left(\left[x_1, \cdots, x_y - \frac{\varepsilon}{2}, \cdots, x_n\right]\right)}{\varepsilon} \quad (2.2)$$

2.4.1 The Covariance Matrix Adaptation Evolution Strategy

The CMA-ES (Covariance Matrix Adaptation Evolution Strategy) [Kern et al., 2004; Hansen, 2006] is an evolutionary algorithm for difficult non-linear non-convex optimization problems in continuous domain. The CMA-ES is a second order approach to estimate a positive definite matrix within an iterative procedure (more precisely: a covariance matrix, that is, on convex-quadratic functions, closely related to the inverse Hessian). This makes the method feasible on non-separable and/or badly conditioned problems. The CMA-ES does not use or approximate gradients and does not even presume or require their existence. This makes the method feasible on non-smooth and even non-continuous problems, as well as on multimodal and/or noisy problems. It turns out to be a particularly reliable and highly competitive evolutionary algorithm for local optimization [Kern et al., 2004; Hansen, 2006].

To use CMA-ES, it is necessary to define a fitness function, the set of variables and the fitness function evaluations it has to compute. As it has

been said, in our problem, the fitness function is the RMSD between the real and the remade protein. The torsion angles are the variables, and the total fitness function evaluations grows according to the required quality of the solutions. In general, the more fitness function evaluations we allow, the better solutions we obtain and the more execution time it is required.

With that configuration, CMA-ES is going to search a set of torsion angles that minimize the difference between both structures, the real and the remade ones. Therefore, the final set of torsion angles can be used to remake a better protein than with the mathematical torsion angles extracted from the real protein.

2.5 Pre-processing

The input to the PSP problem is a sequence of amino acids. The preprocessing phase is going to reduce the search space and to include known information inside the optimization phase. The preprocessing steps are:

1. *Secondary Structure Prediction.* There are many algorithms to predict the secondary structure of a protein [Singh, 2001]. In [Singh, 2001]can be seen that the accuracy is up to 70% in this prediction. The input to those algorithms is the chain of amino acids, and the output is a sequence of symbols, having one symbol for each amino acid. Each symbol describes that an amino acid is in an α Helix, or in a β Strand or the amino acid is out of any secondary structure. Therefore, before this step we know some information about each amino acid.

2. *Super-Secondary Structure Prediction.* As described before, [Sun and Jiang, 1996] give us some information about how define the super-secondary structure by using the short peptide chain between two secondary structures. Therefore, this step has to analyze the output of the Secondary Structure Prediction phase to complete the information of the amino acids in a short connecting peptide.

3. *Creation of the backbone and the side-chain torsion angles variables.* Given the sequence of amino acids and the Table 1.2, it is possible to build the protein representation based on its torsion angles. Each variable has a floating point representation.

4. *Setting the backbone torsion angles constraints.* By Using information given by Secondary and Super-Secondary Structure Prediction, Tables 2.1 and 2.2, and the corresponding representation of the variables, we can set the backbone constraints, that should be the same for any optimization process used to solve the PSP problem.

5. *Rotamer libraries loading.* There is one file with the information of the backbone-dependent rotamer libraries. In this step we have to read the file and introduce all the information in a structure able to work with, in a fast way.

6. *Initial protein loading.* This step has to load a protein to refine it, or to execute an homology based algorithm for PSP in order to include more information to the optimization process. Once the protein is loaded, it is necessary to extract the values of the torsion angles of that protein as seen in the section 2.4.

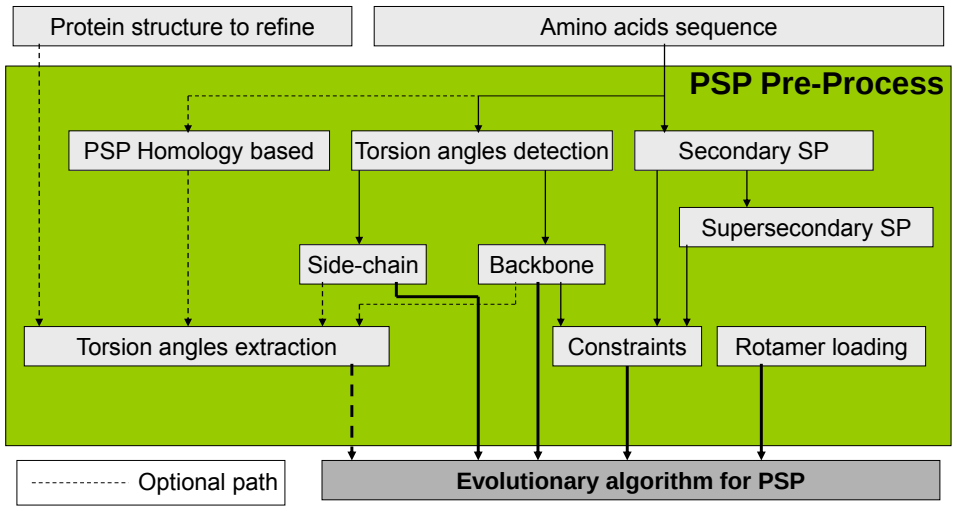


FIGURE 2.10: Pre-processing phase. The optional paths can be used to refine a protein or to use an homology based algorithm for PSP in order to include more information to the optimization process

Preprocesamiento

2.6 Conclussions

In the PSP problem, the search space is very vast. The preprocessing phase, seen in this chapter, allows us to reduce the search space. It is also possible to include information in the next phase by using the rotamer libraries.

In this chapter we have proposed a new method to optimize the torsion angles of a known protein in order to include the information into an optimization algorithm.

Publications with contributions of this chapter:

1. BIOSTEC 2011: *A Method to Improve the Accuracy of Protein Torsion Angles* [Calvo, Ortega, Anguita, Taheri, and Zomaya, 2011c]

2. 10th International Work-Conference on Artificial Neural Networks (IWANN 2009): *Protein Structure Prediction by Evolutionary Multiobjective Optimization: Search Space Reduction by Using Rotamers* [Calvo, Ortega, Anguita, Urquiza, and Florido, 2009c]

Chapter 3

Proposed Evolutionary Optimization Procedure for PSP

In this Chapter, our implementations of PSP procedures based on the multi-objective algorithms NSGA2 and PAES are presented. We also describe some heuristics to improve our initial PSP approaches and a new method to extract torsion angles from a protein based on an optimization method that outperforms the previous method.

3.1 Evolutionary Optimization

The evolutionary optimization paradigm is based in the continuous modification and evaluation of a feasible solution to one problem. That behavior is repeated until the solution is good enough or the maximum number of evaluations is reached. In the last case, the solution is the best found during the search Zitzler and Thiele [1999b]; Zitzler, Deb, and Thiele [2000]; Deb, Pratap, Agarwal, and Meyarivan [2002]; Knowles and Corne [1999]; Bonissone, Subbu, Eklund, and Kiehl [2006]; Hansen [2006].

The evolutionary optimization is used when there is not any theoretical model of the problem. Usually this kind of problems have a vast search space that make impossible to test each and every possible solution. In that sense it is not only to test as many options as it can, but also it has to use some heuristics in order to move to better solutions in each iteration.

The algorithm has to increase the goodness of an evaluation with respect to the previous possible solutions tested. In that process, we can divide that evaluation into tow different cases: a mono-objective approach and a multi-objective approach.

3.1.1 Mono-objective Algorithms

This algorithms have to optimize only one goal in order to solve the problem. This kind of algorithms for PSP are widely studied and many of them can be found in the literature Cutello et al. [2006]; Goldman et al. [1996];

Krasnogor, Hart, Smith, and Pelta [1999]. These algorithms can be mainly divided into Local Search and Global Search.

1. *Local Search*: It is used when the search space is not very big, and it is relatively easy to know where are the best solutions, or we only need to refine a solution. These methods search in the neighborhood of the current solution, and they do not try to move to other areas of the search space. Some examples of local searching are: Hill climbing, Monte Carlo, Evolutionary Strategies Hansen [2006], etcetera.

2. *Global Search*: These methods can search in the whole search space and depending on the heuristics they can explore different areas of the search space in the beginning and performs a local search when a promising area is found, or it can explore in any moment, or another strategy. Some examples of global searcher are: Simulated annealing, Max-Min, Swarm Intelligence, etcetera.

Mono-objective algorithms can also be divided into evolutionary population and evolutionary individuals. The *first* type consists in a population that is modified as a generational evolution. The main idea of these algorithms is to start with a population, generate a new population using a crossover between the first population. This new population is mutated, and then some individuals from the new population can replace others individuals from the first population and the process starts again, this process is shown in figure 3.1.

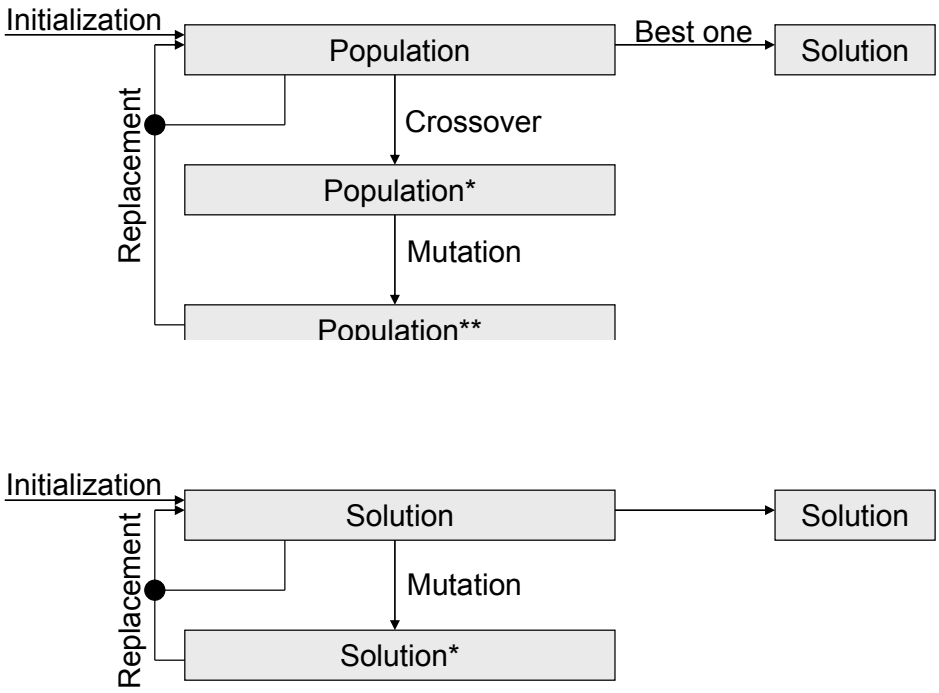


FIGURE 3.2: Basic scheme of an individual evolution based algorithm.

The *second* type of approach considers only one individual which is mutated to move to better search areas. The main idea of this method is to start from an initial solution and mutate the solution in each iteration, and if the new solution satisfies some criteria, it replaces the previous solution. This process is shown in figure 3.2.

Algoritmo evolutivo con un solo individuo

3.1.2 Multi-objective Algorithms

Multi-objective optimization [Deb et al., 2002] can be defined as the problem of finding a vector ($x = [x_1, x_2, ..., x_n]$) that satisfies a given restriction set ($g(x) \leq 0, h(x) = 0$) and optimizes the vector of objectives $f(x) = \{f_1(x), f_2(x), f_m(x)\}$. The objectives are usually in conflict between themselves, thus, optimizing one of them is carried out at the expense of the others. This leads to the need of making a compromise, which implies the concept of Pareto optimality. In a multi-objective optimization problem, a decision vector x* is said to be a Pareto optimal solution if there is not any other feasible decision vector, x, that improves one objective without worsening at least one of the other objectives.

Using a multi-objective optimization procedure for the PSP problem is justified from different points of view. In example, in Figure 3.3 the energy function, E, is defined by adding two potentials, A and B, and one of them is multiplied by a constant, K. In order to exactly determine the energy function E, it is necessary to know the exact value of K. In the other hand, as it is shown in the bottom right part of the Figure 3.3, if the problem is formulated as a multi-objective problem in which each of the potentials is one of the objectives, the Pareto Front will include the searched solution. Obviously, to solve the problem (the minimum of E), it is important to determine the optimun solution from the Pareto Front, but in any case, a set of solutions is provided and some information can be extracted from them.

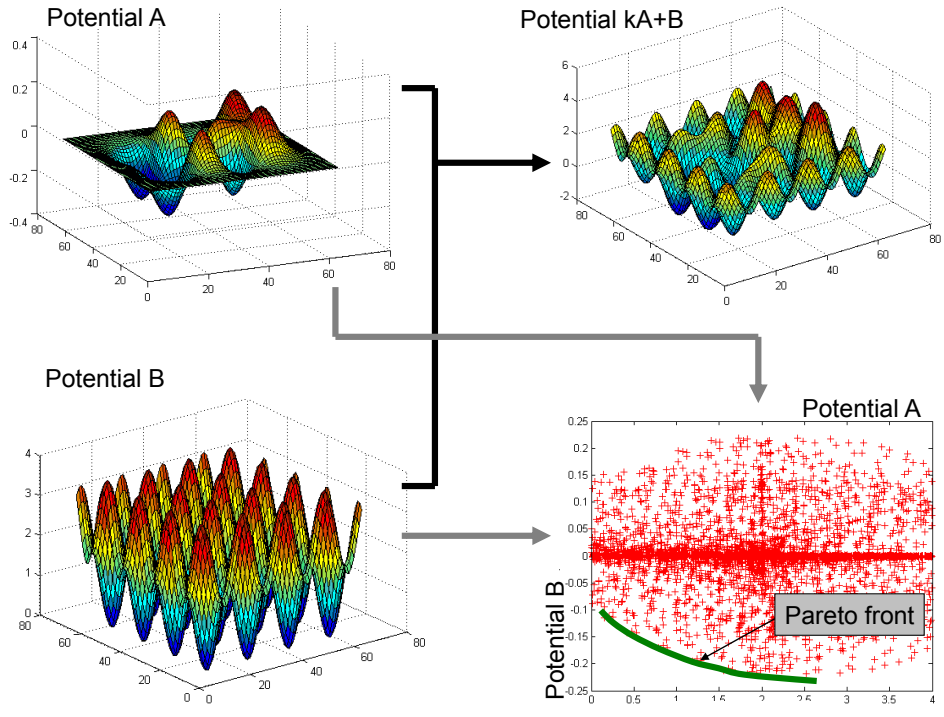


FIGURE 3.3: Comparative between mono-objective and multi-objective algorithms.

A mono-objective approach only generates one solution (or several solutions in case of a multi-modal energy function), the best one, but sometimes an expert needs more than one solution to extract information and select which solution will fit better in each situation. For instance, if we want to buy a car, we do not want only one offer, although it could be the better option. In that case we need a range of options, each one is going to have something better and something worse than other option. With that set of cars, we

can select the best one for us. This is a typical example of a multi-objective problem. Therefore, although a problem could be formulated as a mono-objective problem, it is possible to see it as a multi-objective problem to generate a rank of solutions and let the experts or the consumers decide which one could be better according to their preferences.

Some multi-objective procedures have been proposed [Coello, Lamont, and Veldhuizen, 2007] as MOGA [Fonseca and Fleming, 1993], NSGA [Srinivas and Deb, 1994], NSGA2 [Deb et al., 2002], PAES [Knowles and Corne, 1999], SPEA [Zitzler and Thiele, 1999a] and SPEA2 [Zitzler, Laumanns, and Thiele, 2001]. In this work we are going to focus NSGA2 (Non-dominated Sorted Genetic Algorithm II) and PAES (Pareto Archive Evolution Strategy) to solve a multi-objective formulation of the PSP problem.

Non-dominated Sorted Genetic Algorithm II [Deb et al., 2002] (NSGA2) is a population based algorithm that keeps an elitist set of solutions inside the population built after each iteration of his evolutionary algorithm. In that sense the algorithm can manage better the crossover and a better evolution is usually observed. Another characteristic is that this algorithm includes a crowding operator to generate a diversified set of solutions, to get the Pareto Front that covers a wide range of the solution space. NSGA2 is consider one of the best multi-objective algorithms.

Pareto Archive Evolution Strategy [Knowles and Corne, 1999]: presents a simple (1+1) strategy for multi-objective optimization instead of a population-based approach as the most part of proposed MOEAs. PAES has been well studied in the multi-objective optimization literature and has been applied

to a range of real-world problems. It has been suggested that it may outperform population-based methods for certain applications.

One of the key mechanisms in PAES is the maintenance of an archive of non-dominated solutions. The use of this archive yields a negative efficiency preserving strategy that prevents degradation of solutions. In order to maintain a limited archive size, PAES uses a crowding method based on the division of objective space into a hypergrid. Non-dominated solutions are accepted into the archive if (i) the archive is not full, (ii) they dominate another solution in the archive, or (iii) the archive is full but they occupy a grid position that is less crowded than the grid position of at least one other solution (in this case this other solution will be replaced).

PAES starts with a randomly initialized solution, which is copied into the archive of non-dominated solutions and becomes the current solution. In every iteration, the current solution is mutated and the mutant is accepted if and only if (i) it is not dominated by any other solution in the archive and (ii) it dominates the current solution or occupies a grid position that is at most a crowded as that of the current solution.

3.2 The Multi-objective Optimization Approach to PSP

In this section all the concepts required to apply a multi-objective evolutionary algorithm to the PSP problem are explained:

1. The operators and heuristics that the proposed multi-objective evolutionary algorithm requires.

2. Our implementation of the NSGA2 algorithm to the PSP problem and the proposed pseudo-code.

3. Our implementation of the PAES algorithm to the PSP problem and the proposed pseudo-code.

4. The methods we propose to select solutions from the Pareto front (Decision phase).

3.2.1 Operators and Heuristics

The main components that define an evolutionary algorithm are the initialization phase, the mutation operators and the fitness function. A wide set of optimization algorithms can be devised from adequate definitions of these operators. Therefore, in what follows we define the three components used in our multi-objective approach to the PSP problem.

3.2.1.1 Initialization

The initialization of the population in an evolutionary algorithm constitutes an important step as it can affect the quality of the obtained solution and the iterations required to get it. In [Kubalik and Lazansky, 1999; Maaranen, Miettinen, and Mäkelä, 2004; Rahnamayan, Tizhoosh, and Salama, 2007; Wang, Wu, Wang, Dong, Yu, and Chen, 2009], different approaches for

the initialization of the population are proposed. A preprocessing phase composed by several short executions with small populations to get the individuals of the initial population is considered in [Kubalik and Lazansky, 1999]. The quasi-random procedure described in [Maaranen et al., 2004] allows improvements in the solution quality although no reduction in the iterations required to converge is obtained. The opposition-based learning method described in [Rahnamayan et al., 2007] makes possible to start with populations that accelerate the convergence of the evolutionary algorithm although there is not any a priori information about the solution. In [Wang et al., 2009], the so called space transformation search (STS) strategy is reported, and it is also shown that this approach outperforms the usual random initialization and the opposition-based method [Rahnamayan et al., 2007], with respect to the quality of the found solution.

We propose the use of three initialization methods, random, probabilistic, and template-based:

Random initialization. Our random method sets each backbone torsion angle with a random value according to the constraints of the variable. The side-chain torsion angles are set by using one of the methods explained in the *Torsion Angles Statistic Libraries* section (Section 2.3).

Probabilistic initialization. Our probabilistic method uses the rotamer libraries [Dunbrack and Cohen, 1997] to set each amino acid at its most probable conformation. In these libraries there is a list of the possible backbone values. For each value, the library shows the most probable side-chain conformations, and the number of proteins in the Protein Data

Bank using those conformations. Therefore, by summarizing the number of conformation for each backbone value, it is possible to extract the most common value for the backbone torsion angles of each amino-acid. Once the backbone torsion angles are initialized, the side-chain torsion angles are set by using one of the methods explained in the *Torsion Angles Statistic Libraries* section (Section 2.3).

Template-based initialization. The template-based method gives us a way to include the structure obtained by a template-based approach, like TASSER [Zhang, 2009], in the initialization of our optimization process, thus executing in that case a refinement of that homology result. In the next Section this process is further explained as it can be used to enable an hybrid configuration of our procedure to the PSP problem, because with it we can merge template-based results and *ab initio* techniques.

3.2.1.2 Mutation operators

Other alternatives to insert knowledge in the evolutionary algorithm are the customization of the mutation operators and the inclusion of iterations of local exploitation searching within the evolutionary global exploration. For instance, the mutation and crossover operators could be designed to take into account the constraints in the data structure and to repair unfeasible solutions [Bonissone et al., 2006]. Moreover, hybrid genetic algorithms, such as memetic ones [Moscato, 1999], can be used to combine local search heuristics with crossover operators.

TABLE 3.1: Search space for each angle ϕ and ψ depending on the amino acid position in the secondary structure.

Super-secondary structure	ϕ	ψ
H (α helix)	[-75°, -55°]	[-50°, -30°]
E (β strand)	[-130°, -110°]	[110°, 130°]
undefined	[-180°, 0°]	[-180°, 180°]

TABLE 3.2: Constraints for each angle ϕ and ψ depending on the amino acid position inside the corresponding super-secondary structure.

Super-secondary structure	ϕ	ψ
a	[-150°, -30°]	[-100°, 50°]
b	[-230°, -30°]	[100°, 200°]
e	[30°, 130°]	[130°, 260°]
l	[30°, 150°]	[-60°, 90°]
t	[-160°, -50°]	[50°, 100°]
undefined	[-180°, 0°]	[-180°, 180°]

We have developed three types of mutations. All of them randomly choose an amino acid of the protein to execute their mutation. The first type of mutation is used to explore the search space. It changes the backbone torsion angles randomly taking into account the constraint of these angles defined in tables 3.1 and 3.2. This mutation operator is used intensively in the first part of the optimization process and its frequency is progressively reduced with the execution time. Precisely, expression 3.1 computes the probability to generate a new child by this operator [Cutello et al., 2006]:

$$p = e^{-2i/t}; p = 1...1/e^2 \tag{3.1}$$

In this expression:

1. p is a probability that is set to 1 in the first iteration and it is progressively decreased iteration by iteration.
2. i is the number of fitness function evaluations that has been executed until the present iteration.
3. t is the total number of fitness function evaluations our algorithm is going to execute.

The second mutation operator changes the side-chain of a selected amino acid. It uses a Gaussian distribution to generate the new torsion angles. This distribution is defined from the rotamer library [Dunbrack and Cohen, 1997] explained before. The probability of using this method is also defined by (3.1). The third method makes a little change in the backbone torsion angles of the chosen amino acid. If that mutation implies a change to the side-chain distribution, the new side-chain torsion angles are set to the new Gaussian distribution. In each mutation phase, this mutation method is executed several times. In the first generation, it is applied more frequently than in the last ones, in order to implement a local search. The number of applications of this mutation operator is defined by [Cutello et al., 2006]:

$$n = \left\lfloor \left(1 + \frac{length}{4.0}\right) e^{-2i/t} \right\rfloor \tag{3.2}$$

where:

1. n is an integer number that depends on the number of amino acids in the protein and on the generation. For instance, in our experiments, for a protein with 100 amino acids, we have obtained n=26 in the first iteration and n=3 in the last iteration.

2. i and t take the same values explained in 3.1.

3. *length* is the number of amino acids in the protein.

With these three mutation operators, the proposed procedure performs as a global searcher in the first generations and as a local searcher in the last ones.

3.2.1.3 Fitness Function

Once we know how to mutate a protein, it is necessary to evaluate the quality of each conformation in order to compare the solutions and determine their level of adaptiveness and build an evolutionary algorithm. As it has been said before, we focus our optimization approach in the minimization of the free energy of the protein, and CHARMM [Cutello et al., 2006] or AMBER [Cornell et al., 1995; Wang et al., 2000] give us that information. Therefore, in the following the objectives of the proposed multi-objective algorithms are going to be analyzed.

We use the implementation provided by TINKER [TINKER, 2004] for both force fields: CHARMM and AMBER. This implementation gives us the next energy terms: Bond Stretching (E_1), Angle Bending (E_2), Improper

Torsion (E_3), Torsional Angle (E_4), Van der Waals (E_5), Charge-Charge (E_6), and Continuum Solvation (E_7). The terms E_1 to E_4 are related to the so called bond energies that corresponds to local interactions among the atoms, and the rest evaluates the non-bond energies that account for long-distance forces.

An initial formulation of the multi-objective algorithm could be formulated in terms of two objectives, one for the bond energies and another one for the non-bond energies. Analyzing these energies, it can be seen that the Van Der Waals energy term can hide the other non-bond terms because has a highest change range (Figure 3.3). Therefore, we separate the non-bond energy into two terms, thus, having three objectives: the bond terms, the Van Der Waals term, and the other terms corresponding to the non-bond energies.

These three objectives take into account only the *ab initio* behavior, because it only takes into account energies between atoms, but nothing about previous knowledge. It should be also useful to take advantage of some information provided by template-based algorithms. Thus, it could be interesting to introduce another objective that represents this information. This way, here we propose to add an objective that compares the difference between the template-based conformation and the current conformation, thus, trying to generate optimized proteins similar to template-based one.

A multi-objective approach is usually less efficient if it has to deal with many objectives (more than four). Therefore, a good number of objectives should be two or three. As we have proposed three objectives for the *ab*

initio behavior, the inclusion of the homology objective implies to have four objectives. Thus, we have considered two formulations of the multi-objective protein structure prediction problem that use three objectives as is is described in what follows:

a) Three objectives without initial homology information:

$$\begin{aligned} f &= [f_{bond}, f_{non-bond-1}, f_{non-bond-2}] \\ f_{bond}(X) &= E_1 + E_2 + E_3 + E_4 \\ f_{non-bond-1}(X) &= E_5 \\ f_{non-bond-2}(X) &= E_6 + E_7 \end{aligned} \tag{3.3}$$

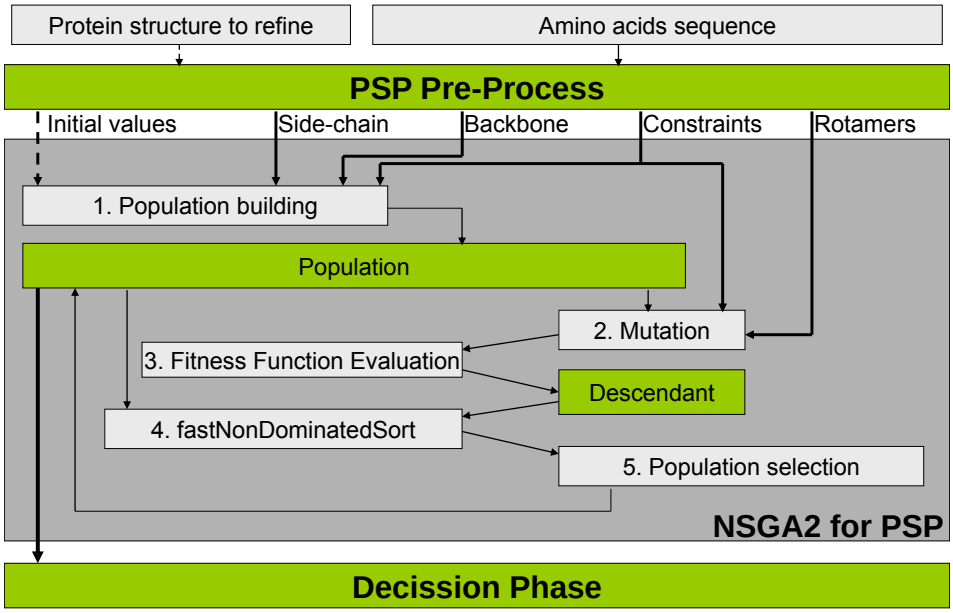
b) Three objectives with initial homology information:

$$\begin{aligned} f &= [f_{bond}, f_{non-bond}, f_{rmsd}] \\ f_{bond}(X) &= E_1 + E_2 + E_3 + E_4 \\ f_{non-bond-1}(X) &= E_5 + E_6 + E_7 \\ f_{rmsd}(X) &= RMSD(initial, X) \end{aligned} \tag{3.4}$$

Where RMSD (3.5) measures the similarity between the two protein structures initial and X.

$$RMSD(a,b) = \sqrt{\frac{\sum_{i=1}^{n} |\tau_{ai} - \tau_{bi}|^2}{n}} \tag{3.5}$$

In (3.5), a and b are the protein structures to be compared, n is the number of atoms in the protein, and τ_{ai} and τ_{bi} are the 3D positions of the atom



NSGA2FIGURE 3.4: PSP by an evolutionary algorithm based on NSGA2.

i in a and b respectively. Before using equation (3.5) it is necessary to fit both structures as much as possible.

Once we have the required operators to execute a multi-objective evolutionary algorithm, its time to describe the optimization algorithm. As it has been said before, we have considered an evolutionary strategy like PAES [Knowles and Corne, 1999] and a population based algorithm like NSGA2 [Deb et al., 2002].

3.2.2 NSGA2

NSGA2 (Non-dominated Sorting Genetic Algorithm II) [Deb et al., 2002] is a multi-objective evolutionary algorithm that is considered one of the best multi-objective algorithms. It uses a population of individuals, and thus it shows the main characteristics of these kind of algorithms such as parent selection, crossover, mutation, and replacement. In the selection phase, it takes into account the crowdedness of different zones in the population, in order to select a well distributed set of parents. This method uses a crowding factor for each individual in the population. Another specific characteristic is the way to keep an elite of individuals in the population across the replacement phases. To do that, it uses a non-dominated sorting procedure to built a non-dominated set among the population defined by parents and descendants. It adds this set of non-dominated solutions to a new population, removes these individuals from the current population, and selects the new set of non-dominated solutions, including them again in the new population, and so on, until the new population is complete. In this way, it keeps the best set of solutions across the iterations.

In order to use NSGA2 to approach the PSP problem, we have removed the crossover phase, because an useful crossover operator that take advantage of the solutions found in the previous iterations is not a trivial task in this problem. It is not easy to crossover two different conformations obtaining descendants that keep the properties of the parents. Due to the non-bond free energy, a crossover between two good individuals can result in very bad descendants. Finally, after some tests, we have discarded the crossover

phase because the results were worse than those obtained without crossover operators.

The pseudo-codes provided in Algorithm 3.2.1 and Algorithm 3.2.2 give the details of our multi-objective procedure.

Algorithm 3.2.1: NSGA2(*sequence*)

$P_0 \leftarrow initialPopulation(sequence)$
while *evaluations*
do
- $Q_t \leftarrow makeNewPop(P_t)$
- $R_t \leftarrow P_t \cup Q_t$
- $F \leftarrow fastNonDominatedSort(R_t)$
- $P_{t+1} \leftarrow \emptyset$
- $i \leftarrow 1$
- **while** $|P_{t+1}| + |F_i| \leq N$
  - **do**
    - $crowdingDistAssignment(F_i)$
    - $P_{t+1} \leftarrow P_{t+1} \cup (F_i)$
    - $i \leftarrow i + 1$
- $Sort(F_i)$
- $P_{t+1} \leftarrow P_{t+1} \cup F_i[1 : (N - |P_{t+1}|)]$
- $t \leftarrow t + 1$

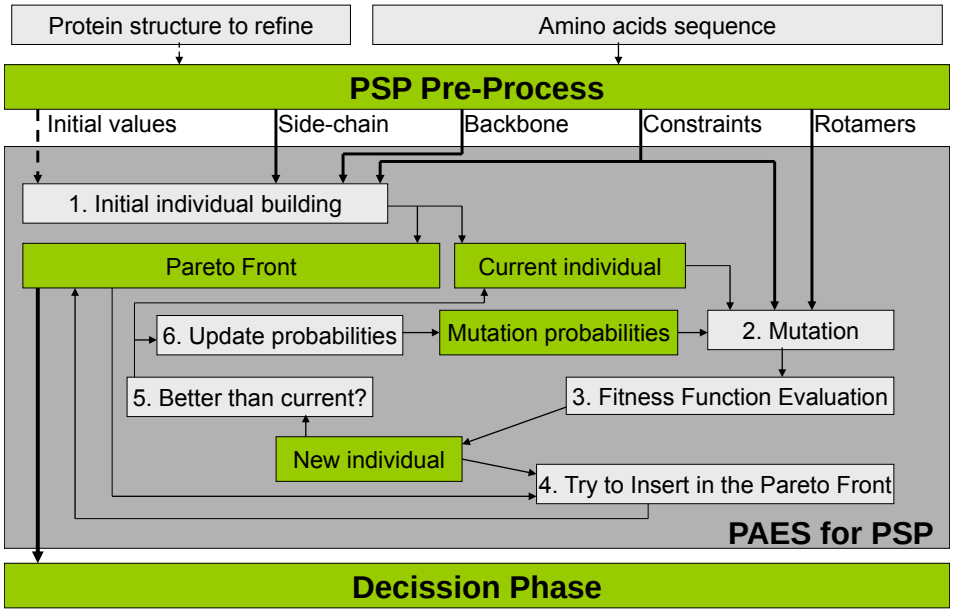
return (P_t)

Algorithm 3.2.2: MUTATION($protein$)

$amino \leftarrow randomAmino(protein)$
$newProtein \leftarrow change(protein, amino)$
return $(newProtein)$

In these Algorithms:

1. The procedure $initialPopulation$ initializes the population of the algorithm.
2. The variable $evaluations$ is the number of fitness function evaluations the algorithm still has to complete.
3. The set P_t is the population in the generation t.
4. The set Q_t is the descendence of the population in the generation t.
5. The set R_t is the union of P_t and Q_t, it is used to select the new population.
6. The procedure $makeNewPop$ makes the new population applying the crossover and the mutation operators. In our approach, we only apply the mutation operator for each protein in the population.
7. The procedure $fastNonDominatedSort$ sorts individuals in R_t by non dominated fronts. In each F_i there are individuals in the same non dominated front.
8. The procedure $crowdingDistAssigment$ calculate the crowding distance for each individual.



PAES

FIGURE 3.5: PSP by an evolutionary algorithm based on PAES. Green boxes are data structures, grey boxes are functions and numbers are the sequence of the execution.

9. The procedure *randomAmino* chooses a random amino acid of the protein.

10. The procedure *change* mutates the amino acid depending on the type of mutation we are executing.

3.2.3 PAES

PAES (Pareto Archived Evolution Strategy) [Knowles and Corne, 1999] is a multi-objective optimization algorithm based on an evolution strategy paradigm that uses only one individual. Its main characteristic is that it works with a population including only one individual. This algorithm keeps a set of non-dominated solutions just as an archive. The main phases for this algorithm are mutation and replacement. The replacement phase compares the current individual and its mutation. If the mutation is better than the current individual for all the objectives, then the new protein replaces the current one. If the mutation is worse than the current for all the objectives, then it is discarded and the algorithm continues using the current individual. Finally, if both of them are non-dominated (better in some objectives and worse in the others), a crowding method is applied to select the individual in the less crowded area [Knowles and Corne, 1999].

Every new protein generated by the mutation operator is checked to determine if it can be inserted in the archive of non-dominated solutions. If the protein structure is inserted in the archive, the algorithm has to compare every protein in this set of non-dominated solutions with the current solution to check if the new protein is better in every objective. If that happens, the structure in the archive that has been compared is deleted from the archive of non-dominated solutions. This way, the proteins stored in the archive are always in a set of non-dominated solutions.

The PAES algorithm repeats this behavior until it achieves the maximum number of fitness function evaluations that has been established. Finally,

the archive is returned as the found approximation to the Pareto Front.

The pseudo-codes provided in Algorithm 3.2.3 and Algorithm 3.2.4 give the details of our multi-objective procedure.

Algorithm 3.2.3: PAES(*sequence*)

$current \leftarrow initialSolution(sequence)$
$pareto \leftarrow [current]$
while $evaluations > 0$
do
- **if** $simplifiedSearchSpace()$
  - **then** $new \leftarrow mutateSimplified(current)$
  - **else**
    - **if** $mutate1()$
      - **then** $new1 \leftarrow mutation1(current)$
    - **if** $mutate2()$
      - **then** $new2 \leftarrow mutation2(current)$
    - $new3 \leftarrow current$
    - **for** $i = 0 \cdots mutate3()$
      - **do** $new3 \leftarrow mutation3(new3)$
    - $new \leftarrow best(new1, new2, new3)$
- $tryInsert(pareto, new)$
- **if** $new == best(new, current)$
  - **then**
    - $current \leftarrow new$
    - $increaseMutationProbability()$
  - **else if** $!dominate(current, new)$
  - **then** $current = lessCrawded(pareto, new, current)$
- $evaluations \leftarrow evaluations - 1$

return $(pareto)$

Algorithm 3.2.4: MUTATIONX($protein$)

$amino \leftarrow randomAmino(protein, probabilities)$
$decreaseProbability(amino)$
$newProtein \leftarrow changeX(protein, amino)$
$decreaseProbabilities()$
return $(newProtein)$

In Algorithms 3.2.3 and 3.2.4:

1. The procedure *initialSolution* either executes a template based algorithm to get a protein conformation like that provided by TASSER [Wu et al., 2007; Zhang, 2009], or executes a probabilistic method to build the initial solution, or executes a random procedure to get a first solution.

2. The procedure $simplifiedSearchSpace$ determines whether or not the algorithm is using a simplified search space. It is determined according to the number of iterations and the percentage of time the algorithm has to be run in that simplified search space. This technique is explained at the end of this chapter.

3. The procedure $mutateSimplified$ performs a mutation in a simplified search space. This technique is also explained at the end of this chapter.

4. The procedure $mutate1$ and $mutate2$ use Equation (3.1).

5. The procedure $mutate3$ executes (3.2) to determine the third mutation that has to be executed.

6. The procedures *mutation*1, *mutation*2, and *mutation*3 execute the three types of mutations implemented. Each one has the structure of the procedure $MUTATIONX(protein)$, where X can be 1, 2 or 3.

7. The procedure *best* returns the best protein among the given ones.

8. The procedure *tryInsert* tries to put the protein in the archive of non-dominated solutions.

9. The procedure *increaseMutationProbability* increases, by using the expression (3.8), the probability of the mutated amino acid to be chosen in the following mutations.

10. The procedure *dominate* determines if one protein is better than the other.

11. The procedure *lessCrowded* selects the protein in a less crowded area of the archive of non-dominated solutions.

12. The procedure *randomAmino* chooses a random amino acid of the protein taking into account the mutation probabilities assigned to each amino acid.

13. The procedure *changeX* mutates the amino acid according to the type of mutation we have selected, where X can be 1, 2 or 3.

14. The procedure *decreaseProbability* decreases the mutation probability of the chosen amino acid according to the expression (3.7).

15. The procedure *decreaseProbabilities* decreases the mutation probabilities of every amino acid according to the expression (3.6).

3.2.4 Pareto Classification and Knowledge Extraction

Once the multi-objective algorithm is executed, we get a set of feasible structures, that approaches the Pareto Front of our multi-objective problem. As we are looking for the specific structure of the target protein, the solution to our problem is not the set of structures given by the Pareto Front, but only one of them. Probably it could be interesting to provide a little set of structures to the experts, but in any case we have to select one or at most a few structures form the obtained Pareto Front. There are some alternatives to select a few representative solutions from a Pareto Front in the PSP problem. Among them, we can:

1. Select the structure with the minimum free energy.
2. Use a method that selects those solution which have a big increment of one objective and a little decrement in the others.
3. Classify the structures in the Pareto Front to extract knowledge from them. Therefore, it is possible to select the most important structures of the Pareto Front acording to the PSP problem. SPICKER [Zhang and Skolnick, 2004] implements a procedure following this idea.

Experimentally, to select the structure with the minimum free energy does not work fine as it can be seen in [Cutello et al., 2006]. The knife method is a general method for multi-objective algorithms. Therefore, it does not manage specific information corresponding to this problem.

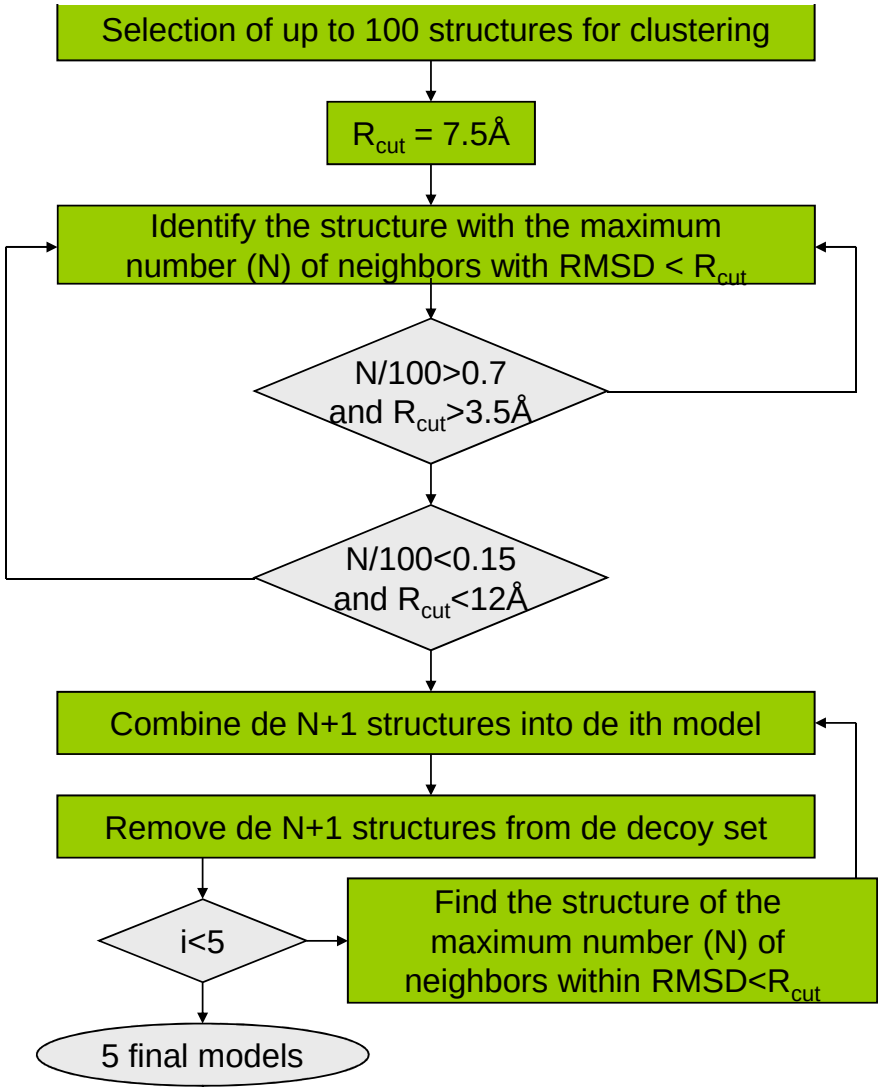


FIGURE 3.6: Flow chart of the SPICKER clustering algorithm [Zhang and Skolnick, 2004].

This way, we have choose SPICKER [Zhang and Skolnick, 2004] for our procedure PITAGORAS-PSP, because this method is used in one of the best approaches to PSP proposed at the moment and, as it manages specific information about the structures, we consider it could provide better approximations to this problem than general methods.

The SPICKER algorithm classifies all the structures into groups, then selects the group with the high number of structures and combines all the structures into one model. Then, it removes all these structures from the initial set, selects again the group with the high number of structures and repeats the same process until five models are generated. This way, by using this software it is possible to select a little set of structures from the found approximation to Pareto Front. The Figure 3.6 shows the whole process of SPICKER and more information can be reached in [Zhang and Skolnick, 2004]

3.3 A new hybrid approach for PSP problem

To be competitive, present *ab initio* methods should include strategies to start from good enough solutions or solutions that aid in the searching process. For instance, as small proteins can be predicted easier than large ones and taking into account that the conformation space grows exponentially with the number of amino acids, many procedures [Zhang, 2009; Roy et al., 2010] divide the proteins into a number of fragments that are predicted separately by searching into fragment structure libraries. Then, the fragments

are assembled through different alternatives that are sampled by a searching or optimization procedure.The hybrid scheme here proposed, as it is based in an evolutionary procedure that requires a population of solutions, uses different strategies to determine the initial population as explained before. Moreover, we also propose some new optimization techniques to improve the prediction quality or/and the computation requirements.

3.3.1 Hybridizing

This approach can be seen as a hybrid algorithm because, as it is shown in Figure 3.7, it uses not only secondary and super-secondary structure prediction, but also a template-based algorithm for PSP. Thus, it mixes *ab initio* and template-based approaches to reach an efficient procedure that takes advantage of the exploration/exploitation characteristics of evolutionary algorithms and includes the knowledge extracted by template-based procedures. This goal can be reached nowadays thanks to the availability of servers that provide this knowledge and can be accessed by Internet. Therefore, our procedure can manage information of homologous protein structures and *ab initio* techniques for this problem. This capability allows our approach to get good results whenever homologous proteins could be found in the Protein Data Bank, but it could also produce good results if such information is not available because it presents the characteristics of an *ab initio* procedure. Moreover, this approach can be also used to refine protein structure predictions generated by other algorithms as it can perform an optimization of the structures using the multi-objective evolutionary algorithm that constitutes its core, along with all the information

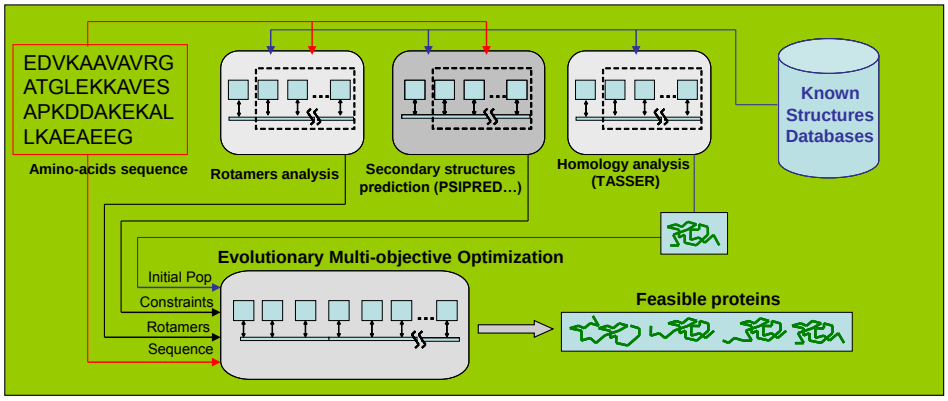


FIGURE 3.7: A hybrid approach to the PSP problem.

it can obtain from the sequence of amino acids from other servers around the Web.

As it is shown in Figure 3.7, it is possible to execute more complex procedures to take into account the information extracted from previously known structures. For example, among the best current approaches for PSP are TASSER [Wu et al., 2007] and ROSETTA [Rohl, Strauss, Misura, and Baker, 2004]. TASSER starts with a template identification process by iterative threading through the program PROSPECTOR_3 [Wu et al., 2007], which is able to identify homologous and analogous templates. Then, the configuration is divided into continuous aligned fragments with more than five residues, and a Monte Carlo sampling procedure is applied to generate different assemblies of these protein fragments. Finally, the clustering program SPICKER is applied for model selection. ROSETTA also combines small fragments of residues (obtained from known proteins) by a

Monte Carlo strategy. This way, in these procedures some kind of template-modeling is firstly applied before a random exploration of the conformation space spanned by different combining alternatives. The solutions provided by these procedures could be included in the initial population of an evolutionary optimization procedure to help in the search process as those solutions encapsulate the information about known structures.

Once an initial protein is obtained by these methods, we need to obtain the values of the torsion angles in order to include them in our initial population. The next subsection describe the method to obtain those torsion angles by extracting the highest part of the available information as possible.

3.3.2 Additional Useful Optimization Techniques

This work proposes the use of two new techniques to improve the performance of the applied evolutionary algorithm. These techniques are focused on *simplified search space* and an *amino acid mutation probability* as it is described in what follows.

3.3.2.1 Simplified search space

In the first part of the EA (for example, the first 10% of the total number of evaluations), the search space used is a simplification of the real one. This search space consists in only one variable, with only four possible values, per amino acid. This way, the EA can travel across the whole simplified

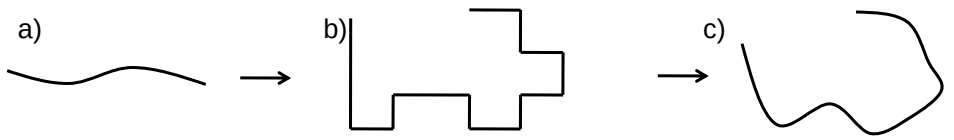


FIGURE 3.8: Protein structure (a) at start (b) after the simplified search space period (c) at end.

search space. After this period, the search space becomes the real one (Fig. 3.8).

3.3.2.2 Amino acid mutation probability

A new procedure to manage the mutation probabilities has been also included. It takes into account that bond energies are independent from the location of the corresponding amino acid, whereas the non-bonded energies depend on the present shape and structure. Taking that information into account, a mutation in one amino acid could affect the bond energy in the same way, independently from the location of the amino acid in the sequence. But the non-bonded energy could be affected in a very different way depending on the location of the amino acid in the sequence and the present structure, as it can be seen in Figure 5.18 (see the arrows). Therefore, an amino acid mutation could determine a very different energy change depending on the present structure of the protein. The method here proposed tries to set highest probabilities of mutation to those amino acids that play an important role in the present structure. To do so, each amino acid has a mutation quality factor that determines its probability to be

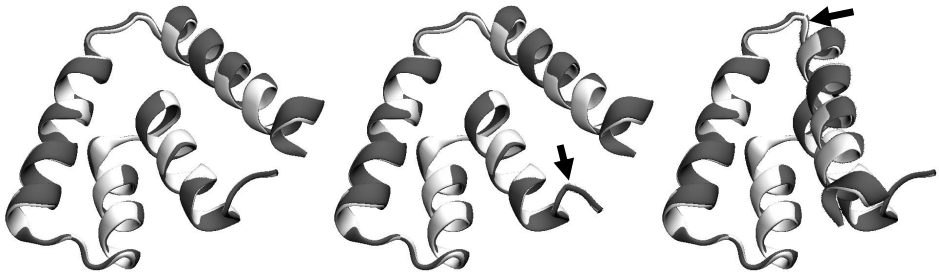


FIGURE 3.9: (left) Current protein. (center) Mutating an amino-acid in one extreme of the sequence. (right) Mutating an amino-acid of the middle of the sequence.

selected for a mutation (and this probability increases with that mutation quality factor).

In each step, all the mutation quality factors are decreased, and after some amount of time, every amino acid have the same probability to be selected. Moreover, in each mutation, depending on the quality of the generated protein, the mutation quality factor for the mutated amino acid can be increased or decreased. With that method, good mutations tend to be repeated to optimize, bad mutations tend to be avoided, and mutations with little effect tend to be avoided until the end of the algorithm. Equation (3.6) shows how to decrease the amino acid probability for all the amino acids, (3.7) represent the decreasing method for the selected protein, and (3.8) describes the probability change according to the quality of the mutated protein, iteration by iteration:

$$prob_{iter} = \frac{prob_{iter-1}}{global} \tag{3.6}$$

$$prob_{iter} = \frac{prob_{iter-1}}{selected} \tag{3.7}$$

$$prob_{iter} = prob_{iter-1} + energyDifference \tag{3.8}$$

As it can be seen in (3.8), as the normalized difference between the energies of new and old protein structures ($energyDifference$) is added to the mutation probability, better mutations will have higher selection probabilities.

3.4 Structure of the proposed hybrid approach to PSP

In this section we describe the software modules that define our framework for approaching the PSP problem. These modules are devoted to the different phases of the PSP, including the pre-processing phase, the torsion angles optimization, the evolutionary algorithms for PSP, and the decision phase. All the information that has been summarized in Figures 3.10 and 3.11 corresponds to the descriptions and explanations given in previous chapters.

The schemes given in Figures 3.10 and 3.11 include all the steps involved in the process, that starts from a given sequence of amino acids that defines the protein and finished with the predicted 3D structures for that protein. From those descriptions it is possible to realize the complexity of the system developed in this research.

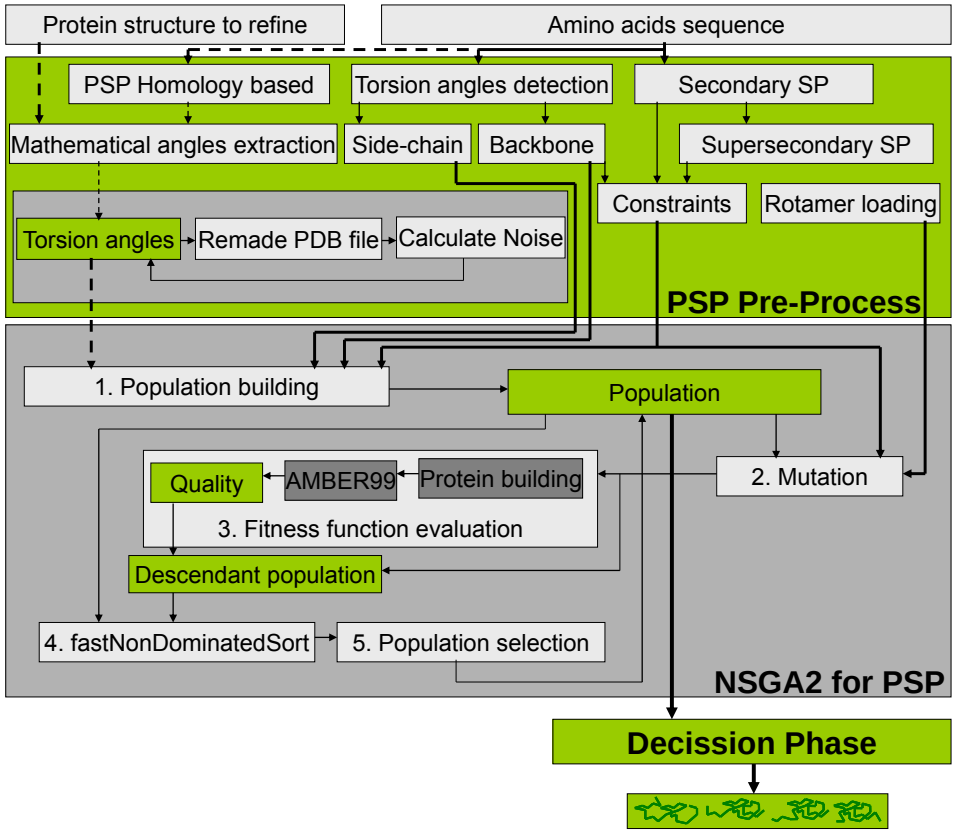


FIGURE 3.10: Sequential scheme of PITAGORAS-PSP based on NSGA2.

In the Figure 4.5 it is shown the whole protein structure prediction process proposed in this work based in the NSGA2 approach. As it has been said before, the sequence amino acids is the input of the overall procedure. There are three main phases in the process: pre-process, optimization approach, and decision phase.

Pre-process. In this phase, as it has been described in Chapter 2, the backbone and side-chain variables are extracted, the secondary structure prediction is obtained, and the rotamers library is loaded. Moreover, an homology-based procedure is executed in order to obtain an initial conformation. All this information is the input of the optimization phase.

NSGA2 for PSP. This is the main phase of the multi-objective approach to PSP. As it has been explained in this Chapter, a NSGA2 procedure has been developed to find a set of non-dominated structures as near as possible to the corresponding Pareto Front.

Decision phase. The last phase tries to obtain the best conformations of those returned by the NSGA2 multi-objective approach. In Chapter 3 the decision phase has been fully explained.

As it can be appreciated in the figure, the global complexity of the proposed approach to solve the PSP problem is very high. It requires a lot of procedures, steps, and middle results, and the final result of the overall process depend not only in the quantity and quality of the external information used along the process, but also in the heuristics and optimization techniques here proposed.

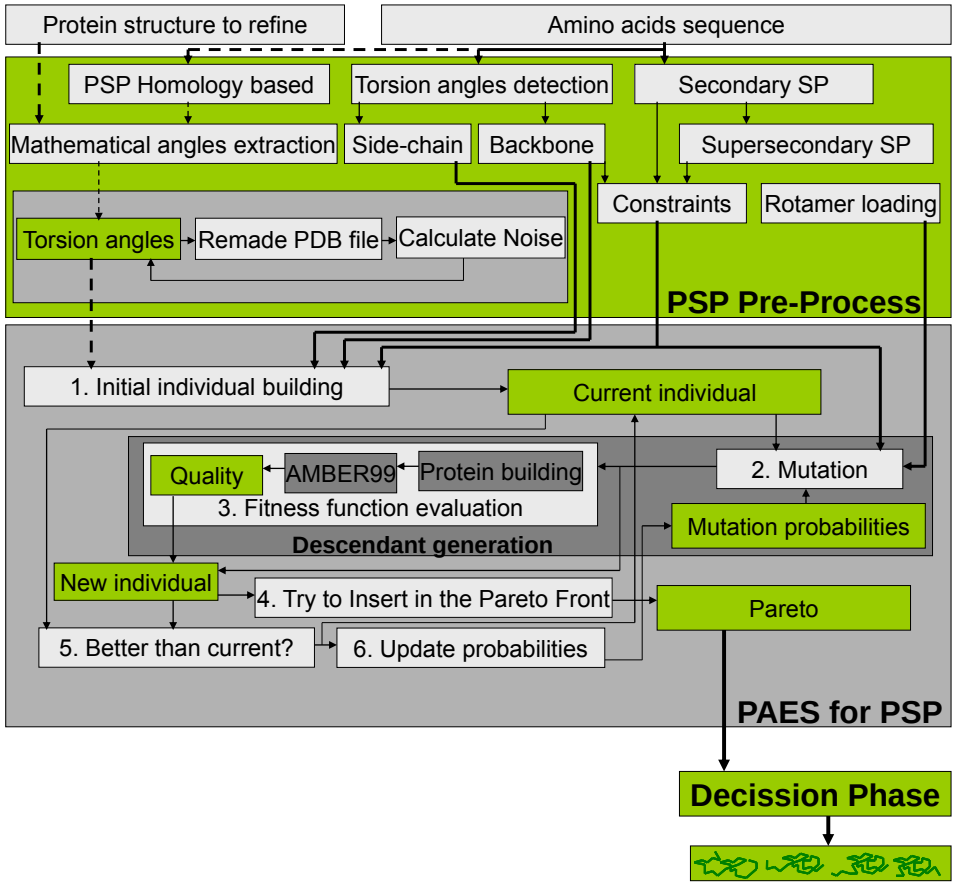


FIGURE 3.11: Sequential scheme of PITAGORAS-PSP based on PAES.

In the Figure 4.9 it is shown the whole protein structure prediction process proposed in this work based in the PAES approach. This scheme has the same structure as the previous one, the only difference is the optimization process, based on the PAES approach in this case.

3.5 Conclusion

Our implementations of PSP procedures based on the multi-objective algorithms NSGA2 and PAES have been presented in this chapter. We also have described some heuristics to improve our initial PSP approaches and a new method to extract torsion angles from a protein based on an optimization method that outperforms the previous method.

Publications with contributions of this chapter:

1. Neurocomputing 2011: *PITAGORAS-PSP: Including domain knowledge in a multi-objective approach for protein structure prediction* [Calvo, Ortega, and Anguita, 2011b]

2. The Journal of Supercomputing 2011: *Comparative of parallel Multi-objective approaches to protein structure prediction* [Calvo, Ortega, and Anguita, 2011a]

3. 4th International Workshop on Practical Applications of Computational Biology & Bioinformatics (IWPACBB 2010): *A Hybrid Scheme to Solve the Protein Structure Prediction Problem* [Calvo, Ortega, and Anguita, 2010a]

4. VII Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados: *Aproximación híbrida paralela para la predicción de estructuras de proteínas* [Calvo, Ortega, and Anguita, 2010b]

Chapter 4

A Speculative Parallel PAES and other Parallel Implementations

The computing time required to predict the structure of a single protein by using a mono-processor computer is very high, it can take several hours or days in a computer with a 1.86 GHz processor and 4 GB of main memory. Therefore, PSP is a clear example of application that requires high performance computing. In this chapter we consider the use of parallel programming to reduce the time to get an acceptable protein 3D structure for the target sequence of amino acids.

Analyzing the computation requirements of our approach to the PSP problem, we have taken into account that it has a hard phase that corresponds to the computation of the fitness function, the protein building from its torsion angles, and the evaluation of the conformation free energy, for all the individuals in the population of the evolutionary algorithm. Nevertheless, the computation of the fitness function for a given individual in the population is independent with respect to the computation of the other fitness function evaluations. As this phase takes around a 90% of the computing time, it is possible to take good speedups by distributing the fitness function evaluations of the population among the processors of a multi-processors computer.

As each fitness function evaluation requires more than half second in a machine with a 1.86 GHz processors and 4 GB of main memory, a population of a hundred individuals could need around one minute of processor. Therefore, distributing this workload among several processors could reduce this time significantly without requiring a high volume of communications. Moreover, the evaluation of each fitness function only requires the torsion angles, and it does not depend on the computation of other fitness functions. Thus, no communications are required if each processor computes the fitness of a subset of individuals in the population.

We have built parallel versions of the two evolutionary alternatives, NSGA2 and PAES, described before to implement our multi-objective optimization procedure for PSP.

4.1 Master-Worker Scheme for NSGA2

As it has been said before, the function fitness evaluation is the hardest phase of the algorithm. After some experiments, we have observed that around a 90% of the computing time is devoted to this part of the procedure. The more individuals we have in the population, the more percentage of computing time is required by this phase. Therefore, the fitness evaluation can be considered the bottleneck of the procedure and it is a good decision to distribute this work among the different processors as it suppose almost all the work to do. Moreover, taking into account that the fitness function evaluation of each protein conformation is independent for the other solutions in the population, the idea of a *master-worker* scheme seems to be interesting.

The Master-Worker scheme uses one processor as the master, and the other processors are workers. A worker receives a task from the master, completes that task, and returns the result back to the master, waiting for a new task. The master processor executes the whole process and has to send tasks to the workers and to receive the results. Therefore, the master manages all the parallel system, and the workers follow orders from the master. In Figure 4.1 it is shown an scheme to illustrate the way we have distributed the workload of our procedure.

We propose three *master-worker* schemes that are described in what follows:

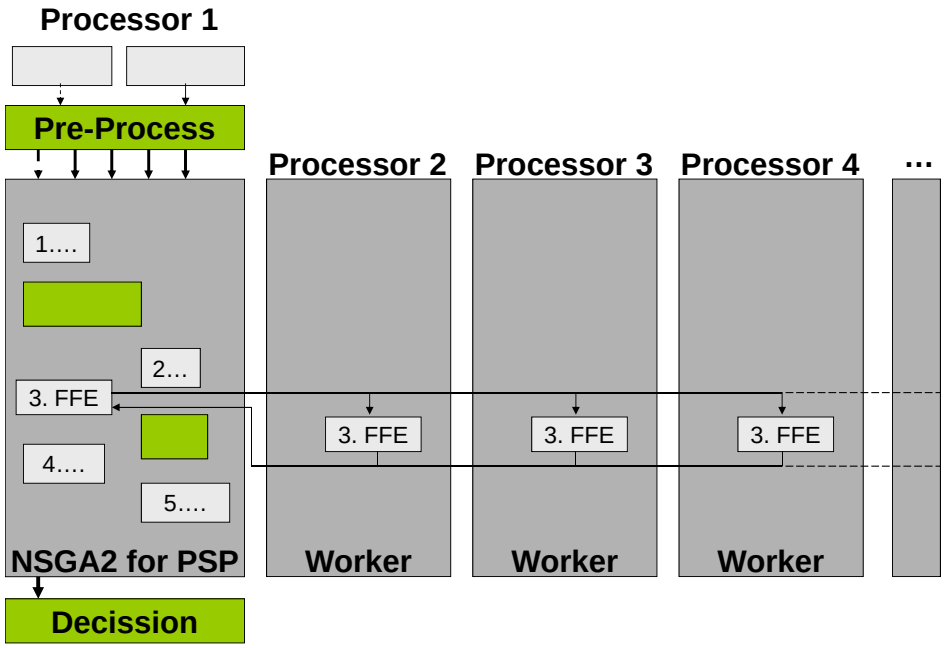


FIGURE 4.1: Parallel scheme to distribute the Function Fitness Evaluation. The *Processor 1* executes the multi-objective procedure described in the Figure 3.4, but it distributes the Fitness Function Evaluation (FFE) among the other Processors, being these processors the workers.

4.1.1 The Master-Worker-1 Scheme

In the *master-worker-1* scheme, the master distributes the individuals of the population among the workers by using a *Round Robin* scheme. Thus, the fitness function evaluations are also distributed among all the workers. This way, if we have n workers, n evaluations of the fitness function are distributed among them. The master waits until the first worker returns its work, once the job is done, the master obtains the fitness of the protein

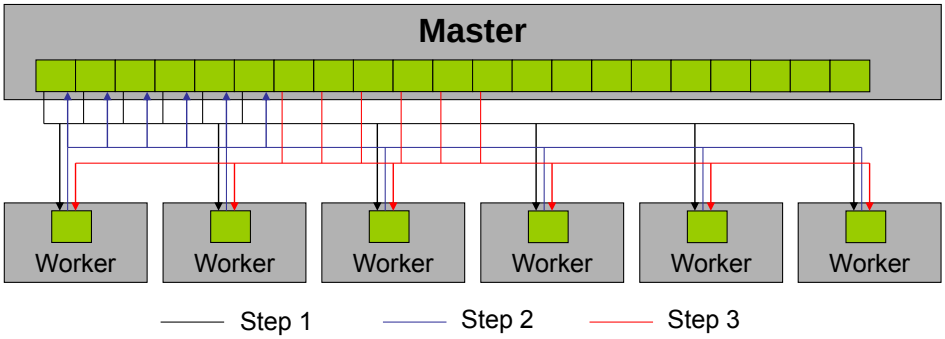


FIGURE 4.2: Load distribution scheme for the master-worker-1 parallel approach. The master sends one conformation to each worker, then receives the answer from each worker, and send a new conformation to each one. This process is repeated until all the conformations are evaluated. This distribution scheme requires 2 messages per each conformation.

conformation given to this worker and gives another conformation to it. Then the master waits for the next worker that finishes its evaluation. As each evaluation of the fitness function requires more or less the same amount of time, this parallel scheme distributes n tasks, then it collects the results after waiting for a while. Then, it distributes another n tasks among the workers. In Figure 4.2 the whole process is represented.

4.1.2 The Master-Worker-2 Scheme

In the *master-worker-2* scheme, the master distributes the evaluations of the fitness function among all the workers by using a block distribution. This way, having n workers and m conformations to evaluate (individuals in the population), all the evaluations of the fitness function are distributed

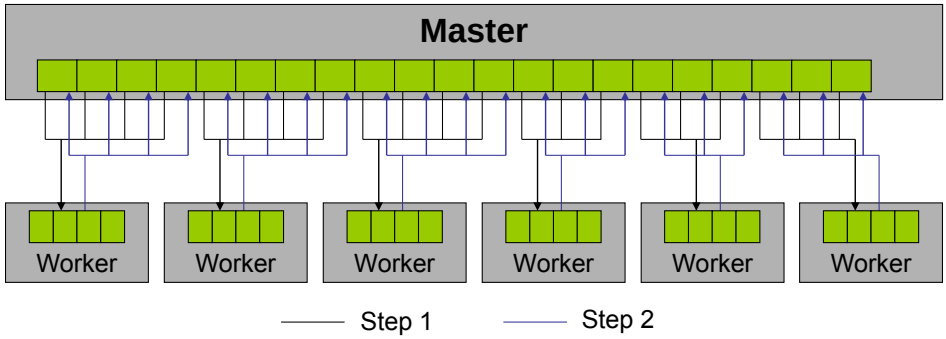


FIGURE 4.3: Load distribution scheme for the master-worker-2 parallel approach. The master sends a set of conformations to each worker, distributing all the conformations. Then receives the answer from each worker. This process is not repeated as every conformation has been distributed in the first step. This distribution scheme requires 2 messages per each worker.

among n workers by using the equation (4.1) to compute the number of contiguous structures the master has to send to each worker in the parallel system:

$$Load_i = min\left(\lceil m/n \rceil, m - min(m, i\lceil m/n \rceil)\right); i = 0..n-1 \tag{4.1}$$

where $Load_i$ is the number of individuals that the master sends to worker i^{th}.

With this distribution, for instance, having 100 structures and 10 workers, each worker receives 10 structures in one message. Using 9 workers, the 8 first processors receive 12 structures and the last worker receives 4 structures. Finally, if we use 18 processors, 6 structures are given for each of

the 16 first processors, 4 structures to the 17th processor, and the last one, does not receive any task.

The formula (4.1) optimizes the communication among master and workers, because $\lceil m/n \rceil$ defines the peak load in at least one processor, thus the maximum computing time required. Once we have a maximum processor time requirement, we have to reduce the communications time, and this is done by reducing the processors without decreasing the maximum processor time. Thus with formula (4.1), it is possible to reduce the number of processors, and the messages required to send and receive their tasks and results, without incrementing the processor time requirements. The improvement achieved by this distribution is more important whenever each task depends on others tasks. In that situations, there are more messages between processors, and the less processors we have, the less communications we need. Therefore, once the maximum computing time is defined, we have to minimize the communication time as much as we can, without increasing the maximum computing time defined before.

Once the load is distributed among the processors, the master waits until the first worker returns its results and move to the next worker. The whole process is shown in Figure 4.3.

4.1.3 The Master-Worker-3 Scheme

This scheme *master-worker-3* is similar to *master-worker-2* but the master is also used as a worker. This approach has been considered because as there is a lot of time between the workload distribution and the reception

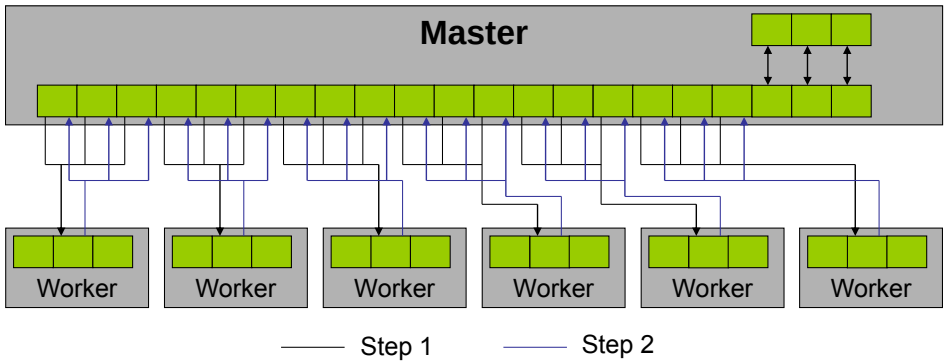


FIGURE 4.4: Load distribution scheme for the master-worker-3 parallel approach. The master sends a set of conformations to each worker, distributing all the conformations and the master process a set of conformation by itself. Then receives the answer from each worker. This process is not repeated as every conformation has been distributed in the first step. This distribution scheme requires 2 messages per each worker. As the master process a set of conformation, the total amount of work for each worker is lower, therefore, it is faster.

of the results, the master can be used to compute some fitness functions. Figure 4.4 corresponds to this alternative.

4.1.4 PITAGORAS-PSP based on NSGA2

In the Figure 4.5 it is shown the whole protein structure prediction process proposed in this work based in the NSGA2 approach. As it has been said before, the sequence amino acids is the input of the overall procedure. There are three main phases in the process: pre-process, optimization approach, and decision phase.

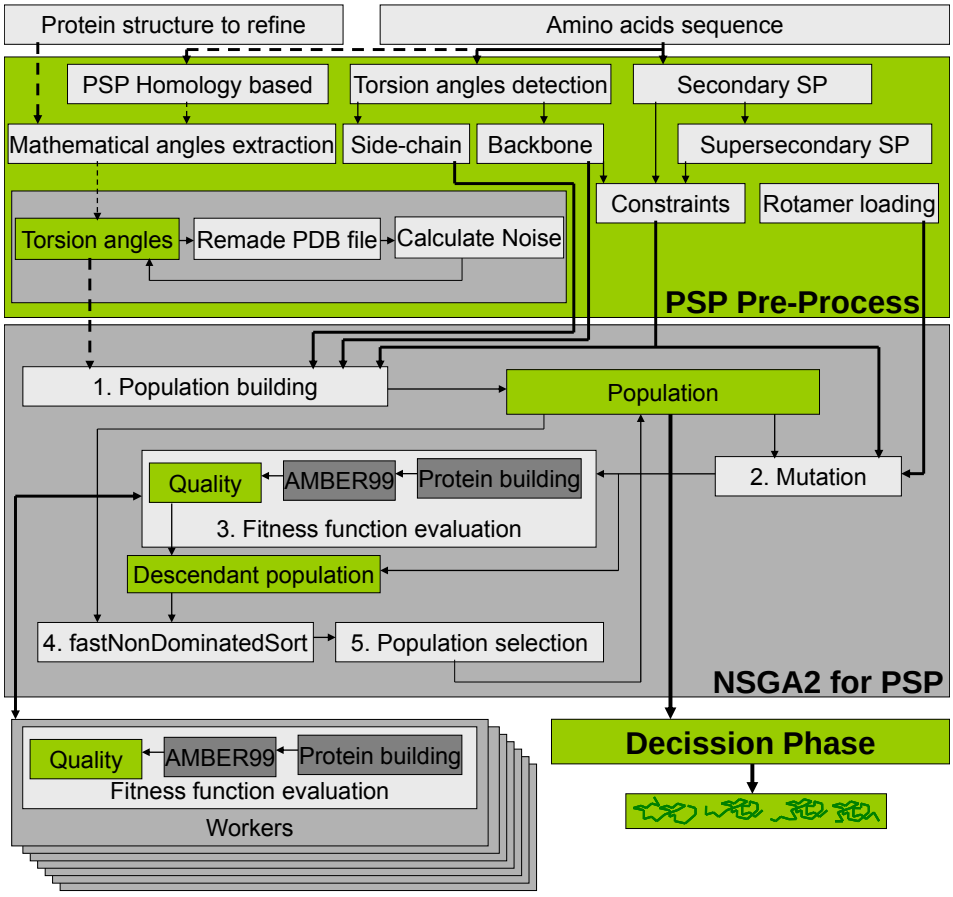


FIGURE 4.5: Global scheme of PITAGORAS-PSP based on NSGA2.

Pre-process. In this phase, as it has been described in Chapter 2, the backbone and side-chain variables are extracted, the secondary structure prediction is obtained, and the rotamers library is loaded. Moreover, an homology-based procedure is executed in order to obtain an initial conformation. All this information is the input of the optimization phase.

NSGA2 for PSP. This is the main phase of the multi-objective approach to PSP. As it has been explained in Chapter 3, a parallel implementation of NSGA2 has been developed to find a set of non-dominated structures as near as possible to the corresponding Pareto Front.

Decision phase. The last phase tries to obtain the best conformations of those returned by the NSGA2 multi-objective approach. In Chapter 3 the decision phase has been fully explained.

As it can be appreciated in the figure, the global complexity of the proposed approach to solve the PSP problem is very high. It requires a lot of procedures, steps, and middle results, and the final result of the overall process depend not only in the quantity and quality of the external information used along the process, but also in the heuristics and optimization techniques here proposed.

4.2 A New Parallel Implementation of PAES

As it has been said before, one of the multi-objective evolutionary algorithms we have implemented is based on PAES [Knowles and Corne, 1999].

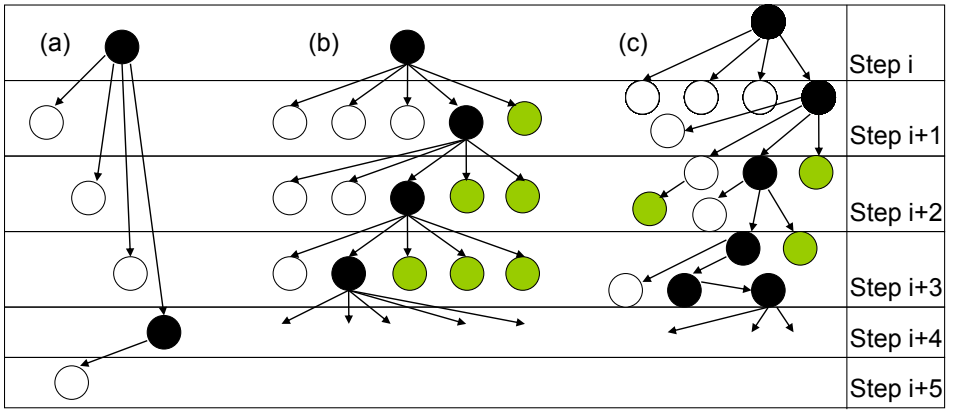


FIGURE 4.6: There are five processors available in each step. The black nodes represent the solutions selected as new parents and the green nodes correspond to wasted work: (a) Sequential PAES (b) Naive Parallel PAES (N-PAES) (c) Speculative Parallel PAES by Adaptive Computation (SP-PAES).

In each iteration, PAES generates a single child and decides whether to select this child or to keep the parent as the current solution. In what follows, a naive parallelization scheme for PAES that preserves the behavior of the sequential PAES is described (N-PAES).

Given a current solution, PAES will frequently need to generate a number of offspring solutions before an acceptable offspring is found, that replaces the current solution (i.e. before we have a change in the generation). Hence, if we have a number of n processors available, we can generate n offsprings and use these processors to simultaneously (in a single time step) generate and evaluate an ordered set of n prospective offsprings for the current solution. The master node then scans the fitness values of all n offsprings in order,

and accepts the first one of these that fulfills the PAES acceptance criterion. In this way, the original PAES strategy is maintained. It is evident that the efficiency of this parallel scheme may vary strongly depending on the number of children generated (i.e. the number of processors available) and the difficulty of the optimization task. When the searching is very easy (e.g. at the beginning of the optimization process) or when a large number of processors are available, the parallel strategy is likely to "waste" a large number of evaluations. Instead, when it is difficult to find a better solution, the strategy is very efficient. In Figure 4.6 (a) we show the difference between a sequential algorithm and a parallel algorithm with the same behavior

In this dissertation we have developed a more elaborated parallelization scheme, which attempts to minimize the number of "wasted" evaluations by limiting the number of offsprings that are generated simultaneously for a given current solution. The discrepancy between the number of offspring generated and the number of processors available can then be used to generate and evaluate the next offspring generations, in an effort to maximize the number of total iterations covered in a single time step.

In each iteration, the algorithm has to take a decision between the parent node and the new node, hence there is two nodes that have to be considered in each decision. Given a parallel time step, we have many decisions to take in this time, and it could happen that one parent node takes part in few decisions. We have to create a new view of the PAES tree to separately represent each decision on the evolution process. In this way, we can allocate the resources in a tree. In Figure 4.6 (c) we show the new prediction tree

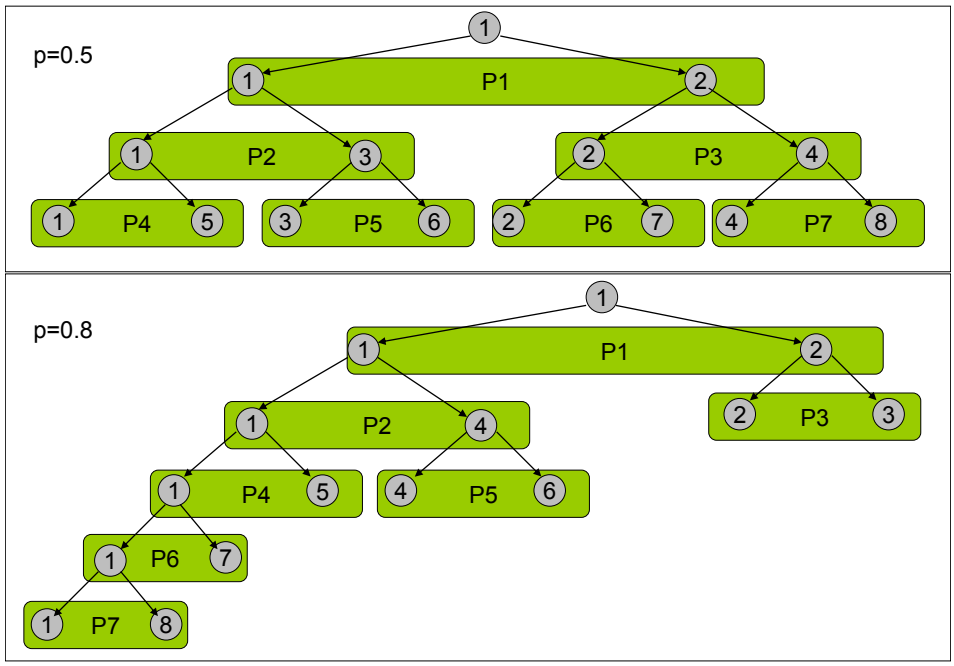


FIGURE 4.7: Prediction trees for p=0.5 (a) and p=0.8 (b). The node number is the number of solution generated and the nodes are distributed among seven processors in this case. Each processor has to generate and evaluate the new node, and select between both nodes.

versus the naive one. In the prediction tree, we copy the parent node to the left child, thus the right child is the real child of the parent. Therefore, initially, in Figure 4.6, the comparison is done between the parent and the child, but in the prediction tree, in Figure 4.7, the comparison involves the two children. Anyway, the comparison is the same in both representations, in the sense that both of them compares the current solution with the mutated one, but in the initial representation the current solution is the parent

and the mutated one is the child, and in the prediction tree, the current solution and the mutated one are both children.

Assuming a fixed probability, for instance $p = 0.5$, of generating a favorable mutation, we can optimally distribute the processors available based on a static evaluation tree, as illustrated at top in Figure 4.7). However, in a realistic optimization scenario, the probability p will be different to 0.5 and it could change with time, resulting in different shapes of the optimal evaluation tree (bottom in Figure 4.7).

As we show in Figure 4.6 (b) and (c), the prediction tree approach could perform better than the naive scheme. It will depend on the quality of the prediction factor.

Analyzing the behavior of the tree parallelization scheme and the naive one, we can see that the naive is going to work fine if the behavior of the problem is keeping in the parent node the mayor part of the time. In that case, we could use a prediction factor $p = 0$, and SP-PAES would work equal than N-PAES. Nevertheless, in the most frequent cases SP-PAES would behalf better than N-PAES.

Algorithm 4.2.1: PARALLELPAES(c)

$urrent \leftarrow initialSolution()$
$procs \leftarrow numberOfProcessors()$
while *evaluations*
do $\begin{cases} p \leftarrow getPrediction() \\ current \leftarrow timeStep(current, procs, p) \end{cases}$
return ($current$)

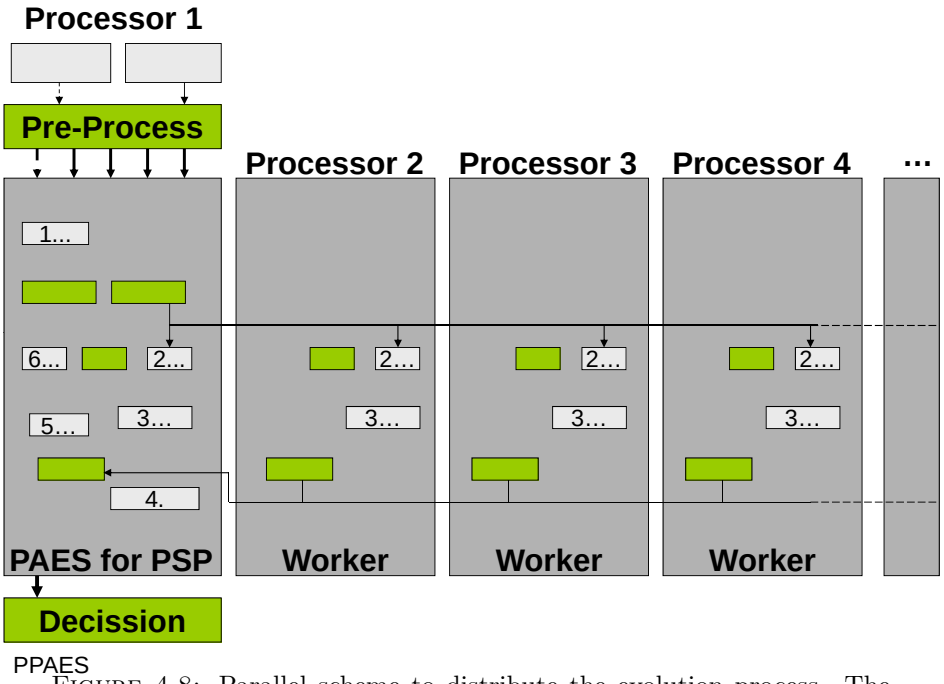


FIGURE 4.8: Parallel scheme to distribute the evolution process. The *Processor 1* executes the multi-objective procedure described in the Figure 3.5, and the other Processors are the workers.

In the Algorithm 4.2.1 is described the main characteristics of SP-PAES as follows:

1. The procedure *numberOfProcessors* returns the number of processors available in the parallel machine.

2. The procedure *getPrediction* returns the adaptive prediction factor in the present branch of the tree. This method has to take into account

the history of the algorithm to calculate what will happen in the following steps. The parameter p defines the prediction tree shape.

3. The procedure *initialSolution* has been explained in Section 3.2.1.1.

4. The procedures *mutate* and *best* have been explained in the sequential PAES algorithm.

Algorithm 4.2.2: TIMESTEP($current, procs, p$)

$child \leftarrow mutate(current)$
if $procs > 1$
  then $\begin{cases} child\_desc \leftarrow timeStep(child, (procs-1) * p) \\ current\_desc \leftarrow timeStep(child, (procs-1) * (1-p)) \end{cases}$
  else $\begin{cases} child\_desc \leftarrow child \\ current\_desc \leftarrow current \end{cases}$
$selected \leftarrow best(current, child)$
if $selected = current$
  then return $(current\_desc)$
if $selected = child$
  then return $(child\_desc)$

4.2.1 PITAGORAS-PSP based on PAES

In the Figure 4.9 it is shown the whole protein structure prediction process proposed in this work based in the PAES approach. This scheme has the

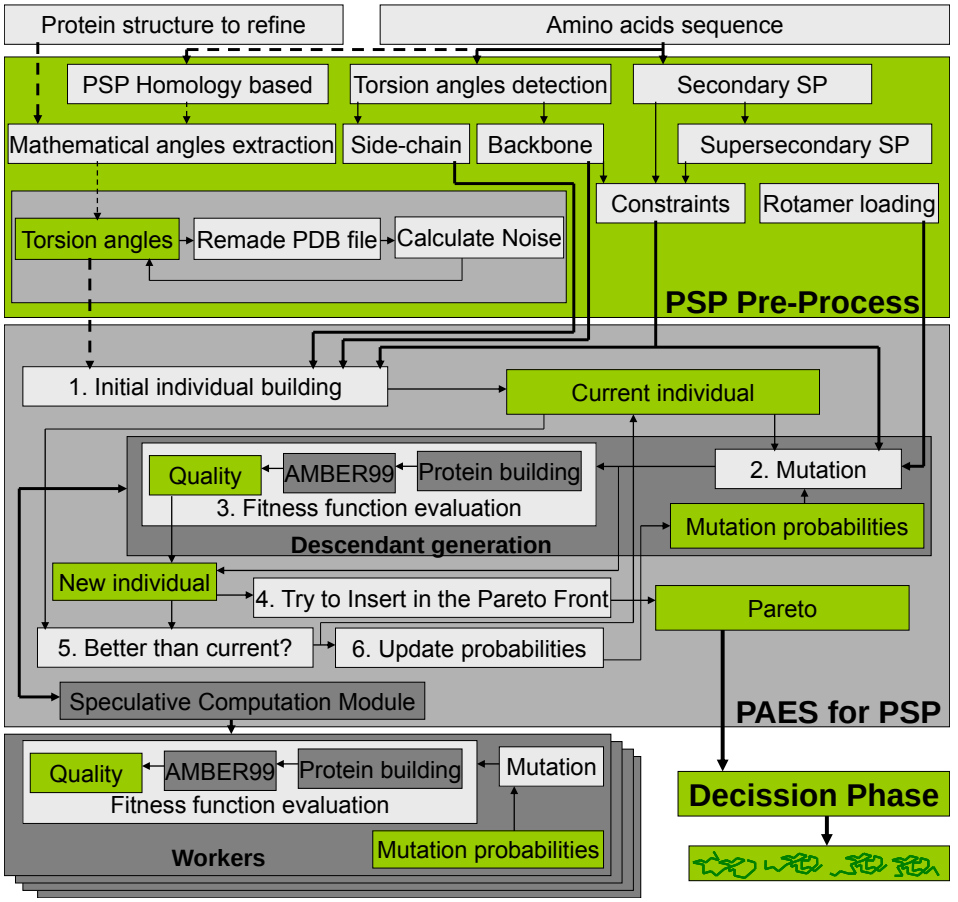


FIGURE 4.9: Global scheme of PITAGORAS-PSP based on PAES.

same structure as the one based on NSGA2, the only difference is the optimization process, based on the PAES approach in this case. In Chapter 3, this optimization procedure have been described, and the same sub-scheme has been used to explain the behavior:

Pre-process. In this phase, as it has been described in Chapter 2, the backbone and side-chain variables are extracted, the secondary structure prediction is obtained, and the rotamers library is loaded. Moreover, an homology-based procedure is executed in order to obtain an initial conformation. All this information is the input of the optimization phase.

PAES for PSP. This is the main phase of the multi-objective approach to PSP. As it has been explained in Chapter 3, a parallel implementation of PAES has been developed to find a set of non-dominated structures as near as possible to the corresponding Pareto Front.

Decision phase. The last phase tries to obtain the best conformations of those returned by the NSGA2 multi-objective approach. In Chapter 3 the decision phase has been fully explained.

The schemes given in Figures 4.5 and 4.9 include all the steps involved in the process, that starts from a given sequence of amino acids that defines the protein and finished with the predicted 3D structures for that protein, using the parallel algorithms described in this Chapter. From those descriptions it is possible to realize the complexity of the system developed in this research.

4.3 Conclusions

As it has been explained, due to the computing requirements of the PSP problem, we propose different methods for parallelize NSGA2 and PAES algorithms applied to the PSP problem. We propose a new method to parallelize the PAES algorithm preserving its original behavior by using speculative computation and adaptive prediction trees.

Publications with contributions of this chapter:

1. The Journal of Supercomputing 2011: *Comparative of parallel Multi-objective approaches to protein structure prediction* [Calvo et al., 2011a]

2. International Conference on Computational and Mathematical Methods in Science and Engineering: *Improving ab-initio protein structure prediction by parallel multi-objective evolutionary optimization* [Calvo, Ortega, and Anguita, 2009a]

3. 17th Euromicro International Conference on Parallel, Distributed and Network-Based Processing: *Parallel Protein Structure Prediction by Multiobjective Optimization* [Calvo and Ortega, 2009]

4. VII Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados: *Aproximación híbrida paralela para la predicción de estructuras de proteínas* [Calvo et al., 2010b]

5. XX Jornadas de Paralelismo: *Alternativas de Optimización Multiobjetivo Paralela en la Predicción de Estructuras de Proteínas* [Calvo, Ortega, and Anguita, 2009b]

6. XIX Jornadas de Paralelismo: *Optimización Multiobjetivo Paralela en la Predicción de Estructuras de Proteínas* [Calvo and Ortega, 2008]

Chapter 5

Experiments and Results

In this Chapter we present the results obtained in this researching work. We evaluate from different tests the results of the torsion angles optimization, the protein structure prediction algorithm, and the performance of the corresponding parallel implementations. First of all we define the experiments (Section 5.1), and finally, the results are presented and analyzed (Section 5.2).

5.1 Experiments Description

This section describes the experimental setup required to evaluate our proposals. We have focused in three different kinds of experiments:

1. Experiments on Torsion Angles Optimization (Section 5.1.1)

2. Experiments on Protein Structure Prediction (Section 5.1.2)

3. Experiments on performance of the corresponding parallel procedures (Section 5.1.3)

5.1.1 Experiments for Torsion Angles Optimization

In order to check the quality of the optimized torsion angles generated by our proposal, we compare the remade protein conformation structure and the original one. To compare two structures we use the RMSD measure (3.5) that gives the similarity between the remade and the native 3D structures, thus lower RMSD values are better.

The proteins 1PLW, 1CRN, 1UTG [RCSB, 2009] and T0513 (CASP8) have been used as benchmarks in this work. The protein 1PLW, also known as enkephalin, has 5 amino-acids (22 torsion angles), 1CRN has 46 amino-acids (194 torsion angles), 72 amino-acids in 1UTG (342 torsion angles) and 69 amino-acids are present in T0513 (327 torsion angles). With this set of proteins we have a complete range of sizes in order to check the behavior from little to big proteins.

As we propose different approaches that use the gradient descent process, we execute them until no movement is possible. In the other hand, when the CMA-ES [Hansen, 2006] approach is applied, it needs the number of fitness function evaluations to execute. In that case we have used different values. More specifically we have considered: 1000, 5000, 20000, and 100000 fitness functions evaluations.

5.1.2 Experiments for Protein Structure Prediction

To evaluate the accuracy of the protein structures determined by our procedures we have also considered the RMSD measure (3.5) that, as it has been indicated previously, gives the similarity between the predicted and the known native 3D structures. We also consider the GDT-TS measure [Zemla, 2003] (Global Distance Test - Total Score) (5.1), that computes the percentage of residues that can fit under distance cutoff of 1, 2, 4, and 8 $\mathring{A}$. Thus, in this cases higher GDT-TS values are better.

$$GDT - TS = 100(fit_1 + fit_2 + fit_4 + fit_8)/4length \tag{5.1}$$

In (5.1) fit_1, fit_2, fit_4, and fit_8 are the number of aligned residues within 1, 2, 4, and 8 $\mathring{A}$, respectively, and $length$ is the number of amino acids in the compared proteins.

In this case, we have run our algorithms by using a benchmark set that includes Free-Modeling proteins of different sizes and characteristics included in the CASP8 set: T0397 (82 amino acids), T0416 (52 amino acids), T0496 (120 amino acids), and T0513 (69 amino acids). The initial population uses TASSER results from CASP8 to avoid new knowledge implicit in the structures databases. We use these known proteins just to compare our results with those from others procedures, but the procedures here proposes could be executed with new proteins, where no structure is previously known.

We have executed the algorithms along 250,000 fitness function evaluations and have selected the solution in the Pareto front using the SPICKER software described in Section 3.2.4.

5.1.3 Experiments for Parallel Performance of the Corresponding Procedures

Our procedure has been implemented in parallel by using Message Passing Interface (MPI), and it has been executed in a cluster with 14 bi-processor nodes connected by Gigabit Ethernet. It includes 28 Intel Xeon Quad Core 5320 processors at 1.86 GHz, with 4 GB DDR2 RAM and 250 GB HD per node.

We have executed it in a range of different processors to observe speed up (5.2) and the efficiency (5.3) of the parallel approach.

$$SpeedUp = \frac{T_1}{T_n} \tag{5.2}$$

$$Efficiency = \frac{SpeedUp}{n} \tag{5.3}$$

In (5.2) and (5.3) T_x is the time required to execute the parallel approach in x processors, and n is the number of processors used in the execution.

5.2 Results

This section presents the results obtained applying the experiments described in the previous section. This section is also organized into three subsections:

1. The results on Torsion Angles Optimization (Section 5.2.1)
2. The results on Protein Structure Prediction (Section 5.2.2)
3. The results on parallel performance of the corresponding procedures (Section 5.2.3)

5.2.1 Results for Torsion Angles Optimization

In Figures 5.1, 5.2, 5.3 and 5.4 it is shown the different algorithms we have considered (included our proposal), and the time and performance results for the proteins used as benchmarks in the optimization of the torsion angles information. As it can be seen in Figures 5.1, 5.2, 5.3 and 5.4, in the 1CRN protein we have reduced the noise up to 70%, more than 80% in 1UTG and more than 90% of improvement is obtained in T0513. Depending on the time and on the algorithm used to get a solution, we can get different torsion angles. As it is shown in the Figures 5.1, 5.2, 5.3 and 5.4, CMA-ES obtains the best results for every single protein with enough running time. Depending on the protein, other methods obtain good solutions with less time than CMA-ES. As it can be seen, the *Method 1* is always among the

two or three worst methods. Analyzing the results we can conclude that according to their effectiveness, the methods can be ordered in this way: CMA-ES, Method 4, Method 3, Method 2, and Method 1, although this could slightly vary depending on the protein.

Figures 5.5, 5.6, and 5.7 show the improvements we can obtain by using CMA-ES, in 1CRN, 1UTG, and T0496 proteins respectively, to remake a protein structure, taking into account the real omega torsion angle. Figures 5.8, 5.9, and 5.10 show the improvements we can obtain by using CMA-ES, in 1CRN, 1UTG, and T0496 proteins respectively, to remake a protein structure, using the ideal value for the omega torsion angle. Each figure is a match of a real protein structure with a remade protein structure using the usual mathematical torsion angles (a) in each figure, and the optimized torsion angles obtained with CMA-ES (b) in each figure.

The use of the ω torsion angle does not seem to influence the result significatively whenever we apply our optimization method to extract torsion angles from a protein, as very good remade proteins have been obtained in any case. To summarize, by using the original mathematical torsion angles the noise can be appreciated, and it is specially significant whenever omega torsion angles are not considered.

We can also compare the refinement capabilities of the best method found depending on the protein size. As it is shown in Figures 5.1, 5.2, 5.3 and 5.4, the less number of torsion angles, the less the improvement it is possible to obtain, as less errors are accumulated. In a short protein, there are not many errors, and thus the remade protein structure is kept similar to the

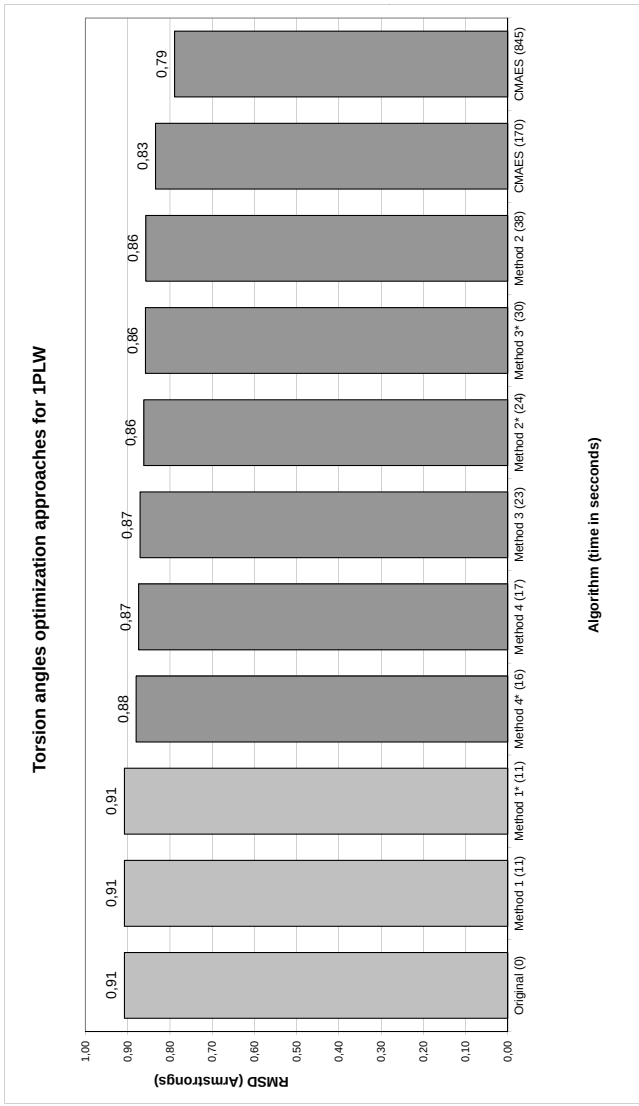


FIGURE 5.1: A comparative graph of all the methods ordered by the time required to get a solution for 1PLW protein. A method with dark column is better than previous methods, but needs more time. The rest of methods work worse. As it can be seen in the graph, the best method is the CMA-ES approach we have proposed to solve this problem.

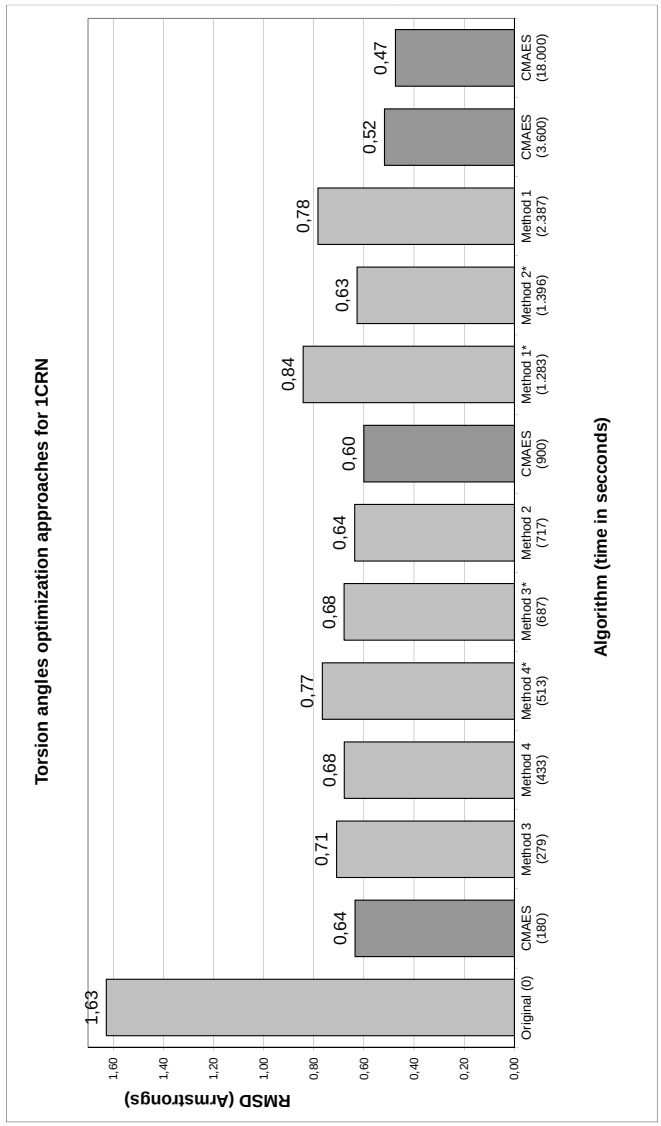


FIGURE 5.2: A comparative graph of all the methods ordered by the time required to get a solution for 1CRN protein. A method with dark column is better than previous methods, but needs more time. The rest of methods work worse. As it can be seen in the graph, the best method is the CMA-ES approach we have proposed to solve this problem.

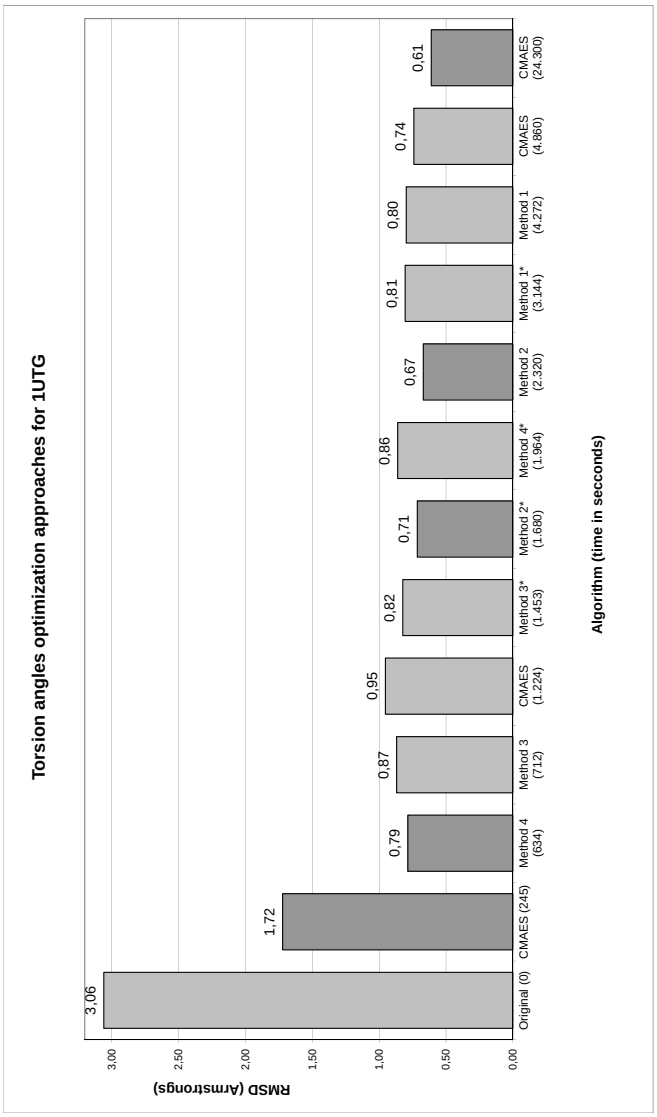


FIGURE 5.3: A comparative graph of all the methods ordered by the time required to get a solution for 1UTG protein. A method with dark column is better than previous methods, but needs more time. The rest of methods work worse. As it can be seen in the graph, the best method is the CMA-ES approach we have proposed to solve this problem.

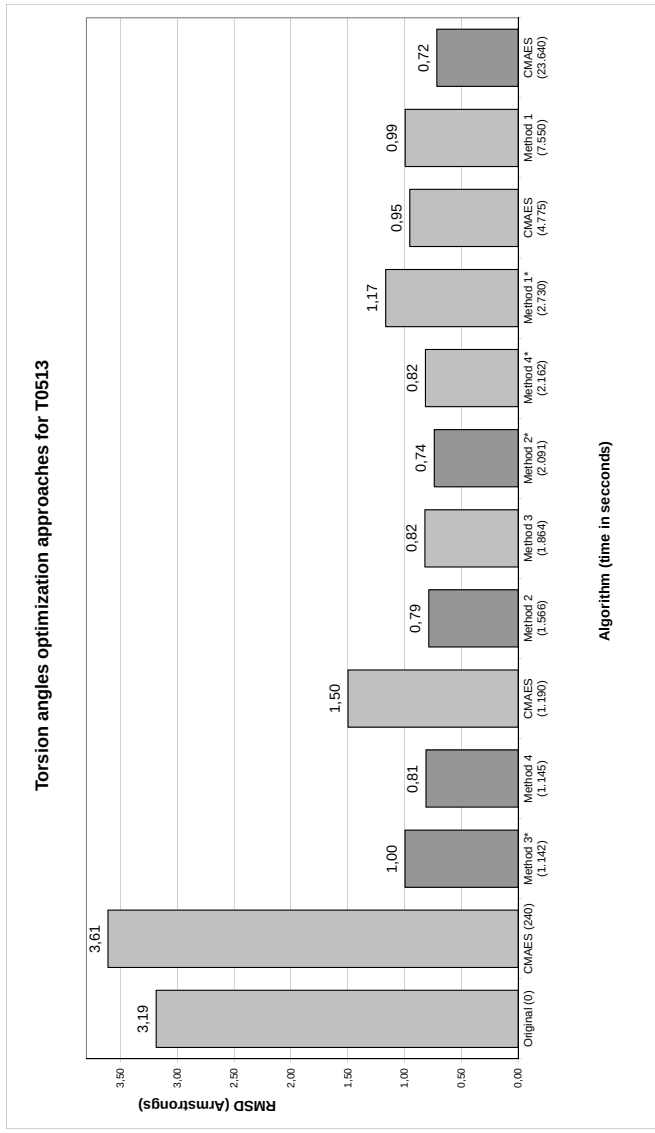


FIGURE 5.4: A comparative graph of all the methods ordered by the time required to get a solution for T0513 protein. A method with dark column is better than previous methods, but needs more time. The rest of methods work worse. As it can be seen in the graph, the best method is the CMA-ES approach we have proposed to solve this problem.

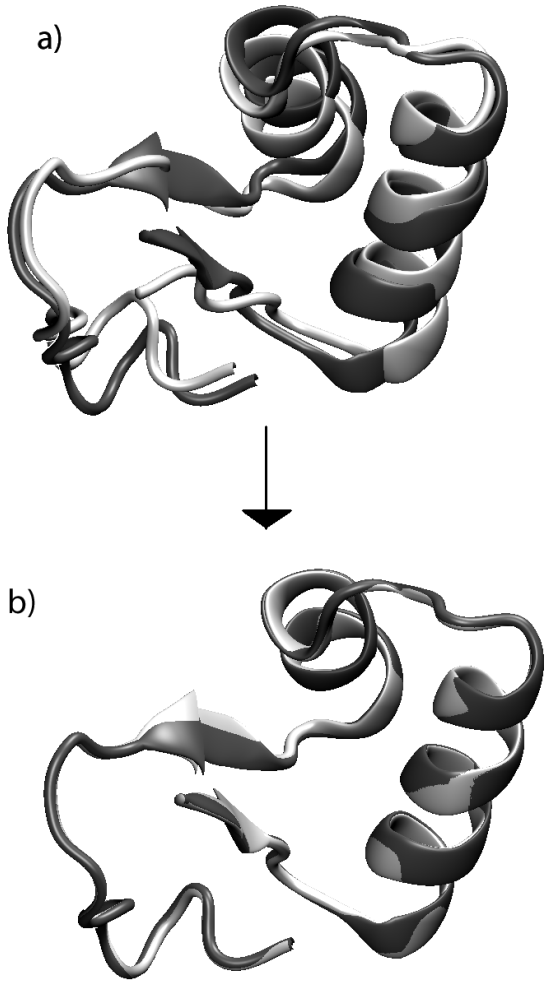


FIGURE 5.5: Improvements in the remade 1CRN protein (46 amino acids), by using the omega torsion angle information. The traditional method, (a), remakes similar structures. In the other hand, our algorithm produces a perfect fitting, (b).

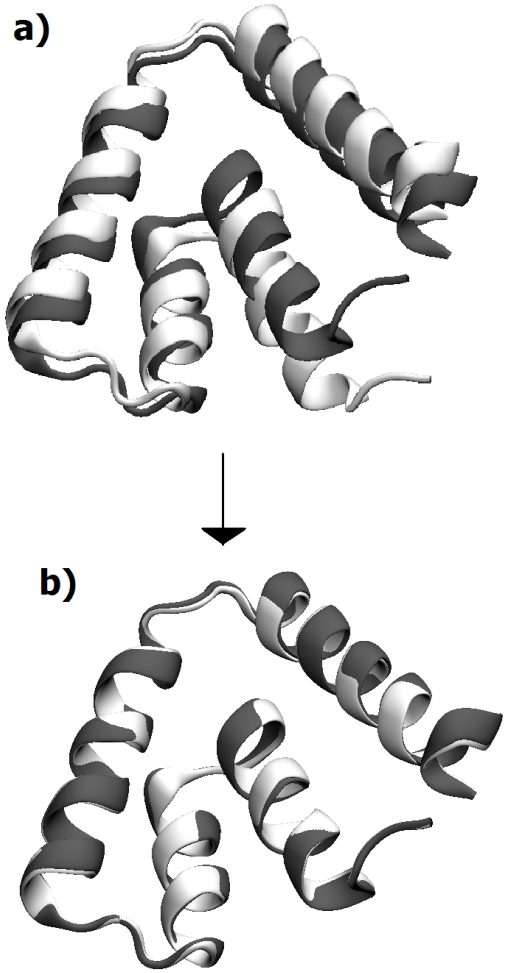


FIGURE 5.6: Improvements in the remade 1UTG protein (72 amino acids), by using the omega torsion angle information. The traditional method, (a), remakes similar structures whenever all the torsion angles are used. Our algorithm produces almost perfect fitting in that protein, (b).

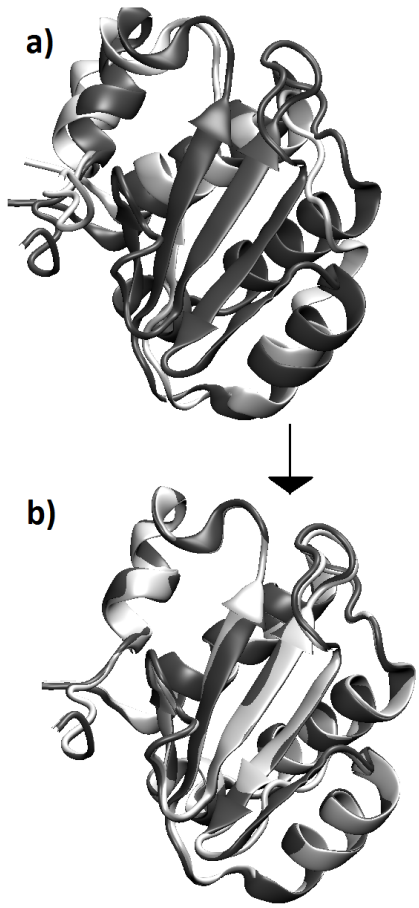


FIGURE 5.7: Improvements in the remade T0496 protein (120 amino acids), by using the omega torsion angle information. The traditional method, (a), is unable to remake similar structures. The remade proteins using the optimized torsion angles, (b), are very similar to the original one. In this protein, as it is bigger than the others, the noise produced by the traditional method is quite high, and the result has nothing to do with the original protein. As it can be seen, our optimization procedure can compensate the cumulative noise and produces good structures.

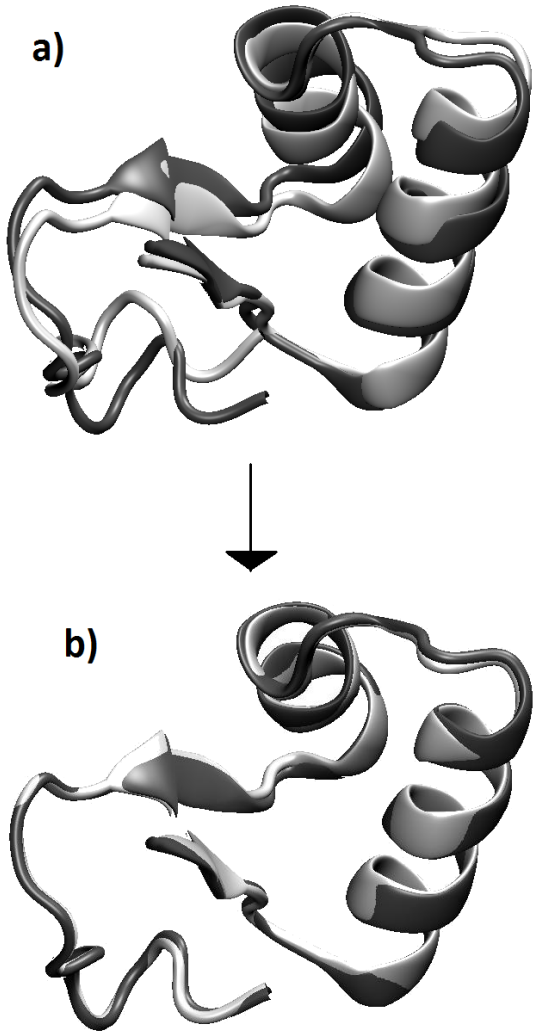


FIGURE 5.8: Improvements in the remade 1CRN protein (46 amino acids) without taking into account the omega torsion angle. The traditional method, (a), produces a lot of noise if we use the ideal value for the omega torsion angles. In the other hand, our algorithm produces almost perfect fitting in that situation, (b).

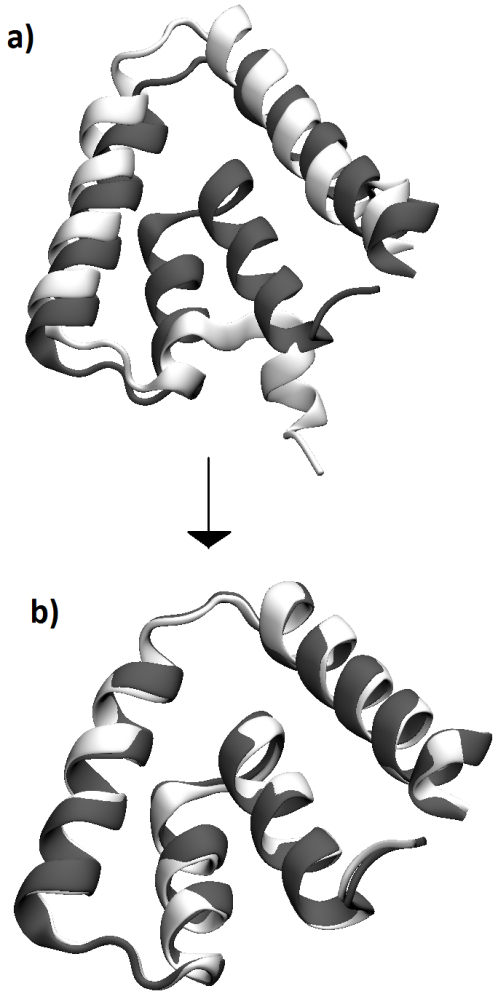


FIGURE 5.9: Improvements in the remade 1UTG protein (72 amino acids) without taking into account the omega torsion angle. The traditional method, (a), produces a lot of noise if we use the ideal value for the omega torsion angles. In the other hand, our algorithm produces almost perfect fitting in that situation, (b).

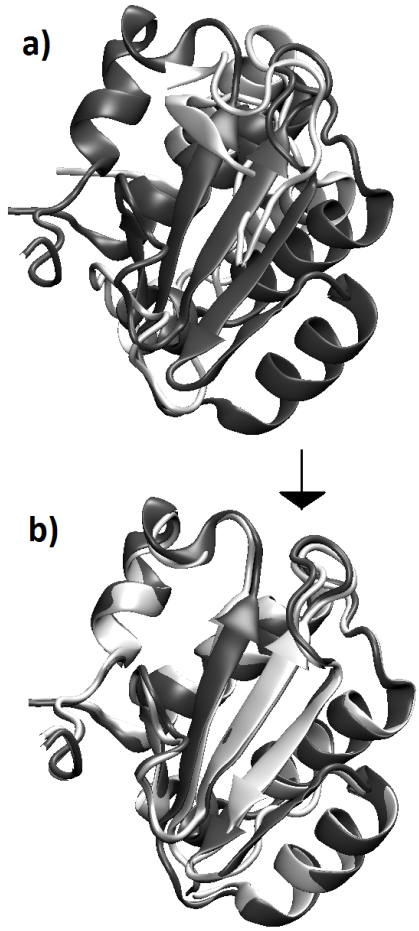


FIGURE 5.10: Improvements in the remade T0496 protein (120 amino acids) without taking into account the omega torsion angle. The traditional method, (a), is unable to remake similar structures either using or not the ideal value in the omega torsion angles. The remade proteins are far from the original protein. The remade proteins using the optimized torsion angles, (b), is very similar to the original one. In this protein, as it is bigger than the others, the noise produced by the traditional method is quite high, and the result has nothing to do with the original protein. As it can be seen, our optimization procedure can compensate the cumulative noise and produces good structures.

original one. Nevertheless in big proteins, a little change in one part of the protein structure can have a big effect in other part of the structure. This effect is graphically shown in Figures 5.5 c), 5.6 c), and 5.7 c).

We also provide the accuracy of this method by showing the deviation of the process. Two proteins have been optimized ten times by the CMA-ES algorithm during 20000 iterations: 3A2B (398 amino acids) and 1L45 (164 amino acids). As it can be seen from Table 5.1, the optimization process here proposed is highly stable.

TABLE 5.1: RMSD of two proteins and deviation of the torsion angles optimization process

Protein	#	Traditional method RMSD	Optimized method RMSD	Deleted noise
3A2B	398	24.567 $\mathring{A}$	1.315 $\pm$ 0.064 $\mathring{A}$	94.6%
1L45	164	5.873 $\mathring{A}$	0.815 $\pm$ 0.029 $\mathring{A}$	86.1%

Finally, the significance of the different methods can be analyzed by an ANOVA test. Every algorithm have been executed more than twenty times, and all the results have been introduced in an ANOVA test. It has been obtained that the probability of these results supposing the null hypothesis is less than 0.01%. Therefore, the results of our CMA-ES proposal are significantly different to that obtained from the other algorithms, as it is shown un Figure 5.11.

This way, CMA-ES algorithm applied to the optimized torsion angles extractor works fine, and it is able to reduce even more than 90% of noise

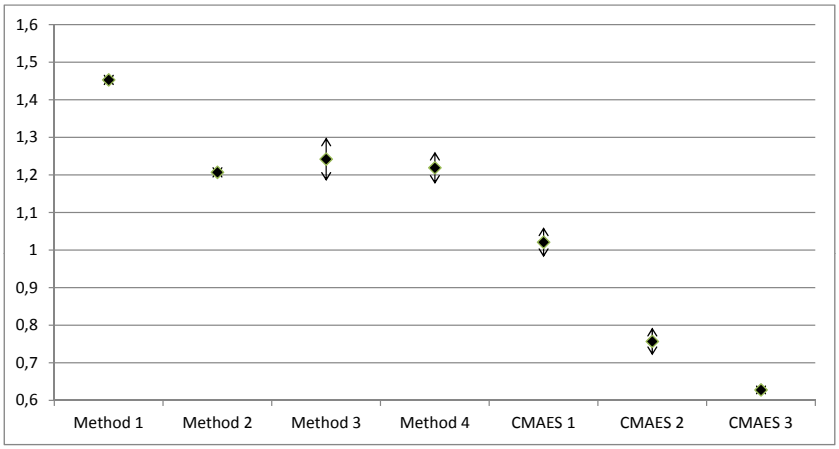


FIGURE 5.11: ANOVA test on the results of torsion angles optimization given by every algorithm. As it can be seen, the results are quite different for each other.

in big proteins. By using that algorithm, we will be able to make accurate protein structure predictors based on templates in algorithms that use angles representation.

5.2.2 Results for Protein Structure Prediction

The quality of the predicted 3D protein structures is evaluated in this subsection.

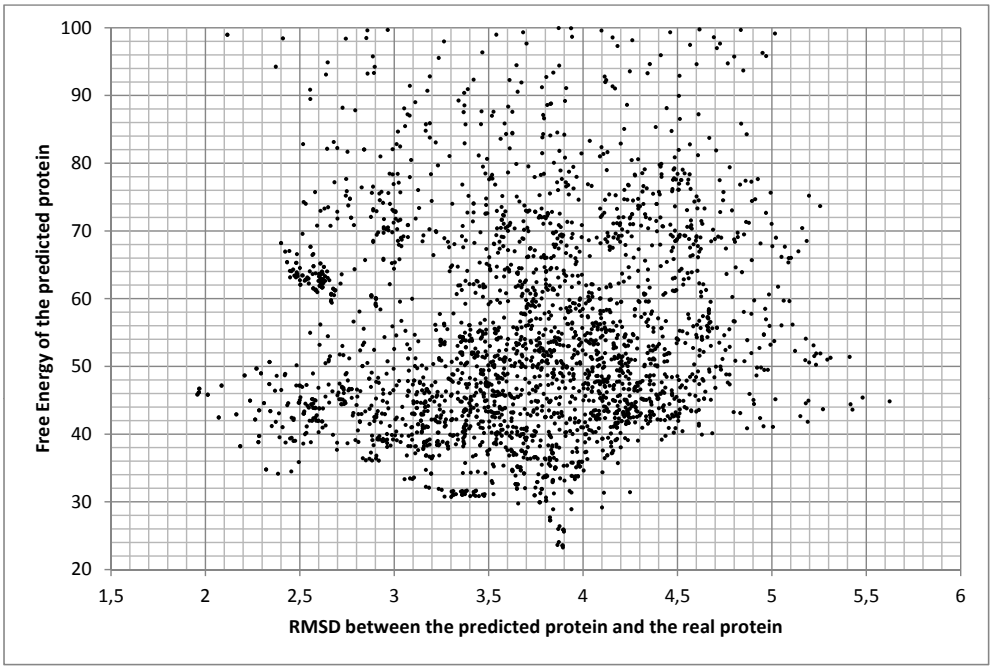


FIGURE 5.12: Each point represent one protein conformation, showing its global free energy versus its RMSD with the real protein. As it can be seen, there is no much information in the free energy to guide the optimization process to reach a good conformation of the sequence of amino acids.

First of all, we have evaluated the relationship between the fitness function and the RMSD of the predicted protein. In Figure 5.12 it is shown the free energy (bonded plus non-bonded) of the protein versus its RMSD. As it can be seen, the global free energy does not represent the final RMSD. We also show in Figures 5.13 and 5.14, the bond energy and non-bond energy respectively. In these figures we can observe that the bond energy has a correlation with the RMSD of the protein. Although the free energy is the

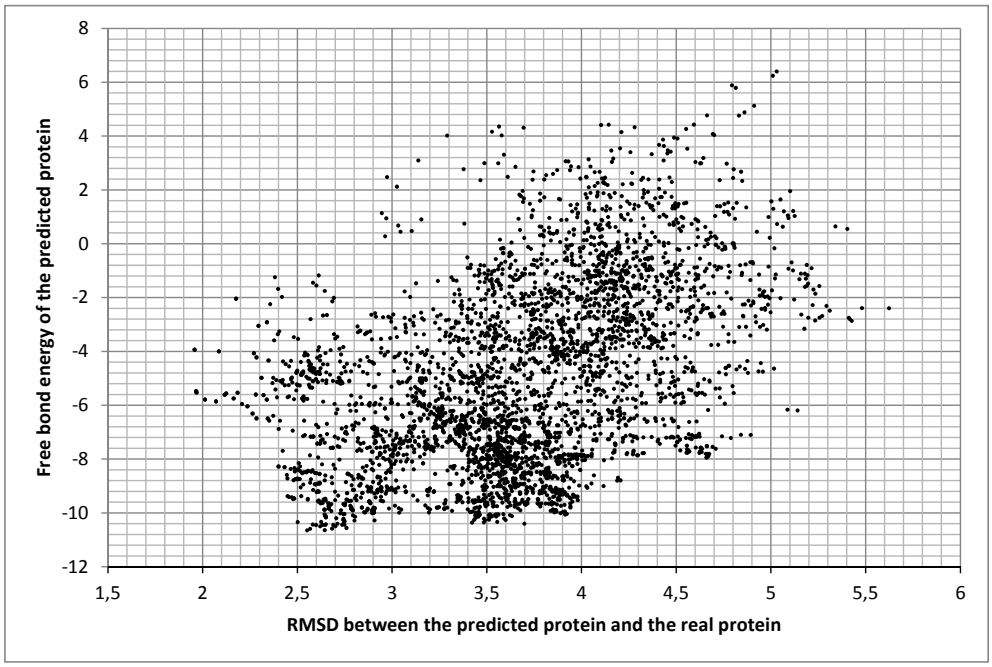


FIGURE 5.13: Each point represent one protein conformation, showing its bonded free energy versus its RMSD with the real protein. As it can be seen, there is a little correlation between the energy and the RMSD to guide the optimization process to reach a good conformation of the sequence of amino acids.

only variable we can optimize, it does not represents the RMSD optimization very well. As it can be seen in the three figures, the best conformation (minimum RMSD) is not the one with the minimum free energy in any case. Therefore, the methods to include external information into the optimization process, and the heuristics that guides the optimization have a very important role in the procedures that try to solve the PSP problem.

Table 5.2 compares the PSP results obtained by our procedure with those

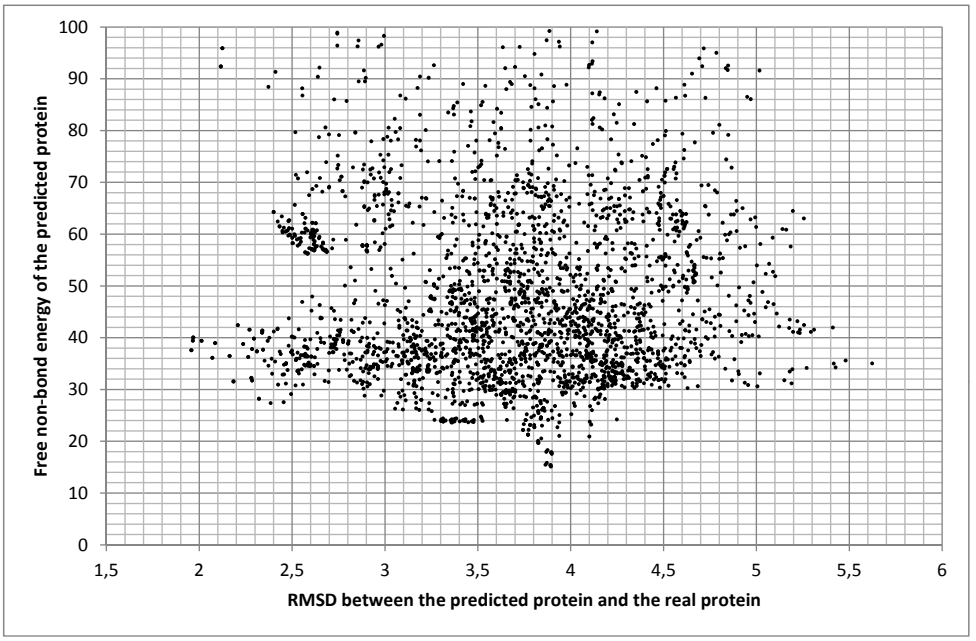


FIGURE 5.14: Each point represent one protein conformation, showing its non-bonded free energy versus its RMSD with the real protein. As it can be seen, there is no much information in the free energy to guide the optimization process to reach a good conformation of the sequence of amino acids.

obtained by the TASSER [Wu et al., 2007] algorithm, which is one of the best PSP procedures available at the moment. The version of PITAGORAS-PSP used in the comparison provided in Table 5.2 is based on the multi-objective optimization procedure PAES.

We have also evaluated PITAGORAS-PSP by using the results obtained in the CASP competition. Figures 5.15 and 5.16 shows the quality of the predicted protein structure for four different proteins. We have highlighted four procedures: PITAGORAS-PSP and three of the best algorithms in the

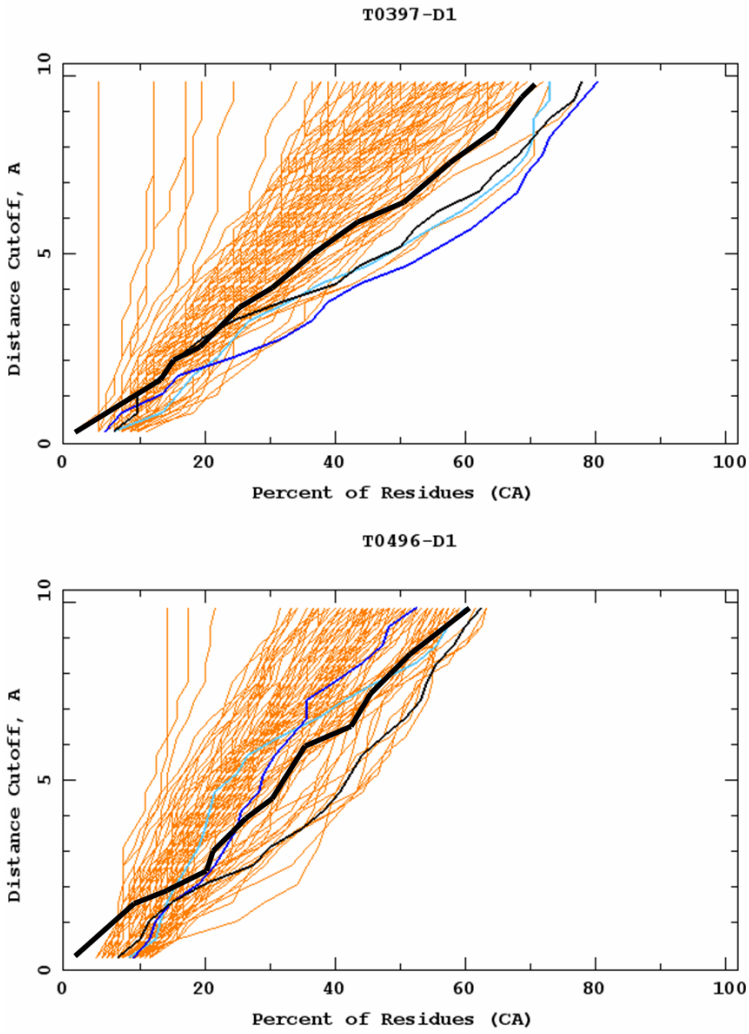


FIGURE 5.15: Comparative with CASP algorithms by using T0397 and T0496 proteins respectively (GDT analysis: largest set of CA atoms, evaluated as percent of the modeled structure, that can fit under DISTANCE cutoff: 0.5 $\mathring{A}$, 1.0 $\mathring{A}$,..., 10.0 $\mathring{A}$). Our algorithm is represented by the thicker line. Other three of the best procedures for T0397 have been selected to compare their relative performances in different proteins.

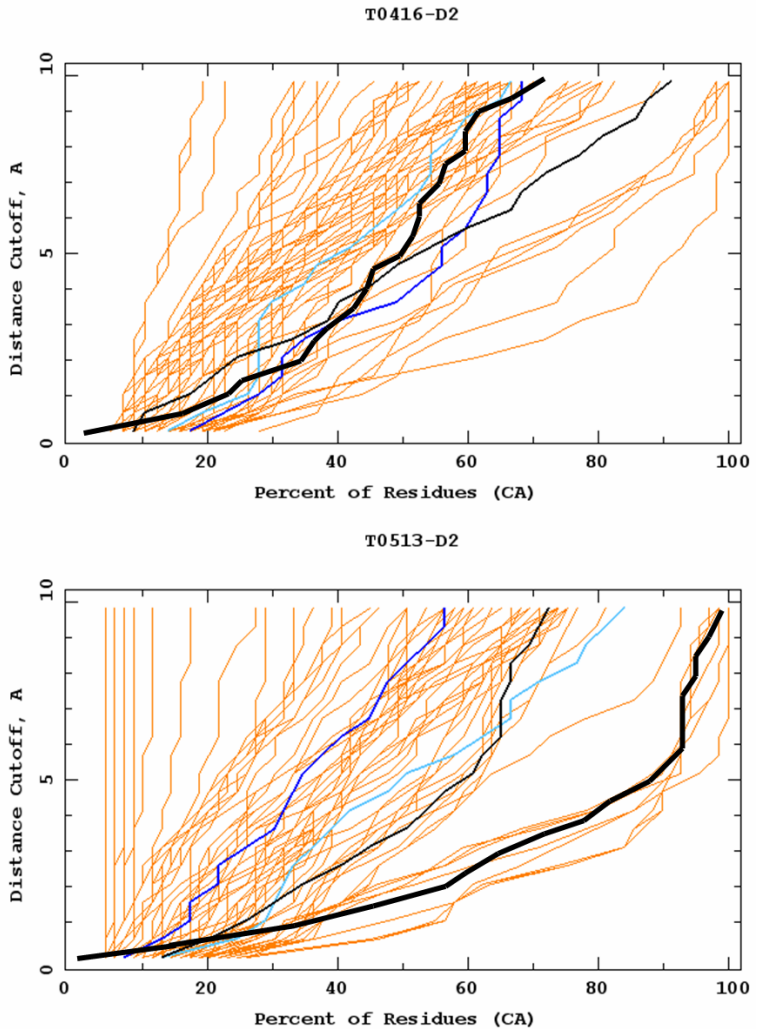


FIGURE 5.16: Comparative with CASP8 algorithms by using T0416 and T0513 proteins respectively (GDT analysis: largest set of CA atoms, evaluated as percent of the modeled structure, that can fit under DISTANCE cutoff: 0.5 $\mathring{A}$, 1.0 $\mathring{A}$,..., 10.0 $\mathring{A}$). Our algorithm is represented by the thicker line. Other three of the best procedures for T0397 have been selected to compare their relative performances in different proteins.

TABLE 5.2: PITAGORAS-PSP versus TASSER solutions in CASP8 (Only one solution is provided by CASP8, thus, no standard deviation can be shown).

Protein	#	PITAGORAS-PSP RMSD	TASSER RMSD	Improvement
T0397	82	*10.981* $\pm$ 0.122 *Å*	11.239 *Å*	2.3%
T0416	52	*9.407* $\pm$ 0.409 *Å*	12.934 *Å*	*27.3%*
T0496	120	11.965 $\pm$ 0.024 *Å*	*11.885 Å*	-0.7%
T0513	69	*4.292* $\pm$ 0.000 *Å*	4.297 *Å*	0%
		GDT-TS	GDT-TS	
T0397	82	28.35 $\pm$ 0.34	*28.96*	2%
T0416	52	*43.27* $\pm$ 0.21	41.23	-5%
T0496	120	23.96 $\pm$ 0.00	23.96	0%
T0513	69	67.03 $\pm$ 0.00	67.03	0%

Free Modeling proteins in the CASP. As it can be seen from Figures 5.15 and 5.16, the best procedure for one protein can be worse than other procedures in another protein. This fact shows that at the moment there is no close solution to the PSP problem, as it is not possible to find approaches that provide the best 3D structure for every known protein. Nevertheless, we can say that PITAGORAS-PSP obtains good protein structure predictions across the considered benchmark set.

5.2.2.1 Simplified search space method

As it has been said in Chapter 3, in the first part of the EA (for example, the first 10% of the total number of evaluations), the search space used is a simplification of the real one. This search space consists in only one

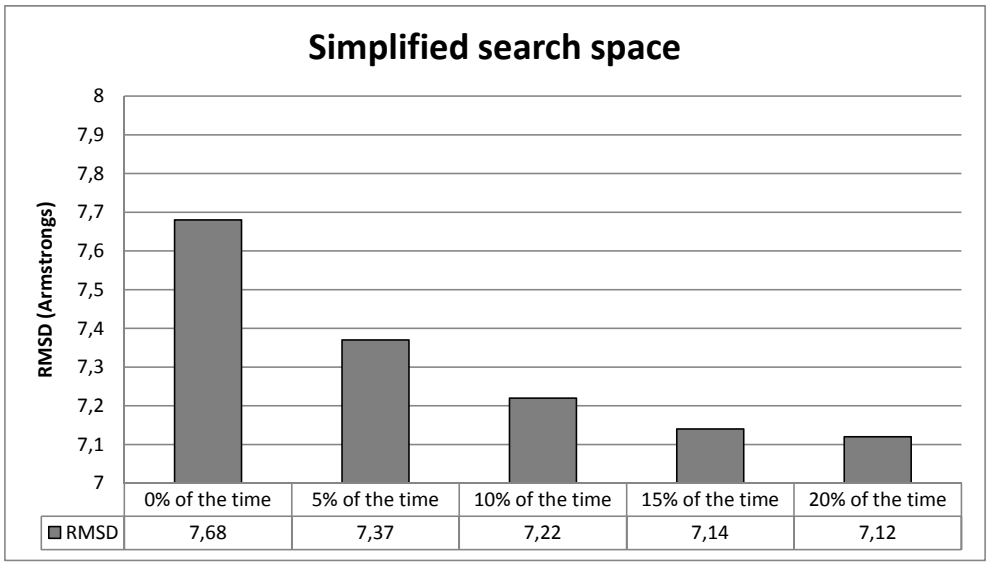


FIGURE 5.17: Usage of the simplified search space method. This method is used during a percentage of the execution time. The Figure shows 0%, 5%, 10%, 15% and 20% of the execution time using this method.

variable, with only four possible values, per amino acid. This way, the EA can travel across the whole simplified search space. After this period, the search space becomes the real one (Fig. 3.8). In Figure 5.17 it is shown the influence of this method depending on the percentage of the time that the algorithm is using it, going from 0% to 20%. The difference between using or not this method is important.

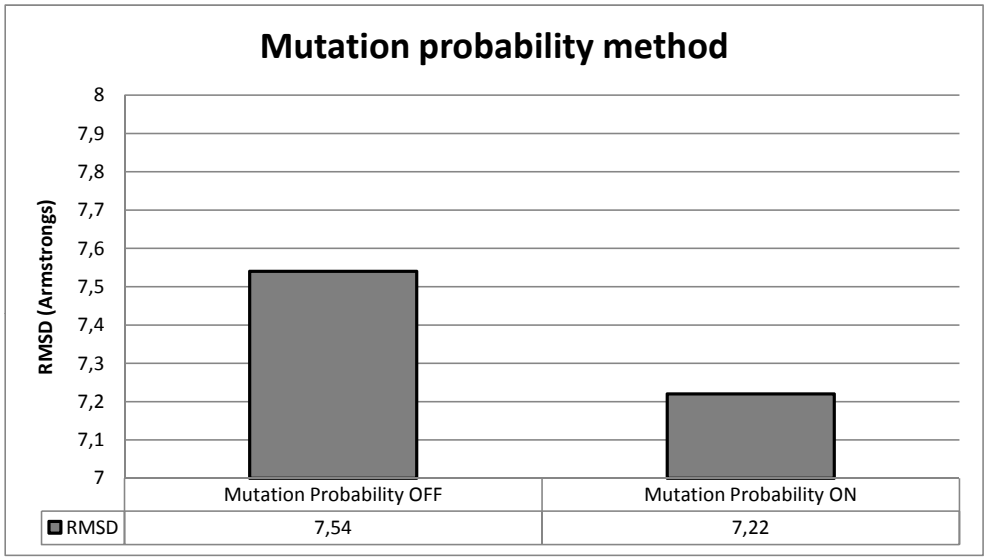


FIGURE 5.18: Usage of the mutation probability method, the Figure compares the results by using or not this method.

5.2.2.2 Amino acid mutation probability

In Chapter 3, the amino acid mutation probability method has been fully explained. As it has been described, an amino acid mutation could determine a very different energy change depending on the present structure of the protein. The method we have proposed tries to set highest probabilities of mutation to those amino acids that play an important role in the present structure. In Figure 5.18 we shows the difference between using or not this method. As it is shown, the method gives us a better result.

5.2.3 Parallel performance analysis

Once the quality of the solutions found by our procedure has been analyzed, the performance figures about the benefits obtained from the different parallel executions are now provided. As we have proposed, two different parallel multi-objective approaches, NSGA2 and PAES, we analyze the parallel performance of both alternatives in the following sections.

5.2.3.1 Parallel NSGA2 performance

Figure 5.19 shows the average computing time required to complete the structure prediction against the number of processors, and Figure 5.20 provides the corresponding speedups. The individuals of the population have been distributed among the processors by using partitions of similar size. As it is shown in Figure 5.20, the obtained speedup is up to 13.62 with 14 processors. In Figure 5.21 it can be seen that the *master-worker-3* efficiency keeps close to 1 among the experiments (using 14 nodes the efficiency is 0.97). In the *master-worker-1* and *master-worker-2* approaches, the efficiency is also high for more than 6 processors.

From these results we can observe than the implemented master-worker scheme, despite its simplicity, obtains high performances in the problem of the Protein Structure Prediction. Therefore, it demonstrates that, as it has been supposed, the fitness function evaluation is the main bottleneck in the PSP problem. We also observe that for our maximum number of

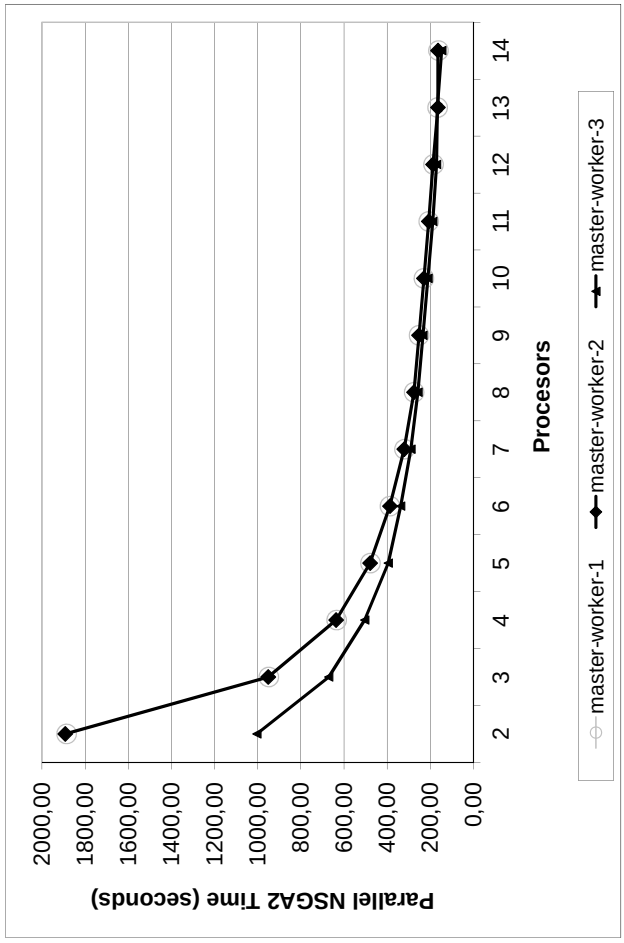


FIGURE 5.19: Execution time of the parallel NSGA2 against the number of processors.

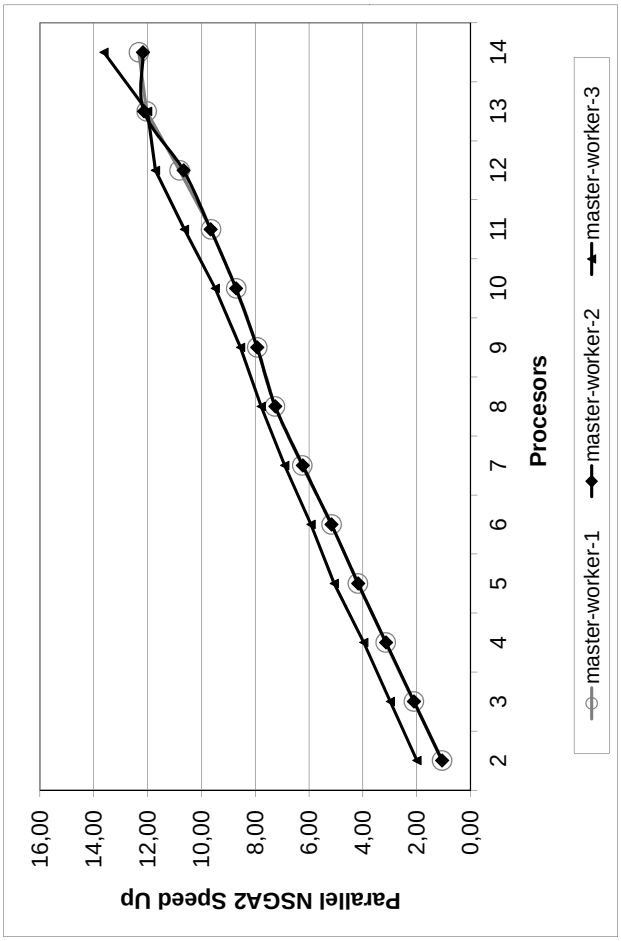


FIGURE 5.20: Speedup of the parallel NSGA2 against the number of processors.

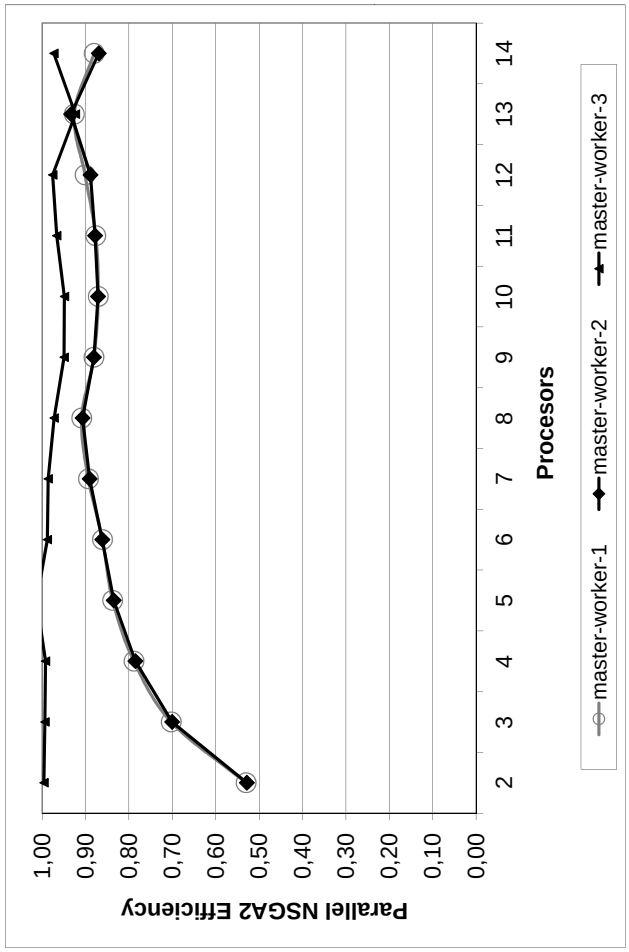


FIGURE 5.21: Efficiency of the parallel NSGA2 against the number of processors.

processors, the parallel system keeps the efficiency, therefore more speedup could be obtained by using more processors.

5.2.3.2 Parallel PAES performance

Figure 5.22 shows the average computing time required to complete the structure prediction problem against the number of processors, and Figures 5.23 and 5.24 provides the corresponding speedups and efficiencies. As it is shown in Figure 5.23 the speedup is up to 17 with 20 processors. In Figure 5.24 it can be seen that the efficiency keeps close to 90% in all the experiments. In Table 5.3 the results of the parallel performance are available and its deviation for the 1PLW protein structure prediction, executed ten times.

Taking into account that results, we can conclude that our proposal is able to extract a significant speedup from a highly sequential algorithm such as the PAES. Therefore, the speculative computation works quite well in this problem. That can be explained by taking into account that improving a given solution is usually a difficult test, and in many iterations the algorithm remains in the same solution. With that behavior, the adaptive speculative computation learns that the best alternative is to create a lot of children from the current solution, and most of them are used in the sequential algorithm.

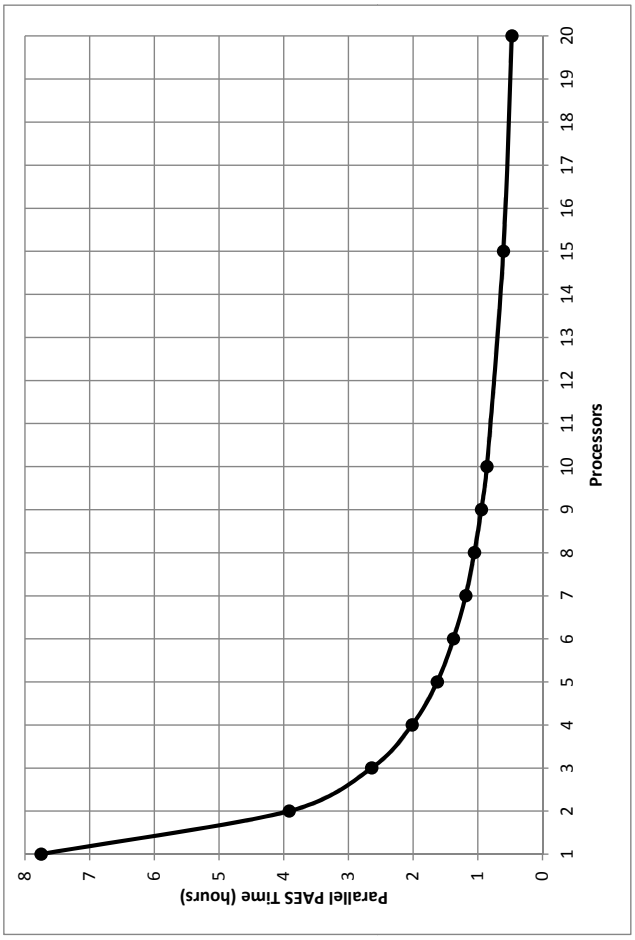


FIGURE 5.22: Execution time of the parallel PAES against the number of processors for the 1PLW protein.

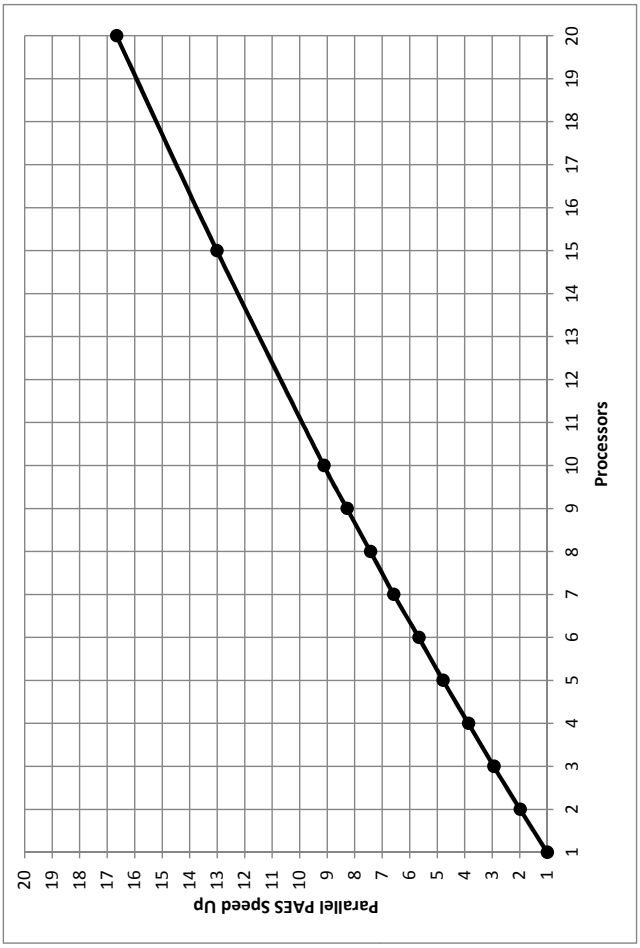


FIGURE 5.23: Speedup of the parallel PAES against the number of processors for the 1PLW protein.

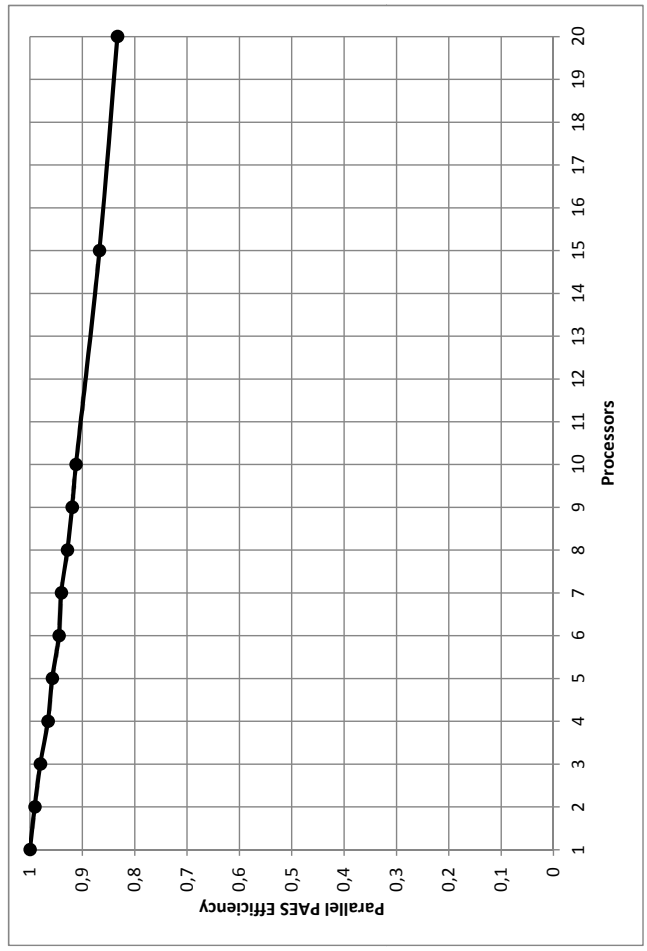


FIGURE 5.24: Efficiency of the parallel PAES against the number of processors for the 1PLW protein.

TABLE 5.3: Time, speed up and efficiency of the Parallel PAES procedure executed in the prediction of the 1PLW protein structure.

Processors	Time	Speed up	Efficiency	Deviation
1	7.75	1.0	100%	0.0%
2	3.91	2.0	99%	0.1%
3	2.64	2.9	98%	0.2%
4	2.01	3.9	97%	0.5%
5	1.63	4.8	96%	0.6%
6	1.37	5.7	94%	0.7%
7	1.19	6.6	94%	0.9%
8	1.05	7.4	93%	1.1%
9	0.94	8.3	92%	1.1%
10	0.86	9.1	91%	1.1%
15	0.60	13.0	87%	1.6%
20	0.47	16.7	83%	1.9%

5.3 Conclusions

We have presented the results obtained in this researching work. We have shown the results of the torsion angles optimization, the protein structure prediction algorithm, and the performance of the corresponding parallel implementations, being most of these results significantly better than other approaches.

Chapter 6

Conclusiones y trabajo futuro

Este capítulo, en una primera sección, resume el estado del conocimiento en predicción de estructuras de proteínas existente antes de comenzar el trabajo presentado en esta memoria. A continuación se resumen las aportaciones de este trabajo realizadas para resolver este problema (Secciones 6.2 y 6.3). Finalmente (Sección 6.4), se se presentan algunos problemas que podrían constituir el trabajo futuro en esta línea de investigación. Algo importante en un problema abierto todavía, como es la predicción de estructuras de proteínas.

6.1 El estado del arte al comenzar este trabajo

El problema de la predicción de estructuras de proteínas pone en común conceptos biológicos y técnicas computacionales. Requiere modelos precisos de la energía de una estructura 3D de una proteína. No obstante, a nivel teórico todavía no se conoce el mecanismo real de plegamiento de una proteína y, por tanto, aún estamos lejos de encontrar una solución completa y eficiente a este problema, para cualquier tipo de proteína. Hasta ahora, los mejores métodos para la predicción de estructuras 3D de proteínas están basados en homologías, como por ejemplo I-TASSER o ROSETTA@HOME. Estos algoritmos suelen tener una primera etapa en la que buscan homologías y extraen conocimiento de las bases de datos existentes, y una segunda etapa en la que optimizan la estructura para tener en cuenta las zonas en las que no se han encontrado homologías. No obstante, estos algoritmos de optimización, son menos eficientes desde el punto de vista de la incorporación del estado del arte en el campo de la optimización. Normalmente, estas soluciones han sido desarrolladas en gran medida por biólogos. Por tanto, si bien son capaces de aunar mucha información específica del problema biológico, en cambio, no aplican métodos de optimización avanzados.

En el campo de las librerías estadísticas de ángulos de torsión, y en la propia extracción de estos ángulos de torsión desde una proteína, actualmente existe un método matemático para calcular estos ángulos y posteriormente desarrollar librerías estadísticas. No obstante, dado que las estructuras conocidas se ven afectadas por cierto ruido, este método matemático sufre

una acumulación de dicho ruido, ocasionando que las proteínas reconstruidas a partir de los ángulos extraídos puedan ser bastante diferentes a las originales. Por tanto, todo sistema que use estas librerías o necesite extraer ángulos de torsión de una proteína, trabajaría con información sujeta a cierto nivel de error.

6.2 Contribuciones de este Trabajo

La principal contribución de este trabajo consiste en un nuevo procedimiento para la Predicción de Estructuras de Proteínas basado en algoritmos evolutivos multi-objetivo. Además, este procedimiento incluye varias estrategias para reducir el espacio de búsqueda, y dispone de distintas estrategias y heurísticas para incrementar la calidad de las soluciones, como son los métodos específicos de inicialización de la población, los operadores de mutación adaptados al problema, las estrategias de evolución, el uso de librerías estadísticas de ángulos de torsión, y una nueva división de objetivos. Comparando nuestros resultados con los de otros algoritmos del estado del arte actual, puede verse que para cierta tipología de proteínas, nuestro método consigue resultados comparativamente similares y en ocasiones incluso mejores que los métodos más eficientes actuales. Además, nuestra implementación paralela del algoritmo permite una reducción considerable del tiempo de ejecución.

Nuestro sistema, llamado PITAGORAS-PSP, es capaz de ejecutar una optimización multi-objetivo avanzada teniendo como punto de partida soluciones obtenidas por sistemas basados en homologías. De esta forma, nuestro sistema puede usarse no solo para predecir estructuras desde cero, sino también para optimizar y refinar estructuras calculadas por otros métodos.

Por otra parte, también hemos propuesto la primera versión paralela del algoritmo multi-objetivo PAES, preservando su comportamiento secuencial. Hasta el momento, para paralelizar este algoritmo se incorporaba una modificación en el mismo para convertirlo en poblacional y así poder distribuir los individuos en varias máquinas. Dado que el algoritmo PAES está basado en poblaciones de un solo individuo, solo hay una solución que va evolucionando en el tiempo. Nuestra propuesta es capaz de paralelizar este algoritmo manteniendo su semántica original y para ello usamos computación especulativa. Aunque la ganancia en velocidad no es muy elevada (solo conseguimos una eficiencia cercana a 0.5), es capaz de ejecutar de forma rápida un algoritmo con dependencias masivas entre iteraciones, y considerablemente dificil de paralelizar.

El trabajo aporta también novedades en el campo de la extracción de ángulos de torsión para una estructura de una proteína dada. Hemos desarrollado un método basado en estrategias de evolución para generar unos ángulos de torsión optimizados que amortigüen y minimicen el ruido de una estructura. Como resultado, proporcionamos un método capaz de generar ángulos de torsión que, si se usan para regenerar la proteína original, consiguen resultados prácticamente iguales a los reales. En cambio, con el método matemático existente hasta ahora, las reconstrucciones de proteínas

medianas o grandes, puede generar estructuras lejanas a las reales. Como conclusión, se puede indicar que con el método aquí propuesto, podrían generarse nuevas librerías estadísticas con información más exacta y, por tanto, de mayor calidad.

6.3 Publicaciones

Los resultados de todo este trabajo han sido publicados en revistas y conferencias tanto nacionales como internacionales:

1. Neurocomputing 2011: *PITAGORAS-PSP: Including domain knowledge in a multi-objective approach for protein structure prediction* [Calvo et al., 2011b]. Chapter 3.

2. The Journal of Supercomputing 2011: *Comparative of parallel Multi-objective approaches to protein structure prediction* [Calvo et al., 2011a]. Chapter 4.

3. BIOSTEC 2011: *A Method to Improve the Accuracy of Protein Torsion Angles* [Calvo et al., 2011c]. Chapter 2.

4. 4th International Workshop on Practical Applications of Computational Biology & Bioinformatics (IWPACBB 2010): *A Hybrid Scheme to Solve the Protein Structure Prediction Problem* [Calvo et al., 2010a]. Chapter 3.

5. International Conference on Computational and Mathematical Methods in Science and Engineering: *Improving ab-initio protein structure prediction by parallel multi-objective evolutionary optimization* [Calvo et al., 2009a]. Chapter 4.

6. 10th International Work-Conference on Artificial Neural Networks (IWANN 2009): *Protein Structure Prediction by Evolutionary Multiobjective Optimization: Search Space Reduction by Using Rotamers* [Calvo et al., 2009c]. Chapter 2.

7. 17th Euromicro International Conference on Parallel, Distributed and Network-Based Processing: *Parallel Protein Structure Prediction by Multiobjective Optimization* [Calvo and Ortega, 2009]. Chapter 4.

8. VII Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados: *Aproximación híbrida paralela para la predicción de estructuras de proteínas* [Calvo et al., 2010b]. Chapter 3.

9. XX Jornadas de Paralelismo: *Alternativas de Optimización Multiobjetivo Paralela en la Predicción de Estructuras de Proteínas* [Calvo et al., 2009b]. Chapter 4.

10. XIX Jornadas de Paralelismo: *Optimización Multiobjetivo Paralela en la Predicción de Estructuras de Proteínas.* Chapter 4. [Calvo and Ortega, 2008]

Y se han obtenido algunas referenciasa estos trabajos, pese a lo recientes de los mismos, como por ejemplo:

1. Applied Microbiology and Biotechnology: *Maximizing the native concentration and shelf life of protein: a multiobjective optimization to reduce aggregation* [Saif, Vinod, and Vandana, 2011]

6.4 Trabajo Futuro

El trabajo futuro para esta línea de investigación, puede dividirse en dos ámbitos: la predicción de estructuras de proteínas y la optimización de ángulos de torsión

6.4.1 Predicción de Estructuras de Proteínas

La predición de estructuras de proteínas (PSP) es un problema que está lejos de su resolución final completa, por lo que aún se requerirá mucho esfuerzo en el futuro para conseguir mejorar las predicciones. Debido a la alta demanda computacional será necesario encontrar implementaciones que aprovechen al máximo la arquitectura de los computadores paralelos. Como además se necesita manejar datos que se encuentran en bases de datos distribuidas geográficamente, hará falta implementar procedimientos eficientes en plataformas de altas prestaciones distribuidas y GRID. El uso de la computación en nube (Cloud Computing) probablemente contribuya a impulsar los avances en este campo porque permite que los investigadores puedan acceder remotamente a plataformas de cómputo y almacenamiento de altas prestaciones con menor coste en tiempo y dinero.

Además, como trabajo futuro queda analizar las casuísitcas en las que cada optimizador que hemos propuesto aporta más valor, para así poder crear un algoritmo que explote más ciertos optimizadores en las proteínas o momentos donde mejores resultados puedan dar, optimizando así los resultados y el tiempo de ejecución.

6.4.2 Base de Datos Online de Ángulos de Torsión Optimizados

Gracias a la definición de un nuevo método para la extracción de los ángulos de torsión de las proteínas, una línea de trabajo futuro podría basarse en la creación de una base de datos a la que pueda accederse online y en la que se volcarían los ángulos de torsión de las proteínas conocidas. Esta base da datos podría usarse para la extracción de un conocimiento más preciso y ayudaría al desarrollo de nuevos métodos que, en el futuro, resuelvan definitivamente el problema de la Predicción de Estructuras de Proteínas, algo decisivo para la lucha contra nuestras enfermedades de enorme incidencia social, como el Alzheimer o el Cáncer.

Chapter 7

Contributions and Future Work

This chapter presents the conclusions and future work to do (which is important if it is taken into account that the PSP is still an open problem). First of all, we start with a brief explanation of the previous situation to this work, then, we present the main contributions and the conclusions of this thesis, and finally, a description of the future work in this research line is provided.

7.1 Previous Situation to this Work

The PSP problem joins biological and computational concepts. It requires accurate and tractable models of the conformation energy. Thus, there is a long way to go to find useful solutions to the problem, more significantly in the case of proteins of high sizes. Up to now, the best procedures to the Protein Structure Prediction are template-based like TASSER or ROSSETA. These methods join an homology approach and an optimization process. Nevertheless, the optimization approach is mono-objective and is less relevant than the initial part based on templates.

In the field of torsion angles and rotamer libraries, the only method to extract torsion angles from a protein was a mathematical one. Therefore, some noise is introduced in the systems.

7.2 Contributions of this Work

Our contribution in this thesis deals with a new procedure for PSP based on a multi-objective evolutionary algorithm. It allows a reduction in the number of variables and provides some strategies to improve the quality of the solutions such as specific initialization methods and mutation operators, the use of rotamer libraries, and three objectives in the multi-objective cost function. Moreover, the parallel implementation of our procedure achieves a relevant reduction in the processing time. By comparing the results of our procedure with those provided by other previously proposed efficient

approaches, such as [Wu et al., 2007]. It has been shown that our methods provide conformations of comparable or even better quality.

In our approach, called PITAGORAS-PSP, it is possible to execute an advanced multi-objective optimization starting with homology information. This behavior can outperform other procedures in the PSP problem. as it has been shown in the benchmark set of proteins considered.

We also have proposed the first parallel approach, to the best of our knowledge, to the PAES multi-objective algorithm that preserves its sequential behavior. This parallelization is based in speculative computation, and although the efficiency is not very high, it can execute faster the algorithm PAES with a massive dependencies among its iterations and thus it is very hard to parallelize.

We also have contributed to the field of extracting torsion angles values from a given protein. We have developed an optimization process based on CMA-ES to extract the optimized torsion angles that minimize the noise. Therefore, better information is extracted and better rotamer libraries can be used.

7.3 Publications

The results of this work have been published in journals and international and national conferences as follows:

1. Neurocomputing 2011: *PITAGORAS-PSP: Including domain knowledge in a multi-objective approach for protein structure prediction* [Calvo et al., 2011b]. Chapter 3.

2. The Journal of Supercomputing 2011: *Comparative of parallel Multi-objective approaches to protein structure prediction* [Calvo et al., 2011a]. Chapter 4.

3. BIOSTEC 2011: *A Method to Improve the Accuracy of Protein Torsion Angles* [Calvo et al., 2011c]. Chapter 2.

4. 4th International Workshop on Practical Applications of Computational Biology & Bioinformatics (IWPACBB 2010): *A Hybrid Scheme to Solve the Protein Structure Prediction Problem* [Calvo et al., 2010a]. Chapter 3.

5. International Conference on Computational and Mathematical Methods in Science and Engineering: *Improving ab-initio protein structure prediction by parallel multi-objective evolutionary optimization* [Calvo et al., 2009a]. Chapter 4.

6. 10th International Work-Conference on Artificial Neural Networks (IWANN 2009): *Protein Structure Prediction by Evolutionary Multi-objective Optimization: Search Space Reduction by Using Rotamers* [Calvo et al., 2009c]. Chapter 2.

7. 17th Euromicro International Conference on Parallel, Distributed and Network-Based Processing: *Parallel Protein Structure Prediction by Multiobjective Optimization* [Calvo and Ortega, 2009]. Chapter 4.

8. VII Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados: *Aproximación híbrida paralela para la predicción de estructuras de proteínas* [Calvo et al., 2010b]. Chapter 3.

9. XX Jornadas de Paralelismo: *Alternativas de Optimización Multiobjetivo Paralela en la Predicción de Estructuras de Proteínas* [Calvo et al., 2009b]. Chapter 4.

10. XIX Jornadas de Paralelismo: *Optimización Multiobjetivo Paralela en la Predicción de Estructuras de Proteínas.* Chapter 4. [Calvo and Ortega, 2008]

Some others publication have mentioned our works, although they have been published a short time ago, in example:

1. Applied Microbiology and Biotechnology: *Maximizing the native concentration and shelf life of protein: a multiobjective optimization to reduce aggregation* [Saif et al., 2011]

7.4 Future Work

The future work to this field is divided into two lines corresponding to Protein Structure Prediction and Torsion Angles Optimization.

7.4.1 Protein Structure Prediction

The PSP problem is far to be solved, in that sense, more work has to be done in order to get better prediction qualities. As these procedures have a very high computational demand, researches must work to take advantages of the parallel computer architectures. Moreover, as these parallel procedures need to work with data distributed, high performance distributed and GRID implementations must be considered. Cloud computing can drive the advances in this field because it allows that the researches access to the state of the art in compute and store platform at lower cost both in time and budget. Cloud computing techniques will also provide an easy access to the knowledge extracted by others approaches in order to include them in our population. Improved procedures to select a solution from the non-dominated set could be also useful to increase the usefulness of multi-objective approaches to PSP.

7.4.2 Online Optimized Torsion Angles Data Base

As we propose a new method to optimize torsion angles, a future work can be based in the creation of an online data base that includes torsion angles. This data base could be used to create a new rotamer library more accurate than the previous ones used in this work.

Bibliography

D.P. Anderson. Boinc: A system for public-resource computing and storage. *Proc. Fifth IEEE/ACM Int'l Workshop Grid Computing*, 2004.

C.B. Anfinsen. Principles that govern the folding of protein chains. *Science*, 181:223–230, 1973.

P.P. Bonissone, R. Subbu, N. Eklund, and T.R. Kiehl. Evolutionary algorithms + domain knowledge = real-world evolutionary computation. *IEEE Trans. On Evolutionary Comput.*, 10 (5):256–280, 2006.

P. Bradley, K.M. Misura, and D. Baker. Toward high-resolution de novo structure prediction for small proteins. *Science*, 309:1868–1871, 2005.

J.C. Calvo and J. Ortega. Optimización multiobjetivo paralela en la predicción de estructuras de proteínas. In *XIX Jornadas de Paralelismo*, 2008.

J.C. Calvo and J. Ortega. Parallel protein structure prediction by multiobjective optimization. In *17th Euromicro International Conference on Parallel, Distributed and Network-Based Processing*, 2009.

J.C. Calvo, J. Ortega, and M. Anguita. Improving ab-initio protein structure prediction by parallel multi-objective evolutionary optimization. In *9th International Conference Computational and Mathematical Methods in Science and Engineering*, 2009a.

J.C. Calvo, J. Ortega, and M. Anguita. Alternativas de optimización multiobjetivo paralela en la predicción de estructuras de proteínas. In *XX Jornadas de Paralelismo*, 2009b.

J.C. Calvo, J. Ortega, M. Anguita, J.M. Urquiza, and J.P. Florido. Protein structure prediction by evolutionary multi-objective optimization: Search space reduction by using rotamers. In *10th International Work-Conference on Artificial Neural Networks (IWANN 2009)*, 2009c.

J.C. Calvo, J. Ortega, and M. Anguita. A hybrid scheme to solve the protein structure prediction problem. In *Advances in Bioinformatics, 4th International Workshop on Practical Applications of Computational Biology & Bioinformatics (IWPACBB 2010)*, pages 233–240, 2010a.

J.C. Calvo, J. Ortega, and M. Anguita. Aproximación híbrida paralela para la predicción de estructuras de proteínas. In *VII Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB2010)*, 2010b.

J.C. Calvo, J. Ortega, and M. Anguita. Comparison of parallel multi-objective approaches to protein structure prediction. *The Journal of Supercomputing*, 58:253–260, 2011a. DOI: 10.1007/s11227-009-0368-4.

J.C. Calvo, J. Ortega, and M. Anguita. Pitagoras-psp: Including domain knowledge in a multi-objective approach for protein structure prediction. *Neurocomputing*, 74:2675–2682, 2011b. doi:10.1016/j.neucom.2011.04.003.

J.C. Calvo, J. Ortega, M. Anguita, J. Taheri, and A. Zomaya. A method to improve the accuracy of protein torsion angles. *BIOSTEC 2011*, 2011c.

CASP. Casp: Protein structure prediction center. 2012. URL `http://predictioncenter.org`.

C. Coello, G. Lamont, and D. Veldhuizen. *Evolutionary algorithms for solving multiobjective problems*. SpringerLink e-books, 2007.

W.D. Cornell, P. Cieplak, C.I. Bayly, I.R. Gould, K.M. Merz Jr, D.M. Ferguson, D.C. Spellmeyer, T. Fox T, J.W. Caldwell, and P.A. Kollman. A second generation force field for the simulation of proteins, nucleic acids, and organic molecules. *J. Am. Chem. Soc.*, 117:5179–5197, 1995.

V. Cutello, G. Narcisi, and G. Nicosia. A multi-objetive evolutionary approach to the protein structure prediction problem. *J. R. Soc. Interface*, 3:139–151, 2006.

V. Cutello, G. Nicosia, M. Pavone, and J. Timmis. An immune algorithm for protein structure prediction on lattice models. *IEEE Trans. On Evolutionary Computation*, 11(1):101–117, February 2007.

R. Day, J. Zydallis, and G. Lamont. Solving the protein structure prediction problem through a multiobjective genetic algorithm. *Nanotech*, 2:32–35, 2002.

K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.

R.L. Dunbrack and F.E. Cohen. Bayesian statistical analysis of protein sidechain rotamer preferences. *Protein Sci*, 6:1661–1681, 1997.

C.M. Fonseca and P.J. Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, 1993.

N. Goldman, JL. Thorne, and D. Jones. Using evolutionary trees in protein secondary structure prediction and other comparative sequence analyses. *J. Mol. Biol.*, 263:196–208, 1996.

J. Handl, D.B. Kell, and J. Knowles. Multiobjective optimization in bioinformatics and computational biology. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 4(2):279–292, April 2007.

N. Hansen. The CMA evolution strategy: a comparing review. In J.A. Lozano, P. Larranaga, I. Inza, and E. Bengoetxea, editors, *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*, pages 75–102. Springer, 2006.

S. Kern, S.D. Müller, N. Hansen, D. Büche, J. Ocenasek, and P. Koumoutsakos. Learning probability distributions in continuous evolutionary algorithms–a comparative review. *Natural Computing*, 3(1):77–112, 2004.

Joshua Knowles and David Corne. The pareto archived evolution strategy : A new baseline algorithm for pareto multiobjective optimisation. *Proceedings of the Congress on Evolutionary Computation*, 1:98–105, 1999.

N. Krasnogor, W.E. Hart, J. Smith, and D.A. Pelta. Protein structure prediction with evolutionary algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference*, 1999.

J. Kubalik and J. Lazansky. Genetic algorithms and their tuning. *Computing Anticipatory Systems*, pages 217–229, 1999.

Arthur M. Lesk. *Introduction to Protein Architecture.* Oxford University Press, 2000. ISBN 0-19-850474-8.

Arthur M. Lesk. *Introduction to Bioinformatics.* Oxford University Press, 2002. ISBN 0–19–927787-7.

C. Levinthal. Are there pathways for protein folding? *Journal de Chimie Physique et de Physico-Chimie Biologique*, 65:44–45, 1968.

H. Maaranen, K. Miettinen, and M.M. Mäkelä. A quasi-random initial population for genetic algorithms. *Computers and Mathematics with Applications*, 47:1885–1895, 2004.

P. Moscato. Memetic algorithms: a short introduction. *New Ideas in Optimization*, pages 219–234, 1999.

S. Rahnamayan, H.R. Tizhoosh, and M.M.A. Salama. A novel population initialization method for accelerating evolutionary algorithms. *Computers & Mathematics with Applications*, 53:1605–1614, 2007.

S. Raman, R. Vernon, J. Thompson, M. Tyka, R. Sadreyev, J. Pei, D. Kim, E. Kellogg, F. DiMaio, O. Lange, L. Kinch, W. Sheffler, B. Kim, R. Das, N.V. Grishin, and D. Baker. Structure prediction for casp8 with all-atom refinement using rosetta. *Proteins: Structure, Function, and Bioinformatics*, 77(S9):89–99, 2009.

RCSB. Pdb (protein data bank). 2009. URL `www.rcsb.org/pdb/`.

C.A. Rohl, C.E.M. Strauss, K.M. Misura, and D. Baker. Protein structure prediction using rosetta. *Methods in Enzymology*, 383:66–93, 2004.

A. Roy, A. Kucukural, and Y. Zhang. I-tasser: a unified platform for automated protein structure and function prediction. *Nature Protocols*, 5:725–738, 2010.

K. Saif, B. Vinod, and P. Vandana. Maximizing the native concentration and shelf life of protein: a multiobjective optimization to reduce aggregation. *Applied Microbiology and Biotechnology*, 89(1):99–108, 2011.

M. Singh. *Predicting Protein Secondary and Supersecondary Structure.*, chapter 29. CRC Press, 2001. ISBN 0–8493–8597–0.

N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1994.

Z. Sun and B. Jiang. Patterns and conformations of commonly occurring supersecondary structures (basic motifs) in protein data bank. *Journal of Protein Chemistry*, 15(7), 1996.

M. Taufer, C. An, A. Kerstens, and CL. Brooks. Predictor@home: A 'protein structure prediction supercomputer' based on global computing. *IEEE Transactions on Parallel and Distributed Systems*, 17(8), 2006.

TINKER. Software tools for molecular design. 2004. URL `http://dasher.wustl.edu/tinker/`.

The UniProt. The universal protein resource (uniprot) 2009. *Nucleic Acids Res.*, 37:169–174, 2008.

H. Wang, Z. Wu, J. Wang, X. Dong, S. Yu, and C. Chen. A new population initialization method based on space transformation search. In *Fifth Int. Conference on Natural Computation*, pages 332–336, 2009.

J. Wang, P. Cieplak, and P.A. Kollman. How well does a restrained electrostatic potential (resp) model perform in calculating conformational energies of organic and biological molecules? *Journal of Computational Chemistry*, 21(12):1049–1074, 2000.

John C. Wooley and Yuzhen Ye. *A Historical Perspective and Overview of Protein Structure Prediction.* 2007. DOI: 10.1007/978-0-387-68372-0 1.

S. Wu, J. Skolnick, and Y. Zhang. Ab initio modelling of small proteins by iterative tasser simulations. *BMC Biol.*, 5, 2007.

A. Zemla. Lga: a method for finding 3d similarities in protein structures. *Nucleic Acids Research*, 31:3370–3374, 2003.

Y. Zhang. I-tasser: Fully automated protein structure prediction in casp8. *Proteins*, 59:100–113, 2009.

Y. Zhang and J. Skolnick. Spicker: Approach to clustering protein structures for near-native model selection. *Journal of Computational Chemistry*, 25:865–871, 2004.

E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999a.

E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–217, 1999b.

E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation Journal*, 8 (2):125–148, 2000.

E. Zitzler, M. Laumanns, and L. Thiele. Spea2: Improving the strength pareto evolutionary algorithm. *Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, pages 95–100, 2001.