Department of Computer Science
University of Copenhagen

Algorithms for Protein Structure Prediction

Ph.D. Thesis

Martin Paluszewski

October 2008

Acknowledgements

My acknowledgements go to all the people who I have been working with during this Ph.D.-study. This is especially my supervisor, Associate Professor Pawel Winter who also encouraged me to begin my Ph.D.-study. Special thanks go to Professor Kevin Karplus and the rest of his lab for making our collaboration in Santa Cruz possible. I would like to thank my research collaborators and paper co-authors Thomas Hamelryck and Rasmus Fonseca for valuable contributions and discussions. With the background in computer science, my knowledge of proteins was minimal before I began this study. Fortunately, my lovely wife Hanne Hammerbak has an educational background in biology and could answer most of my questions involving protein biology. Our wonderful sons Bjarke and Toke are my biggest motivation for doing research that hopefully will have a positive effect in their time.

Abstract

The problem of predicting the three-dimensional structure of a protein given its amino acid sequence is one of the most important *open* problems in bioinformatics. One of the carbon atoms in amino acids is the C_α-atom and the overall structure of a protein is often represented by a so-called C_α-trace.

Here we present three different approaches for reconstruction of C_α-traces from predictable measures. In our first approach [63, 62], the C_α-trace is positioned on a lattice and a tabu-search algorithm is applied to find minimum energy structures. The energy function is based on half-sphere-exposure (HSE) and contact number (CN) measures only. We show that the HSE measure is much more information-rich than CN, nevertheless, HSE does not appear to provide enough information to reconstruct the C_α-traces of real-sized proteins. Our experiments also show that using tabu search (with our novel tabu definition) is more robust than standard Monte Carlo search.

In the second approach for reconstruction of C_α-traces, an exact branch and bound algorithm has been developed [67, 65]. The model is discrete and makes use of secondary structure predictions, HSE, CN and radius of gyration. We show how to compute good lower bounds for partial structures very fast. Using these lower bounds, we are able to find global minimum structures in a huge conformational space in reasonable time. We show that many of these global minimum structures are of good quality compared to the native structure. Our branch and bound algorithm is competitive in quality and speed with other state-of-the-art decoy generation algorithms.

Our third C_α-trace reconstruction approach is based on bee-colony optimization [24]. We demonstrate why this algorithm has some important properties that makes it suitable for protein structure prediction.

Our approach for model quality assessment (MQA) [64] makes use of distance constraints extracted from alignments to templates. We show how to use CN probabilities in an optimization algorithm for selecting good distance constraints and we introduce the concept of non-contacts. When comparing our algorithm with state-of-the-art MQA algorithms on the CASP7 benchmark, our algorithm is among the top-ranked algorithms. We are currently participating in CASP8 MQA with this algorithm.

Preface

I began my Ph.D.-studies in June, 2005 at the Department of Computer Science, University of Copenhagen under supervision of Associate Professor Pawel Winter. At that time our algorithms and optimization group had much experience with network problems, production planning, packing problems and transportation problems, but little experience with computational biology. We therefore decided, to begin attacking problems in computational biology using our expertise for solving complex optimization problems. One of the main purposes of my Ph.D.-study therefore was to identify suitable problems in the field of protein structure prediction and get our group involved in the field. In 2007 I was working together with professor Kevin Karplus for 6 months. He is the head of the protein structure prediction group at the University of California, Santa Cruz (UCSC). During that period, I learned much about many of the problems that exist in the field of protein structure prediction and I was introduced to the field of protein model quality assessment.

General Outline

This thesis summarizes the research I have been involved with during my Ph.D.-study. It consists of papers together with a text describing the background of our work.

The most important parts of this Ph.D.-thesis are the papers included in Chapters 9 – 12. They contain detailed descriptions of the algorithms we have developed and all our research results. For people in the field of protein structure prediction, no prerequisites for reading the papers should be needed. As is the case with most scientific papers, they are written *by* experts in the field *to* other experts in the field. The introduction and background text here (pages 9 to 95) is therefore aimed at scientists and students with little or no background in bioinformatics. I therefore also allow myself to be less formal in this text. It does not contain the details of our research and should therefore be weighted less than the papers in the evaluation of this thesis.

Most chapters contain an introductionary description of the topic and a few illustrative examples of important results in the literature. The chapters also contain sections called *Our Research*, which describe how we apply the concepts and results of the given chapter in our research.

Contents

The following introduction text and papers constitute this thesis:

1. **Introduction and Background.** Chapters 1 – 8
   M. Paluszewski.
2. **Chapter 9**. Paper:
   M. Paluszewski, T. Hamelryck, and P. Winter. Reconstructing Protein Structure From Solvent Exposure using Tabu Search. *Algorithms for Molecular Biology*, 1:20+, October 2006.
   Status: Published
3. **Chapter 10**. Paper:
   M. Paluszewski and P. Winter. Protein Decoy Generation using Branch and Bound with Efficient Bounding. *Proc. of 8th International Workshop on Algorithms in Bioinformatics, WABI*, 2008.
   Status: Published
4. **Chapter 11.** Paper:
   M. Paluszewski and K. Karplus. MQA using Distance Constraints from Alignments. *Proteins, Structure, Function and Bioinformatics, (accepted)*, 2008.
   Status: To appear
5. **Chapter 12.** Paper:
   R. Fonseca, M. Paluszewski, and P. Winter. Protein Structure Prediction using Bee Colony Optimization Metaheuristic (draft).
   Status: Work in progress

The appendix contains the following technical report, extended abstract and posters:

1. **Appendix A.** Paper:
   M. Paluszewski and P. Winter. EBBA: Efficient Branch and Bound Algorithm for Protein Decoy Generation. *Department of Computer Science, Univ. of Copenhagen*, 08(08), 2008.
   Status: Published
2. **Appendix B.** Extended Abstract:
   R. Fonseca, M. Paluszewski, and P. Winter. Protein Structure Prediction using Bee Colony Optimization Metaheuristic. *Meta'08.*
   Status: To appear
3. **Appendix C.** Poster:
   M. Paluszewski, T. Hamelryck, and P. Winter. Protein Structure Prediction using Tabu Search and Half-sphere-Exposure Measure. *RECOMB (poster)*, 2006.
   Status: Abstract published
4. **Appendix D.** Poster:
   M. Paluszewski and P. Winter. Branch and Bound Algorithm for Protein Structure Prediction using Efficient Bounding. *RECOMB (poster)*, 2007.
   Status: Abstract published

Contents

Chapter 1

Introduction

A protein is a complex molecule consisting of thousands of atoms that interact with each other and with surrounding molecules. Proteins are very important molecules in all living organisms and are often referred to as *the molecules of life*. Knowing the native structure of proteins is fundamental for our understanding of their functionality, since protein structure directly determines protein function. Today, the three-dimensional structures of proteins are found using X-ray crystallography or *nuclear magnetic resonance* (NMR) experiments. These methods are quite expensive and they can be very time consuming. Furthermore, these methods cannot be be applied to all proteins, especially many of the membrane proteins [93, 59].

1.1 Protein Structure Prediction is an Open Problem

One of the most important *open* problems in bioinformatics is therefore to find an algorithm that takes an amino acid sequence as input and outputs the native structure of the protein (i.e., the three-dimensional coordinates of all atoms) as illustrated in Figure 1.1. This problem is called the *protein structure prediction* problem. A closely related problem is the *protein folding* problem, which is concerned with the prediction of the atomic positions during the actual folding of the protein. A solution to the protein folding problem is therefore also a solution to the protein structure prediction problem and is therefore considered to be a much harder problem. Even though scientists have been trying to solve the protein structure prediction problem since the 1960's, no algorithm is able to predict the structure of proteins in general (in reasonable time). The database of amino acid sequences with known structures (PDB) is growing rapidly. So, algorithms that are based on so-called homology modeling are often able to predict the structure of proteins that have a homolog counterpart in the database. In general, however, we cannot assume that a protein has a homolog counterpart with known structure and so-called *ab initio* or *de novo* algorithms are therefore trying to use more fundamental properties that do not require the specific knowledge of other proteins.

Progress in the field of protein structure prediction will also have a great medical relevance. If we learn how the amino acid sequence is related to the

native protein structure, engineers could design new enzymes working as drugs for various diseases [85]. Protein design is in principle the reverse of protein structure prediction.

In this thesis, several classical algorithms for protein structure prediction and model quality assesments are described together with our latest research.

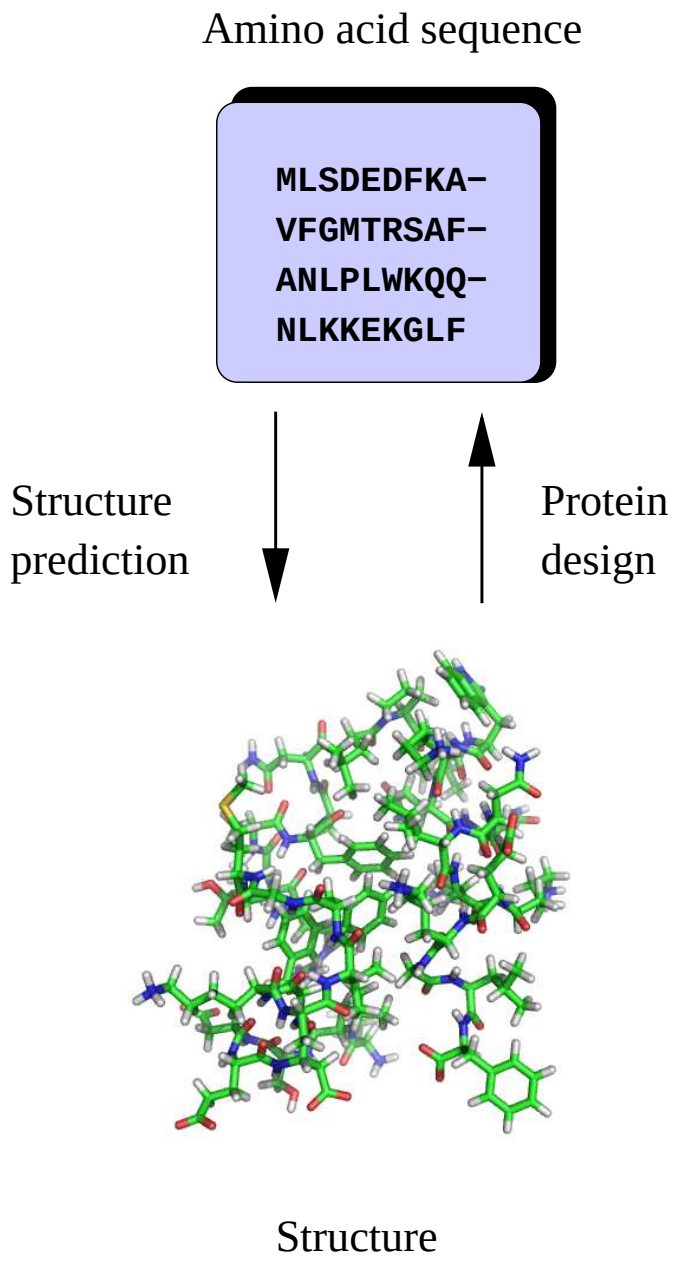


Figure 1.1: Protein structure prediction and protein design.

Chapter 2

Proteins

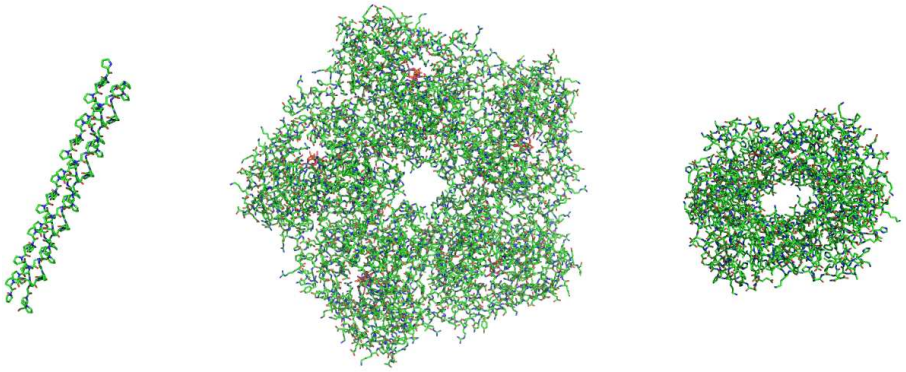
2.1 Categories of Proteins

The diversity of protein functionality is very large and proteins are typically grouped into three categories:

- **Structural proteins.** These are building blocks in skin, hair, nails, muscles etc. An example is the protein *collagen*, Figure 2.1(A), which is bundled in collagen fibers and is the main component in skin, bones and teeth.
- **Enzymatic proteins.** These proteins are catalysts in chemical reactions. An example is the enzyme *glutamine synthetase*, Figure 2.1 (B), which plays an important role in the metabolism.
- **Functional proteins.** These proteins can be thought of as small machines with a very specific task. An example is the protein *hemoglobin*, Figure 2.1 (C), which is responsible for transportation of oxygen in blood.

2.2 Protein Synthesis

A protein is a chain of smaller molecules, called amino acids. Proteins consists of 20 standard amino acids, but there are more of them in nature. Proteins are distinguished by their specific sequence of amino acids. The chain of amino acids is assembled in the living cell - this process is called *protein synthesis*. As illustrated in Figure 2.2, protein synthesis begins in the nucleus with transcription. Here, one part of the *deoxyribonucleic acid* (DNA) strand is copied and encoded into a molecule called messenger *ribonucleic acid* (mRNA). The mRNA is then transported out of the nucleus membrane and translated into a chain of amino acids by the ribosomes in the cytoplasm. The translation of mRNA nucleic acids into amino acids is specified by rules called the *genetic code*. There are four different nucleotides, so mRNA (and DNA) can be represented by strings of an alphabet of 4 letters. Usually A,G,C and T for DNA and A, G, C and U for mRNA. The genetic code is a mapping from every triplet of nucleotides -



(A) (B) (C)

Figure 2.1: (A) Collagen fibers (structural protein). (B) Glutamine synthetase (enzyme). (C) Hemoglobin (functional protein).

giving 64 combinations - into one of the 20 amino acids as illustrated in Figure 2.3. The construction of the actual amino acid chain, also called the polypeptide chain, occurs when neighbouring amino acids react and create peptide bonds (see Figure 2.4).

2.3 Protein Structure

The atoms in the peptide bonds are fixed in a plane called the amid plane. So, the main flexibility of the polypeptide chain, comes from the backbone bonds connected to each C_α atom (see Figure 2.5). A proteins degree of freedom is not only given by the ϕ and ψ angles - most of the side chains also have bonds with several possible dihedral angles. However, modifications in the side chains are local compared to the ϕ and ψ angles which describe the overall path of the backbone chain.

It is generally believed that already during translation the polypeptide chain begins to fold. When the polypeptide chain folds, the amino acids move according to the degree of freedom, such that the Gibbs free energy of the system is minimized. This is described in more detail in section 2.4. In most cases, the chain folds to the *native* structure in the order of milliseconds [12].

2.3.1 Levels of Protein Structure

It is the atomic configuration of the native structure that determines the properties and functionality of a protein. Protein structure is typically described in four levels:

- **Primary structure.** The polypeptide chain is made from amino acids in the ribosomes and the primary structure of a protein describes this

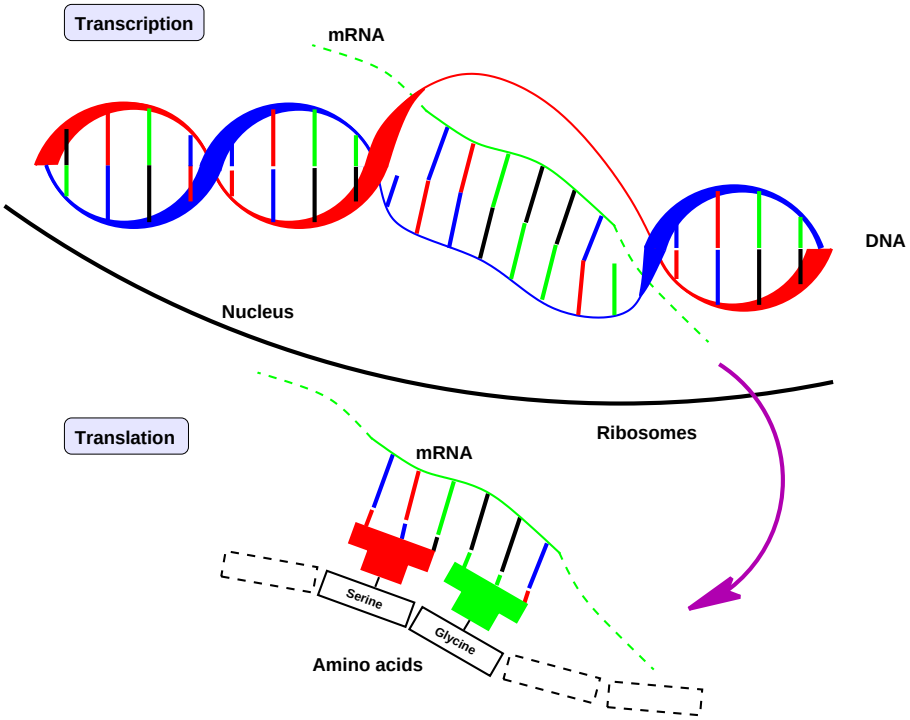


Figure 2.2: Protein synthesis begins with transcription, where a part of DNA is copied to a complementary mRNA molecule. The mRNA molecule is then used in the translation phase where the polypeptide chain is constructed.

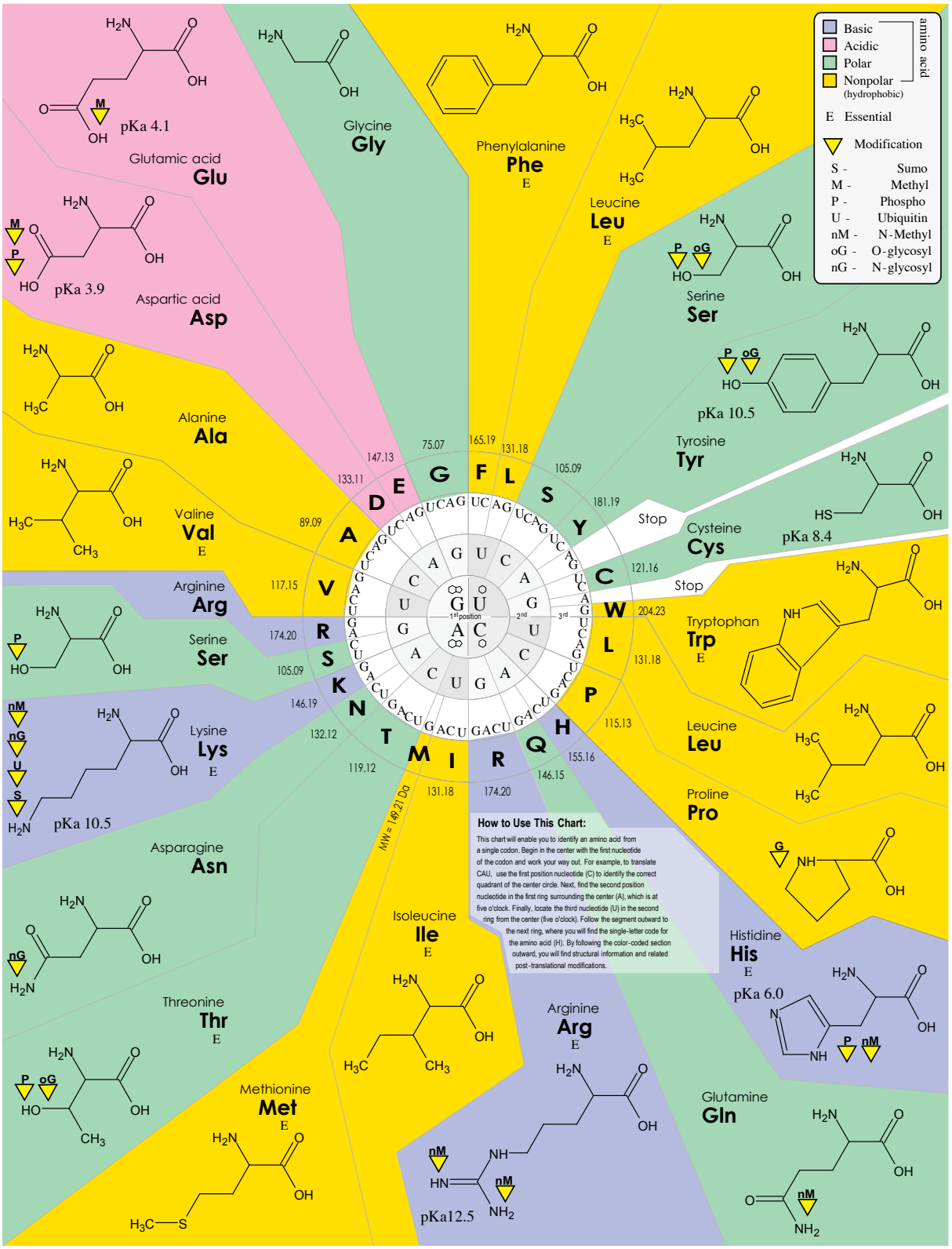


Figure 2.3: Illustration of the genetic code. Image from Wikipedia (http://en.wikipedia.org/wiki/Image:GeneticCode21.svg).

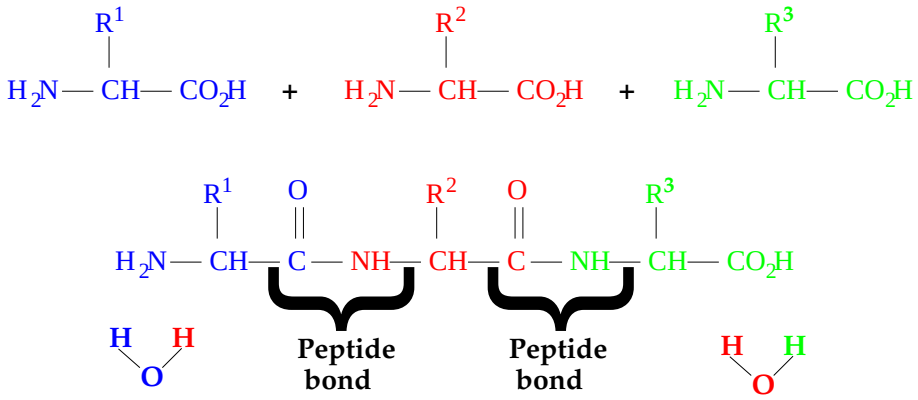


Figure 2.4: Three amino acids are joined by two peptide bonds. In this reaction, two water molecules are released. The R^i-molecules correspond to the side-chains of the amino acids.

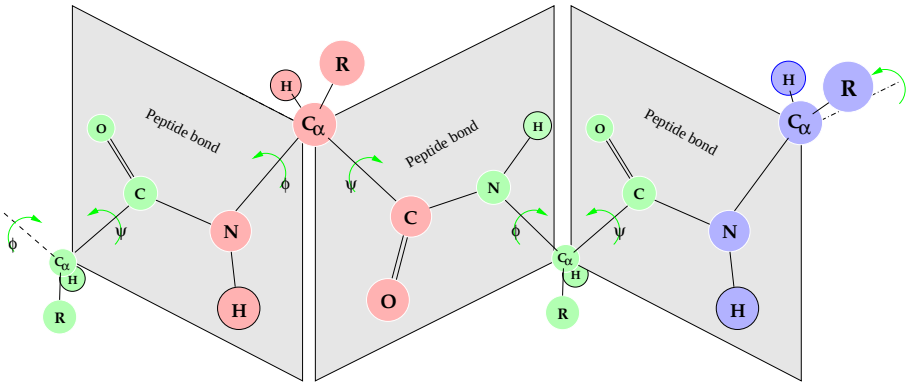


Figure 2.5: Structure of the polypeptide backbone. The backbone atoms in a peptide bond are all fixed in a plane and these planes can *rotate* according to the ϕ and ψ angles. ϕ is the dihedral angle between N and C_α and ψ is the dihedral angle between C and C_α. Each internal amino acid therefore contributes with two degrees of freedom and the end-chain amino acids contribute with one degree of freedom. The additional degree of freedom of the amino acid side-chains depend on the amino acid type.

sequence of amino acids. The primary structure starts at the N-terminal.

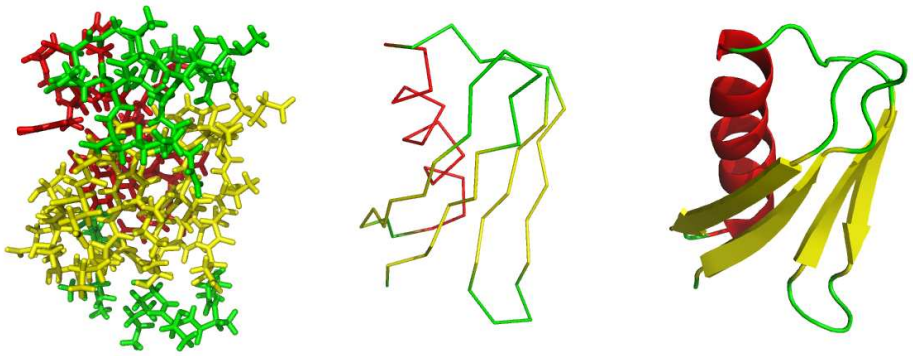
- **Secondary structure.** Some local structure patterns are very often observed in proteins because they lead to stable, low energy, structures. The most frequent of those are the *alpha helices* and the *beta sheets*, but loops and other helices are usually also considered to be secondary structure. The local structure of a backbone that is not a secondary structure element is called a *random coil*. In Figure 2.6 three different visual representations of *protein G* is shown. Protein G contains one alpha helix and one beta sheet with 4 beta strands. It is difficult to see these secondary structures when all protein bonds are printed (A), however in Figure 2.6(B) and Figure 2.6(C) only the backbone atoms are printed and the secondary structure patterns appear more clearly. The arrangement of a specific secondary structure combination is called a *motif* (i.e. sheet, turn, sheet) and a more general description of a secondary structure arrangement is called a *fold*. An example of a typical fold is the *beta-barrel* as illustrated in Figure 2.7.

- **Tertiary structure.** This is the full description of a folded polypeptide chain. In principle the tertiary structure is the 3D-coordinates of the atoms in the native structure of the protein. However, proteins are often not completely stable, so these 3D-coordinates often correspond to the most observed state or the crystallized state of the protein.

- **Quarternary structure.** Some proteins consist of several polypeptide chains which are assembled in a more complex molecule (often called a protein complex). The description of how these folded polypeptide chains are assembled is called the quarternary structure of the protein.

2.4 Thermodynamic Hypothesis

In the late 50' and early 60' Anfinsen et al. [5] studied what happens to the protein ribonuclease when it is first denatured (unfolded) and thereafter renaturated (folded). They observed that different conformations of the unfolded polypeptide chain of ribonuclease always fold to the same native state and they postulated the *thermodynamic hypothesis*. It states that the native state of a protein, in its normal environment, is the structure with lowest Gibbs free energy. This property is fundamental for the understanding of protein folding and is why it is believed that the native state of a protein can be predicted just from the knowledge of its amino acid sequence. Note that the thermodynamic hypothesis has been verified on many different proteins later, however it is observed that some proteins receive help to fold from specialized proteins called chaperones [21]. It is still discussed whether or not the existence of chaperones conflicts with Anfinsens thermodynamic hypothesis.



(A) (B) (C)

Figure 2.6: 3 different visualizations of *protein G*. (A) Sticks, (B) Backbone trace, (C) Cartoon. The sticks visualization shows a *stick* between pairs of bonded atoms. The backbone visualization shows lines between atoms on the backbone. The cartoon visualization clearly shows helices strands and coils in a stylistic figure.

2.5 Protein Folding

If we assume that Anfinsens thermodynamic hypothesis is valid, then proteins contain everything that is needed to fold to the structure with minimum free energy - the native structure. When the chain folds, it must undergo changes mainly in the neighbouring bonds of the C_α-atoms but it is not known in details how these changes occur. The modifications of the chain over time correspond to some path in the energy landscape (Figure 2.8). However, it is not known how many folding pathways exist for a given protein and how these pathways are found by the protein.

2.5.1 Levinthal's Paradox

In 1969 Levinthal [51] argued that proteins could not use a completely random search or exhaustive search since the number of possible conformations of an amino acid chain is astronomical high. This argument is called *the Levinthal paradox* even though it was never believed that proteins actually do fold using completely random search or exhaustive search. Instead, it is very likely that the energy landscape is funnel-like such that the forces acting on an arbitrary unfolded chain, eventually move the atoms in their native positions without having to cross very large energy barriers [19].

2.5.2 Properties of the Energy Landscape

If the thermodynamic hypothesis is perfectly true, then the energy landscape must have the following properties:

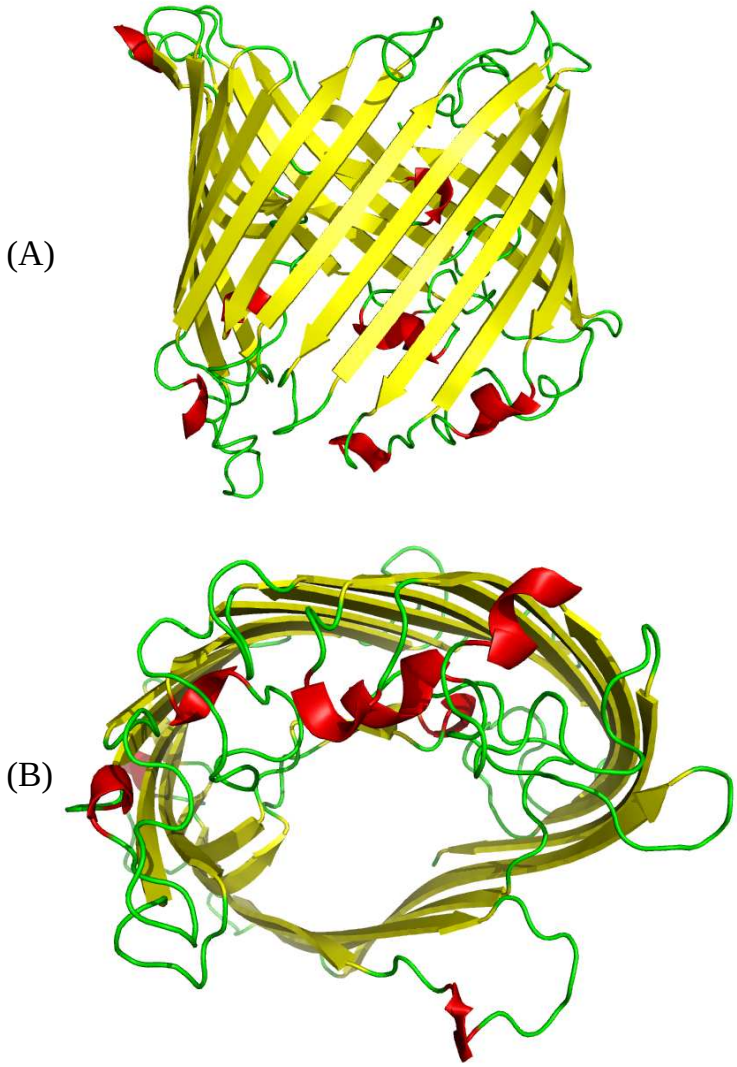


Figure 2.7: The Figure shows one chain of the sucrose-specific porin (PDB: 1A0S). (A) shows the typical beta-barrel fold where beta-strands are arranged antiparrallel. (B) the protein has a hollow center like a barrel.

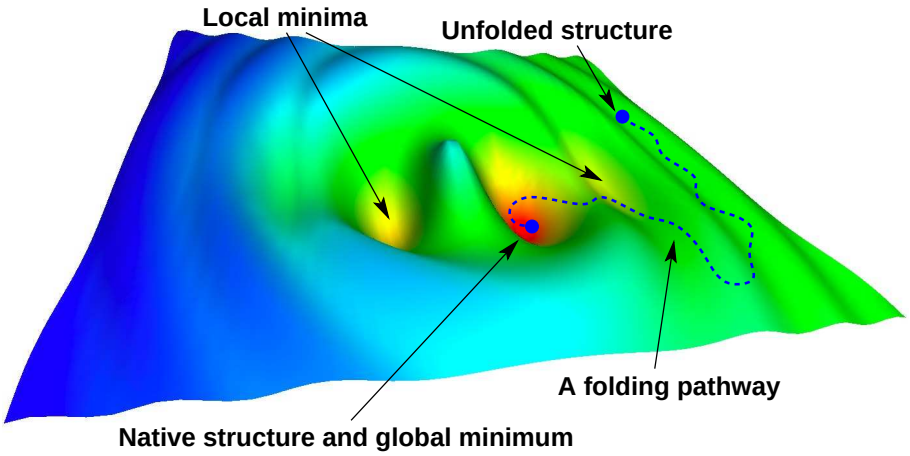


Figure 2.8: Illustration of an energy landscape and a folding pathway. For proteins, the energy landscape corresponds to a multi-dimensional hypersurface that depends on the positions of the atoms. Each point in the energy landscape therefore corresponds to a conformation with an associated energy.

1. **Uniqueness.** There should be only one structure with minimum free energy. Otherwise a polypeptide chain could have several native structures.
2. **Stability.** Small changes in the structure should not give rise to large energy changes. It is known that proteins often fluctuate around the native state. This fluctuation would require much energy if there were large energy barriers around the native structure.
3. **Accessibility.** The path from an unfolded state to the native state should not contain very large barriers. It is assumed that virtually all unfolded states of the polypeptide chain are able to reach the native state. Crossing very large barriers require much energy and it would be difficult, or impossible, for the polypeptide chain to reach the native state.

In Figure 2.9 an illustration of a protein folding pathway is shown. The figure shows 6 snapshots from an unfolded state to the native structure of the protein.

2.6 Chapter Summary

Proteins are complex molecules that exist in all living organisms. They are constructed in the cell in a process called *protein synthesis*. Proteins are chains of smaller molecules called amino acids. There are 20 different amino acids in nature and it is the sequence of amino acids of the protein that eventually determines the shape and functionality of the protein. Typically, in the order of milliseconds, the chain of amino acids folds to the native structure of the protein. It is believed that the native structure has minimum free energy. It is not known in details how proteins move from an unfolded structure to the

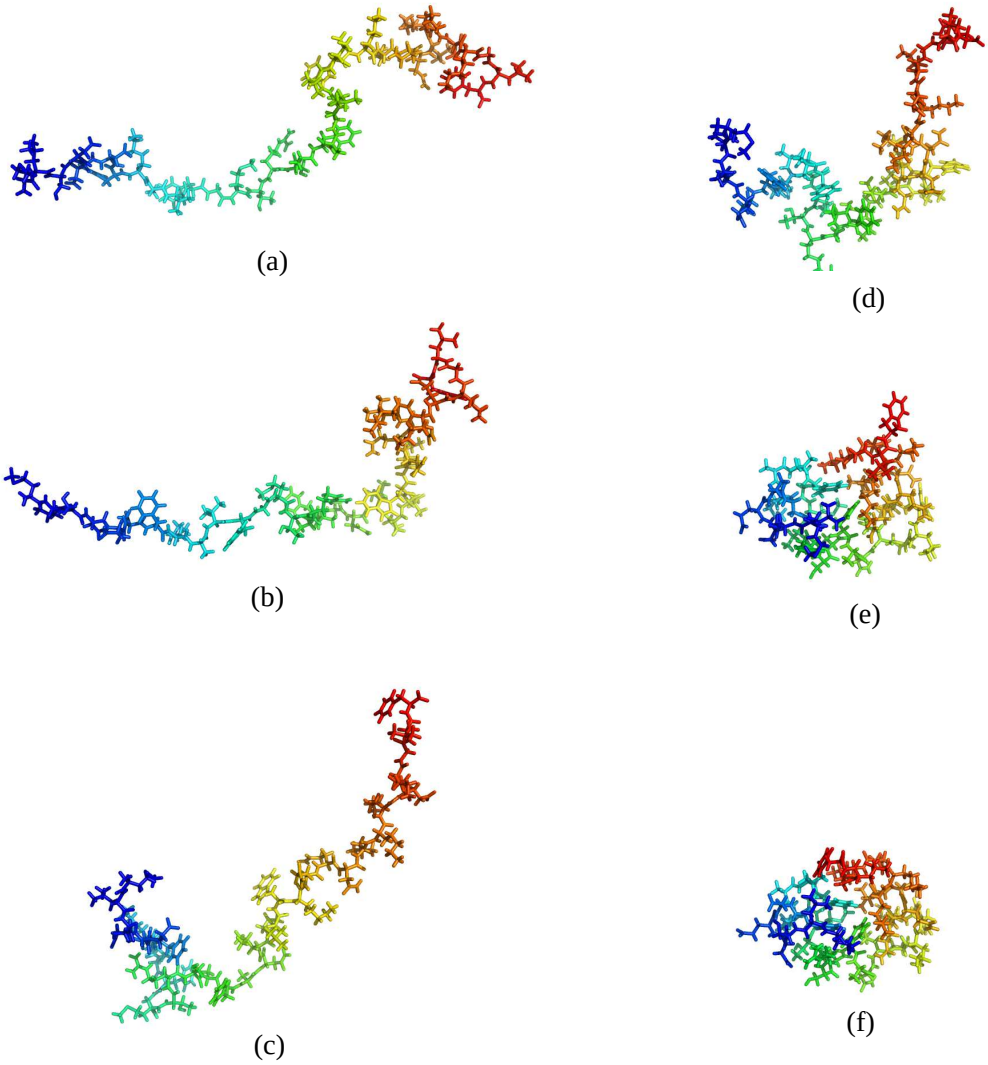


Figure 2.9: What folding of Villin Headpiece might look like in 6 snapshots. From an unfolded state (a) to the folded native structure of Villin Headpiece (f). *Note that these structures are found by simulating an unfolding process from (f) to (a) and are therefore not snapshots of a real protein folding. The structures are therefore just a qualified guess of how the steps of a protein folding could look like. Simulation data is provided by Kresten Lindorff-Larsen.*

native structure, but it is believed that the forces of nature create a funnel-like energy landscape.

Chapter 3

Protein Structure Prediction - An Introduction

In the previous chapter, we described the experiments by Anfinsen and argued that all information needed for folding a protein is contained in the amino acid sequence. In the right environment, the chain of amino acids can therefore be considered as a self-assembling machine. When all information needed is stored in the amino acid sequence, a compelling idea is of course to write an algorithm that takes the amino acid sequence as input and outputs the tertiary structure of the protein. In this chapter, we briefly introduce the various concepts of protein structure prediction and describe the two major problems that we are faced with. This chapter also describes various methods for evaluation of prediction quality. The next chapters 4 and 5 describe in more detail secondary structure prediction and tertiary structure prediction respectively.

3.1 Protein Folding Problem

There are basically two very different approaches for computing the native structure of a protein. One approach is to mimic nature such that the actual folding pathway is computed. The idea is to start with an unfolded chain and apply a physics based energy function (Figure 3.1) such that the atoms move according to Newton's laws of motion. If this can be done with enough accuracy and speed, the simulation would eventually reach a structure similar to the native structure of the protein. This problem is called *the protein folding problem* and is often studied using molecular dynamics. A solution to the protein folding problem would therefore give both the atomic pathways of folding and the native structure of the protein. The main difficulty with the naïve molecular dynamics approach is that the forces acting on the atoms are not constant, they depend on the positions of all atoms. All simulations using this approach are therefore approximations of the real pathway. Experiments show that it requires extremely small time steps to give a realistic simulation and small timesteps require many computations of the forces acting on the atoms. The longest protein that has been correctly folded using molecular dynamics is the Villin Headpiece. It has 36 residues, and using the *folding@home* massively distributed program,

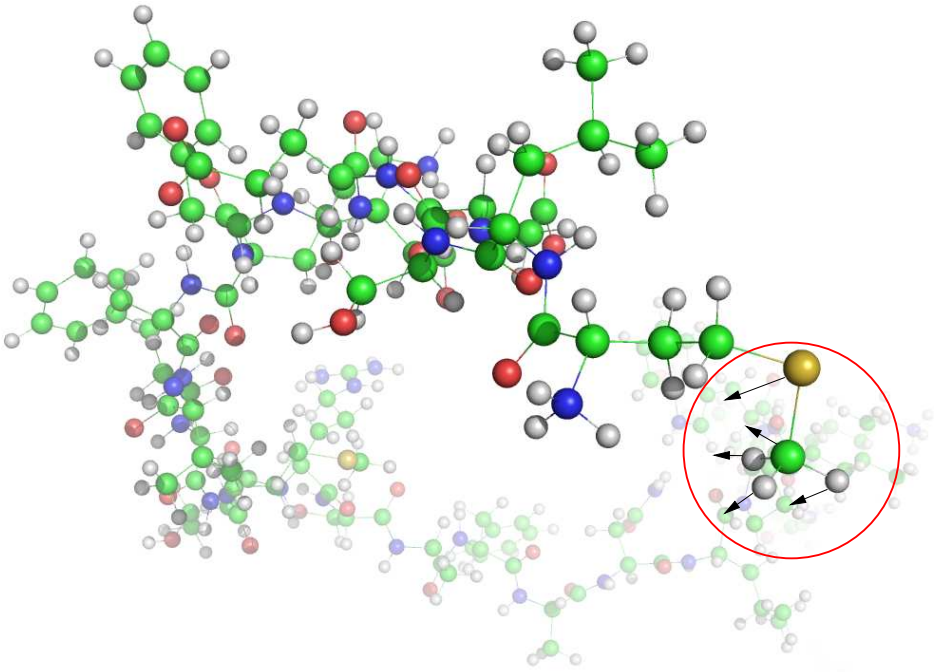


Figure 3.1: The forces acting on the atoms are computed and the positions of the atoms are moved accordingly.

the Pande group was able to simulate 500 μs of folding using 200.000 CPU's[32]. With exceptions of the smallest peptides, this approach is therefore currently not feasible.

3.2 Protein Structure Prediction

Fortunately, for most applications the knowledge of the native structure is more important than the folding pathway. Another main approach is therefore to disregard the actual folding pathway and predict the native structure from amino acid sequence using another method than nature does. This problem is called *the protein structure prediction problem*. Both of these problems are still unsolved. The main reasons for this are the following two obstacles.

3.3 Major Obstacle 1: Energy Function

We currently believe that in nature, the motion of the atoms are described by *quantum mechanical* (QM) rules. The most accurate and obvious choice of energy function would therefore be a QM-based energy function. However, even for small molecules (like a single amino acid) the QM-based energy is very difficult and time demanding to compute. Since proteins are very large molecules, the use of purely QM-based energy functions are currently not feasible. Another approach is therefore to use energy functions that are easier to compute and to

some extend approximate the real QM energy of the protein. Many of the popular approximate energy functions used today are weighted functions of several energy terms, i.e. bond energies, electrostatic forces, van der Waals forces etc. Each of these functions depends on pairs of bonded or non-bonded atoms. These functions are of course fast to compute, but it is a crude assumption that the energy of a protein can be described as a sum of functions that only depend on pairs of atom positions.

The side chains of amino acids are either *hydrophobic* or *hydrophilic* because of their polarity. Consequently, since many proteins are water soluble the main forces in protein folding are concerned with the hydrophobic packing of the protein core. Water soluble proteins therefore tend to have the hydrophobic (nonpolar) amino acids buried in the core of the protein where they are isolated from water. Even though the hydrophobic/hydrophilic forces are consequences of the natural QM energy function, they are often handled by rewarding hydrophobic core packing explicitly [48].

3.4 Major Obstacle 2: The Conformational Space

The conformational space of a protein is huge. In Section 2.3 we showed that each amino acid in the chain gives rise to two degrees of freedom (not counting the degrees of freedom of the side-chain). So, even crude discretizations of the degrees of freedom result in an astronomical large conformational space. Finding the structure with minimum free energy can therefore not be done using complete enumeration. However, in [67] (described in more details in Section 7.5.2), we show that it is possible to exploit the structure of our energy function to efficiently bound large regions of the conformational space. We therefore consider these two obstacles to be intertwined.

3.5 Other Structure Prediction Problems

It is easier to predict the *secondary structure* of a protein compared to the *tertiary structure*. The reason for this is that secondary structure is mainly considered to be a local property of the chain. As described in the next chapter, neural networks using a local window of amino acids are successful in predicting secondary structures. Another category of structure prediction is the side-chain positioning problem [22] (Figure 3.2). This problem is also easier than the tertiary structure prediction. The main reason is that only a few configurations of each side-chain are energy favorable. These configurations are called rotamers and algorithms like SCWRL[14] are able to assign rotamers to the side-chains of a backbone with high accuracy in reasonable time.

3.6 Methods for Evaluation of Predictions

To date, no algorithm is able to *exactly* compute the secondary, tertiary or quarternary structure from the primary structure. Nevertheless, algorithms can predict these structures with various degrees of accuracy. In the literature,

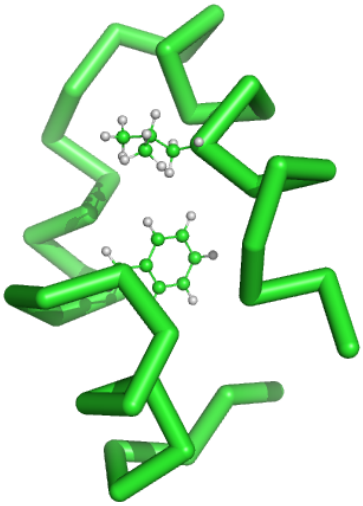


Figure 3.2: In the side-chain positioning problem, the whole backbone is fixed and the problem is to assign the correct orientation (rotamer) of the side-chains. Here two of the side-chains with correct rotamers are shown.

different measures for evaluation of prediction quality have been proposed. Here we describe two measures for secondary structure prediction evaluation (Q_3, CC) and three measures for tertiary structure prediction (RMSD, GDT and AC). In all cases, these measures quantify the similarity between two structures in a single number.

3.6.1 Q_3

A simple and widely used measure for secondary structure prediction quality is the Q_3 score [9]. The Q_3 score of a secondary structure classification is simply the percentage of correctly predicted assignments of secondary structure. Figure 3.3 shows an example of a secondary structure prediction being compared with the exact secondary structure derived from the native structure. The correct assignments are highlighted. While the Q_3 score is easy to compute and interpret, it is not always the best indicator of the prediction quality. Most proteins contain more coil structure than helical and sheet structure and an overprediction of coil is therefore not penalized properly. A prediction algorithm that only predicts coil would, on average, achieve a higher Q_3 score than a prediction algorithm that only predicts helices which is not reasonable.

3.6.2 CC

A perhaps better method for evaluation of secondary structure prediction is the correlation coefficient (CC) [56]. The CC is a number between -1 and 1 and is defined as follows.

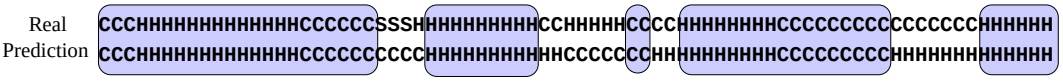


Figure 3.3: An example of a secondary structure prediction together with the exact secondary structure derived from the native state of the protein. H corresponds to *helix*, S corresponds to *sheet* and C corresponds to *coil*. The Q_3 score in this example is $56/76 \simeq 74\%$. The shown example is the secondary structure prediction of Calbindin (4ICB) using PSIPRED [57]. This prediction is used in our algorithms [67, 66, 24] described in Chapter 7.

$$CC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FN)(TP + FN)(TN + FP)(TN + FN)}}$$

where TP is the number of true positives, TN is the number true negatives, FP is the number of false positives and FN is the number of false negatives. If we want to measure the CC of helical prediction, TP would correspond to the number of correctly predicted helices, TN the number of correctly predicted non-helices, FP the number of wrongly predicted helices and FN the number of wrongly predicted non-helices. A CC of -1 means that all predictions are wrong. A CC of zero means that the prediction is 'random' and a CC of 1 is a perfect prediction.

3.6.3 RMSD

It is often important to measure the similarity between two protein backbones. In some search algorithms we only want to consider protein backbones that are different to some extent [63] or perhaps we want to compare a predicted structure with the native structure. The *Root Mean Square Deviation* (RMSD) measure is often used for backbone prediction quality. In literature, two versions of RMSD are generally used:

- Coordinate RMSD is

$$CRMSD = \sqrt{\frac{\sum_{i=1}^{n} |\bar{a_i} - \bar{b_i}|^2}{n}}$$

  where a and b are two vectors of coordinates (usually C_α-coordinates) that should be compared. CRMSD can be computed for all positions of a and b in space, however CRMSD is usually computed for the optimal superposition between a and b. The task of finding an optimal superposition of two point sets can be tackled by various techniques. One technique used in many implementations is the quaternion method described by S. Kearsley [39]. It runs in $O(n)$ time where n is the length of the vectors being compared.

- Distance RMSD is

$$DRMSD = \sqrt{\frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(|\bar{a_i} - \bar{a_j}| - |b_i - b_j|)^2}{(n(n-1))/2}}$$

  This definition of DRMSD does not depend on the relative positions of a and b, and it is therefore not necessary to compute the optimal superposition.

3.6.4 GDT

It has long been known by CASP organizers and CASP assessors that RMSD is not always well-suited for evaluation of protein structure predictions, especially long and difficult targets. The reason is that RMSD is a global measure that does not reward partially correct models. To illustrate the problem with RMSD, consider a structure where 70% of the first residues can be aligned with the target and the last 30% is positioned completely different from the target (Figure 3.4). Often such a prediction is considered to be a good prediction (if no close homologoues are known), however, RMSD does not reflect this very well. The Global Distance Test (GDT) [97] measure was proposed in 1999, and it avoids this problem. Basically, GDT computes the percentage of residues having C_α-atoms that are within a certain distance cutoff from the target in an optimal superposition. With an appropriate distance cutoff, GDT would therefore be 70 in the example illustrated in Figure 3.4. By varying the cutoff-value, a very descriptive plot of structural similarity can be generated as in the example shown in Figure 3.5. It is often not trivial to decide what the distance cutoff should be when GDT is used as a score function. The GDT score is therefore often reported using an average of different cutoff distances; typically 1, 2, 4 and 8 Å.

3.6.5 AC

None of the measures described above consider the positions of side chains. For most applications, it is more important to correctly predict the overall backbone path i.e. C_α-trace than correctly predict the positions of side chains. In our paper *Reconstructing protein structure from solvent exposure using tabu search* [63] we introduced a very simple measure for evaluating the implicit directions of side chains. Given only the C_α-trace, we wanted a simple and fast measure to evaluate if side-chains eventually would be pointing out of the protein (surrounded by water) or pointing towards the interior of the protein. We came up with the following *angle correlation* (AC) that has the following definition

$$AC = \frac{\sum_{i=1}^{n}\theta(\bar{a_i}, \bar{b_i}) - \theta(\bar{c_i}, \bar{d_i})}{n}$$

where $\bar{a_i}$ is the vector pointing from the i'th C_α-atom to the geometric center and $\bar{b_i}$ is the vector pointing in the direction of the i'th C_α-atoms side chain (as defined in Figure 3.6). Vectors c_i and d_i are the corresponding vectors

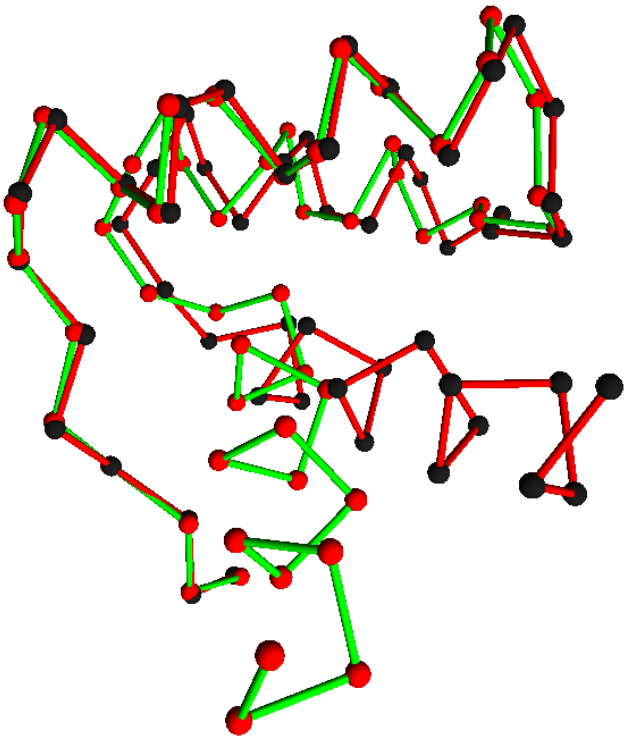


Figure 3.4: Two C_α-traces have almost equal positions of C_α-atoms when the first 70% of the residues are compared. This gives a GDT$\simeq$70 even for small cutoff distances. The RMSD of the best superposition of all C_α-atoms is 3.1. If RMSD is measured for the 70 % of the first residues only, then RMSD would be 0.3.

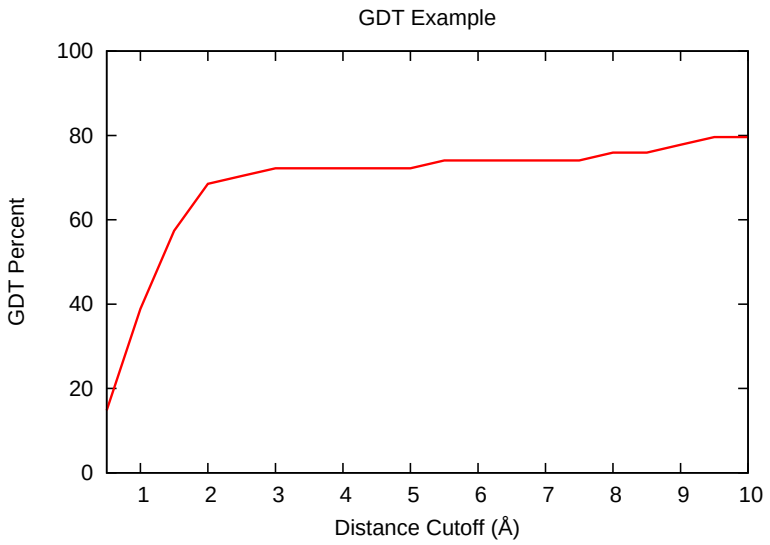


Figure 3.5: GDT is computed for various distance cutoffs. The plot shows that for cutoff distances (>2 Å) the similarity between the structures is almost stable on 70% in this example.

in the native structure. θ measures the angle between the two vectors. Zero AC is perfect correlation, 90° is random correlation and 180° is perfect 'anti'-correlation. There are many examples where a structure can have a good RMSD and a bad AC (or the opposite). This can be problematic when side-chains are positioned on structures that we only selected because of good RMSD or good GDT. In these cases, side-chain positioning might be impossible or perhaps give a wrong full tertiary structure. Figure 3.7 shows a typical RMSD vs. AC plot from [63]. In Figure 3.8 superpositions of two structures with the native structure is shown. Both structures have good RMSD but structure (a) has bad AC and structure (b) has good AC.

3.7 Critical Assessment of Techniques for Protein Structure Prediction (CASP)

Critical Assessment of Techniques for Protein Structure Prediction (CASP) [60] is an experiment that determines the current state of art for protein structure prediction algorithms. Every two years the CASP organizers collect unpublished protein structures from crystallographers. The main idea is therefore to blind test current algorithms for protein structure prediction on these *secret* protein structures. The predictors (participants) of CASP therefore only know the amino acid sequences of these proteins and are asked to predict the native structure using their algorithm. When the predictions have been submitted to the CASP organizers, the secret protein structures are revealed and the different algorithms are evaluated.

There are different evaluation categories at CASP. The one just described

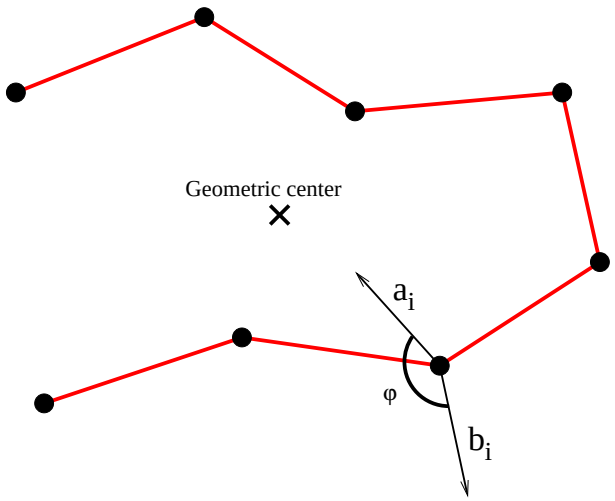


Figure 3.6: An illustration of the vectors involved in the computation of AC.

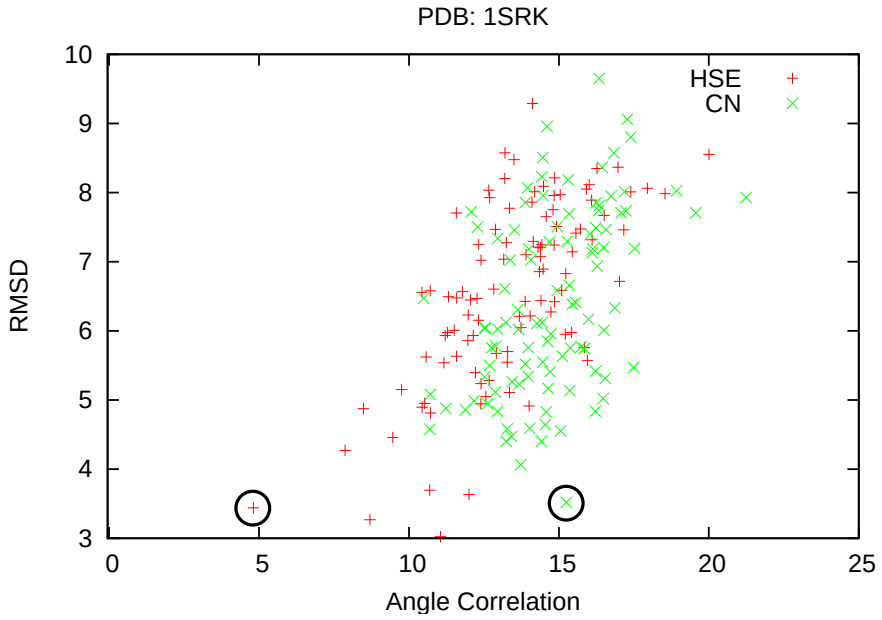
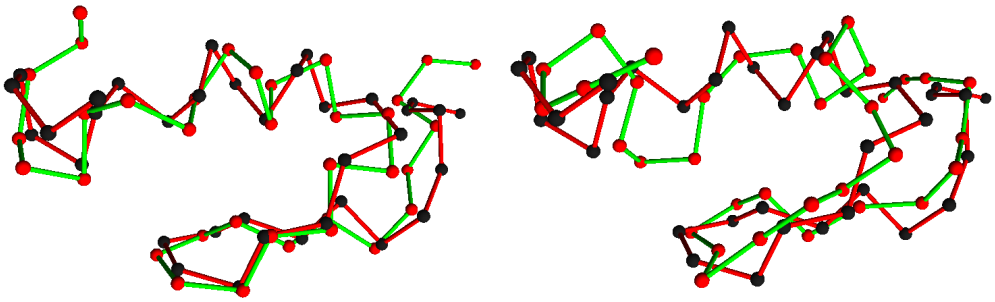


Figure 3.7: An RMSD vs AC plot from [63]. In this experiment, 100 low energy structures are found using two different energy functions (HSE-based and CN-based). The HSE-low-energy structures tend to have a slightly better RMSD and a significant better AC than the CN-optimized structures. The structures marked with circles are shown in Figure 3.8.



(a) (b)

Figure 3.8: The red structure is the C_α-trace of 1SRK and the green structures are the C_α-traces of two of our predictions. The left green prediction is the HSE-optimized structure and the right green prediction is the CN-optimized structure. The optimized structures are marked with circles in Figure 3.8. Both predictions have similar RMSD but different AC.

is the so-called tertiary structure prediction category, which is also considered to be the most important since protein structure determines protein function. Another category is the *model quality assessment* (MQA) category. Here a set of alternative structures (models) for the same amino acid sequence is given. The task of the MQA predictors is therefore to estimate the quality of these models without knowing the native structure of the protein. A good MQA algorithm would therefore be able to rank the alternative models correctly. This is important because most algorithms for protein structure prediction do not only predict one structure but a whole set of structures. MQA is described in more details in Chapter 6.

3.8 Our Research

In our research projects, we do not consider the actual protein folding problem. To attack obstacle 1, our energy functions are all based on knowledge based measures found by algorithms that are trained on a large set of proteins. When databases of proteins with known structures grow we also expect our energy function to be more accurate. Our energy functions are described in more detail in Chapter 7. To overcome obstacle 2, we apply a advanced search heuristics (Section 7.5.1) and in Section 7.5.2 we describe our approach for implicit sampling of the whole conformational space.

We use most of the methods for evaluations of predictions described here.

In [63, 62] the predicted structures are very small (up to 35 residues) and the global measure, RMSD, is appropriate for evaluation of prediction quality. We also show that the so-called HSE optimized structures have a better angle correlation using the AC measure. In [67, 66, 24, 65] we obtain the secondary structure prediction from PSIPRED [57]. Even though we argued that the CC in many cases is more appropriate for measuring secondary structure prediction quality, we use the Q_3 instead. The reason for this is, that we want to show that the secondary structure predictions are comparable in quality with other PSIPRED predictions which are usually evaluated using Q_3. Another reason is that people are also more confident with the Q_3 score because it is easier to interpret. In our MQA approach [64], we use the GDT measure to evaluate our correlation between predicted quality and real quality (GDT). We currently participate in CASP8 with our MQA approach which is described in more detail in Section 6.3.

3.9 Chapter Summary

In the right environment, the amino acid chain is a self-assembling machine. All information needed to compute the tertiary structure is stored in the amino acid sequence. The problem of computing the folding pathway of the amino acid chain from an unfolded structure to the native structure is called the protein folding problem. An easier problem is to disregard the folding pathway and *just* compute the native structure of the protein given the amino acid sequence. Even though both problems are extremely important they are still unsolved. There are two main reasons for this. One problem is that the natural energy function is very difficult or even impossible to compute and all known approximations of the natural energy function do not have the required properties. Another problem is that the conformational space is very big and difficult to search. Even though these problems have not been solved, there is a vast number algorithms for computing folding pathways and predicting protein structure. However, either they never give reasonable results or they can only predict the tertiary structure of a small subset of proteins. For evaluation of algorithms for protein structure prediction, many measures have been proposed. In this chapter, some of the most popular measures are described together with their advantages and disadvantages. Every second year the state-of-the-art algorithms are assessed at CASP. Here, the algorithms are blind-tested on a number of protein structures and their tertiary structure prediction are evaluated when the native structures of the proteins are revealed.

Chapter 4

Secondary Structure Prediction

Given an amino acid sequence, secondary structure prediction is the task of mapping each amino acid into one of the secondary structure categories. These categories are typically helix, strand and coil. The first secondary structure prediction algorithms, developed in the 1960s and 1970s, were based on the properties of the single amino acids [28, 52, 81], even though there are some correlation between amino acid type and secondary structure, the accuracy of these algorithms is poor. Later, a significant improvement in accuracy came when the algorithms considered segments of contiguous amino acids for prediction of secondary structure. Among the best performing algorithms in the 1980's [78] and 1990's [82] were algorithms based on neural networks. Today, one of the best performing algorithms (PSIPRED) is based on feed-forward neural networks combined with evolutionary information [57]. In the following text it is briefly described how the feed-forward neural network can be used for prediction of secondary structure.

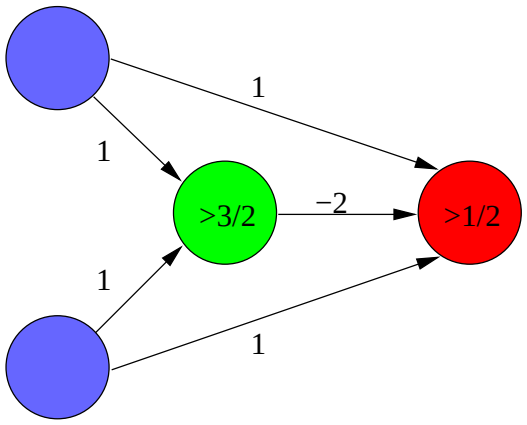
4.1 Neural Networks

Artificial neural networks are usually very rough models of natural neural networks that exist in most animals. These biological neural networks (brains) are very complex and are not considered in this text. However, the terminology used to describe artificial neural networks use some of the words that describe biological neural networks (like neurons and synapses). In the following text, an artificial neural network is therefore just referred to as a *neural network*.

The main motivation for using a neural network for solving biological problems compared to other methods is their ability to handle inconsistent and noisy data.

4.1.1 The Neuron and the Synapses

The simplest unit in a neural network is the neuron. The neurons can be connected to each other, and these connections are called *synapses*. A synapse has a weight called the *synaptic strength*. The synaptic strengths are usually determined by training (Section 4.2). The neuron has a transfer function that



XOR	0	1
0	0	1
1	1	0

Figure 4.1: This neural network can calculate the XOR function. The two left-most neurons are *input* neurons and the right-most neuron is the *output* neuron. In this example, the values of the input neurons are binary. The value of the other neurons are calculated according to Equation 4.1. The transfer function of the neurons are step functions returning either 0 or 1 depending on the threshold of the neurons (shown on the non-input neurons). The synaptic strengths are shown on the edges and the XOR function is shown in the table. The neurons in this example do not have biases.

determines the value of the neuron based on the values of the other incoming neurons together with the synaptic strengths. In standard implementations, the value of neuron i is:

$$y_i = f\left(\sum_j v_{ji} y_j + v_i\right) \tag{4.1}$$

where f is the transfer function, v_{ji} is the synaptic strength from neuron j to neuron i, v_i is the neurons bias and y_j is the value of neuron j. A small example of a neural network is shown in Figure 4.1.

4.1.2 Transfer Functions

The value of the neuron depends on the transfer function as shown in Equation 4.1. In the example in Figure 4.1, the transfer function is a step function, but more often continuous and differentiable transfer functions are used. One of the most applied transfer functions is the sigmoidal function

$$y(x) = \frac{1}{1 + e^{-x}} \tag{4.2}$$

In the limits ($\pm\infty$), the sigmodial function behaves like a step function and near $x = 0$ the sigmoidal function is almost linear.

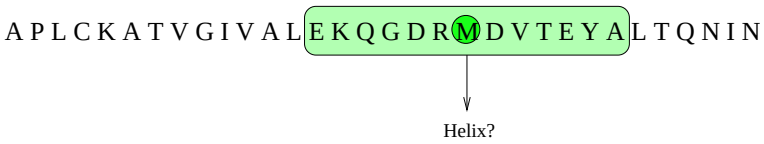


Figure 4.2: A window slides over the amino acid sequence and the neural network predicts the secondary structure of the central amino acid.

4.1.3 Feed-Forward Architecture for Secondary Structure Prediction

This neural network architecture has three layers. An input layer, a hidden layer and an output layer. The input to the neural network is a segment of amino acids called a *window* and the output is the classification of the central amino acid in the window (Figure 4.2). The encoding of an amino acid is *orthogonal*, meaning that there is a neuron for each of the 20 amino acids. Windows at the beginning or the end of an amino acid sequence contain slots with no amino acid, so an additional neuron is used to represent non-existent amino acids. The neurons of the input layer are therefore clustered in groups of 21 neurons as illustrated in Figure 4.3. The size of the window and the number of neurons in the hidden layer are variables and good values must be determined experimentally. The output layer typically has two neurons, one corresponds to the classification of a secondary structure class (i.e. helix) and the other corresponds to other classes (i.e. strand and coil). Similar neural networks for other secondary structure classes can be made. The prediction of an amino acid window can be determined by the *winner takes it all* strategy, meaning that the output neuron with highest value determines the classification.

4.2 Training the Network

When *training the network*, we want to adjust the parameters of the network such that it has optimal performance. In most cases the architecture is fixed and the weights of the synapses are adjusted. When a neural network is used for classifying input data (i.e. a window of amino acids) into secondary structure, it is natural to compare the output of the neural network with the correct classification. For this purpose, *training data* is used to compute an error depending of the network output and the correct output. When training the network, the synaptic weights are therefore adjusted such that this error becomes minimal. The naïve approach to this task is to try all combinations of synaptic weights and choose the set of weights that gives the minimum error. However, in theory the synaptic weights are continuous and even crude discretizations give a huge number of combinations. It is therefore necessary to use another strategy for determining the weights of the synapses.

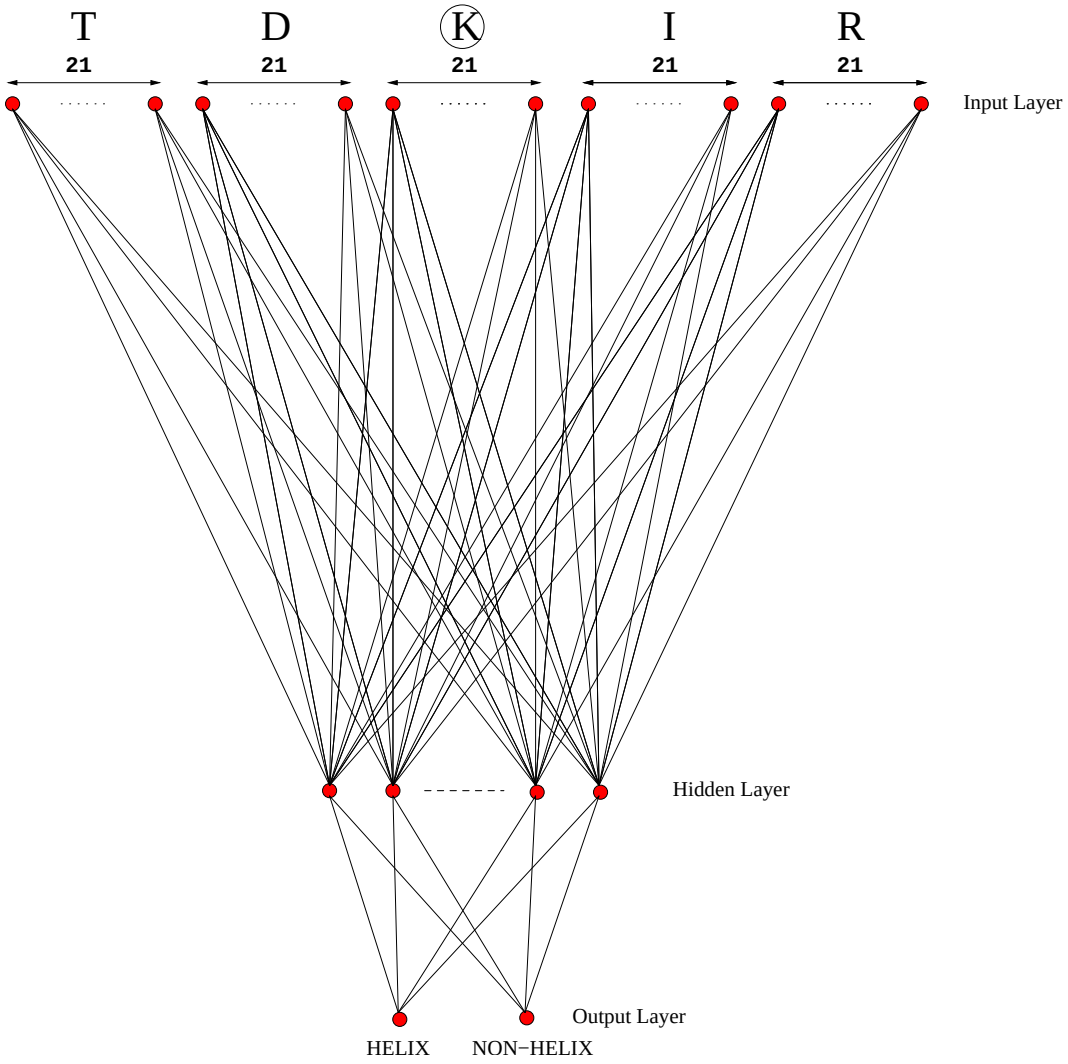


Figure 4.3: Three-layered neural network.

4.2.1 Backpropagation

Backpropagation is an algorithm for adjusting the synaptic weights such that the error becomes small. Backpropagation does not guarantee to find the optimal set of weights, but nevertheless, it is the most applied algorithm for training feed-forward networks. Basically, the backpropagation algorithm computes the gradient of the error function efficiently. The gradient is a vector, that points in the direction where the function grows most. The opposite direction of the gradient therefore indicates where the function declines the most. By calculating the gradient of the error function and adjusting the synaptic weights accordingly it is possible to minimize the error function. Refer to [80] for more details about the backpropagation algorithm.

4.3 Our Research

In most of our research we make use of secondary structure predictions. These predictions are often reliable and contain much information about the native structure of the protein. We do currently not make use of our own neural network for secondary structure prediction (a standard feed-forward network with backpropagation training). Instead, we make use of online webservers because they are more accurate.

In our papers [67, 66, 24], we use predictions of secondary structure to reduce the conformational space. The secondary structure elements are used as rigid segments and it is therefore important that the secondary predictions are as good as possible. We therefore use the PSIPRED webserver for this task, which is based on the feed-forward network and is generally believed to give the best performance. In [67, 66, 24] we also use contact number and half-sphere-exposure measures from neural network predictions. In [64] the contact number probability distributions used by our model quality assessment algorithm are found using feed-forward neural networks. Section 7.5.2 describes in more detail our approach of fixing secondary structure segments and using contact number predictions. Chapter 6 describes in more detail how contact number probabilities are applied for model quality assessment.

4.4 Chapter Summary

Secondary structure prediction is a problem that has received much attention in the literature. Therefore, many different algorithms have been proposed for this problem. In the 1980's neural networks showed great promise for solving this problem. Today, one of the best algorithms (PSIPRED) is based on feed-forward neural networks together with evolutionary information. Secondary structure prediction has many applications and many of the tertiary structure prediction algorithms rely heavily on good secondary structure predictions.

Chapter 5

Tertiary Structure Prediction

Tertiary structure prediction is the task of predicting the native structure of a protein given the amino acid sequence. A vast number of algorithms that attack this problem are described in the literature. In this chapter, some of the fundamental approaches are briefly described (molecular dynamics and homology modeling).

Our approaches for protein structure prediction are based combinatorial optimization. We therefore devote Chapter 7 to prediction algorithms from the literature based on combinatorial optimization and describe our research in this context.

5.1 Molecular Dynamics

The typical way of doing molecular dynamics on proteins is to consider the atoms as spheres with a given radius and mass. The energy function is often a sum of different terms that represent both bonded and non-bonded interactions. The motion of the atoms is then assumed to follow Newton's laws of motion

$$\mathbf{F}_i = m_i \mathbf{a}_i = m_i \frac{\partial^2 \mathbf{r}_i}{\partial t^2} \tag{5.1}$$

where $\mathbf{F}_i$ is the force vector acting on atom i, m_i is the mass of atom i and a_i is the acceleration vector of atom i. The force acting on atom i can also be derived from the potential energy U, such that

$$\mathbf{F}_i = -\frac{\partial}{\partial r_i U} \tag{5.2}$$

When combining these two equations, we get the differential equations that describe the motion of atom i when we know the energy function U

$$m_i \frac{\partial^2 \mathbf{r}_i}{\partial t^2} = -\frac{\partial}{\partial r_i U} \tag{5.3}$$

where $\mathbf{r}_i$ are the coordinates of the i'th atom and t is the time. If Equation 5.3 could be solved analytically for all atoms, we would end up with a function, $\mathbf{r}_i(t)$ for each atom, that describes the exact coordinates of the atoms as a

function of the time given the energy function U. In such case, protein structure prediction would be easy, because the coordinates of the native structure would be $\lim_{t\to\infty} \mathbf{r}_i(t)$. However, proteins are complex systems with thousands of atoms that interact. The energy function therefore depends on the position of all atoms, and no analytical solution are known to exist. Therefore, Equation 5.3 must to be solved numerically. There are many different approaches to how this can be done, however they are all approximations.

5.1.1 The Verlet Algorithm

One of the most straightforward numerical integration algorithms is the Verlet algorithm which is described here. For other more precise algorithms, refer to [76]. The Verlet algorithm, and many other numerical integration algorithms use the Taylor expansions of the energy function:

$$\mathbf{r}_i(t+\delta t) = \mathbf{r}_i(t) + \mathbf{v}_i(t)\delta t + \frac{1}{2}\mathbf{a}_i(t)\delta t^2 \tag{5.4}$$

$$\mathbf{r}_i(t-\delta t) = \mathbf{r}_i(t) - \mathbf{v}_i(t)\delta t + \frac{1}{2}\mathbf{a}_i(t)\delta t^2 \tag{5.5}$$

The sum of these equations is

$$\mathbf{r}_i(t+\delta t) + \mathbf{r}_i(t-\delta t) = 2\mathbf{r}_i(t) + \mathbf{a}_i(t)\delta t^2 \tag{5.6}$$

The position $\mathbf{r}_i$ of atom i a small timestep δt from the current time t is therefore

$$\mathbf{r}_i(t+\delta t) = 2\mathbf{r}_i(t) - \mathbf{r}_i(t-\delta t) + \mathbf{a}_i(t)\delta t^2 \tag{5.7}$$

which is a function of the current positions and the previous positions of the atoms.

A deterministic molecular dynamics simulation could therefore start with some unfolded amino acid chain and iteratively update the positions of the atoms using Equation 5.7. One of the problems with this approach (as described in Section 3.1) is that a proper simulation needs extremely small values of δt, typically in the order of femto seconds 10^{-15}. Even though some proteins fold very fast (in the order of micro seconds), molecular dynamics is still not a feasible approach for protein structure prediction. However, molecular dynamics can be a useful tool for learning more about the mechanics of protein folding.

5.2 Homology Modeling

Many proteins have a high degree of structural similarity among different species. These proteins often have important functionalities in the living cell and are therefore needed in many life forms. When protein sequences from different species have a high degree of similarity, they are said to be *conserved* or *homologous*. In the paper by Chotia and Lesk [16] it is shown experimentally that homologous proteins with a high sequence identity are generally more similar in structure than proteins with less sequence identity. The main technique in homology modeling is therefore to find sequences with high sequence identity

to the target sequence in the database of proteins with known structure (i.e. PDB). One or more of these structures are chosen as *templates* and are used to predict the structure of the target.

Many protein sequences have a homologue counterpart in PDB, and for these proteins, homology modeling can be a successful prediction algorithm. Typical steps in homology modeling are, 1) template recognition, 2) target/template alignment, 3) model building and, 4) model quality assessment [55]. In the following sections, procedures 1 to 3 are described briefly. Since we have made some contributions in the field of model quality assessment [64], we devote the next chapter to this topic.

5.2.1 Template Recognition

When searching a database for templates, different algorithms are typically used such that:

- Close homologues are identified with fast and simple algorithms such as FASTA [70] and BLAST [1].
- Remote homologues are identified with more sophisticated algorithms such as SAM-T06 [37] and PSI-BLAST [2].
- No homologues could be found. This might either be because the algorithms cannot detect the homologues or because no homologues exist in the database.

BLAST and SAM-T06 are among the most popular algorithms for finding homologues and are briefly described here. We also use the SAM-T06 server for finding templates in our model quality assessment algorithm.

BLAST

Typically, one needs to query a sequence against a large database with millions of amino acids and detect the sequences with highest alignment score. One way to do this is to run the well-known Smith-Waterman algorithm [88] on the query sequence and all database sequences and return the highest scoring sequence(s). However, because of the size of the databases and the time complexity of the Smith-Waterman algorithm, this approach is often infeasible.

Basic Local Alignment Search Tool (BLAST) was developed by Altschul et al. and published in 1990. Like the Smith-Waterman algorithm, it computes a sequence alignment between two strings (i.e. amino acid sequences or nucleotide sequences) and assigns an alignment score. While the Smith-Waterman algorithm is exact (it always computes the best local alignment) BLAST is a heuristic algorithm and typically runs several orders of magnitude faster than the Smith-Waterman algorithm.

The main increase in speed comes from the fast pre-filtering of sequences. BLAST first checks if the query sequence contains a subsequence (typically three amino acids) that scores at least T when aligned with a subsequence in

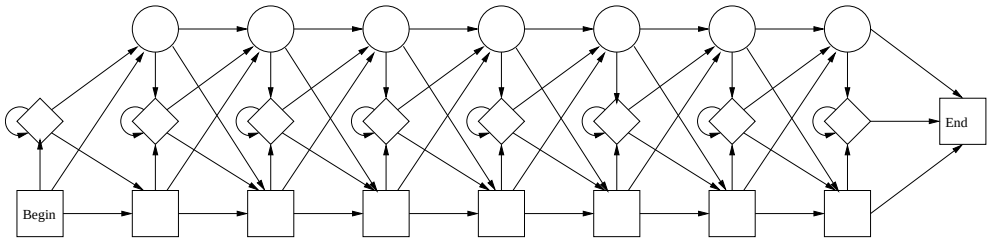


Figure 5.1: A typical HMM topology for biological sequence analysis.

the database sequence. If that is not the case, the database sequence is expected to be insignificant and is discarded. The threshold parameter T therefore determines the sensitivity of BLAST. In the next step, the sequence of three amino acids is extended in both directions to improve the alignment score even further. Again, if the extended alignment score is below some threshold, the database sequence is discarded. Finally, if the database sequence is not pre-filtered, a local alignment is being computed. Using this approach, many of the database sequences that eventually would give a low alignment score are pre-filtered which is much faster than computing the whole local alignment.

HMM for Database Search (SAM-T06)

Sequence Alignment and Modeling system (SAM) [37] is a collection of programs mainly for creating and using hidden Markov models (HMMs). The construction of HMMs and homology detection is automated by the online server called SAM-T06 which is currently the best performing SAM server. The SAM-T06 server can be used for various kinds of local structure prediction and tertiary structure prediction. Our algorithm for extracting distance constraints from alignments (described in Section 6.3) uses templates and alignments found by SAM-T06, SAM-T2K and SAM-T04. The SAM-servers have a lot of features, however, the central part of SAM is the construction and use of HMMs. Here, we therefore only describe how HMMs can be used for template detection.

An HMM is a statistical objects that have been used in many applications, especially speech recognition. In 1994 Krogh et al. [43] described how HMMs can be applied for various tasks in protein modeling. An HMM can be illustrated as a graph, or more specifically, a finite state machine as shown in Figure 5.1. The topology of an HMM can be different from the illustration, but the figure shows the typical HMM topology for biological sequence analysis that was introduced in [43].

A path in the HMM begins at the *begin* node and ends at the *end* node. Such a path can only follow the directed edges, and edges never point backwards. This architecture is therefore also called a left to right architecture. A path in the HMM corresponds to an aligned sequence, possibly with gaps and insertions. The generated aligned sequence depends on the nodes traversed by the path. There are three types of nodes in the HMM:

1. Square nodes correspond to matches.

2. Diamond nodes correspond to insertions.

3. Circular nodes correspond to deletions.

Each edge has an associated transition probability, and the square nodes have letter emission probabilities (these probabilities are not shown in the figures). Figures 5.2 and 5.3 show examples of sequences aligned to an HMM using paths from the start node to the end node. Note that given a sequence, there is an exponential number of paths in the HMM and the HMM therefore represents and exponential number of alignments. Each of these paths has an associated probability and it is usually the path/alignment with highest probability that is interesting. One of the applications of HMMs is to generate multiple alignments. If we assume that the aligned sequences shown in Figures 5.2 and 5.3 are high probability paths then the resulting multiple alignment of the sequences is:

```
-A-VPtjC--
KApVA---LK
```

In the example above, extra deletions have been inserted to align the match states (capital letters).

There are advantages and disadvantages of using HMMs for multiple alignments compared to other multiple alignment algorithms. One of the major disadvantages is, that it usually is very time expensive to train the HMM on the appropriate set of sequences. However, this only needs to be done once, and the following alignments of sequences can be done in $O(N^2)$ time compared to other multiple alignment algorithms that are often NP-hard [96].

There are many other applications of HMMs for biological sequence analysis. The application of HMMs that we use for template detection in [64] is the following. HMMs are trained on sets of sequences that are known to be structurally related (a family of proteins). This training adjusts the transition and emission probabilities such that protein sequences of a given family have a high probability in the corresponding HMM. Given a sequence with unknown structure, the path in the HMM with maximum probability can be interpreted as the alignment of the sequence with unknown structure to the family of proteins that the HMM was trained on. The probability of the path indicates if the sequence with unknown structure is a member of the protein family or not.

Training an HMM can to some extent be compared with training a neural network described in Section 4.2. Instead of adjusting the synaptic weights for neural networks, the transition- and emission probabilities are set systematically in accordance with the training set. Several training algorithms have been proposed in the literature and the most used is the expectation minimization algorithm [10].

5.2.2 Target-template Alignment

When the templates have been identified, perhaps using one or more of the algorithms just described, the target is aligned to template. In an alignment

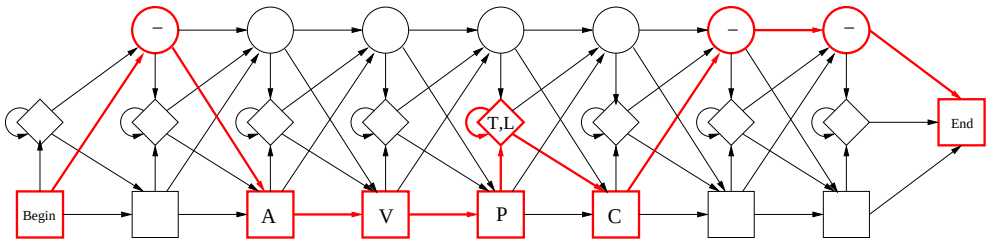


Figure 5.2: The amino acid sequence $AVPTLC$ is aligned to the HMM. The shown path in the HMM generates the alignment $-AVPtlC--$. Capital letters correspond to matches and small letters correspond to insertions.

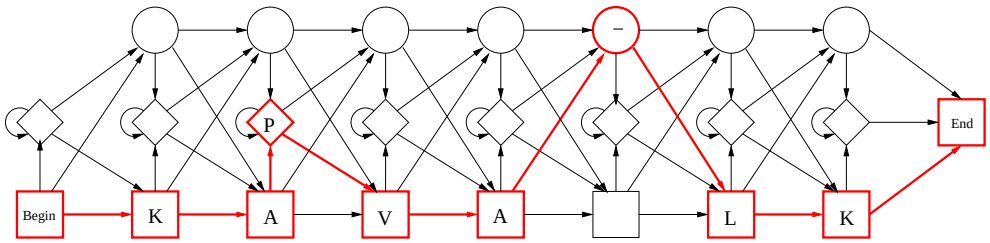


Figure 5.3: The amino acid sequence $KAPVALK$ is aligned to the HMM. The shown path in the HMM generates the alignment $KApVA-LK$.

some of the residues in the target correspond to residues in the template. In most algorithms for template detection, an alignment is also computed. In Figure 5.4 an alignment between a target and a template is shown. Except for exact template matches, such alignments will have deletions and/or insertions in the alignment. If the template and alignment are correct, the matching residues provide a lot of information about the target structure. In Figure 5.5 the C_α-atoms of the matching residues are shown. The coordinates are from the template and thus provide a large amount of information about the global structure of the target.

5.2.3 Model Building

Using the information from the alignments to the templates, the next step is to build full atom models. There as a vast number of approaches to this in the literature. One is to consider short conserved fragments from the templates and assemble them such that some energy function is minimized [87]. Another approach is to extract geometric constraints from the alignments to the templates and find structures that satisfy the constraints [83]. In the model building step, one also often needs to assign coordinates to residues that are not represented in the alignments. This problem is called the *loop closure problem* [11].

The last step in homology modeling is to pick the best of the models generated in the model building step. This is called model quality assessment and is described in more detail in the next chapter.

```
-MVKFACRAITRGRAEGEALVTKEYISFLGGIDKETG-IVKE
m------------ITTGKVWKFGDDISTDEITPGRYNl--TK

DC-E-------IKGESV------------AGRILVFPGGKG-
DPk-elakiaf------ievrpdfarnvrPGDVVVAGKNFGi

-ST--VGSYVLLNLRKNGVAPKAIINKKTETIIAVGAAMAE-
gSSreSAALALKAL---GI---------------AGVIAEs

-------------IPLVEVRDEKFFEAVKTGDRVVVNADEGY
fgrifyrnainigIPLLLGKTEG----LKDGDLVTVNWETGE

V----ELIELEHHHHHH------------------------
VrkgdEILMFEPLE---dflleivreggileyirrrgdlcir
```

Figure 5.4: The blue sequence is the template and the red sequence is the target. The figure shows an alignment between the two sequences. Insertions correspond to small letters, matches correspond to capital letters and deletions correspon to '-'.

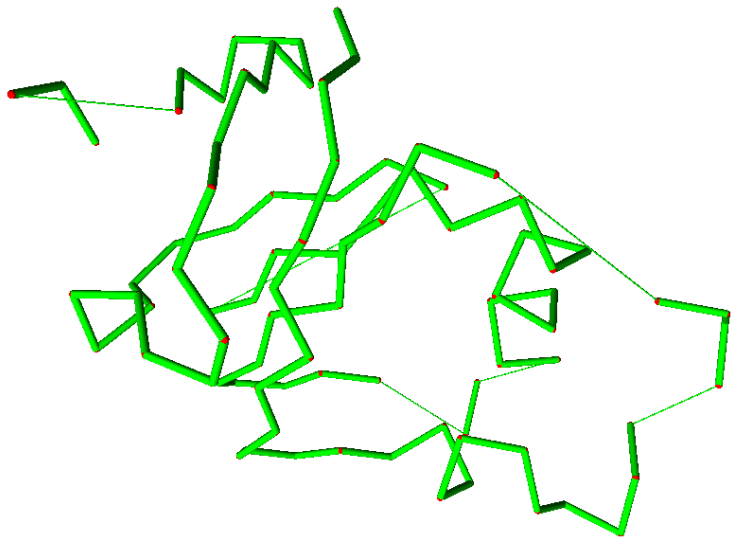


Figure 5.5: The trace of C_α-atoms of the matching residues in Figure 5.4.

5.3 Our Research

Our algorithms for protein structure prediction are based on combinatorial optimization which are described in more details in Chapter 7. However, our algorithm for model quality assessment is based on homology modeling as described in this chapter. Our MQA algorithm does not build models, so the last model building step is omitted. We find templates and alignments using SAM (an HMM). These alignments to templates are used for constructing a set of distance constraints. The distance constraints (between C_β-atoms) are then used in a score function to assess the models in question. Our algorithm for model quality assessment is described in more detail in the next chapter and in [64].

5.4 Chapter Summary

There is a vast number of different approaches for attacking the tertiary structure prediction problem. Some of the basic (non-combinatorial) techniques are molecular dynamics and homology modeling which are described in this chapter. Molecular dynamics is a somewhat naïve approach for computing the atomic trajectories of protein folding. Nevertheless, using molecular dynamics the native structures of small peptides have been predicted. Homology modeling is probably one of the most successful protein structure prediction approaches. Many proteins have one or more so-called homologue counterparts with known structure. A basic idea is therefore to detect these homologue proteins and use them as templates for later model building. In our algorithms for protein structure prediction, we use techniques from combinatorial optimization which are covered in Chapter 7. However, our MQA algorithm is heavily based on techniques from homology modeling.

Chapter 6

Model Quality Assessment

In the previous chapter, it was described how model quality assessment (MQA) often is a natural step in many algorithms for protein structure prediction. MQA algorithms are typically more general and can be used to assess arbitrary models for some target. For example, consider a biologist who sequenced a gene and wants to know the tertiary structure of the corresponding protein. The biologist would of course first query PDB to see if the protein has been analyzed before. If that is not the case, she might want to use a tertiary structure prediction server. There are many online prediction servers available and she might end up using the I-TASSER webserver, because she knows it performed best at the latest CASP7 [98]. On the other hand, the I-TASSER prediction server, does not *always* give the best prediction result among available servers. Another approach is therefore to query many prediction servers known to perform well and select the best *model* generated. This approach, of course, brings up another problem. It is not trivial to determine which model in a set of alternative models is *best* without knowing the native structure. This is the model quality assessment problem. Often, MQA is not only about determining the best model, MQA is often stated as the problem of assigning a score [0:1] to each alternative model of a target, such that the score *correlates* with the *real quality* of the model (Figure 6.1). The *real quality* of the model is usually GDT (as defined in section 3.6), but alternative measures have been used. How the scores should *correlate* with the real quality is not clear and is discussed in more details in the following section.

6.1 Correlation

It is not easy to agree on what measure of quality should be used for evaluating MQA. The reason for this is of course that MQA are used in different contexts. Here we briefly describe three correlation measures; Pearson's r, Spearman's ρ and Kendall's τ. There are other measures of evaluating an MQA, such as the ability to select the best model. Refer to [6] for a description of other MQA evaluation measures.

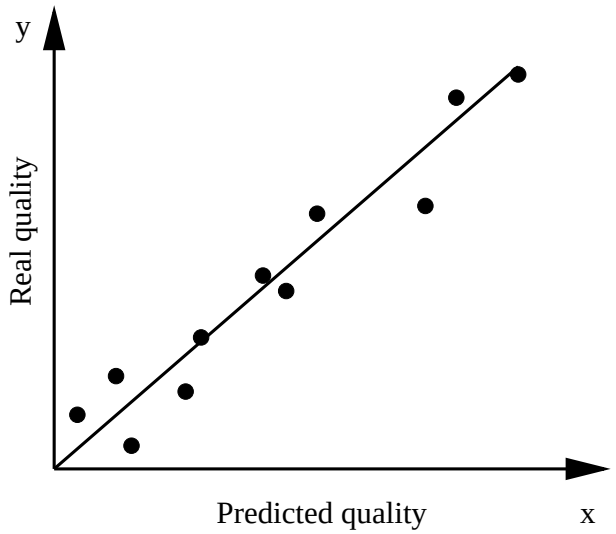


Figure 6.1: Illustration of an MQA with good linear correlation. The points correspond to alternative models for a specific target. The *predicted quality* is the assignment of scores from the MQA algorithm and the *real quality* is the similarity with the model and the native structure (perhaps in terms of GDT). Since the native structure is typically not known when doing MQA, a plot like this can only be made when the native structure is known and the MQA is evaluated.

6.1.1 Pearson's r

When the MQA category was first introduced at CASP7, the MQA algorithms were evaluated using Pearson's r which can be defined as:

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}},$$

where (x_i, y_i) corresponds to pairs of (predicted quality, real quality) for all alternative models in the set. $\bar{x}$ and $\bar{y}$ are the average values over all models of x and y correspondingly.

Pearson's r measures the degree of linear correspondence between two variables with a number r in $[-1 : 1]$. Pearson's r would therefore be high on the linearily correlated points shown in Figure 6.1. In our MQA paper [64] we claim that Pearson's r is inappropriate for evaluation of MQA, because we generally do not care about the *linearity* of the MQA prediction. An example of what we think could be a perfect MQA is shown in Figure 6.2. Even though the MQA in this ad-hoc example is able to pin-point the best model and perfectly rank all models, it has a low Pearson's r because the points are not linearily correlated. To avoid this inappropriate evaluation, we therefore propose to use Spearman's ρ or even better, Kendall's τ as correlation measure for MQA.

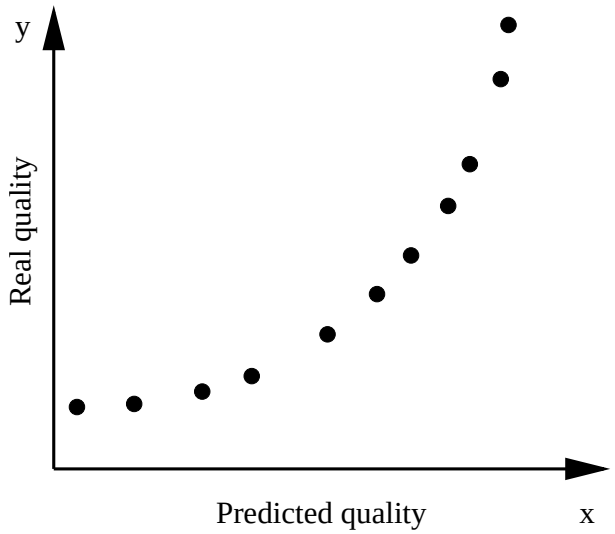


Figure 6.2: Illustration of an artificial MQA with a low Pearson's r.

6.1.2 Spearman's ρ

Spearman's ρ is a special case of Pearson's r. When computing Spearman's ρ, the raw data is first converted to ranks and then Pearson's r is computed for the ranks. Spearman's ρ is therefore maximum for the perfect correlated points in Figure 6.2.

6.1.3 Kendall's τ

Kendall's τ also measures the correspondence between two rankings and is defined as

$$\tau = \frac{4P}{n(n-1)} - 1,$$

where n is the number of points and P the number of concordant pairs. A pair of points is said to be concordant if

$$\text{sign}(X_A - X_B) = \text{sign}(Y_A - Y_B)$$

If two random points (A and B) are chosen and $X_A > X_B$ then Kendall's τ is proportional to the probability that $Y_A > Y_B$. We prefer Kendall's τ over Spearman's ρ, because it is more interpretable, and in our paper [64] we show examples where Kendall's τ agrees more with our intuition of a good MQA than Pearson's r and Spearman's ρ.

6.2 Algorithms for MQA

The ability to assess the quality of a protein model is a fundamental problem in the field of protein structure prediction and many different algorithms have been described in the literature. Recently the CASP organizers recognized the

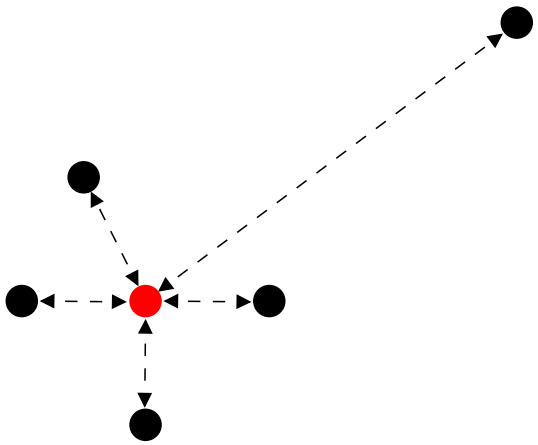


Figure 6.3: An illustration of 6 models where 5 of the models are clustered (according to some arbitrary metric). The red model is the highest scoring model, because it has the lowest mean distance to the other models.

importance of MQA and therefore introduced the MQA category in CASP7 where 26 groups participated. Two algorithms (Pcons and Lee) showed to be superior to the rest of the groups and we briefly describe these two algorithms here. We also describe two new MQA algorithm (that did not participate in CASP7). One is based on *support vector regression* (SVR) and the other uses a new weight optimization algorithm. Finally we briefly introduce our algorithm for MQA, which is described in more details in [64].

6.2.1 Pcons

Pcons [94, 95] is a consensus algorithm that measures the similarity of each model to the other models. Pcons uses LGscore [18] as a similarity measure, but any similarity measure can in principle be applied. The score of a model therefore corresponds to the average similarity between the model and the other models in the set. A model that is very similar to many other models in the set would therefore score high. Any consensus algorithm, like Pcons, depends on the quality of the input set of the models. Even if the input set *does* contain a very good model, the consensus approach might fail if the input set also have a large number of bad and structural similar models. When assessing models from good automated prediction servers (like in CASP7 MQA), the input set often contains many good models which makes consensus approaches appropriate.

6.2.2 Lee's Algorithm

The second best CASP7 MQA algorithm was the Lee algorithm. The basic idea in Lee's algorithm is very simple. First a tertiary structure prediction of the target is made (The Lee group of course use their own prediction algorithm). Then the similarity between each model in the set and Lee's prediction is measured and the models are scored accordingly (Figure 6.4). If the tertiary structure

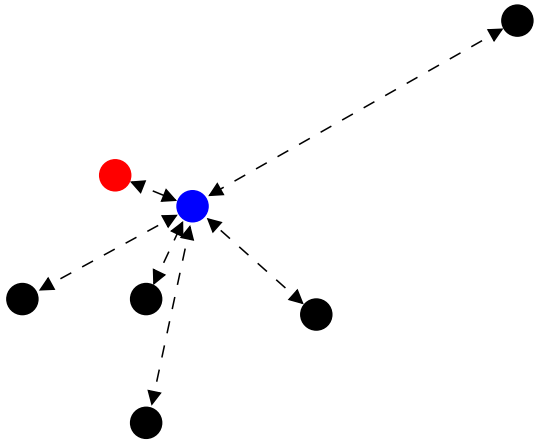


Figure 6.4: An illustration of 6 models to be assessed and one predicted model (blue) by Lee's tertiary structure prediction algorithm. The red model is closest to the predicted model (in terms of GDT) and is therefore given the highest score.

prediction is good (which is often the case with Lee's predictions) this approach is of course successful. In the opposite case, Lee's algorithm is known to produce very bad MQAs when their tertiary structure prediction is wrong [64].

6.2.3 Support Vector Regression

An example of a new MQA algorithm that did not participate in CASP7 is the support vector regression algorithm by Qiu et al. [79]. The idea is to consider many features that somehow describe the quality of the models. The SVR algorithm by Qiu et al., considers a total of 25 features divided in two categories; 4 consensus based features and 21 structural features. The consensus based features include a score function similar to the Pcons approach and the structural features are computed from the individual models (i.e. score functions based on pairwise atomic interactions, hydrophobic packing, angle preferences etc.). The purpose of the algorithm, is to end up with a linear function of these features such that it approximates the GDT of the models:

$$\mathrm{GDT_a}(x) = w_1 f_1(x) + w_2 f_2(x) + \cdots + a_n f_n(x) + b$$

In the equation above, $\mathrm{GDT_a}$ is the approximated GDT computed by the function. The feature functions f_i, $1 \leq i \leq n$, depend on the model x and $\bar{w}$ and b are parameters that must be set appropriately. A good linear function therefore minimizes the error between the real GDT ($\mathrm{GDT_r}$) and $\mathrm{GDT_a}$. To accomplish this, the weights are adjusted using the machine learning technique, SVR. When treating the problem as an SVR problem, a training set is used for solving the convex optimization problem:

$$\text{Minimize} \qquad \tfrac{1}{2}||w||^2 \tag{6.1}$$

$$\begin{aligned} \text{Subject to} \quad & \mathrm{GDT_r}(i) - \mathrm{GDT_a}(i) - b \leq \epsilon && (6.2)\\ & \mathrm{GDT_a}(i) - \mathrm{GDT_r}(i) + b \leq \epsilon && (6.3)\\ & \forall i = 1, 2, \cdots, n && (6.4) \end{aligned}$$

Where w are the weights of the features. $GDT_a(i)$ is the approximate GDT of the i'th training example and $GDT_r(i)$ is the real GDT of the i'th training example. When solving the problem stated above, we find a solution where the errors are within the predefined range ϵ and the sum of the squared weights is minimal. In practice, however, it is inappropriate to predefine ϵ. If ϵ is too small, the problem might not contain any feasible solutions and if ϵ is too large, the function might generate many errors. A more useful alternative formulation used by Qiu et al. is therefore:

$$\begin{aligned} \text{Minimize} \quad & \tfrac{1}{2}||w||^2 + \textstyle\sum_{i=1}^{n} C_i(\zeta_i + \hat{\zeta}_i) && (6.5)\\ \text{Subject to} \quad & \mathrm{GDT_r}(i) - \mathrm{GDT_a}(i) - b \leq \epsilon + \zeta_i && (6.6)\\ & \mathrm{GDT_a}(i) - \mathrm{GDT_r}(i) + b \leq \epsilon + \hat{\zeta}_i && (6.7)\\ & \zeta_i, \hat{\zeta}_i \geq 0 \qquad \forall i = 1, 2, \cdots, n && (6.8) \end{aligned}$$

Where ζ and $\hat{\zeta}$ are variables that make sure that a feasible solution always exists. The constants C_i, $1 \leq i \leq n$, are predefined and correspond to a trade-off between the weight minimization and the error minimization. In the implementation by Qiu et al., they use a higher weight on high ranked models because they want the algorithm to perform better on good models. The SVR algorithm is illustrated in Figure 6.5.

Not surprisingly, after solving the optimization problem, it turns out to be a consensus feature that is given the highest weight. Qui et al. claim that their MQA algorithm outperforms all MQA algorithms at CASP7.

6.2.4 Weight Optimization

Other algorithms for learning the weights of a linear function of features have been proposed in literature. Here, the weight optimization approach from Archie et al. [6] is briefly described. This algorithm is interesting in this study, since some of the features are alignment constraints from our MQA algorithm [64] described in the next section. The optimization algorithm consists of a number of so-called *rebalancing* steps. The basic idea is to divide the features in two sets $(f_1, \cdots, f_m)$ and $(f_{m+1}, \cdots, f_z)$ and let the cost function depend on a parameter $(0 \leq p \leq 1)$ such that:

$$\mathrm{Cost}(x) = p(w_1 f_1(x) + \cdots + a_m f_m(x)) + (1-p)(w_{m+1} f_{m+1}(x) + \cdots + a_z f_z(x))$$

For fixed weights, the idea is to find a value of p that gives the *best* cost for some training set. The best costs in this context are values that correlate well with GDT. The parameter p that optimizes the correlation between Cost(x) and GDT in the above equation is determined using Brent's method [76]. When p is determined, the weights $w_1, \cdots, w_m$ are multiplied by p and the weights

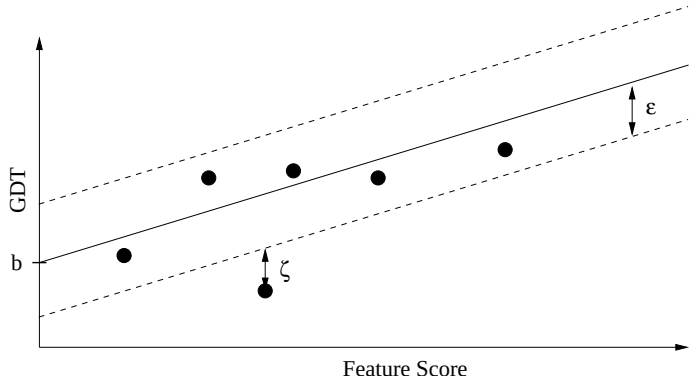


Figure 6.5: For illustration purposes, only one feature is considered here. The slope of the solid line therefore corresponds to the weight of the feature, and the y-axis intersection corresponds to the b parameter. In this example all but one model are within the ϵ range of the line shown and a feasible solution to the problem in Equation 6.1 therefore does not exist (for this slope and b-value). When solving the alternative formulation in Equation 6.5, the ζ of the outlying model is positive and the solution becomes feasible.

$w_{m+1}, \cdots, w_z$ are multiplied by $(p-1)$. This optimization algorithm begins with an initialization of the parameters (see [6] for details) and continues with a number of the rebalancing steps until no improvements in correlation can be found.

When no consensus features are used, this MQA algorithm performs slightly better than the other MQA algorithms described in this chapter. In this case, the most significant features are the alignment constraints (Section 6.3). When model consensus features are added, the MQA algorithm performs significantly better than all other MQA algorithms, and the most significant features, of course, become the consensus based features.

6.3 Our Research

We have developed an algorithm for MQA and tested it on the CASP7 benchmark. The algorithm is described in details in [64] and an overview is briefly described here.

6.3.1 Overview

There are 5 main steps in the MQA algorithm which are also illustrated in Figure 6.6:

a. The input to the MQA algorithm is the amino acid sequence of the target and a set of alternative models for the target.

b. We use SAM_T06 for detecting homologues and computing the alignments. SAM_T06 also returns an E-value for each template found. Tem-

plates with low E-value are on average more correct than templates with high E-value.

c. SAM_T06 also computes the alignments to the templates. For each residue pair with chain separation greater than or equal to 9, we store the distance between C_β-atoms from the alignments if the distance is less than 8 Å. We therefore end up with a length × length - table with sets of observed distances between C_β-atoms as illustrated in Figure 6.7. Then a weighted average for all entries in the residue × residue table is computed. The weights associated with each weight depend on the E-value as described in [64]. The resulting table (Figure 6.8) contains a so-called *desired distance* and a *confidence-value* (weight) of the desired distance. This weight is in the interval [0:1]. If a pair of C_β-atoms has been in contact in many high quality alignments, the weight of the constraint is high (near 1) and if the pair of C_β-atoms has been in contact in few low quality alignments the weight is low (near 0).

d. Each entry in the table generates a so-called *distance constraint*. If the entry has a desired distance, the distance constraint is a function with minimum value in the desired distance as shown in Figure 6.9. Otherwise, the entry generates a so-called non-contact as shown in Figure 6.10.

e. The final model cost function is the weighted sum of all distance constraints. Given a model, the distance between each pair of C_β-atoms is measured and the corresponding distance constraint value is computed. The weighted sum of all distance constraint values is the cost of the model. We rescale the costs such that they correspond to scores in the interval [0:1] where 1 corresponds to the best scoring model and 0 is the worst scoring model.

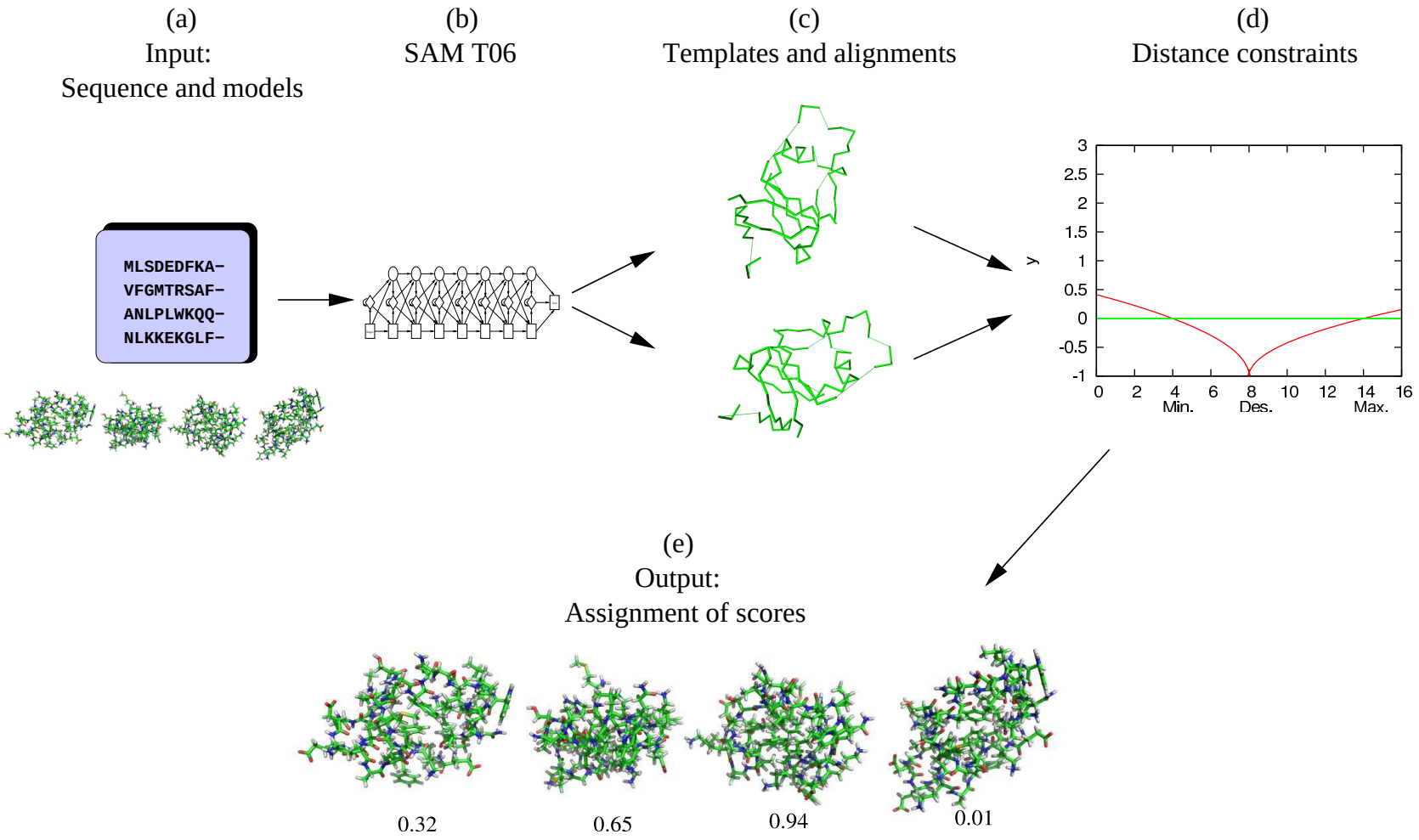


Figure 6.6: Overview of our MQA algorithm.

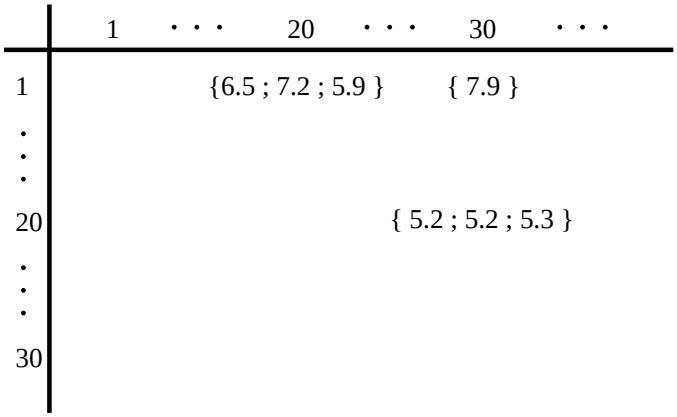


Figure 6.7: A table of observed distances (≤ 8) between pairs of C_β-atoms is constructed.

6.3.2 Optimization

Figure 6.11(a, c, e) shows examples of the quality of the distance constraints for three targets. Target T0314 is known to be one of the most difficult targets of CASP7 in terms of prediction quality. The main reason is that no good homologues have been detected. T0365 is template-based and considered to be medium difficult. T0346 is a so-called high-accuracy template-based target and is the easiest target of CASP7. At least for the template based targets, the figure shows a clear correspondence between constraint weight and constraint quality. It is therefore an obvious improvement strategy to select and use only the high weight constraints. In [64] we describe two selection strategies. One is to select and use only the high weight constraints. When evaluating our MQA algorithm using this approach, the performance is slightly improved compared to using all constraints. However, the most useful selection strategy we have tested is an optimization technique based on contact number probability distributions. We use the feed-forward neural network called *predict-2nd* [38] to predict the probability of the residue having various numbers of contacts. Figure 6.13 illustrates an example of such a prediction of a contact number distribution. One of the objectives in the optimization approach is therefore to select a subset of the constraints such that the total contact probability is maximized. The other objective is to maximize the average weight of the constraints selected. We use a simple greedy approach to find solutions to this problem as described in more details in [64]. The consequence of selecting the constraints using the optimization approach is illustrated in Figures 6.11 (b,d,f) and 6.12 (b,d,f). It is clear that we are able to filter away many of the wrong constraints, which eventually leads to better correlations of MQA.

6.3.3 Evaluation

We have compared our MQA algorithm with other MQA algorithms in the literature, including the two best ranked algorithms at CASP7 and the MQA

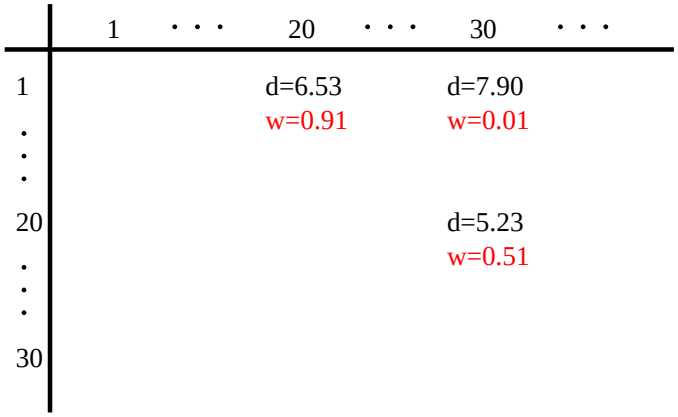


Figure 6.8: A weighted average distance is computed together with the confidence-weight of the distance. In this ad-hoc example there is a high confidence that the C_β-atoms of residue 1 and residue 20 are near 6.53 Å from each other.

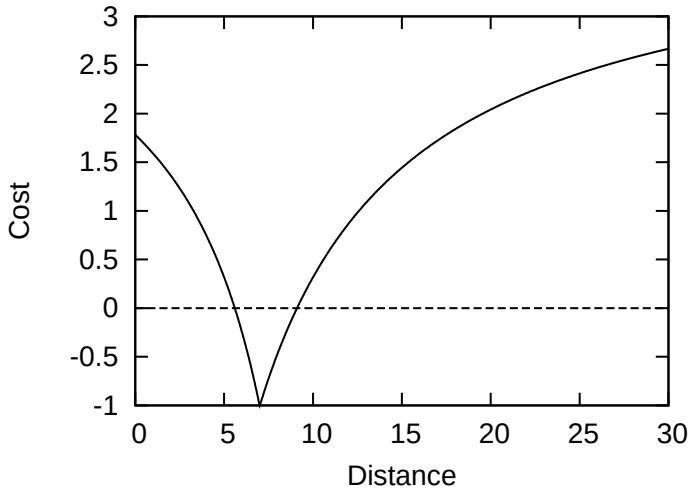


Figure 6.9: The cost function of the distance constraint where the desired distance is 7 Å.

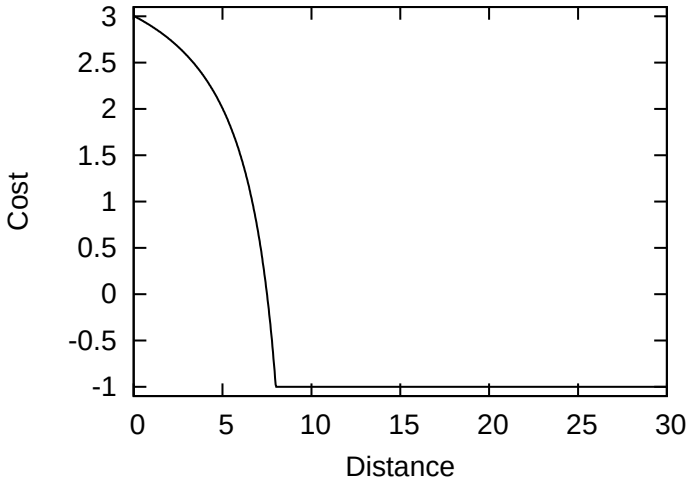


Figure 6.10: A plot of the non-contact cost function. These cost functions correspond to pairs of residues where contacts between C_β atoms have not been observed in any alignment.

algorithm by Qiu et al. Table 6.1 shows the results of this comparison (refer to [64] for details). The *Constraints (Consensus)*-row in the table is our MQA algorithm when the distance constraints are extracted from the models instead of alignments. This corresponds to a model consensus approach which again shows the best performance on the CASP7 MQA benchmark. The *Undertaker* row is from the weight optimization algorithm described in Section 6.2.4, where 73 different features from the Undertaker protein structure prediction program are used. Among these features are the alignment constraints which proves to be the most significant of the features.

6.4 Chapter Summary

Model Quality Assessment (MQA) is the problem of assigning a quality measure to alternative models of a target without knowing the native structure. It is a natural step in many algorithms for protein structure prediction and other applications. The MQA category has recently been presented at CASP7. It is not a trivial task to evaluate an MQA algorithm. Several correlation methods have been proposed and we argue that Kendall's τ is one of the most appropriate measures for evaluating MQA algorithms. The best MQA algorithm at CASP7 was a consensus based algorithm that scored the models according to their mean distance to other models (in terms of LGscore). Consensus based algorithms require a set of good models (to derive consensus from) and can therefore not be used for assessing few models. In the extreme case where the quality of one or two models should be assessed, it does not make sense to use a consensus approach. Our approach for MQA does not have this requirement. The score function we use, is based on distance constraints from alignments. Our MQA algorithm therefore performs best on template based targets. We also show how

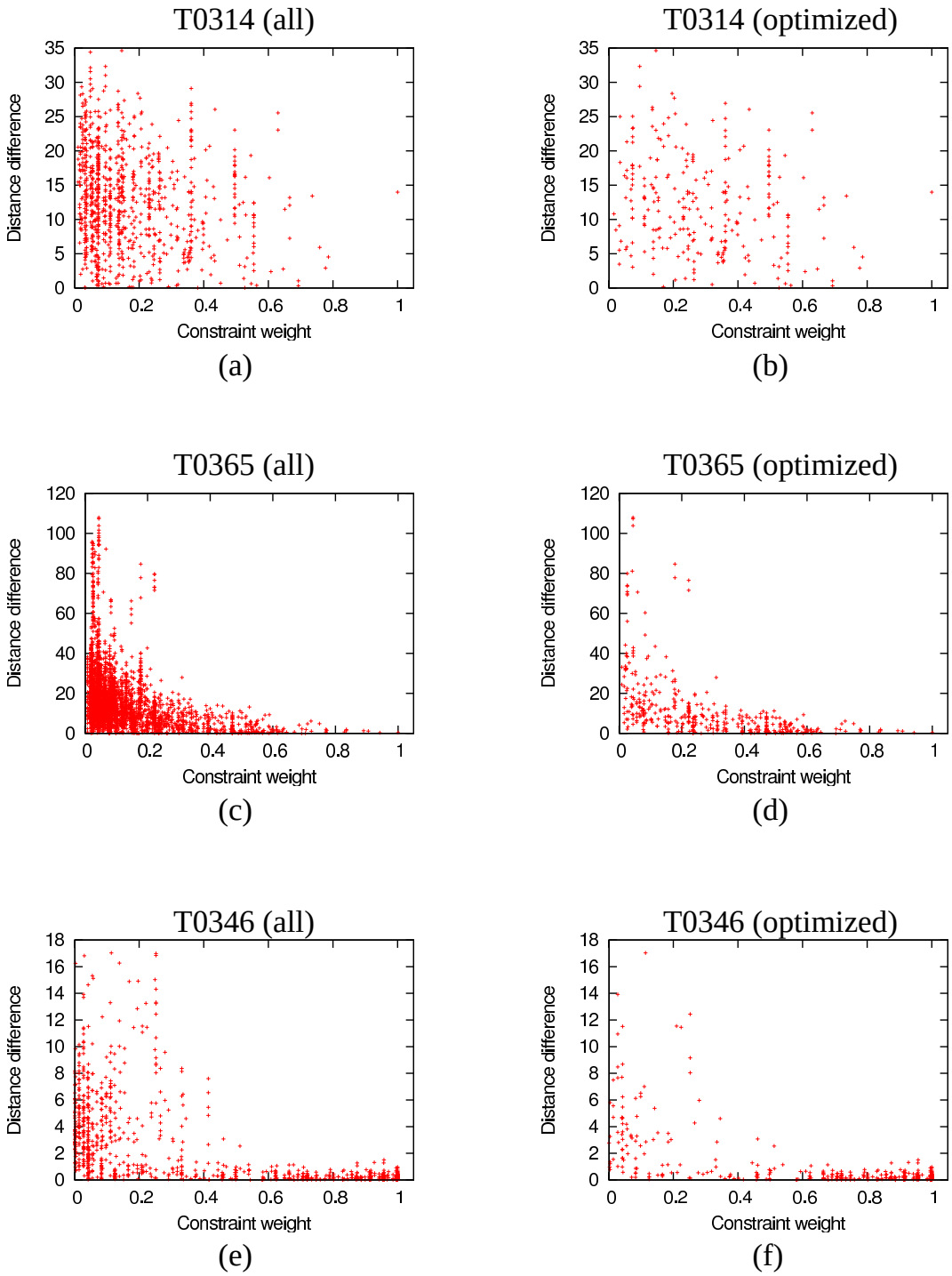


Figure 6.11: Quality of distance constraints for three targets ranging from very difficult (T0314) to very easy (T0346). The *distance difference* is the absolute value of the difference between the *desired distance* of the constraint and the *real distance* in the native structure.

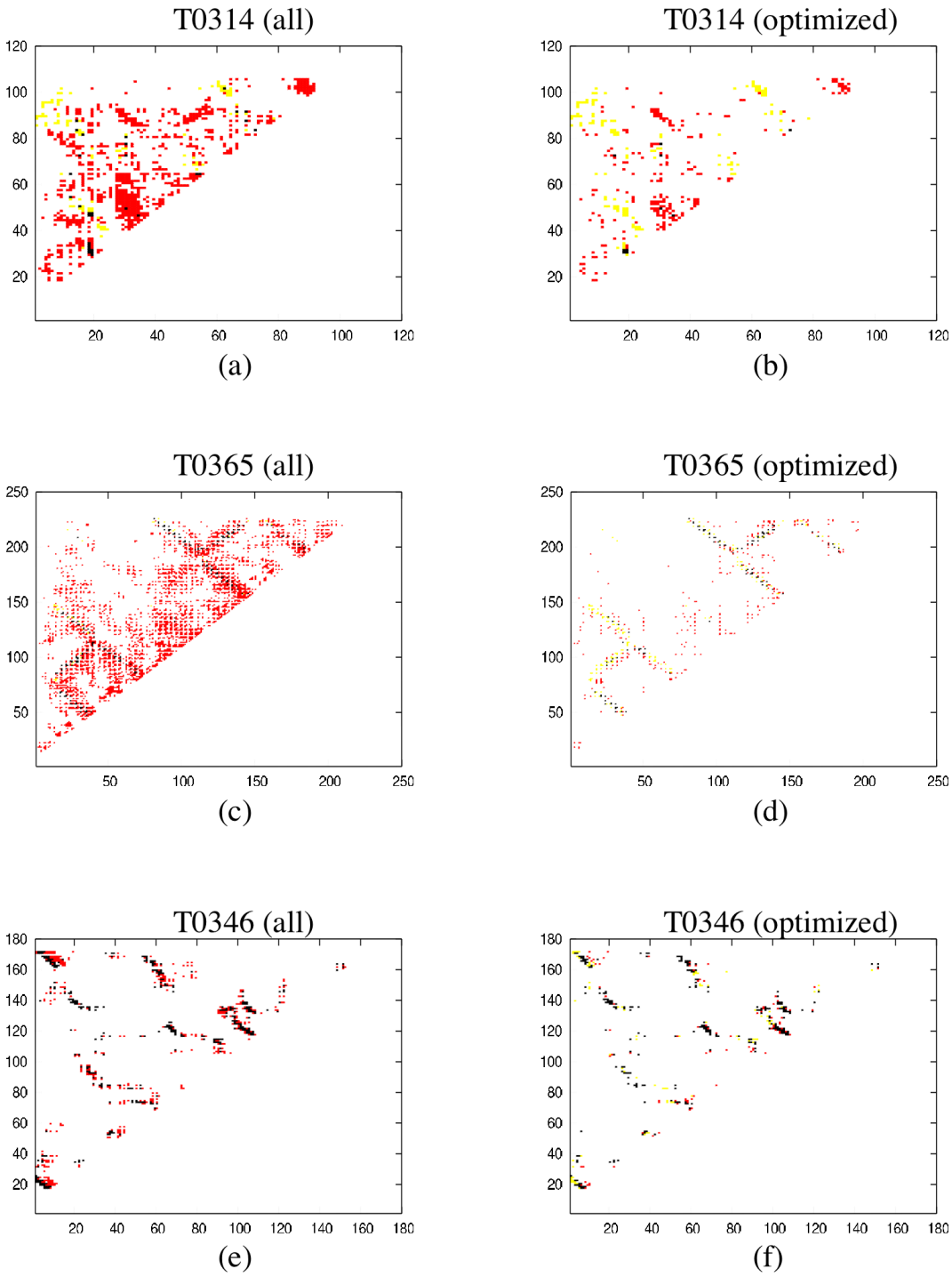


Figure 6.12: Contact maps for three targets ranging from very difficult (T0314) to very easy (T0346). Black points correspond to contacts in the native structure that are correctly predicted (a distance constraint is generated). Yellow points correspond to contacts in the native structure that we missed (no distance constraint generated) and red points correspond to pairs of residues not in contact but having a distance constraint. White points corresponds to correctly predicted non-contacts. From the figures, it is clear that mainly the red points are being filtered away by the optimzation algorithm, while leaving the black points.

Group	$\bar{\tau}$	$\bar{r}$
Constraints (Consensus)	**0.62**	**0.86**
Undertaker (with align. constr.)	**0.62**	**0.86**
Lee	0.59	0.81
Qiu	0.58	0.85
Constraints (Aligments)	0.57	0.83
Pcons	0.56	0.85

Table 6.1: The table shows the average Kendall's τ and average Pearson's r for MQA with each algorithm compared, ranked using Kendall's τ. The average values are on a per-target basis. The *Constraints (Alignments)* row is the results of MQA with distance constraints from alignments. The *Constraints (Consensus)* row is the results of extracting the constraints from the models to be assessed. The *Undertaker* row is the results of using all Undertaker cost functions including the alignment constraint sets. Lee, and Pcons are top ranked MQA algorithms presented at CASP7 (groups 556 and 634 respectively). Qiu is the SVR MQA algorithm. In this table only the full backbone models are evaluated. Our distance constraints perform worse on models with missing backbone atoms, because a subset of the distance constraints can not be evaluated. In [6] the Undertaker constraints (together with the alignment constraints) are also evaluated on all models including the broken models.

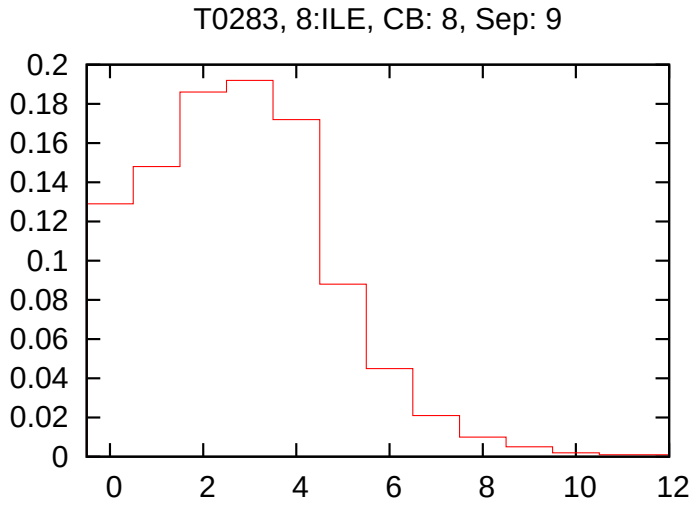


Figure 6.13: A contact number probability distribution for residue number 8 (isoleucine) of target T0283. The plot shows that there is maximum probability that the C_β-atom of the residue is in contact (≤ 8) with three other C_β-atoms (with chain-separation 9 or more). The probability of 10 or more contacts for this residue is almost zero.

to apply contact number predictions for selecting good distance constraints.

Chapter 7

Structure Prediction using Combinatorial Optimization

The protein structure prediction problem can be treated as a standard combinatorial optimization problem which is one of the classical disciplines in computer science. A combinatorial optimization problem consists of a mathematical object which has different discrete states. Each of these states has an associated value which is defined by a so-called objective function. The solution to the combinatorial optimization problem is the state with the global minimum (or maximum) value of the objective function. When treating the protein structure prediction problem as a combinatorial optimization problem, we therefore need to discretize the different structures of the polypeptide chain and associate an objective value to every state. The objective value should somehow represent the real Gibbs free energy of the polypeptide chain. However, it is not trivial to discretize the polypeptide chain in a reasonable manner. In nature there is an infinite number of possible structures of a polypeptide chain, so any discretization is more or less unnatural. When we treat the protein structure prediction problem as a combinatorial optimization problem, we therefore sacrifice some realism to achieve computational tractability.

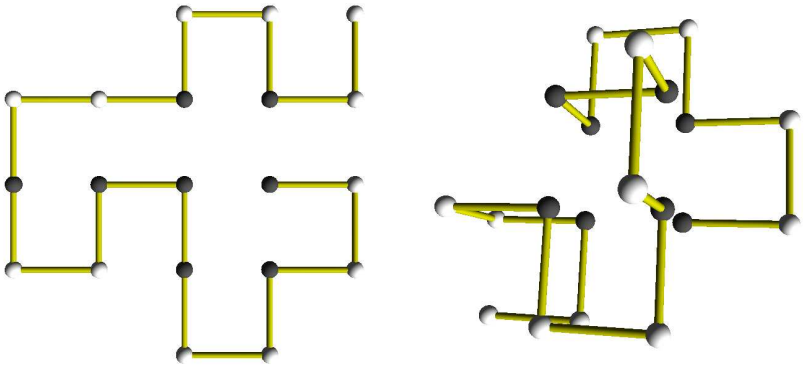
7.1 Discrete Representations of a Polypeptide Chain

A straightforward discretization of the polypeptide chain is to treat the amid planes as rigid objects and only allow discrete values of the ϕ and ψ angles (shown in Figure 2.5 page 17). Note that even a very rough discretization of 4 different values of the dihedral angles gives an astronomical number of possible structures. The number N of possible structures using this discretization (not counting clashing structures or other unnatural structures) is

$$N(L) = 4^{2L-2}$$

where L is the number of amino acids in the polypeptide chain. The number of conformations for a protein of 200 amino acids therefore is:

$$N(200) = 4^{398} \simeq 4 * 10^{239}$$



(a) (b)

Figure 7.1: (A) The string of beads on a 2D quadratic lattice. (B) The string of beads on a 3D cubic lattice. The 2D lattice representation is of course very unnatural because no protein is 2D, however, this lattice can be useful for evaluation and analysis of algorithms. The 3D lattice representation is more natural but still considered to be a very rough discretization.

so, even with this very rough discretization, the number of possible structures are many times bigger than the number of atoms in the universe - and we have not even considered the degree of freedom of the side chains.

7.1.1 Discretization using Lattices

One of the much simpler representations is to only consider one atom per residue. This representation is often called *a string of beads*. Each bead typically represents either the C_α-atom or C_β-atom of the amino acids. Discretization of the string of beads can be done in many ways. One of the widely used discretizations is to force each bead to be positioned on a lattice (Figure 7.1).

The advantages of using a lattice for the discretization of a string of beads are many:

- There is a finite number of structures, and complete enumeration is possible for small chains.
- Comparison of structures is easy. Structures are different if they have a different path in the lattice and it is easy to check if two structures are rotational identical.
- It is easier to do exact computations and rounding errors can often be eliminated by considering lattice coordinates instead of space coordinates.
- Many algorithmic problems can be solved very efficiently. Collision detection occurs only when beads occupy the same lattice node. Finding

the neighbours of a bead can be done by only searching the neighbouring lattice nodes. Local moves can be computed very fast.

The only important disadvantage of using a lattice to represent the chain of beads is that real proteins in nature are *not* represented in lattices. Lattice models therefore only approximate real proteins to some extent. However, in section 7.5.1, we show how more complex lattices can reduce this problem.

7.1.2 The HP-model

In 1989 Lau and Dill presented the well-known HP-model [48]. In their model the string of beads corresponds to a polypeptide chain where each residue is classified in the two categories; hydrophobic nonpolar (H) or hydrophilic polar (P). Hydrophobicity is one of the important properties of amino acids and was introduced in Section 3.3 page 26. In the original HP-model, the string of hydrophobic and hydrophilic beads is only allowed to occupy 2D-cubic lattice nodes and the score-function is the number of hydrophobic (H) neighbours in the lattice. Using complete enumeration they find the structure(s) with maximum number of H-neighbours and they argue that these structures share some properties with real proteins. Structures with maximum number of H-neighbours tend to have a high compactness and a hydrophobic core like water soluble proteins. The 2D-structure in Figure 7.1 is an example of a solutions in the HP-model where the black nodes correspond to hydrophobic amino acids and the white nodes correspond to hydrophilic amino acids. Lau and Dill did not use any optimization technique other than complete enumeration when they presented their algorithm in 1989. In the next section it is described how to find good or even optimal solutions when complete enumeration is not feasible.

7.2 Solving Combinatorial Optimization Problems

Many real-world problems can be treated as combinatorial optimization problems. Typical examples are the traveling salesman problem [45], vehicle routing problem [45] and knapsack problem [40]. These examples are all known to be NP-hard which is often the case for combinatorial optimization problems. Hart and Istrail [30] also showed that finding optimal structures in one of the simplest formulations of the protein structure prediction problem, the 2D HP-model, is NP-hard. When dealing with NP-hard problems, people generally use three different approaches.

a. **Exact algorithms.** Even though a problem is NP-hard, it might be possible to solve realistic problem instances in reasonable time. For some problems, much is known about the structure of optimal solutions, which can be used for constructing efficient exact algorithms. This is the case for the HP-model. Even though it is NP-hard, Backofen et al. [7] are able to compute fast exact solutions to instances with up to 200 residues in the HP-model using their theory of compact hydrophobic cores.

b. **Approximation algorithms.** An approximation algorithm runs in polynomial time and is able to give a guarantee on the quality of the solution. Not all NP-hard problems can be approximated, Hart and Istrail [30], however, showed that the HP-model can be approximated with a guaranteed energy within 3/8 of optimal energy.

c. **Heuristics.** Heuristic search algorithms do not give a guarantee on the solution quality. Nevertheless, in practice, heuristic algorithms are preferred for many combinatorial optimization problems. This is mainly because they are easy to implement and imperically show good solution qualities.

7.3 Metaheuristics for Protein Structure Prediction

In the following sections, different popular metaheuristics are described. Metaheuristics are heuristics that are more general and can be applied to a broad range of optimization problems. The metaheuristics included here (Monte Carlo Search, Tabu search and Bee colony optimization) are just a small subset of metaheuristics proposed in the literature. They are chosen because we use them in our research described in Section 7.5.

7.3.1 Monte Carlo Search

In protein structure prediction, the metaheuristic Metropolis Monte Carlo Search [58] (here we just name it Monte Carlo Search) is one of the most applied metaheuristics. This can be explained by the simplicity of the metaheuristic and the similarity with physical systems. Monte Carlo-based search algorithms differ from molecular dynamics algorithms by being nondeterministic. For these algorithms *randomness* is important and that is why they are named after the famous casino in Monaco. There are many variants of Monte Carlo search (simulated annealing [42], replica exchange [90], Markov chain Monte Carlo [3], etc.). The standard approach is to maintain a single current structure and iteratively apply small random changes to it in each iteration. If the energy of the modified structure is lower than the current structure, the modified structure is automatically accepted as the current structure. If the energy of the modified structure is higher than the current structure, the modified structure is accepted with some probability. The standard probability of accepting a modified structure s', given the current structure s is

$$P(s'|s) = e^{-\frac{U(s')-U(s)}{T}}$$

where U is the energy function and T is the temperature. For high temperatures, almost all solutions are accepted and for low temperatures almost only the improving solutions are accepted. When applying the MC metaheuristic, it is important to determine a suitable temperature. A widely used variant of MC is called *simulated annealing* (SA) where the temperature is gradually decreased during the SA run.

The small changes applied to a structure defines a *neighbourhood* of structures. As the name indicates, neighbouring structures should be close in the

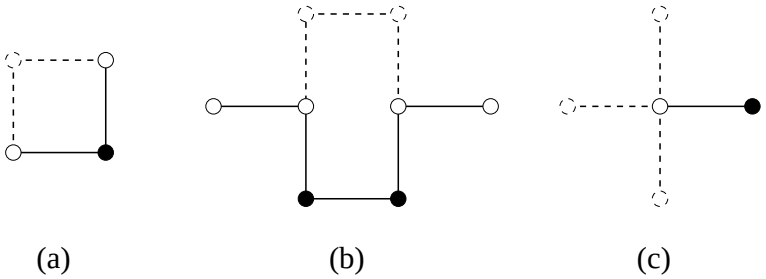


Figure 7.2: Three examples of moves in a quadratic lattice. The black bead is being moved into the position of the dashed white bead. The positions of other solid white beads are fixed. (a) Corner move. (b) Crankshaft move. (c) The end move (three different moves only allowed by the two end beads). These moves are 2D version of the move set by Sali et al. [84]

solution space and therefore share some properties. A neighbourhood is often defined by a *move set*. In Figure 7.2 an example of a very simple move set for the string of beads on a 2D cubic lattice is illustrated.

7.3.2 Tabu Search

One of the most successful metaheuristic in combinatorial optimization is *tabu search* (TS). It has shown to be successful for many applications like vehicle routing [25], VLSI routing [89, 53], packing problems [77] etc., but it has not been given much attention in the field of protein structure prediction. The search paradigms presented so far (molecular dynamics (MD) in Section 5.1 page 43 and MC) have roots in physics. However, many models for protein structure prediction are not based on physics. They are often extremely simplified and discretized - and their energy functions might not even contain any physical-derived terms. In these cases, it is therefore not likely that MD or MC could simulate the real folding pathways. In most cases the only interesting structures are those with low energy. Like MD and MC, tabu search do provide a pathway of examined solutions, however it should not be given a physical interpretation.

TS was first described by Fred Glover in 1989 [26]. It is a local search algorithm with a memory. A local search algorithm iteratively chooses some solution that improves the current solution. At some point it therefore ends in a local optimum where no neighbourhood solution can improve the current solution. This local optimum might also be global, but this can usually not be determined. The risk of getting trapped in local minima can be reduced using a memory. In the most simple implementations of TS, the memory consists of previously visited solutions stored in a so-called tabu-list. When choosing the new neighbourhood solution, the memory is scanned to make sure that it has not been visited before. This simple tabu definition was used by Oakley et al. [61] for prediction of protein aggregation with modest success. Tabu search can also be used for marking regions of the search space as tabu. This is often done by defining some of the attributes of visited solutions tabu. In most TS

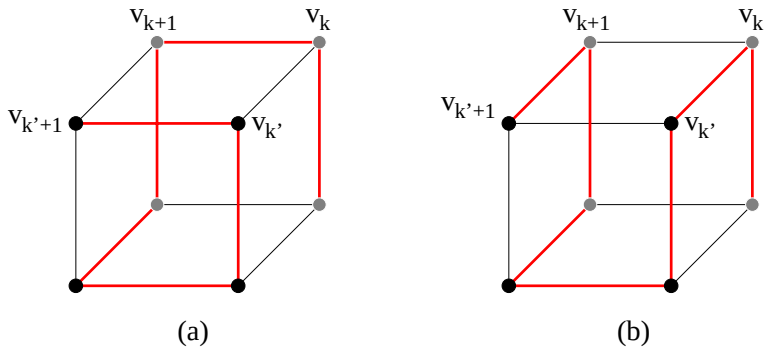


Figure 7.3: (a) shows a path where the conditions for an edge patch is present. (b) shows the result of an edge patch.

algorithms the tabu list is considered to be short term memory and is therefore implemented using a FIFO queue with a predefined length.

One of the few applications of TS on protein structure prediction is the study by Pardalos et al. [69]. They use a cubic lattice to represent the string of beads as a self-avoiding path. The objective function is a simple statistically derived potential energy. The neighbourhood of structures consists of the so-called *edge patches*. An edge patch is a global change (compared to the local moves illustrated in Figure 7.2) to the self-avoiding path in the lattice and it can be made when the following conditions are present:

Let $v_1, v_2, \cdots, v_n$ be lattice nodes visited by the string of beads in this order. If $k < k'$ and $v_k, v_{k'}$ are neighbours and v_{k+1} and $v_{k'+1}$ are also neighbours. Then a new path in the lattice can be generated by adding the two edges (v_k, $v_{k'}$) and (v_{k+1}, $v_{k'+1}$), and deleting the two edges (v_k, v_{k+1}) and ($v_{k'}, v_{k'+1}$). An example of an edge patch is shown in Figure 7.3.

An edge patch is characterized by the edges that are inserted and the edges that are removed. In the TS algorithm by Pardalos et al., the applied edge patches are inserted in the tabu list and future moves are prevented from applying previously used edge patches. Using this strategy, a region of the conformational space is made tabu with each edge patch in the tabu list. The advantage of marking regions tabu compared to just making previously visited solutions tabu, is that the search escapes local minima faster. However, we also face the risk that good solutions are made tabu just because they share some properties with bad solutions. The edge patch move just described can make very large changes to the structure. The risk is therefore, that a structure is defined as tabu, even though it structurally does not have any similarities with a previously visited structure. This risk is reduced by the *aspiration criteria*; if a solution in the neighbourhood is better than the best observed solution, it is accepted even if it is tabu.

In our paper *Reconstructing Protein Structure from Solvent Exposure using Tabu Search* [63] we minimize the risk of making good structures tabu by introducing a new tabu definition. It is directly based on structural differences as described in more detail in Section 7.5.1.

7.3.3 Artificial Intelligence

artificial intelligence (AI) is often considered to be either strong AI or weak AI. Strong AI is said to be comparable or even better than human intelligence. However, there is no good definition of strong AI that is widely agreed on. The main reason is probably that *intelligence* does not have a real scientific definition. My favourite description of strong AI is from Alan Turing [91]. If a machine can pass the Turing test, then it is strong AI. In short, the Turing test is about whether or not a machine can answer arbitrary questions, such that a person (giving the questions) cannot reliably distinguish the machine from a person. So, according to Turing, if a machine answers questions like a human, then it is strong AI. None of the techniques described in this thesis, are designed to act as human beings and are therefore considered to be weak AI. Weak AI, are in general algorithms that try to simulate human or animal traits. Examples of such traits are *learning*, *reasoning*, *planning* etc. However, algorithms that make use of general natural phenomena like evolution or swarming are usually also considered to be weak AI.

Many of the algorithms used for solving combinatorial optimization problems are considered to be weak AI. This is also the case for some of the algorithms in computational biology. Three of the AI techniques described in this thesis are so-called *supervised learning* algorithms. These are artificial neural networks (Section 4.1), hidden Markov models (Section 5.2.1) and support vector regression (Section 6.2.3). All of these algorithms have in common that they are presented with a number of examples and are supposed to *learn* the general patterns in the set of examples. These algorithms differ much in how they learn from the examples and how they represent the knowledge they have learned. However, they are all capable of handling incomplete and uncertain data in the set of examples. The tabu search metaheuristic presented in this Chapter is also considered to be weak AI, simply because of the memory used to escape local minima.

Swarm Intelligence

In Section 7.5.3 we describe our approach for protein structure prediction using a search strategy borrowed from the foraging behaviour of honey bees. Such an algorithm is called *swarm intelligence* (SI) and is also considered to be AI. In nature, some animals *swarm* to achieve survival and reproductive benefits. The specific way a species of animals swarm varies, but the term is usually used to describe a group of animals (usually insects, fish or birds) that moves in the same direction and behaves similar to environmental changes. SI typically consists of a number of individuals called *agents*. Such agents are able to work independently in the environment, but usually make decisions based on communication with other agents or changes in their environment. In the case of our bee colony optimization approach, agents correspond to honey bees. A honey bee is able to collect nectar without the help of other bees, but it can also communicate with other bees (using the so-called *waggle dance*) to tell other bees about the positions of good flower beds. Another example of a swarm is an ant colony.

When using *ant colony optimization* (ACO), the agents correspond to ants who seek to find food close to their colony. Ants seek for food randomly and also leaves a so-called *pheromone trail*. This pheromone trail attracts the other ants from the colony such that trails with high pheromone contents have a higher probability of being used by the ants. The pheromone also evaporates. Long trails take a long time to travel and therefore eventually end up having lower pheromone concentration than shorter trails. ACO is suitable for solving graph problems such as the TSP [20], but it has also been applied to the protein structure prediction problem in [86].

7.4 Exact Algorithms

In 1968 Levinthal postulated that the conformational space of proteins is too large to be searched exhaustively [51]. It is therefore believed that proteins use folding pathways to reach the native structure. This is probably also one of the main reasons why many computational approaches to protein structure prediction use heuristic search algorithms to generate folding pathways. The main problems with these algorithms are that they get trapped in local minima and they cannot give a guarantee on the solution quality.

Levinthal's postulate is of course widely acknowledged. However, just because nature uses folding pathways it does not necessarily mean that computational approaches to structure prediction must use the same technique. There have been a few *exact* algorithms for protein structure prediction proposed in the literature. These algorithms implicitly search the conformational space exhaustively using advanced optimization techniques. Something that proteins in nature, of course, cannot do.

In computer science it has been known for decades that optimal solutions indeed can be found for many realistic problems, even though the solution space is very big. Some instances of problems like *vehicle routing problems* (VRP) [45], *knapsack problems* (KP) [40] and the Steiner tree problem [31, 13, 68] can be solved in reasonable time using advanced techniques from combinatorial optimization. The fact that the conformational space of proteins is astronomical large, therefore should *not* be a reason for avoiding exact algorithms.

The main advantage of using exact algorithms is that they guarantee to find the global optimum in the given model and do therefore not have the problem of getting stuck in local optimum. One of the disadvantages is that efficient exact algorithms are often much more difficult to design compared to simple search heuristics like MC and TS. The model and energy functions are therefore typically simplified such that bounds can be computed efficiently.

In this section some of the exact algorithms for protein structure prediction that exist in the literature are briefly described. We begin with an exact algorithm in the HP model, continues with the αBB algorithm and end with a description of our own exact algorithm.

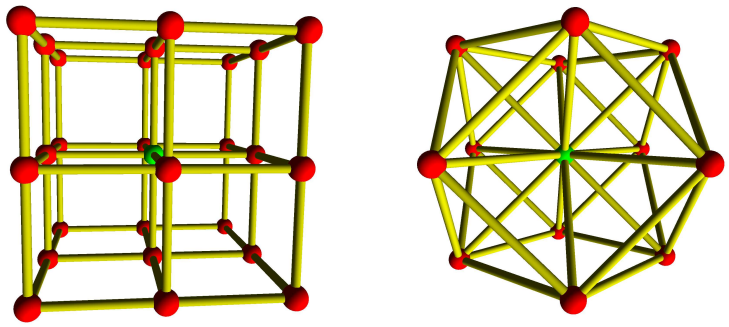


Figure 7.4: (a) The *simple cubic* (SC) lattice where each lattice node has 6 neighbours. (b) the more complex *face centered cubic* (FCC) lattice where each lattice node has 12 neighbours. The lattice vectors of the SC lattice are $(\pm 1, 0, 0), (0, \pm 1, 0), (0, 0, \pm 1)$, and the lattice vectors of FCC are all combinations of $(\pm 1, \pm 1, 0), (\pm 1, 0, \pm 1), (0, \pm 1, \pm 1)$.

7.4.1 Exact Structure Prediction in the HP-model

In the paper by Backofen and Will [7] an efficient constraint-based algorithm is presented. Their algorithm can solve large instances of problems in the *simple cubic* (SC) lattice and the *face-centered-cubic* (FCC) lattice (Figure 7.4). Their objective is to maximize the number hydrophobic neighbours of the string of beads. This is the same objective as for the classic HP-model, but Backofen and Will use more complex and realistic lattices.

The basic idea of their approach comes from the observation that optimal solutions often have near the maximum number of hydrophobic contacts that is possible in a lattice. So, by knowing only the number of hydrophobic amino acids, they can precompute the expected energy. Furthermore they can precompute the so-called maximally compact cores of the string of beads. This is illustrated in Figure 7.5 with a simple example. When generating a solution, all that is needed is to thread the remaining hydrophilic amino acids on the hydrophobic core. However, this might not be possible if the optimal solution does not have a maximally compact core. In that case, the algorithm iteratively threads the hydrophilic amino acids on less compact cores until it succeeds. Refer to [7] for details about computing compact cores and threading the hydrophilic amino acids on the hydrophobic cores.

Using their exact algorithm it is possible to find the global minimum structure of proteins up to 200 residues in less than a minute on a Pentium 4. Backofen and Will do not report the similarity of the global minimum structures with the native structures. This is probably because there are no similarity due to the simple energy function.

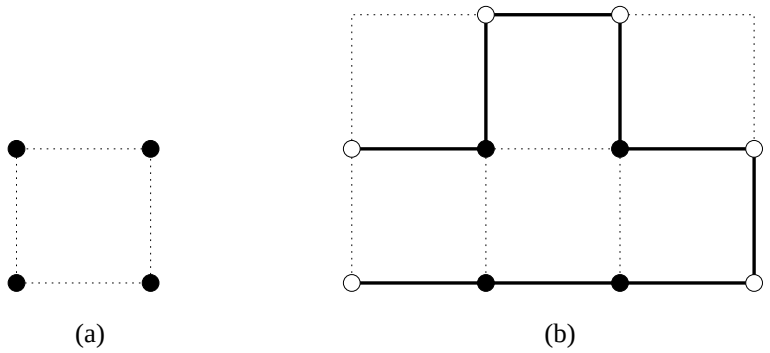


Figure 7.5: Consider the string PHPPHPPHHP with 4 hydrophobic amino acids and 6 hydrophilic amino acids. (a) shows the most compact core of 4 hydrophobic amino acids which can be precomputed. Figure (b) shows a possible threading of the string of beads to the core. When such a threading is found (if possible), the solution is optimal.

7.4.2 The αBB Algorithm

The α *Branch and Bound* (αBB) algorithm is one of the few algorithms for protein structure prediction that is based on the branch and bound paradigm. In this section, the general branch and bound paradigm is described, and the αBB by Maranas et al. [54, 4] is introduced. Our branch and bound approach for protein structure prediction is described in Section 7.5.2. Maranas et al. used the αBB algorithm to predict the global minimum energy structures of small molecules (up to 14 atoms). In [23] Eyrich et al. extended the αBB algorithm such that it can handle proteins with hundreds of atoms.

The Branch and Bound Paradigm

The branch and bound approach is an algorithm for solving various combinatorial optimization problems. It was first described in 1960 by Land et al. [44] where they used it for solving linear programming problems. Today, branch and bound algorithms are mostly used for solving NP-hard problems.

One of the basic techniques of the branch and bound paradigm is the recursive subdivision of the solution space into smaller sets. Such a recursive subdivision can also be represented by a tree, where the union of the children nodes represents the solution space of the parent node. Subdivison of the solution space is called *branching*. Branch and bound also require the computations of upper and lower bound estimates for a particular subdivision. A lower bound is a number that is equal to, or lower than, any solution value in the set. The upper bound is a value that is greater than, or equal to, the minimum solution value in the set. Obviously, a particular solution set cannot contain a global minimum solution if the lower bound is higher than an upper bound in any solution set. When such a situation occurs, the whole solution set can be disregarded (bounded) without explicitly considering each solution.

When developing a branch and bound algorithm for a particular problem,

the main goals are therefore to develop an appropriate branching scheme and an efficient lower bound algorithm. A good lower bound algorithm is therefore fast (compared to complete enumeration of the subset) and computes bounds that are close to the minimum value solutions in the sets; a so-called *tight bound*.

αBB

In [54] Maranas et al. represent molecules using a list of dihedral angles between bonded atoms. The energy function is the Lennard-Jones potential function [50]. Branching is done by considering the dihedral angle with the widest range and constructing two new subspaces corresponding to splitting the chosen dihedral angle in two separate intervals. Note that the dihedral angles here are treated as continuous variables and not discrete as for real combinatorial optimization problems.

The lower bound is computed by using theory from convex optimization theory. Maranas et al. develop a theory that shows how to compute a lower bound function, L, given the energy function V. The lower bound function, of course, has the property that it is less than or equal to the energy function for all conformations in the set. Furthermore, L is convex such that a local minimum of L is also a global minimum. In each node of the branch and bound tree, the lower bound is computed by minimizing L which can be done using standard techniques for solving convex optimization problems. The upper bound is the corresponding value of V (the conformation where L is minimum). Maranas et al. are able to find global minimum energy solutions but only for small molecules. However, Eyrich et al. [23] extend the αBB algorithm by using fixed secondary structure elements and are able to work on real sized proteins. In [23] all results are reported only for secondary structure segments derived from the native structure of the protein. As described in Chapter 4, secondary structure prediction is far from perfect and Eyrich et al. do not show how the performance of the αBB algorithm is, when the secondary structure predictions have errors.

7.4.3 Protein Threading

Another large category of tertiary structure prediction algorithms are the *threading* based algorithms. In a typical threading algorithm, a set of structures representing different folds are known. Using the threading algorithm, the most likely fold of an amino acid sequence can be identified and used as a template for later model building.

This task is accomplished by a *threading* of the amino acids of the protein with unknown structure on each of the fold structures. A threading is found by aligning the amino acid sequence to a fold structure, possibly using gaps and insertions, as illustrated in Figure 7.6. Each possible threading has an energy and the threading algorithm finds the threading with minimum energy. Such a minimum energy threading is found for each of the fold structures and the threading with minimum energy over all fold structures is assumed to be the fold structure of the amino acid sequence with unknown structure. The number of possible threadings on a single fold structure is exponential and in [46], the

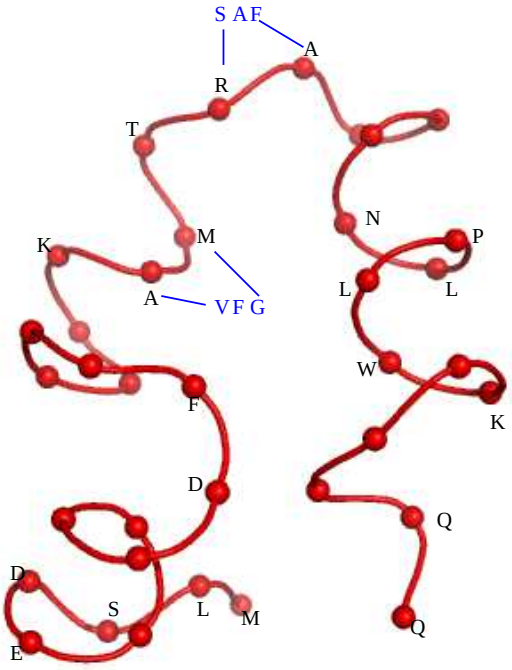


Figure 7.6: The sequence *MLSDEDFKAVFGMTRSAFANLPLWKQQ* is aligned to the fold structure. The blue residues correspond to insertions. The residues with no letters correspond to deletions.

problem is shown to be NP-hard for energy functions with a sum pairwise terms.

7.4.4 Example of an Exact Threading Algorithm

One possible threading model is described in [47]. Here, a so-called *core structural model* (CSM) is made for all fold structures. A CSM consists of a sequence of loops and secondary structure elements. The secondary structure elements have fixed length and the loop regions have a minimum and maximum length. A valid threading of an amino acid sequence is therefore an assignment of subsequences to each loop region and secondary structure element such that the intervals are satisfied (Figure 7.7). The energy function both consists of positions of single amino acids and pairwise terms:

$$f(T) = \sum_i g_1(i, t_i) + \sum_i \sum_{j>i} g_2(i, j, t_i, t_j)$$

Where g_1 is a function depending on the core segment i located at the t_i'th amino acid in the sequence. Likewise g_2 is a function of pairs of core segments

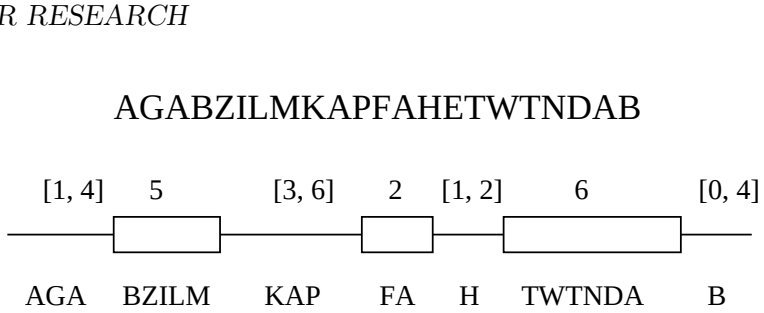


Figure 7.7: An example of a core structural model. The lines correspond to variable length loop regions and the boxes correspond to secondary structure elements. The upper intervals correspond to the number of residues allowed in the regions and the letters show an example of a valid threading of sequence $AGABZILMKAPFAHTWTNDAB$.

and their positions in the sequence. The actual values of the energy could depend on secondary structure prediction, amino acid burial, hydrophobicity, statistical derived potentials, etc. The problem of finding the threading with global minimum energy is NP-hard because of the pairwise terms and in [47] the problem is solved using the branch and bound technique.

7.5 Our Research

We have developed three different approaches for reconstructing C_α-traces using techniques from combinatorial optimization. Our first approach described in Section 7.5.1 applies exact values of half-sphere-exposure (HSE) to reconstruct C_α-traces using Monte Carlo search and Tabu search. In Section 7.5.2 we describe our exact approach using a branch-and-bound technique for decoy generation. In this approach we also apply predicted measures and the algorithm can therefore be considered as *de novo*. In Section 7.5.3 we describe our artificial intelligence approach. It is based on a swarm intelligence which mimics the foraging behaviour of honey bees. All of these approaches are briefly described here. For more details refer to the corresponding papers.

7.5.1 Paper: Reconstructing Protein Structure from Solvent Exposure using Tabu Search

In our paper *Reconstructing Protein Structure from Solvent Exposure using Tabu Search* [63] we compare the performance of *Monte Carlo* (MC) search and *tabu search* (TS) on a simple protein structure prediction problem. In addition to the MC and TS comparison we also estimate the information contents of the newly introduced *half-sphere-exposure* (HSE) measure [29]. Figure 7.8 contains a short description of HSE from the paper.

Determining the information contents of the HSE measure or the CN measure is important. Both measures can be predicted from amino acid sequence with reasonable accuracy and can therefore be used for de novo prediction. However, in the paper described here we only use exact measures computed from the native structure of the protein.

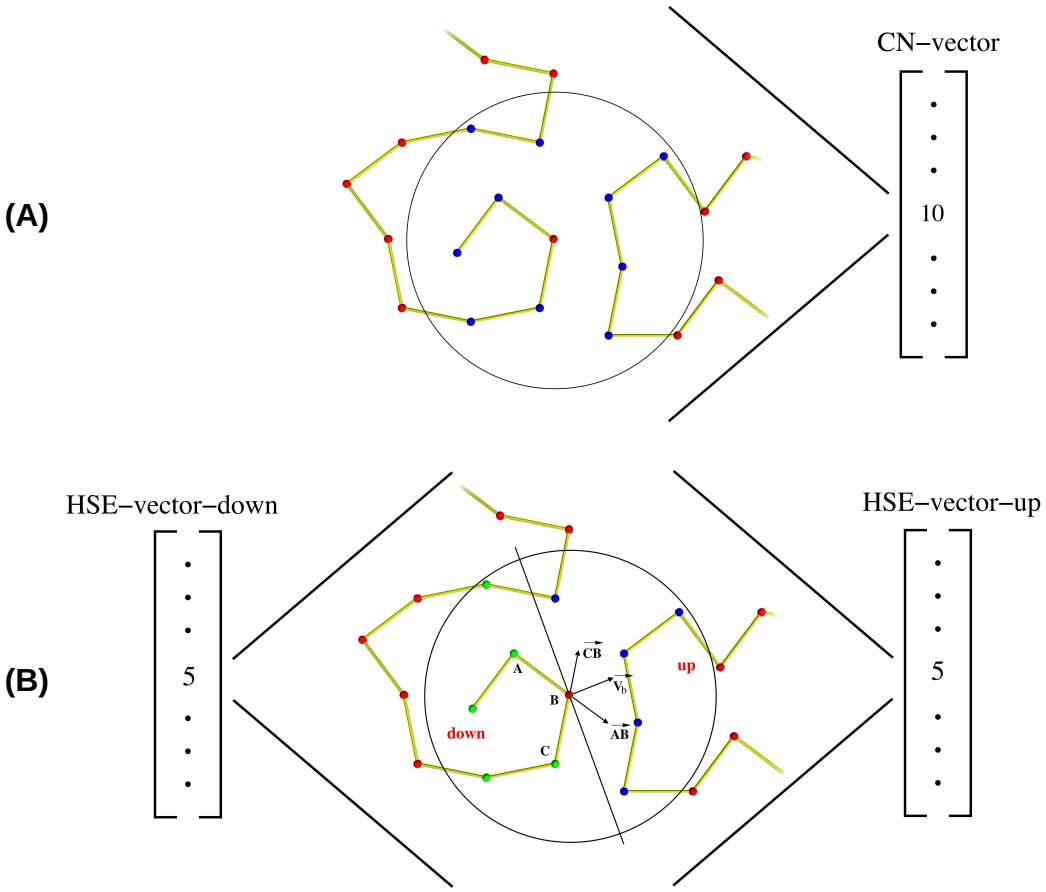


Figure 7.8: *(From [63]).* The extent to which an amino acid in a protein is accessible to the surrounding solvent is highly dependent on the type of amino acid. In general, hydrophilic amino acids tend to be near the solvent accessible surface, while hydrophobic amino acids tend to be buried in the core of the protein. To measure this effect, several solvent exposure measures have been proposed [49, 27, 17, 15, 73, 74, 75], and one of these is the contact number measure (CN) [75]. The CN of a residue is the number of C_α atoms in a sphere centered at the C_α atom of the residue in question (Figure A). The CN of all residues of a protein is called the CN vector. The CN vector is well conserved and can be predicted with high accuracy [41]. While the CN measure uses a single sphere centered at the C_α-atom, the HSE measure considers two hemispheres. Two values, an up and a down value, are associated with each residue, corresponding to the upper and lower hemisphere. The geometry of the HSE construction is shown schematically in Figure B. Given the positions of 3 consecutive C_α atoms (A, B, C), the approximate side-chain direction $\vec{V_b}$ can be computed as the sum of $\vec{AB}$ and $\vec{CB}$. The plane perpendicular to $\vec{V_b}$ cuts the sphere centered at B in an upper and a lower hemisphere. The up and down HSE values measure two fundamentally different environments of an amino acid, one of them corresponding to the neighbourhood of the side chain [29]. The HSE measure compares favorably with other solvent exposure measures in terms of computational complexity, sensitivity, correlation with the stability of mutants and conservation. An important advantage of the HSE measure is that it can be calculated from C_α-only or other simplified protein models. Therefore, it forms an attractive alternative to the use of the CN measure in protein structure prediction methods [87].

Our strategy for evaluating the information contents of HSE and comparison of the algorithms is the following. A random C_α-trace is constructed and iteratively improved by either MC or TS such that the HSE vector of the C_α-trace is similar or close to the HSE-vector of the real protein. This is done by minimizing the energy function

$$\mathrm{E}(A,B) = \sqrt{\frac{\sum_{i=1}^{N}\left((A_{u_i}-B_{u_i})^2+(A_{d_i}-B_{d_i})^2\right)}{2N}},$$

where $\{A,B\}_{u_i}$ and $\{A,B\}_{d_i}$ are the up and down values of the i'th index of the HSE-vectors. N is the length of the vectors. $\mathrm{E}(A,B)$ is the energy of structure S_A where A is the HSE vector of S_A and B is the HSE vector derived from the native structure. These definitions are easily specialized to the CN measure.

To discretize the conformational space of the C_α-trace, the C_α-atoms are confined to be positioned on a lattice. In addition to the simple 2D quadratic lattices and 3D cubic lattices already discussed in Figure 7.1, we also consider the more complex lattices FCC (Figure 7.4(b)) and *high coordination* (HC) lattice (Figure 7.9). A structure is represented by a list of directions in the lattice for all but the first C_α-atom. The move set used by both algorithms, MC and TS, consists of all possible changes of up to three consecutive directions. Using this terminology, the simple move set described in Figure 7.2 contains respectively 2 changes (A), 3 changes (B) and 1 change (C). In addition to this move set, we also allow one index change at a non-endpoint. This results in a translation of the string of beads after the index change.

The MC algorithm is implemented as described in section 7.3.1 and the TS algorithm uses a new tabu definition. The trivial tabu definition is to store previously visited solutions in a tabu list and then prevent the algorithm from visiting them. When using this tabu definition, our experience is that it takes very long time to escape local minima. This is mainly because the tabu list needs to be filled with all structures in a neighbourhood around a local minimum before it can escape the local minimum. When using complex lattices, there are many different structures that are almost structural equal and the tabu list therefore becomes very large before the local minimum is escaped. This is a problem we reduce by defining the concept of explicit- and implicit tabu structures (see Figure 7.10).

In paper [63] there are three experiments which are briefly described here.

a. The first experiment determines suitable values of the tabu difference (ϵ) and the tabu list size. This is done by running the TS algorithm on 20 initially random structures for a small peptide and optimize each structure until a zero-energy structure is found or a maximum of 15 minutes have passed. An average energy of the 20 final structures is computed and plotted in Figure 7.11. When the tabu difference is zero, the TS algorithm behaves as a regular TS algorithm where only previously visited structures are tabu. The figure therefore shows, that our use of implicit tabu structures improves the performance of the TS algorithm considerably on this experiment. On the other hand, one should be careful that

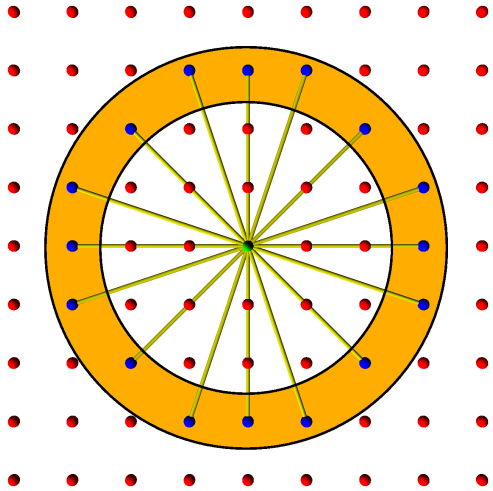


Figure 7.9: This is a 2D illustration of the *High coordination* (HC) lattice. A high coordination lattice has an underlying cubic lattice with unit length less than $3.8/N$ Å for some integer $N > 1$. Cubic lattice points are connected in the high coordination lattice if their Euclidean distance is between $3.8 \pm \beta$ for some $\beta > 0$. The high coordination lattices used in our experiments are named HC4 and HC8 corresponding to their N value (4 and 8). The β value is 0.2 for all applied HC lattices. The figure shows a 2D high coordination lattice with $N = 3$ and $\beta = 0.4$.

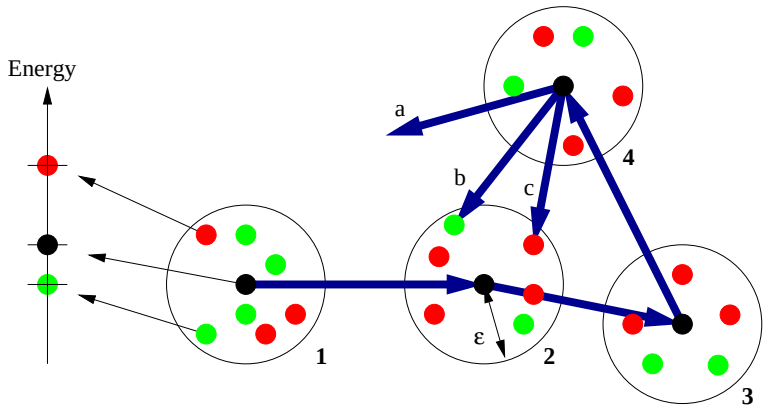


Figure 7.10: *(From [63]).* We keep a list of previously visited structures in a so-called explicit tabu list. Each structure in the explicit tabu list defines a set of implicit tabu structures. Given a structure E in the explicit tabu list, a structure I is said to be implicit tabu if the distance-RMSD (dRMSD) between E and I is less than ϵ and the energy of I is greater than or equal to the energy of E. The adjustable parameter ϵ is called the tabu difference. The figure illustrates a sequence of visited structures (black points) in a solution space. Only the visited structures are inserted in the explicit tabu list. The additional green and red points correspond to structures within ϵ dRMSD of the explicit tabu structures. Green points are structures with lower energy and red points are structures with higher energy than the explicit tabu structure. When choosing a new solution in the neighbourhood three things can happen (as illustrated in the figure), a) A solution is more than ϵ dRMSD away from all explicit tabu structure. b) the solution is within ϵ dRMSD, and the energy is lower than the explicit tabu structure, c) the solution is within ϵ dRMSD, and the energy is higher than the explicit tabu structure. Structures that comply with case c are said to be implicit tabu and cannot be visited. Note that when $\epsilon = 0$ the search heuristic works as a regular TS heuristic since only visited structures become tabu. The use of implicit tabu structures is new in the context of protein structure prediction.

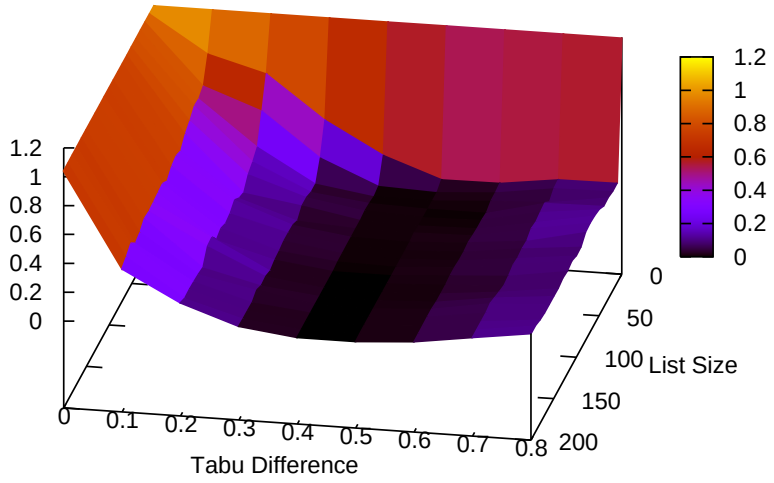


Figure 7.11: Average energy for different combinations of tabu difference and tabu list size.

the tabu difference does not become too large and consequently marks good solutions tabu. In the example shown in the figure, the best value of the tabu difference is between 0.4 and 0.5.

b. The second experiment compares the performance of TS and MC. This is done using the same test framework as in experiment *a* but with different lattices. The results are shown in Figure 7.12. In our paper [63] the y-axis is linear, however, the logarithmic y-axis used here more clearly describes the performance of both algorithms.

c. The purpose of the third experiment is to evaluate the information contents of the CN measure and HSE measure. This is done by using the TS algorithm, such that a number of structures with minimum energy is found. These structures with low energy are then compared to the native structure of the protein in terms of RMSD. In [63] this is done for 5 small proteins. For some of the proteins, we find many different structures with zero or near zero energy which indicates that the energy landscape has many global and local minima.

Based on the experiments, we conclude that the use of implicit tabu structures can increase the performance of TS based search algorithms. Our results also show that TS has a better performance than a typical MC algorithm on this type of problem. However, it is important to note that there are many alternative versions of MC that we have not tested. The HSE vs. CN experiments clearly show that the information contents of HSE is higher than CN. In the paper we also show that the approximate directions of the side-chains are much closer to the native structure for the HSE-optimized structures.

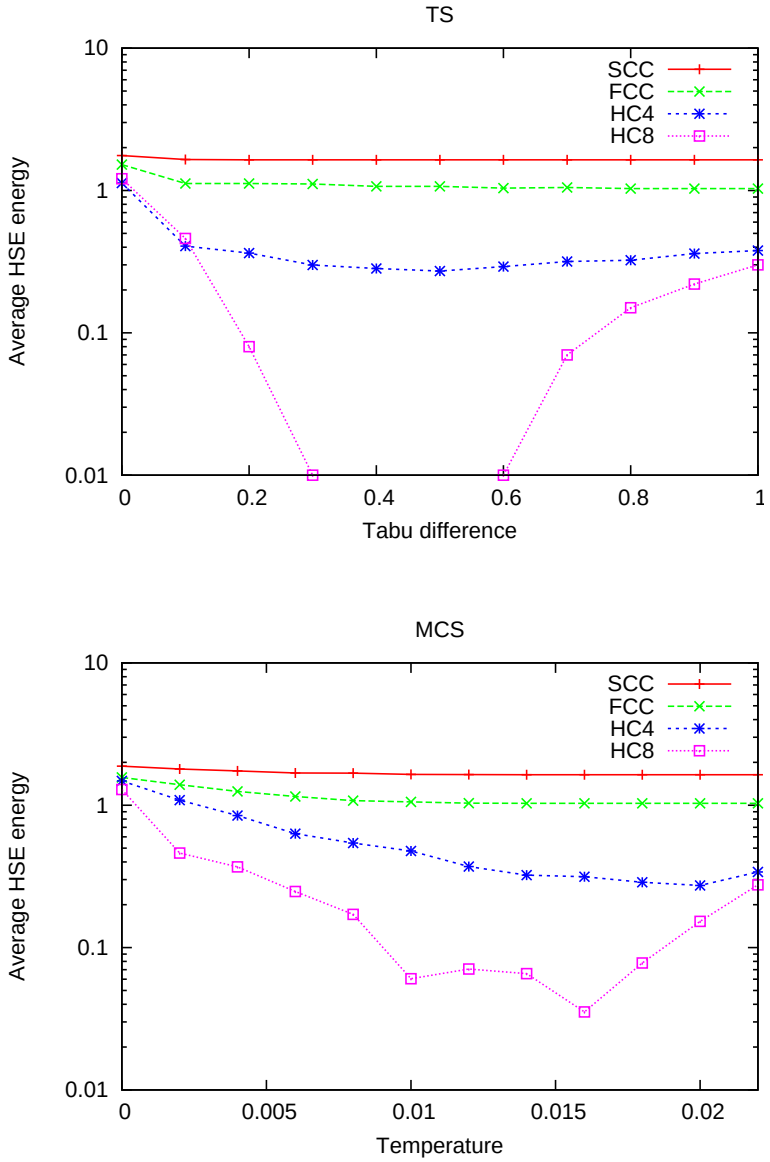


Figure 7.12: Average energy using lattices of different complexity vs. adjustable algorithm parameters. The upper plot shows the average performance of the TS algorithm for a range of tabu differences. For tabu differences 0.4 and 0.5 the average HSE energy is 0 (not shown in a the logarithmic plot). The lower plot shows the performance of the MC algorithm for a range of different temperatures. There are no temperature for the MC algorithm that gives the same performance as the TS algorithm when the tabu difference is between 0.4 and 0.5.

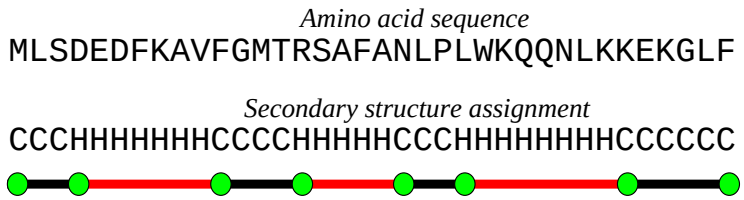


Figure 7.13: The secondary structure is predicted from the amino acid sequence and used for creating segments.

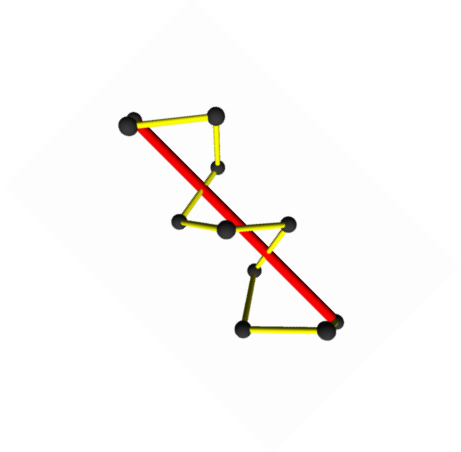


Figure 7.14: A coil segment with valid positions of C_α-atoms.

7.5.2 Paper: Protein Decoy Generation using Branch and Bound with Efficient Bounding

As described in the previous section, the HSE/CN-based energy function has many local minima. Furthermore, the information contained in the HSE or CN measure is not enough to accurately reconstruct C_α-traces of large proteins. In the study described here, we attack these problems by using an exact algorithm that guarantees to find structures with global minimum energy. We also add more predictable information in the form of secondary structure classifications and predicted compactness (radius of gyration).

The Model

The basic idea in our exact approach [66, 67, 65] is to reduce the complexity of the model as illustrated in Figure 7.13. First, the secondary structure of the protein is predicted using PSIPRED [57], then this prediction is used for creating the segments of secondary structure. The purpose of a segment is to define an approximate path in space for the amino acids that it represents. This is done by positioning the first and last amino acids of the segment at the two end points of the segment. Figure 7.14 shows an illustrations of a helix segment together with valid positions of the C_α-atoms of the amino acids.

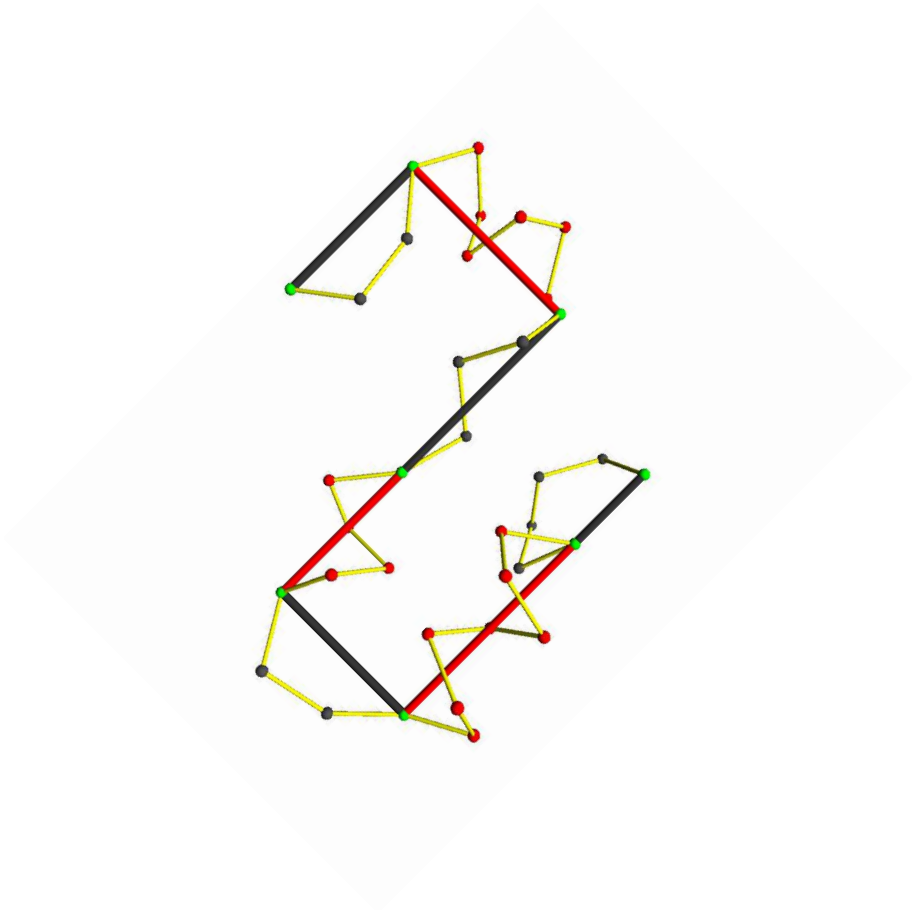


Figure 7.15: An example of a complete structure. In this example two types of secondary structure is represented. The red segments correspond to helices and the black segments correspond to coils.

We discretize our model by allowing these segments to only have a set of predefined directions. There is a trade-off between the number of allowed directions and computational tractability. Our ad-hoc experiments show that we can solve problems in reasonable time when using the 12 uniformly distributed directions from the FCC lattice as described in Figure 7.4(b). In addition to the discretization of the allowed directions of the segments, we also allow a limited set of valid positions of C_α-atoms of a segment. These are called *segment structures*. For helices and sheets, we generate u such segment structures by rotating one structure (having perfect geometry) around the axis defined by the segment. For coil segments, we query a library of coil fragments, to find the most similar sequences and use them as segment structures. A structure represented by this model is called a *complete structure* and is illustrated by the example in Figure 7.15. Refer to [66] for more details about the model and generation of segment structures.

Solving the Problem

Given an amino acid sequence with m segments and u possible segment structures for each segment, the total number of complete structures, N, allowed by this model is

$$N = 4 \times 11^{m-2} \times u^m \tag{7.1}$$

In the equation above, symmetric structures are not counted twice. The total number of complete structures is exponential and too high for a complete enumeration even for small proteins. On the other hand, we have designed a model with many geometric constraints which allows us to compute lower bounds in the branch and bound paradigm efficiently. Our branch and bound algorithm is called: *Efficient Branch and Bound Algorithm* (EBBA) throughout this text. A node in the branch and bound tree corresponds to a partial solution where one or more directions of the segments have been fixed and zero or more segment structures have been fixed. Even though such a partial solution represents a high number of structures, they are heavily geometric constrained. Lower bounds can be computed by taking advantage of these geometric constraints as described in [66, 67].

Experiments and Results

We have tested EBBA on 6 proteins. In all cases we find structures with similar CN and HSE vectors compared with the predicted CN and HSE vectors (example in Figure 7.16). However, even though the CN and HSE vectors matches to some extend, the corresponding structures are not always similar. In other words; the lowest energy structures are in many cases different from the native structures. EBBA is therefore modified such that it returns the 10.000 global minimum energy structures and we show that in this set, good decoys exist for all proteins in our benchmark. EBBA should therefore not directly be used for protein structure prediction, but it is a successful decoy generator compared with other decoy generators in the literature [67, 66].

7.5.3 Paper: Protein Structure Prediction using Bee Colony Optimization Metaheuristic

Our latest approach for protein structure prediction is inspired by swarms of honey bees. This research is work in progress, but contains important results in the context of this study and is therefore included here. The draft of the paper is in Chapter 12 page 151.

In nature, honey bees collect nectar to produce honey in the hive. Honey is the main food source for the bees, so an important task in a bees life is to collect as much nectar as possible. Nectar is produced by flowers in limited amounts. The perfect place for a nectar collecting bee therefore is a flower field. Since bees do not have a map of flower fields in the neighbourhood of their hive, evolution has provided them with a search strategy to maximize the collection of nectar. Before describing this strategy, notice the similarity with the protein structure

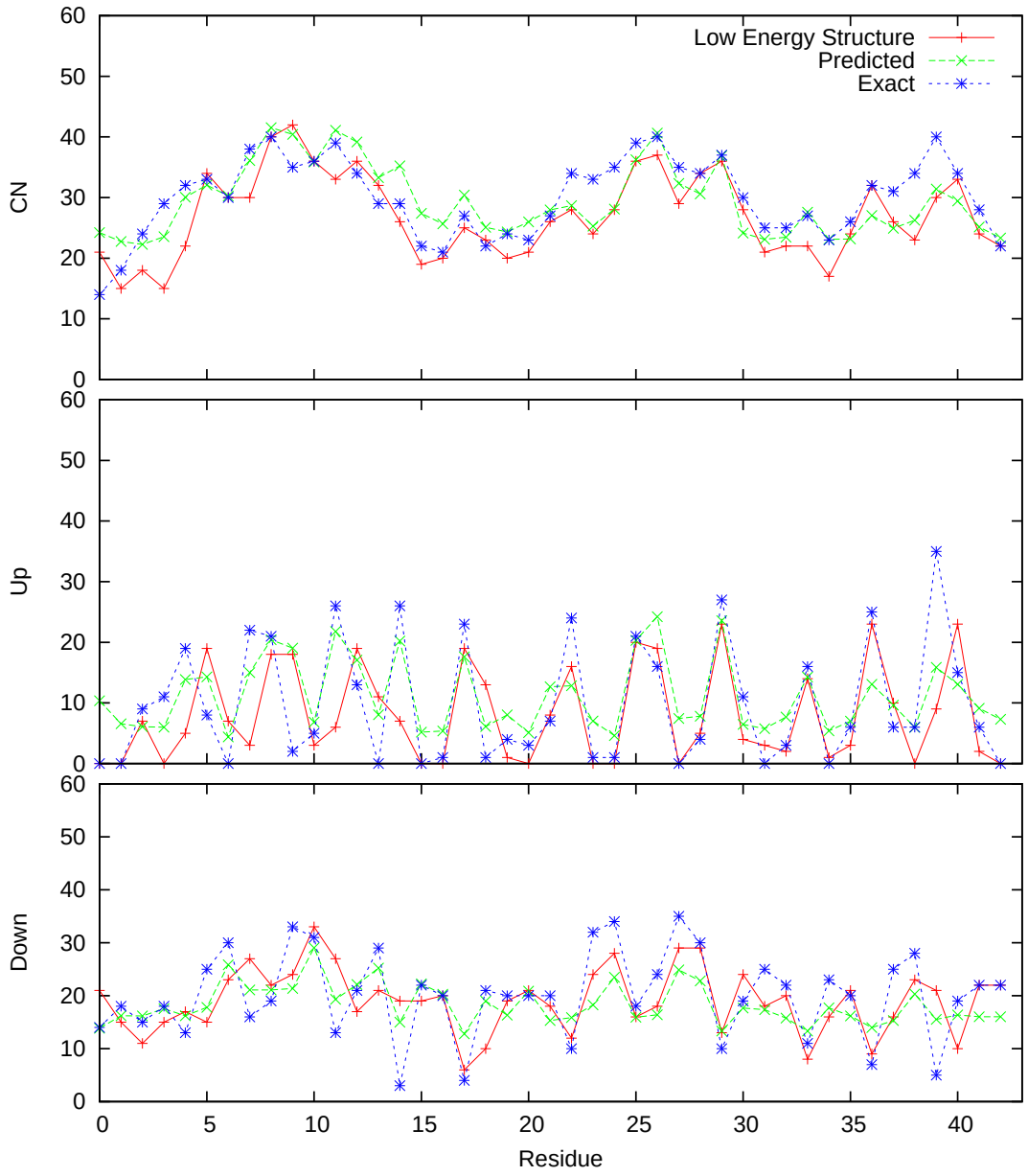


Figure 7.16: An example of CN, up and down vectors for the protein 1FC2. The *low energy structure* is found by EBBA. The *predicted* vector is the prediction from LAKI [92]. The *exact* vector is derived from the native structure and only used for evaluation.

prediction problem or other optimization problems in general. As illustrated in Figure 2.8 page 21, protein structure prediction is about finding the global minimum structure in an energy landscape. Honey bees do not know where the good flower beds are, and we do not know where the minimum energy structures in the conformational landscape are. A simple idea is therefore to apply the bees search strategy to the protein structure prediction problem.

The use of the foraging behaviour of honey bees in combinatorial optimization problems was proposed simultaneously in 2005 by Pham et al. [71] and Karaboga et al. [33]. They later published a number of applications and results of the so-called *bee colony optimization* (BCO) algorithm [72, 36, 35, 34]. In [24] we present our approach for protein structure prediction using BCO. The idea of using BCO for protein structure prediction is not entirely new. In [8] Bahamish et al. used BCO for finding the native state of the 5-residue peptide *met-enkalphin*. The native structure of small polypeptide-chains having only 5 residues is usually not considered to be difficult to predict. Our BCO algorithm is therefore the first algorithm in literature that can handle real-sized proteins (proteins up to 136 residues are considered).

The Bees' Strategy

The basic bee strategy is to organize the swarm such that *many* bees are deployed at high quality flower beds with much nectar and *few* bees search for new flower beds or harvest nectar from low quality flower beds. To accomplish these tasks, the strategies of a bee can roughly be divided in three categories: scouts, workers and onlookers. Here we briefly describe their task in nature and how they correspond to search operations in our BCO algorithm.

- *A **scout** bee in nature:*
  It flies in random directions and eventually finds a flower bed. It collects nectar from the nearby flowers and returns to the hive. In the hive, it performs a so-called *waggle dance* that communicates the amount of nectar in the flower bed and the position of the flower bed to the other bees.

- *A **scout** bee in the algorithm:*
  It corresponds to a valid structure constructed randomly. The energy of the random structure corresponds to the amount of nectar in the flowerbed. The waggle dance corresponds to evaluating the energy and storing the structure in a data structure.

- *An **onlooker** bee in nature:*
  It watches the waggle dances performed by either scout bees or worker bees and decides to collect nectar in a flower bed as shown by one of the waggle dances. If the waggle dance indicates a high quality flower bed, the chance of selecting that particular flower bed is higher.

- *An **onlooker** bee in the algorithm:*
  It is first assigned to an existing protein structure (corresponding to a worker bee or scout bee). This assignment depends on the energy of the

structure such that the chance of being assigned to a low energy structure is higher. Then the onlooker bee is deployed at a structure in the neighbourhood of the assigned structure (possibly found by local search heuristic started at the assigned structure). If the neighbourhood structure is better than the assigned structure, the onlooker replaces the worker bee and becomes a worker bee. Otherwise it flies back to the hive.

- *A **worker** bee in nature:*
  Worker bees are like onlooker bees except that they do not consider other bees waggle-dances. Instead they just fly back to their old flower beds to collect more nectar. If the nectar in their flower bed depletes, they are redeployed as scouts.

- *A **worker** bee in the algorithm:*
  It corresponds to a solution in the conformational space. If the solution is improved (by an onlooker bee) the worker bee is redeployed as either a scout bee or onlooker bee. Otherwise, the worker bee represents the same solution. If some onlooker bee has not improved the site of a worker bee for a pre-specified number of iterations, the site of the worker bee is said to be *exhausted*. In that case, the worker bee is redeployed as a scout bee.

We do not consider scout bees, worker bees and onlooker bees as individual objects in our algorithm. We use the concepts described above to maintain a set of so-called working sites and onlooker sites. This is illustrated in Algorithm 1.

Algorithm 1: BEE-COLONY-OPTIMIZATION

input : S, W, O, $StopT$, $Exhaust$
output: A low energy structure

1 Create $S + W$ working sites by random *(corresponds to deploying S scouts and W worker bees. Worker bees are initially deployed randombly like scouts)*
2 **while** *Stopping criterion is not met* **do**
3 Promote O sites as onlooker sites *(Assign onlooker bees to flower beds using onlooker selection strategy)*
4 **for** *Each onlooker site* **do**
5 Find a neighbourhood site *(using onlooker site improving strategy)*
6 If the neighbourhood site is better than the onlooker site, move the onlooker site to the neighbourhood site. *(corresponds to redeploying the onlooker bee as a worker bee and sending the old worker bee back to the hive*
7 **end**
8 Make all onlooker sites working sites
9 Abandon the S worst working sites and create S new working sites *(using the scout bee strategy)*
10 If a working site has not been improved by an onlooker bee in $Exhaust$ iterations, abandon the working site and construct a new working site *(Corresponds to depletion of nectar and redeployment of worker bees as scout bees)*
11 **end**
12 **return** The best observed working site

Note that step 6 in the algorithm above, might not necessarily be hill climbing which is used here. It would be interesting to test the performance of a strategy where worse solutions can be selected with some probability (i.e. using the Monte Carlo acceptance criteria).

Experiments and Results

We have made experiments where the BCO algorithm is run on the same model as used in EBBA (Section 7.5.2). BCO often finds the optimal solution faster than EBBA - but not always. There are also examples where BCO does not find optimal structures in the 48 hours time limit (optimal solutions were found by EBBA in less than 48 hours for the proteins tested). When using such low complexity models, that can be solved to optimality in reasonable time, EBBA is therefore the preferred algorithm. However, EBBA cannot solve large problems in reasonable time (in terms of model complexity and protein length). If we use a higher complexity model by increasing the allowed directions and rotations, the BCO algorithm is able to find structures with better energy than EBBA.

We have also compared the BCO algorithm with a simple simulated annealing algorithm which uses the same move set as BCO and the cooling scheme is chosen such that it spends the same amount of time as the BCO algorithm (48 hours). The results show that BCO outperform SA by finding structures with

lower energy than BCO. Refer to [24] Chapter 12, page 151 for a table of the results.

7.6 Chapter Summary

When treating the protein structure prediction problem as a combinatorial optimization problem, we typically need to discretize the conformations of the polypeptide chain. Different discretizations have been proposed in the literature; some use a predefined set of allowed angles of the ϕ and ψ angles. Others, confine the C_α-atoms to be positioned on a lattice. Even though discretization gives a reduced and finite number of possible structures, it is often not feasible to find the minimum energy structure using complete enumeration because of the exponential number of structures. It has also been shown that even one of the simplest formulations of the protein structure prediction problem (the HP-model) is NP-hard. In this chapter we show examples of heuristic and exact algorithms for solving various formulations of the protein structure prediction problem. We also briefly introduce our own algorithms for protein structure prediction that are based on techniques from combinatorial optimization.

Chapter 8

Conclusions and Future Directions

The protein structure prediction problem remains a very difficult problem to solve. There has been some progress and successes in the field. However, these are mainly for proteins with homologue counterparts in PDB. While homology based algorithms usually improve when the PDB grows, it is generally not the case for *de novo* prediction algorithms. In my oppinion, structure prediction of template *free* proteins is much more interesting and intellectual challenging than structure prediction of template based proteins. If we want to predict the structure of template free proteins, we have to learn the real mechanisms behind protein folding.

While it is not possible to state exactly why the protein structure prediction problem is so difficult, we know of at least two sub problems that must be solved. One sub problem is to find a computational tractable energy function that approximates the natural energy reasonably well. The other problem is to develop a search algorithm that finds the low energy structures in a huge conformational space. Here, these sub problems are stated as being two seperate problems. However, it is quite possible that they are very intertwined. In the exact algorithm developed during this study, we are able to implicitly search the whole conformational space. However, this can not be done for arbitrary energy functions, so in this case, the energy function and the search algorithm cannot be considered seperately.

I am convinced that the protein structure prediction problem will be solved, such that the native state of *any* amino acids sequence can be predicted with high accuracy. This will probably take some decades of research and perhaps require new computational paradigms. While there are some progresses from year to year, it is not clear if the solution to the protein structure prediction problem will come from many small improvements or one revolutionary idea. I therefore think that it is important to support high risk science in this field, such that untraditional approaches can be developed and tested.

8.1 Main Contributions

The main contributions to the field of protein structure prediction and model quality assessment in this study, in a non-prioritized order, are:

- We show that the half-sphere-exposure measure is more information rich compared to the traditional contact number measure [63].
- We have proposed a new tabu definition and we show that it gives a better performing search algorithm compared to the traditional tabu definition [63].
- We have proposed a discrete and flexible model for representing C_α-traces [66, 67].
- We have developed a branch and bound algorithm (EBBA) that is able to find the lowest energy solutions in this discrete model in reasonable time [66, 67]. This is mainly because of the efficient computation of lower bounds.
- We have developed a heuristic algorithm based on the foraging behaviour of bees to find low energy structures in the discrete model [24].
- We show how to extract distance constraints from alignments and use them for model quality assessment [64].
- We show how to select a good subset of those distance constraints using information from distributions of contact number probabilities [64].

8.2 Future Directions

The future directions of this study are many and only a few of the most promising ideas are listed here.

- Even though EBBA is a *de novo* algorithm, it could be interesting to make use of techniques from homology modeling. One simple idea is to detect the best template in PDB and fix the conserved segments. EBBA should therefore only work on the unconserved parts which eventually results in a much simpler problem. Another, more flexible, way of using homology techniques is to use the distance constraints found by our MQA algorithm.
- Improvements of the energy function of EBBA could improve the quality of the global minimum structures considerably. It would therefore be interesting to use an additional physics based energy function. However, it is not trivial to compute tight lower bounds for energy functions based on residue pairwise functions, so this would require more research.
- The optimization algorithm for selecting a good subset of the distance constraints could be improved considerably. The current algorithm is based

on a greedy approach that finds a local optimum. We should test if improving the optimization algorithm eventually would improve the MQA algorithm.

In my opinion, the most promising idea is the improvements of EBBA. Exact algorithms for protein structure prediction have received very little attention in the literature. This is probably because they are more difficult to develop than heuristic algorithms. However, we basically show that it *is* possible to implicitly sample the whole conformational space with a proper discretization and therefore attack the second major problem described in Chapter 3. A more detailed energy function that allows for tight lower bound computations is therefore on top of my wish list.

Bibliography

[1] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *J Mol Biol*, 215(3):403–410, October 1990.

[2] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucleic Acids Res*, 25(17):3389–3402, September 1997.

[3] C. Andrieu, N. de Freitas, A. Doucet, and M. Jordan. An introduction to MCMC for machine learning, 2003.

[4] I. P. Androulakis, C. D. Maranas, and C. A. Floudas. αBB: A global optimization method for general constrained nonconvex problems. *Journal of Global Optimization*, 7(4):337–363, 1995.

[5] C. B. Anfinsen. Principles that govern the folding of protein chains. *Science*, 181(96):223–230, July 1973.

[6] J. Archie and K. Karplus. Applying Undertaker cost functions to model quality assessment. 2008. Manuscript in preparation.

[7] R. Backofen and W. Sebastian. A constraint-based approach to fast and exact structure prediction in three-dimensional protein models. *Constraints*, 11(1):5–30, January 2006.

[8] H. A. A. Bahamish, R. Abdullah, and R. A. Salam. Protein conformational search using bees algorithm. In *Asia International Conference on Modelling and Simulation*, pages 911–916. IEEE Computer Society, 2008.

[9] P. Baldi and S. Brunak. *Bioinformatics: The Machine Learning Approach, Second Edition (Adaptive Computation and Machine Learning)*. The MIT Press, August 2001.

[10] L. E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics*, 41(1):164–171, 1970.

[11] W. Boomsma and T. Hamelryck. Full cyclic coordinate descent: Solving the protein loop closure problem in Calpha space. *BMC Bioinformatics*, 6, June 2005.

[12] C. Branden and J. Tooze. *Introduction to Protein Structure*. Garland Publishing, January 1999.

[13] M. Brazil, D. A. Thomas, J. F. Weng, and M. Zachariasen. Canonical forms and algorithms for Steiner trees in uniform orientation metrics. *Algorithmica*, 44:281–300, 2006.

[14] A. A. Canutescu, A. A. Shelenkov, and R. L. Dunbrack. A graph-theory algorithm for rapid protein side-chain prediction. *Protein Sci*, 12(9):2001–2014, September 2003.

[15] S. Chakravarty and R. Varadarajan. Residue depth: A novel parameter for the analysis of protein structure and stability. *Structure*, 7(7):723–32, 1999.

[16] C. Chothia and A. M. Lesk. The relation between the divergence of sequence and structure in proteins. *EMBO J*, 5(4):823–826, April 1986.

[17] M. L. Connolly. Solvent-accessible surfaces of proteins and nucleic acids. *Science*, 221(4612):709–13, 1983.

[18] S. Cristobal, A. Zemla, D. Fischer, L. Rychlewski, and A. Elofsson. A study of quality measures for protein threading models. *BMC Bioinformatics*, 2(5), 2001.

[19] K. A. Dill and H. S. Chan. From Levinthal to pathways to funnels. *Nat Struct Biol*, 4(1):10–19, January 1997.

[20] M. Dorigo, V. Maniezzo, and A. Colorni. The ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics*, 26(1):29–41, 1996.

[21] J. Ellis. Proteins as molecular chaperones. *Nature*, 328:378–379, July 1987.

[22] O. Eriksson, Y. Zhou, and A. Elofsson. Side chain-positioning as an integer programming problem. In *WABI '01: Proceedings of the First International Workshop on Algorithms in Bioinformatics*, pages 128–141, London, UK, 2001. Springer-Verlag.

[23] V. A. Eyrich, D. M. Standley, A. K. Felts, and R. A. Friesner. Protein tertiary structure prediction using a branch and bound algorithm. *Proteins*, 35(1):41–57, 1999.

[24] R. Fonseca, M. Paluszewski, and P. Winter. Protein structure prediction using bee colony optimization metaheuristic. *Work in progress.*

[25] M. Gendreau, A. Hertz, and G. Laporte. A tabu search heuristic for the vehicle routing problem. *Manage. Sci.*, 40(10):1276–1290, October 1994.

[26] F. Glover and M. Laguna. Tabu search. In C. Reeves, editor, *Modern Heuristic Techniques for Combinatorial Problems*, Oxford, England, 1993. Blackwell Scientific Publishing.

[27] J. Greer and B. L. Bush. Macromolecular shape and surface maps by solvent exclusion. *Proc Natl Acad Sci U S A*, 75(1):303–7, 1978.

[28] A. V. Guzzo. The influence of amino acid sequence on protein structure. *Biophysical Journal*, 5:809–822, 1965.

[29] T. Hamelryck. An amino acid has two sides: A new 2D measure provides a different view of solvent exposure. *Proteins*, 59(1):38–48, April 2005.

[30] W. E. Hart and S. Istrail. Fast protein folding in the hydrophobic-hydrophilic model within three-eights of optimal. In *STOC '95: Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 157–168, New York, NY, USA, 1995. ACM.

[31] R. Hwang, D. Richards, and P. Winter. *The Steiner Tree Problem*, volume 53. North-Holland, Annals of Discrete Mathematics, 1992.

[32] G. Jayachandran, V. Vishal, and V. S. Pande. Using massively parallel simulation and markovian models to study protein folding: Examining the dynamics of the villin headpiece. *The Journal of chemical physics*, 124(16), April 2006.

[33] D. Karaboga. An idea based on honey bee swarm for numerical optimization technical report-TR06. Technical report, Erciyes University, Engineering Faculty, Computer Engineering Department, November 2005.

[34] D. Karaboga, B. Akay, and C. Ozturk. Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks. In *MDAI*, pages 318–329, 2007.

[35] D. Karaboga and B. Basturk. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *J. of Global Optimization*, 39(3):459–471, 2007.

[36] D. Karaboga and B. Basturk. On the performance of artificial bee colony (ABC) algorithm. *Appl. Soft Comput.*, 8(1):687–697, 2008.

[37] K. Karplus, S. Katzman, G. Shackleford, M. Koeva, J. Draper, B. Barnes, M. Soriano, and R. Hughey. SAM-T04: What's new in protein-structure prediction for CASP6. *Proteins*, September 2005.

[38] Sol Katzman, Christian Barrett, Grant Thiltgen, Rachel Karchin, and Kevin Karplus. Predict-2nd: A tool for generalized local structure prediction. 2008. Manuscript in preparation.

[39] S. K. Kearsley. On the orthogonal transformation used for structural comparisons. *Acta Cryst.*, 1989.

[40] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems*. Springer, Berlin, Germany, 2004.

[41] A. R. Kinjo and K. Nishikawa. Recoverable one-dimensional encoding of three-dimensional protein structures. *Bioinformatics*, 21(10):2167–70, 2005.

[42] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science, Number 4598, 13 May 1983*, 220, 4598:671–680, 1983.

[43] A. Krogh, M. Brown, I. S. Mian, K. Sjölander, and D. Haussler. Hidden Markov models in computational biology. Applications to protein modeling. *J Mol Biol*, 235(5):1501–1531, February 1994.

[44] A. H. Land and A. G. Doig. An automatic method of solving discrete programming problems. *Econometrica*, 28(3):497–520, 1960.

[45] G. Laporte and I. H. Osman. Routing problems: A bibliography. *Annals of Operations Research*, V61(1):227–262, December 1995.

[46] R. H. Lathrop. The protein threading problem with sequence amino acid interaction preferences is NP-complete. *Protein Eng*, 7(9):1059–1068, September 1994.

[47] R. H. Lathrop and T. F. Smith. Global optimum protein threading with gapped alignment and empirical pair score functions. *Journal of Molecular Biology*, 255(4):641–665, February 1996.

[48] K. F. Lau and K. A. Dill. A lattice statistical mechanics model of the conformational and sequence spaces of proteins. 22(10):3986–3997, 1989.

[49] B. Lee and F. Richards. The interpretation of protein structures: Estimation of static accessibility. *J Mol Biol*, 55:379–400, 1971.

[50] J. E. Lennard-Jones. Cohesion. *Proceedings of the Physical Society*, 43(5):461–482, 1931.

[51] C. Levinthal. Are there pathways for protein folding? *J Chim Phys*, 65:44–45, 1968.

[52] P. N. Lewis, N. Go, M. Go, D. Kotelchuck, and H. A. Scheraga. Helix probability profiles of denatured proteins and their correlation with native structures. *PNAS*, 65(4), 1970.

[53] R. M. Mahmood and Sadiq M. S. A parallel tabu search algorithm for optimizing multiobjective VLSI placement.

[54] C. D. Maranas and C. A. Floudas. Global minimum potential energy conformations of small molecules. *Journal of Global Optimization*, 4:135 – 170, 1994.

[55] M. A. Marti-Renom, A. C. Stuart, A. Fiser, R. Sánchez, F. Melo, and A. Sali. Comparative protein structure modeling of genes and genomes. *Annu Rev Biophys Biomol Struct*, 29:291–325, 2000.

[56] B. W. Matthews. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442–451, October 1975.

[57] L. J. Mcguffin, K. Bryson, and D. T. Jones. The PSIPRED protein structure prediction server. *Bioinformatics*, 16(4):404–405, April 2000.

[58] N. Metropolis and S. Ulam. The Monte Carlo method. *Journal of the American Statistical Association*, 44(247):335–341, 1949.

[59] A. S. Moffat. X-RAY CRYSTALLOGRAPHY: Opening the door to more membrane protein structures. *Science*, 277(5332):1607–1608, 1997.

[60] J. Moult, K. Fidelis, B. Rost, T. Hubbard, and A. Tramontano. Critical assessment of methods of protein structure prediction (CASP)–round 6. *Proteins*, 61 Suppl 7:3–7, 2005.

[61] M. T. Oakley, J. M. Garibaldi, and J. D. Hirst. Lattice models of peptide aggregation: Evaluation of conformational search algorithms. *Journal of Computational Chemistry*, 26(15):1638 – 1646, 2005.

[62] M. Paluszewski, T. Hamelryck, and P. Winter. Protein structure prediction using tabu search and half-sphere exposure measure. *RECOMB (poster)*, 2006.

[63] M. Paluszewski, T. Hamelryck, and P. Winter. Reconstructing protein structure from solvent exposure using tabu search. *Algorithms for Molecular Biology*, 1:20+, October 2006.

[64] M. Paluszewski and K. Karplus. MQA using distance constraints from alignments. *Proteins, Structure, Function and Bioinformatics, (accepted, to appear)*, 2008.

[65] M. Paluszewski and P. Winter. Branch and bound algorithm for protein structure prediction using efficient bounding. *RECOMB (poster)*, 2007.

[66] M. Paluszewski and P. Winter. EBBA: Efficient branch and bound algorithm for protein decoy generation. *Technical report. Department of Computer Science, Univ. of Copenhagen*, 08(08), 2008.

[67] M. Paluszewski and P. Winter. Protein decoy generation using branch and bound with efficient bounding. *Proc. of 8th International Workshop on Algorithms in Bioinformatics, WABI*, 2008.

[68] M. Paluszewski, P. Winter, and M. Zachariasen. A new paradigm for general architecture routing. *Great Lakes Symposium on VLSI, Proceedings of the 14th ACM Great Lakes Symposium on VLSI*, pages 202 – 207, 2004.

[69] P. M. Pardalos, X. Liu, and G. L. Xue. Protein conformation of a lattice model using tabu search. *Journal of Global Optimization*, 11(1):55–68, 1997.

[70] W. R. Pearson. Rapid and sensitive sequence comparison with FASTP and FASTA. *Methods Enzymol*, 183:63–98, 1990.

[71] D. T. Pham, A. Ghanbarzadeh, E. Koc, S. Otri, S. Rahim, and M. Zaidi. The bees algorithm. Technical report, Manufacturing Engineering Centre, Cardiff University, UK, 2005.

[72] D. T. Pham, E. Koc, A. Ghanbarzadeh, S. Otri, S. Rahim, M. Zaidi, J. Phrueksanant, J. Lee, S. Sahran, M. Sholedolu, M. Ridley, M. Mahmuddin, H. Al-Jabbouli, A. H. Darwish, A. Soroka, M. Packianather, and M. Castellani. The bees algorithm - a novel tool for optimisation problems. Technical report, Cardiff University - Manufacturing Engineering Centre (MEC), 2006.

[73] A. Pintar, O. Carugo, and S. Pongor. Atom depth as a descriptor of the protein interior. *Biophys J*, 84(4):2553–61, 2003.

[74] A. Pintar, O. Carugo, and S. Pongor. Atom depth in protein structure and function. *Trends Biochem Sci*, 28(11):593–7, 2003.

[75] G. Pollastri, P. Baldi, P. Fariselli, and R. Casadio. Prediction of coordination number and relative solvent accessibility in proteins. *Proteins*, 47(2):142–53, 2002.

[76] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in Fortran*. Cambridge University Press, January 1992.

[77] V. Pureza and R. Morabito. Some experiments with a simple tabu search algorithm for the manufacturer's pallet loading problem. *Computers & Operations Research*, 33(3):804–819, March 2006.

[78] N. Qian and T. J. Sejnowski. Predicting the secondary structure of globular proteins using neural network models. *J Mol Biol*, 202(4):865–884, August 1988.

[79] J. Qiu, W. Sheffler, D. Baker, and W. S. Noble. Ranking predicted protein structures with support vector regression. 71(3):1175–1182, 2008.

[80] R. Rojas. *Neural Networks: A Systematic Introduction*. Springer-Verlag New Your, Inc., 1996.

[81] B. Rost. Review: Protein secondary structure prediction continues to rise. *Journal of Structural Biology*, 134(2-3):204–218, May 2001.

[82] B. Rost and C. Sander. Prediction of protein secondary structure at better than 70% accuracy. *Journal of Molecular Biology*, 232(2):584–599, July 1993.

[83] A. Sali and T.L. Blundell. Comparative protein modelling by satisfaction of spatial restraints. *Journal of Molecular Biology*, 234(3):779–815, December 1993.

[84] A. Sali, E. Shakhnovich, and M. Karplus. Kinetics of protein folding. a lattice model study of the requirements for folding to the native state. *J Mol Biol*, 235(5):1614–1636, February 1994.

[85] O. Schueler-Furman, C. Wang, P. Bradley, K. Misura, and D. Baker. Progress in Modeling of Protein Structures and Interactions. *Science*, 310(5748):638–642, 2005.

[86] A. Shmygelska and H. Hoos. An ant colony optimisation algorithm for the 2D and 3D hydrophobic polar protein folding problem. *BMC Bioinformatics*, 6(1), 2005.

[87] K. T. Simons, C. Kooperberg, E. Huang, and D. Baker. Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions. *J Mol Biol*, 268(1):209–25, 1997.

[88] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197, March 1981.

[89] L. Song and A. Vanelli. A VLSI placement method using tabu search.

[90] R. H. Swendsen and J. S. Wang. Replica Monte Carlo simulation of spin-glasses. *Physical Review Letters*, 57(21):2607+, November 1986.

[91] A. M. Turing. Computing machinery and intelligence. *Mind*, 59:433–460, 1950.

[92] B. Vilhjalmsson and T. Hamelryck. Predicting a new type of solvent exposure. ECCB, Computational Biology Madrid 05, P-C35, Poster, 2005.

[93] P. Walian, T. A. Cross, and B. K. Jap. Structural genomics of membrane proteins. *Genome Biology*, 5, 2004.

[94] B. Wallner and A. Elofsson. Pcons5: Combining consensus, structural evaluation and fold recognition scores. *Bioinformatics*, 21(23):4248–4254, December 2005.

[95] B. Wallner and A. Elofsson. Prediction of global and local model quality in CASP7 using Pcons and ProQ. *Proteins*, 69(Suppl. 8), 2007.

[96] L. Wang and T. Jiang. On the complexity of multiple sequence alignment. *J. Comput. Biol.*, 1(4):337–348, 1994.

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

[98] Y. Zhang. I-TASSER server for protein 3D structure prediction. *BMC Bioinformatics*, 9:40+, January 2008.

Chapter 9

Paper: Reconstructing Protein Structure From Solvent Exposure using Tabu Search

M. Paluszewski, T. Hamelryck, and P. Winter. Reconstructing Protein Structure from Solvent Exposure using Tabu Search. *Algorithms for Molecular Biology*, 1:20+, October 2006.
Status: Published

Algorithms for Molecular Biology BioMed Central

Research

Open Access

Reconstructing protein structure from solvent exposure using tabu search

Martin Paluszewski*[1], Thomas Hamelryck[2] and Pawel Winter[1]

Address: [1]Department of Computer Science, University of Copenhagen, Universitetsparken 1, 2100 Copenhagen, Denmark and [2]Bioinformatics Center, Institute of Molecular Biology, University of Copenhagen, Universitetsparken 15 building 10, 2100 Copenhagen, Denmark

Email: Martin Paluszewski* - palu@diku.dk; Thomas Hamelryck - thamelry@binf.ku.dk; Pawel Winter - pawel@diku.dk

* Corresponding author

Published: 27 October 2006

Algorithms for Molecular Biology 2006, **1**:20 doi:10.1186/1748-7188-1-20

Received: 30 March 2006
Accepted: 27 October 2006

This article is available from: http://www.almob.org/content/1/1/20

Abstract

Background: A new, promising solvent exposure measure, called *half-sphere-exposure* (HSE), has recently been proposed. Here, we study the reconstruction of a protein's C_α trace solely from structure-derived HSE information. This problem is of relevance for *de novo* structure prediction using predicted HSE measure. For comparison, we also consider the well-established contact number (CN) measure. We define energy functions based on the HSE- or CN-vectors and minimize them using two conformational search heuristics: *Monte Carlo simulation* (MCS) and *tabu search* (TS). While MCS has been the dominant conformational search heuristic in literature, TS has been applied only a few times. To discretize the conformational space, we use lattice models with various complexity.

Results: The proposed TS heuristic with a novel tabu definition generally performs better than MCS for this problem. Our experiments show that, at least for small proteins (up to 35 amino acids), it is possible to reconstruct the protein backbone solely from the HSE or CN information. In general, the HSE measure leads to better models than the CN measure, as judged by the RMSD and the angle correlation with the native structure. The angle correlation, a measure of structural similarity, evaluates whether equivalent residues in two structures have the same general orientation. Our results indicate that the HSE measure is potentially very useful to represent solvent exposure in protein structure prediction, design and simulation.

Background

The extent to which an amino acid in a protein is accessible to the surrounding solvent is highly dependent on the type of amino acid. In general, hydrophilic amino acids tend to be near the solvent accessible surface, while hydrophobic amino acids tend to be buried in the core of the protein. To measure this effect, several solvent exposure measures have been proposed [1-7], and one of these is the *contact number measure* (CN) [7]. The CN of a residue is the number of C_α atoms in a sphere centered at the C_α atom of the residue in question (Figure 1). The CN of all residues of a protein is called the *CN vector*. The CN vector is well conserved and can be predicted with high accuracy [8].

Recently, a new promising solvent exposure measure, called *half-sphere-exposure* (HSE), has been proposed [9]. While the CN measure uses a single sphere centered at the C_α atom, the HSE measure considers two hemispheres. Two values, an *up* and a *down* value, are associated with

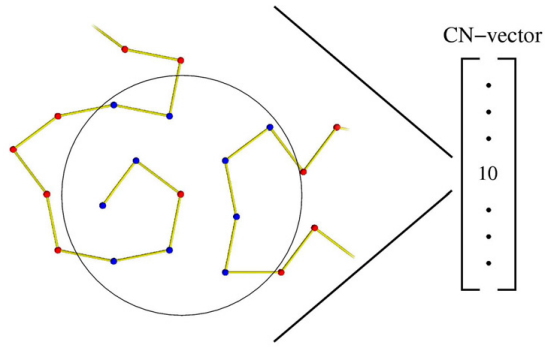


Figure 1
CN. The contact number (CN) of a residue.

each residue, corresponding to the upper and lower hemisphere. The geometry of the HSE construction is shown schematically in Figure 2. The up and down HSE values measure two fundamentally different environments of an amino acid, one of them corresponding to the neighbourhood of the side chain [9]. The HSE measure compares favorably with other solvent exposure measures in terms of computational complexity, sensitivity, correlation with the stability of mutants and conservation. An important advantage of the HSE measure is that it can be calculated from C_α-only or other simplified protein models. Therefore, it forms an attractive alternative to the use of the CN measure in protein structure prediction methods [10].

Here, we study if it is possible to reconstruct a protein's C_α trace solely from a CN vector or an HSE vector. These vectors are obtained from the protein's known native state

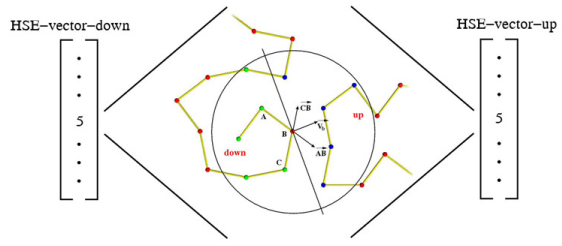


Figure 2
HSE. Given the positions of 3 consecutive C_α atoms (*A*, *B*, *C*), the approximate side-chain direction $\vec{V}_b$ can be computed as the sum of $\overrightarrow{AB}$ and $\overrightarrow{CB}$. The plane perpendicular to $\vec{V}_b$ cuts the sphere centered at *B* in an upper and a lower hemisphere.

and our goal is therefore to evaluate the information contents of these measures. This problem could become important for *de novo* structure prediction, for example if predicted HSE values are used as restraints. Preliminary results show that the HSE measure can be predicted with reasonably high accuracy[11]. Reconstruction of a protein structure from a predicted HSE vector might thus be an attractive way of approaching the sequence-to-structure problem. Predicted CN-/HSE vectors are expected to have errors compared to the exact vectors. The results presented in this paper are based on exact vectors and therefore provide an upper bound on the information contents of predicted CN-/HSE vectors. If protein structure prediction was carried out on a *predicted* HSE vector only, it is expected that the results would not be better than the results presented in this paper. It would therefore be natural to add other predictable information such as secondary structure, radius of gyration etc. to a structure prediction system using predicted HSE vectors. The problem of reconstructing protein structure from vectors of one-dimensional structural information has been studied before. Kinjo et al.[12] used exact vectors of *secondary structure* (SS), CN and *residue-wise contact order* (RWCO) together with refinement using the AMBER force field to reconstruct native like structures. Their results show that SS and CN information without the use of RWCO is not enough to reconstruct native like structures. Unfortunately, prediction methods for the RWCO measure only have moderate performance as compared to SS and CN[12].

Porto et al.[13] described an algorithm for reconstructing the contact map (CM) from its principal (one-dimensional) eigenvector. However, methods for predicting a high quality eigenvector are not likely to exist. Here, we only consider measures that potentially can be predicted with high accuracy. Furthermore we only use one type of measure (either CN or HSE), which is important for evaluating the information content of a measure. To this end, we compare structure reconstruction using an energy function based on the HSE measure with an energy function that uses the well-established CN measure.

If an approximate CN-/HSE vector is obtained from a prediction method, there might be no structure that exactly realizes the vector. In that case, we are interested in finding a structure with a CN- or HSE-vector *similar* to the predicted vector. Therefore we define energy functions based on the HSE- or CN-vectors and minimize them using two conformational search heuristics: *Monte Carlo simulation* (MCS) and *tabu search* (TS). MCS has been widely used for protein structure prediction, and TS has been applied with great success to many optimization problems, but has rarely been used for protein structure prediction [14-16].

In this article, the radius of the HSE sphere is chosen to be 12 Å for all experiments. The *optimal* radius has yet to be determined, both in terms of predictability and reconstructability. If the radius is too small, important residue pairs might be overlooked. On the other hand, if the radius is too large, many irrelevant residues are considered. In this respect, 12 Å seems to be a good compromise [9].

The rest of the article is organized as follows. In the next section we describe the energy function based on the HSE measure. Then the protein abstraction and lattice model are discussed. In section *Heuristics*, we present the two conformational search heuristics, MCS and TS. In section *Lattice experiments*, MCS and TS are evaluated in lattices of different complexity. Finally, we evaluate the information content (that is, to what extent they can be used to reconstruct a protein structure) of the HSE and CN measures using TS and a high complexity lattice.

HSE energy function

The similarity of two HSE vectors A and B of length N can be measured using the following RMS deviation:

$$\mathrm{RMSD}(A,B) = \sqrt{\frac{\sum_{i=1}^{N}((A_{u_i} - B_{u_i})^2 + (A_{d_i} - B_{d_i})^2)}{2N}},$$

where $\{A,B\}_{u_i}$ and $\{A,B\}_{d_i}$ are the up and down values of the i'th index. RMSD (*A*, *B*) can be used to describe the *energy* of structure S_A where A is the HSE vector of S_A and B is the HSE vector of the native structure. These definitions are easily extended to the CN measure. The energy functions are the only optimization criteria used by the MCS and TS algorithms.

The protein model

The HSE and CN energy functions only depend on the positions of the C_α atoms in the protein backbone. This allows us to simplify the problem by considering a protein as a chain of connected points representing the positions of the C_α atoms. Furthermore, to reduce and discretize the conformational space of the protein, we require the C_α atoms of the chain to be positioned on a 3D lattice. A lattice can be defined as a set of basis vectors corresponding to the directions to the neighbouring nodes. The basis vectors of the *simple cubic lattice* (SCC) are the cyclic permutations of [± 1,0,0] ([1,0,0], [-1,0,0], [0,1,0], [0,-1,0], [0,0,1], [0,0,-1]) and the basis vectors of the *face centered cubic lattice* (FCC) are the cyclic permutations of [± 1, ± 1,0] ([1,1,0], [1,0,1], [1,-1,0], [1,0,-1], [-1,1,0], [-1,0,1], [-1,-1,0], [-1,0,-1], [0,1,1], [0,1,-1], [0,-1,1], [0,-1,-1]). This gives 6 basis vectors for SCC and 12 for FCC as illustrated in Figure 3. The length of an edge between two neighbouring nodes is taken to be 3.8 Å which is the average distance between two consecutive C_α atoms in proteins.

Lattice models are widely used for studying the fundamental properties of protein structure[17]. Such models have for example provided invaluable insights on topics such as the validity of pairwise energy functions[18], the evolution of protein superfamilies[19] and the importance of local structural bias in the determination of a protein's fold[20]. Many lattice models have been proposed and evaluated in the literature. Not surprisingly, experi-

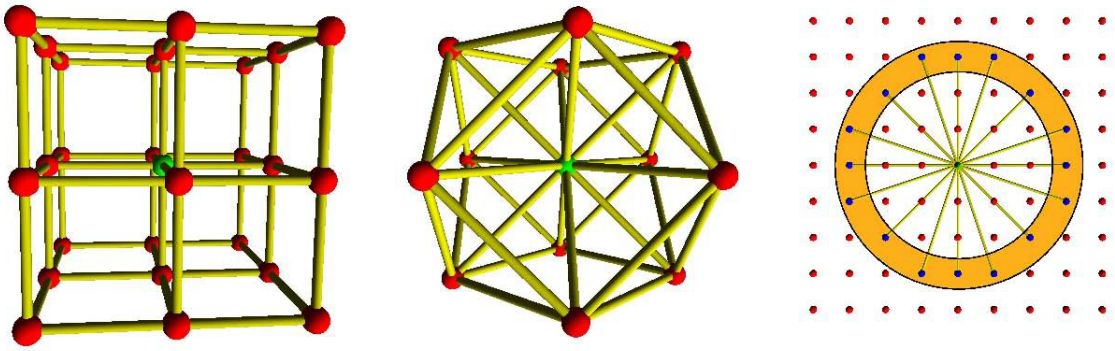


Figure 3
Lattices. Interior nodes of the SCC and FCC lattices are connected to respectively 6 and 12 neighbouring nodes. Nodes of high coordination lattices have many neighbours because of variable edge size.

Algorithms for Molecular Biology 2006, **1**:20 http://www.almob.org/content/1/1/20

ments show a high correlation between the number of basis vectors of a lattice and its ability to represent a protein backbone[21,22]. When deciding on a lattice model, one must always consider the trade-off between the reduction of the conformational space and the quality of the structure representation. Therefore, in section *Lattice experiments* we evaluate four different lattices of various complexity: The SCC lattice, the FCC lattice and two *high coordination* (HC) lattices with 54 and 390 basis vectors, respectively.

A high coordination lattice has an underlying cubic lattice with unit length less than $3.8/N$ Å for some integer $N > 1$. Cubic lattice points are connected in the high coordination lattice if their Euclidean distance is between $3.8 \pm \varepsilon$ for some $\varepsilon > 0$. The high coordination lattices used here are named HC4 and HC8 corresponding to their N value (4 and 8). The ε value is 0.2 for all HC lattices. Figure 3 shows an illustration of a 2D high coordination lattice with $N = 3$ and $\varepsilon = 0.4$. High coordination lattices have previously been used for protein structure prediction[23,24]. Note that the SCC and FCC lattices both have the excluded volume property, meaning that atoms at two different lattice points will never collide. This property does not necessarily hold for high coordination lattices, and collisions must therefore be detected explicitly.

Heuristics

We apply two iterative search heuristics for minimization of the HSE energy. One of them is the tabu search *metaheuristic* proposed by F. Glover in 1989[25,26]. A metaheuristic is a general framework that can be specialized to solve various optimization problems. For many problems in Operations Research (OR), tabu search is the metaheuristic of choice. However, for protein structure prediction, tabu search has only been given a modest amount of attention[14-16].

In Algorithm 1 and 2 (Figures 5 and 6) the pseudo code for tabu search is shown. TS is basically a local improvement heuristic where the best structure in a neighbourhood is repeatedly selected. However, memory is used to prevent cycling in local minima. A previous TS implementation [16] inserts visited structures into a *tabu list* and only consider new structures if they are not in the tabu list. We have found that extending the tabu definition improves the performance considerably. Here, we still keep a list of previously visited structures in a so-called *explicit tabu list*. Each structure in the explicit tabu list defines a set of *implicit tabu structures*. Given a structure E in the explicit tabu list, a structure I is said to be implicit tabu if the distance-RMSD (dRMSD) between E and I is less than ε and the energy of I is greater than or equal to the energy of E. The adjustable parameter ε is called the *tabu difference*. Figure 4 illustrates a sequence of visited structures (black points) in a solution space. Only the visited structures are inserted in the explicit tabu list. The additional green and red points correspond to structures within ε dRMSD of the explicit tabu structures. Green points are structures with lower energy and red points are structures with higher energy than the explicit tabu structure. When choosing a new solution in the neighbourhood three things can happen, *a)* A solution is more than ε dRMSD away from all explicit tabu structure. *b)* the solution is within ε dRMSD, and the energy is *lower* than the explicit tabu structure, *c)* the solution is within ε dRMSD, and the energy is *higher* than the explicit tabu structure. Structures that comply with case *c* are said to be *implicit tabu* and cannot be visited. Note that when $\varepsilon = 0$ the search heuristic works as a regular TS heuristic since only visited structures become tabu. The use of implicit tabu structures is new in the context of protein structure prediction. However, in TS implementations for OR problems it is a common technique to make features of a solution tabu, such that regions of the search space become tabu.

We have also applied standard Monte Carlo simulation (MCS) for minimizing the HSE energy. MCS heuristics are stochastic and therefore differ from TS by being nondeterministic. An MCS iteration consists of randomly choosing a protein conformation in the neighbourhood of a current conformation. For a fixed temperature T, the new protein conformation is accepted with the probability

$$p = e^{-\Delta E/T},$$

where ΔE is the difference between the energy of the current conformation and the new conformation. A protein conformation is modelled as a list of N vectors, where N is the number of C_α atoms of the protein. The neighbourhood of both MCS and TS consists of conformations resulting from changes of one, two or three consecutive indices. A single index change results in a new structure where one part of the structure is fixed and the other part is translated. Two or three indices are changed locally such that the parts of the structure before and after the changing indices are fixed. All local index changes between two lattice points can be stored in a table to speed up the computation time significantly.

Lattice experiments

Here, we evaluate TS and MCS on lattices of different complexity. The purpose of the experiments in this section is to tune the parameters (lattice type, tabu difference, temperature). In the next section we fix the parameters to their optimal values found here and compare the HSE and CN measures on different proteins. For each lattice, the heuristics are initialized with 20 random conformations using different parameter values. The variable parameter of MCS is the temperature and the variable parameter of TS is the

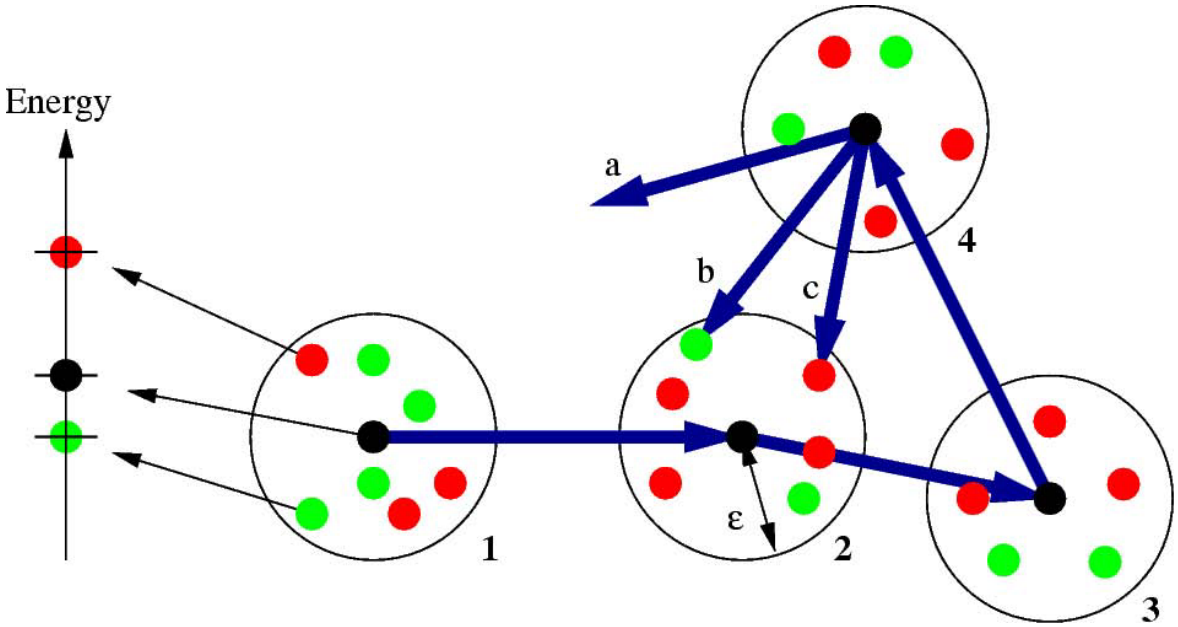


Figure 4
Explicit- and implicit tabu structures. Black points represent explicit tabu structures and red points represent implicit tabu structures.

Algorithm 1 Tabu Search

```
1: bestStructure, s ← random_conformation()
2: bestCost ← cost(bestStructure)
3: while not stop() do
4:    N ← compute_neighbours(s)
5:    sort N with respect to cost
6:    for all i ∈ N do
7:       if cost(N_i) < bestCost then
8:          bestCost ← cost(N_i)
9:          s, bestStructure ← N_i
10:         break loop
11:      end if
12:      if not Tabu(N_i, Q) then
13:         s ← N_i
14:         break loop
15:      end if
16:   end for
17:   pushback( Q, s )
18: end while
19: return bestStructure
```

Figure 5
Algorithm 1.

tabu difference. Each run is stopped after 15 minutes and the structure with the lowest observed HSE energy is reported. To get reasonable running times for these experiments, the HSE energy is based on the native structure of the small protein *Protegrin 1* (1PG1, 18 residues). Tables 1 and 2 show the results of the lattice experiments for the TS and MCS heuristics. There is a row for each lattice type and data columns show the average HSE energy found over the 20 runs for the various parameters. In the SCC lattice, structures with the same HSE energy are found in all 20 runs (tabu difference 0.4 and 0.5), but the best observed HSE energy is rather high. The reason is that the SCC lattice is very coarse grained and low energy structures therefore do not exist in this lattice. For lattices of increasing complexity, the ability to find structures with lower energy increases. TS and MCS seem to perform equally well in low complexity lattices. However, in high coordination lattices, the TS heuristic performs slightly better than MCS on average. For the lattice with highest complexity (HC8) TS found zero energy structures for all 20 runs, this robustness was not observed for the MCS heuristic. These results indicate that conformational search heuristics using the HSE measure require high complexity lattices or off-lattice models with a high degree of freedom. Furthermore, TS is slightly more robust that MCS in high coordination lattices. The results of experiments with variable tabu list size

Algorithm 2 Tabu(Structure s, TabuList Q)

```
1: for all i ∈ Q do
2:    if cost(s) > cost(Q_i) AND RMSD(s, Q_i) ≤ ε then
3:       return true
4:    end if
5: end for
6: return false
```

Figure 6
Algorithm 2.

and variable tabu difference in the HC8 lattice are shown in Figure 7. The figure shows that the tabu list size should generally be more than 50 elements, and there is no gain of having a very long list.

Comparison of HSE and CN measures

In the previous section, experiments on a small protein show that minimization of the HSE energy in high coordination lattices leads to structures with HSE vectors that are very similar (or equal) to the native structure. In this section, experiments on proteins of varying size are done using the TS heuristic with tabu difference 0.4 and the HC8 lattice. The energy functions are based on the HSE vectors of native structures as described in section *HSE energy function*. In addition to the HSE energy, the CN energy is considered for comparison. The main purpose of the experiments is to examine the reconstructability of a protein's backbone solely from the information stored in the HSE-/CN vectors.

Each TS run is started from a random structure which is iteratively improved as described in section *Heuristics*. For these experiments we want to start TS on 100 random structures that are as different from each other as possible. Therefore, to effectively sample the search space, 10000 random conformations are initially generated. Ideally, from this set of 10000 conformations, we would like to choose the set of 100 conformations such that the minimum RMSD between any two conformations is maximized. This problem is generally known as the p-dispersion problem and is NP hard[27]. Solving this problem to optimality is therefore not feasible, so we use a greedy heuristic to find a good set of 100 different random conformations. The greedy heuristic works by first picking a random conformation. The following 99 conformations are then picked one at a time, such the minimimum RMSD to any of the already picked conformations is maximized.

For each protein, the energy function based on its native structure is minimized for each of the 100 random start-

Table 1: Average HSE energy for *Protegrin 1* using *TS* on various lattices and tabu differences.

	Tabu difference										
Lattice	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
SCC	1.76	1.65	**1.64**	**1.64**	**1.64**	**1.64**	**1.64**	**1.64**	**1.64**	**1.64**	**1.64**
FCC	1.52	1.12	1.12	1.11	1.07	1.07	1.04	1.05	**1.03**	**1.03**	**1.03**
HC4	1.13	0.41	0.36	0.30	0.28	**0.27**	0.29	0.32	0.32	0.36	0.38
HC8	1.21	0.46	0.08	0.01	**0.00**	**0.00**	0.01	0.07	0.15	0.22	0.30

The best averages for each lattice type are boldfaced. The column with 0.0 tabu difference corresponds to the results of a regular TS implementation with no implicit tabu structures.

Table 2: Average HSE energy for *Protegrin 1* using *MCS* on various lattices and temperatures.

	Temperature											
Lattice	0.000	0.002	0.004	0.006	0.008	0.010	0.012	0.014	0.016	0.018	0.020	0.022
SCC	1.88	1.80	1.74	1.68	1.68	1.65	**1.64**	**1.64**	**1.64**	**1.64**	**1.64**	**1.64**
FCC	1.57	1.39	1.25	1.15	1.08	1.05	**1.03**	**1.03**	**1.03**	**1.03**	**1.03**	**1.03**
HC4	1.48	1.09	0.85	0.63	0.54	0.48	0.37	0.32	0.31	0.29	**0.27**	0.34
HC8	1.29	0.46	0.37	0.25	0.17	0.06	0.07	0.07	**0.04**	0.08	0.15	0.28

The best averages for each lattice type are boldfaced.

ing conformations and the structures with lowest energy are reported. The search is stopped after 12 hours or if the energy reaches zero. Zero energy means that a structure with exactly the same HSE- or CN vector as the native structure is found (but not necessarily identical structures).

To evaluate the quality of the structures with low energy, the RMSD with the native structure and *angle correlation* [28,29] is used. Angle correlation is a measure with the following definition. For each C_α let $\vec{V}_\alpha$ be the vector pointing in the side chain direction (see Figure 2). Let $V_{\alpha mc}^{\rightarrow}$ be the vector pointing in the direction of the mass center, and let θ_α be the angle between $\vec{V}_\alpha$ and $V_{\alpha mc}^{\rightarrow}$. The angle correlation measure is the average of the differences in θ_α between the optimized structure and the native structure. Zero angle correlation is perfect correlation, 90° is random correlation and 180° is perfect 'anti'-correlation. Note that the CN- and HSE vectors of a structure are identical to the vectors of the mirror of the structure. Therefore, in the following results, if the RMSD between a structure and its native mirror image is smaller we report this value instead. All computations were performed on a 236 nodes Dell Optiplex GX260 cluster (2,4 GHz P4, 512 Mb RAM).

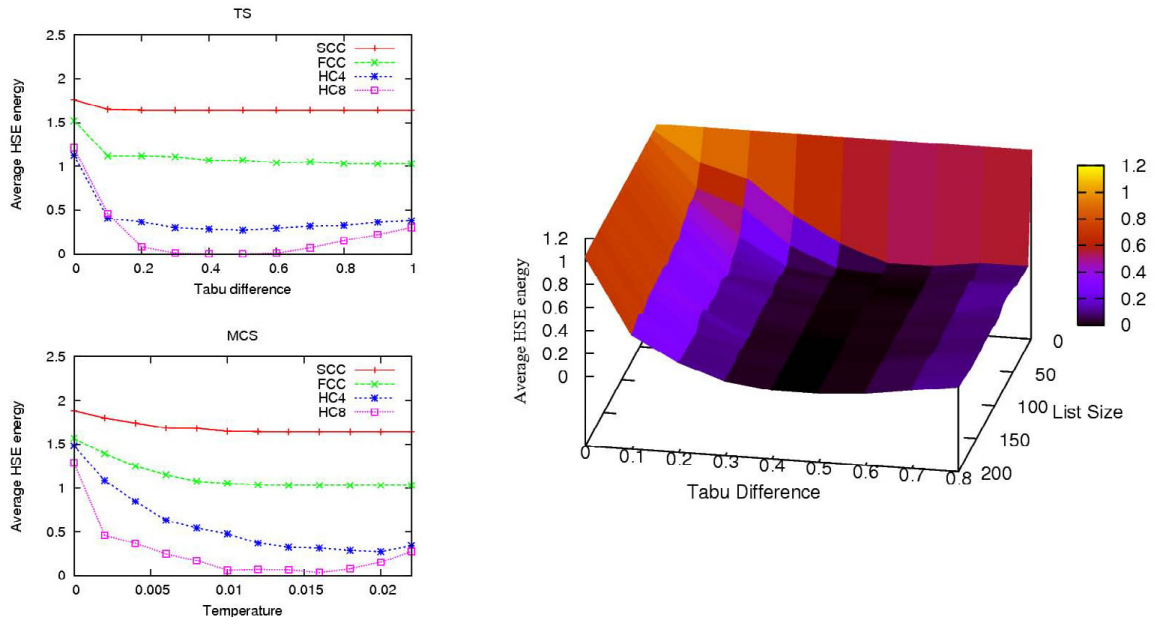


Figure 7
Lattice experiments. The two first plots show the values in table 1 and 2. The right figure shows the average HSE energy on HC8 with variable tabu list size and variable tabu difference.

Table 3: Comparison of the HSE- and CN measures for various proteins.

Residues	Measure	< 7 Å RMSD	< 6 Å RMSD	< 5 Å RMSD	< 4 Å RMSD	< 3 Å RMSD	< 2 Å RMSD	lowest RMSD	lowest energy
				Human Endothelin (1EDN)					
21	CN	100	100	98	60	18	0	**2.09**	**0.00**
	HSE	100	100	100	93	65	37	**0.88**	**0.00**
				Tryptophan Zipper 1 (1LE0)					
13	CN	100	100	100	100	100	22	**1.38**	**0.00**
	HSE	100	100	100	100	100	67	**0.95**	**0.00**
				Third Zinc Finger (1SRK)					
35	CN	60	42	17	(1SRK) 1	0	0	**3.52**	**0.00**
	HSE	56	33	13	5	0	0	**3.02**	**0.33**
				Mu-Conotoxin GIIA (1TCH)					
23	CN	100	100	97	63	23	5	**1.58**	**0.00**
	HSE	100	100	100	97	61	38	**0.91**	**0.00**
				Pandinus Toxin (2PTA)					
35	CN	59	32	14	3	0	0	**3.17**	**0.00**
	HSE	58	44	17	11	2	0	**2.66**	**0.33**

Results and discussion

The results of the HSE and CN comparisons are shown in Table 3. The table shows how many of the 100 HSE/CN minimized conformations are below a certain RMSD threshold. The associated RMSDs and energy values of the 100 conformations are also shown. In Figures 8 to 12, histograms show the RMSD and energy distribution of the CN- or HSE-optimized structures. The histograms reveal that most of the lowest energy structures are similar to the native structure. This trend is much more prevalent for the HSE-optimized structures. Based on the histograms, we conclude that the CN-/HSE-energy functions have a large smooth minimum around the structure of the native state and few smaller local minima scattered around the conformational space.

Scatter plots show the angle correlation vs. RMSD. The Figures also show the best HSE- and CN-optimized structures superimposed on the native structure. The yellow backbone is the native structure, the red backbone is the best HSE optimized structure and the green backbone is the best CN optimized structure.

The CN and HSE comparisons show that low HSE-energy structures are generally closer to the native structure than low CN-energy structures, this both in terms of RMSD and angle correlation. A backbone structure with a good angle correlation implies that the general orientation of the residues is accurate. The plots show that this property is much more prevalent in HSE-optimized structures. Existing protein structure prediction methods that use the CN measure could therefore benefit from using the HSE measure instead of the CN measure.

Here we have developed a lattice model for protein structure prediction using the CN-/HSE energy functions. The search heuristic is based on TS with a novel tabu definition and the results indicate that TS performs better than MCS for this problem. TS with this new tabu definition might also be applied with success for other protein structure optimization problems.

Lattice experiments suggest that near zero energy structures only exists in high coordination lattices. Therefore, when using the HSE measure the model should have a high degree of freedom. All results are found using small proteins (the largest protein has 35 amino acids). When using larger proteins, it becomes very time consuming to find low energy structures and they are often not native like.

We have shown that it is possible to reconstruct the backbone of small proteins using the HSE vector of the native structure. Obviously, a predicted HSE vector would have some errors or noise as compared to the exact HSE vector. A future research project could therefore be to analyze the reconstructability of a protein backbone using HSE vectors with various degree of noise. Other directions could be to consider a more detailed energy function using other predictable information such as secondary structure. Another option could be to enforce protein-like geometry, using for example angular constraints.

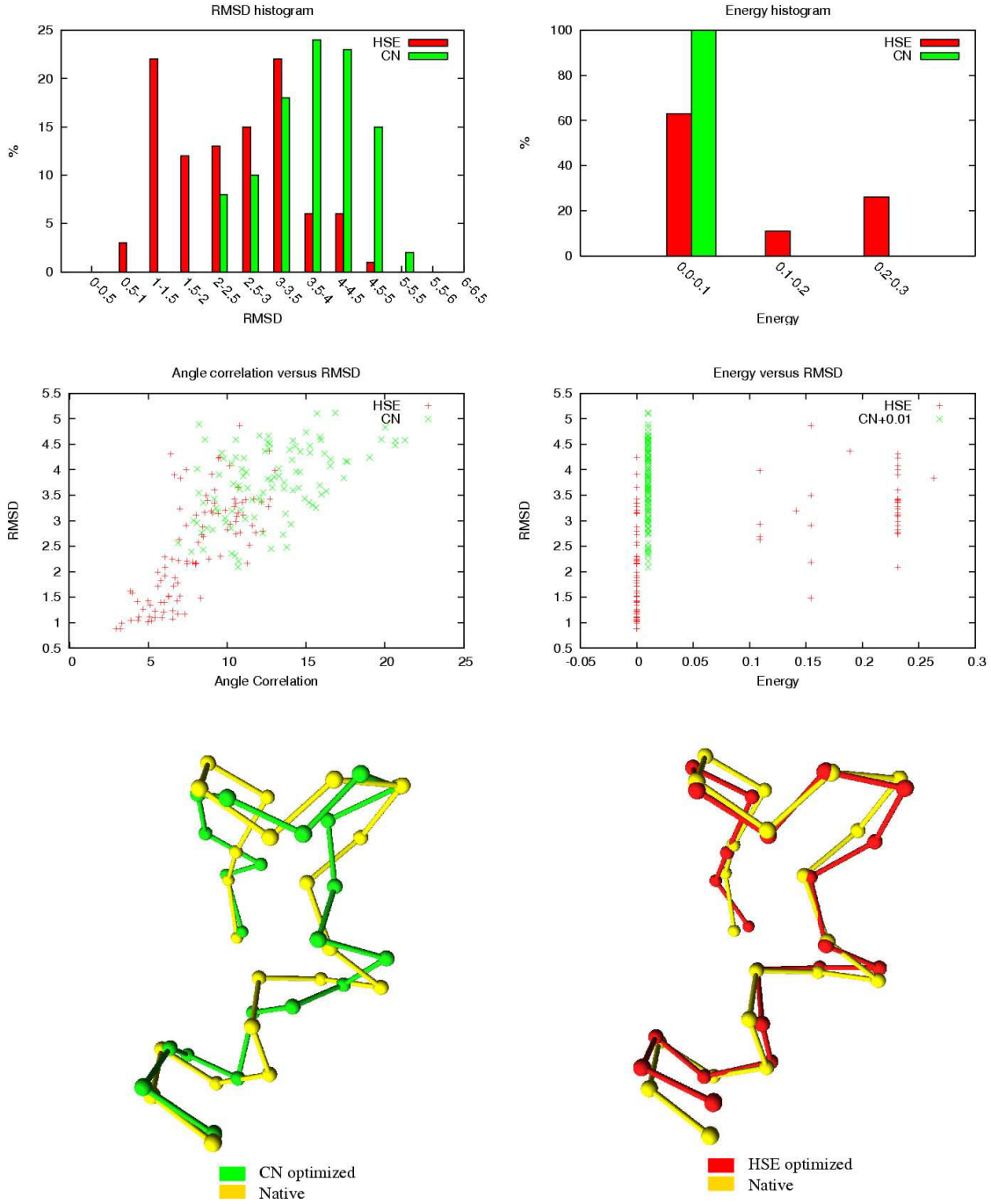


Figure 8
Human Endothelin (1EDN), 21 residues. In the energy versus RMSD plot, the CN values have an offset of 0.01 for better illustration.

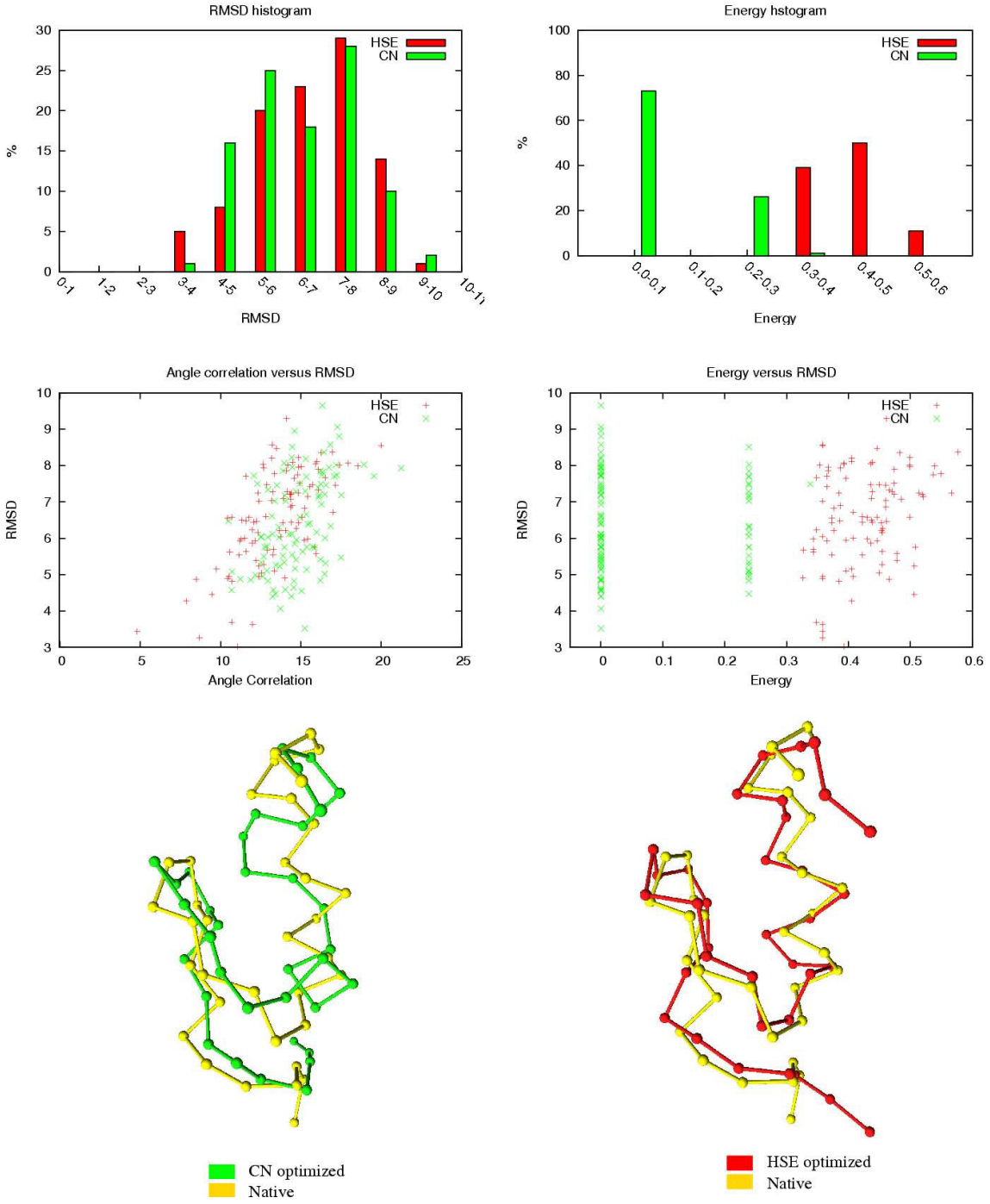


Figure 9
Third Zinc Finger (1SRK). 35 residues.

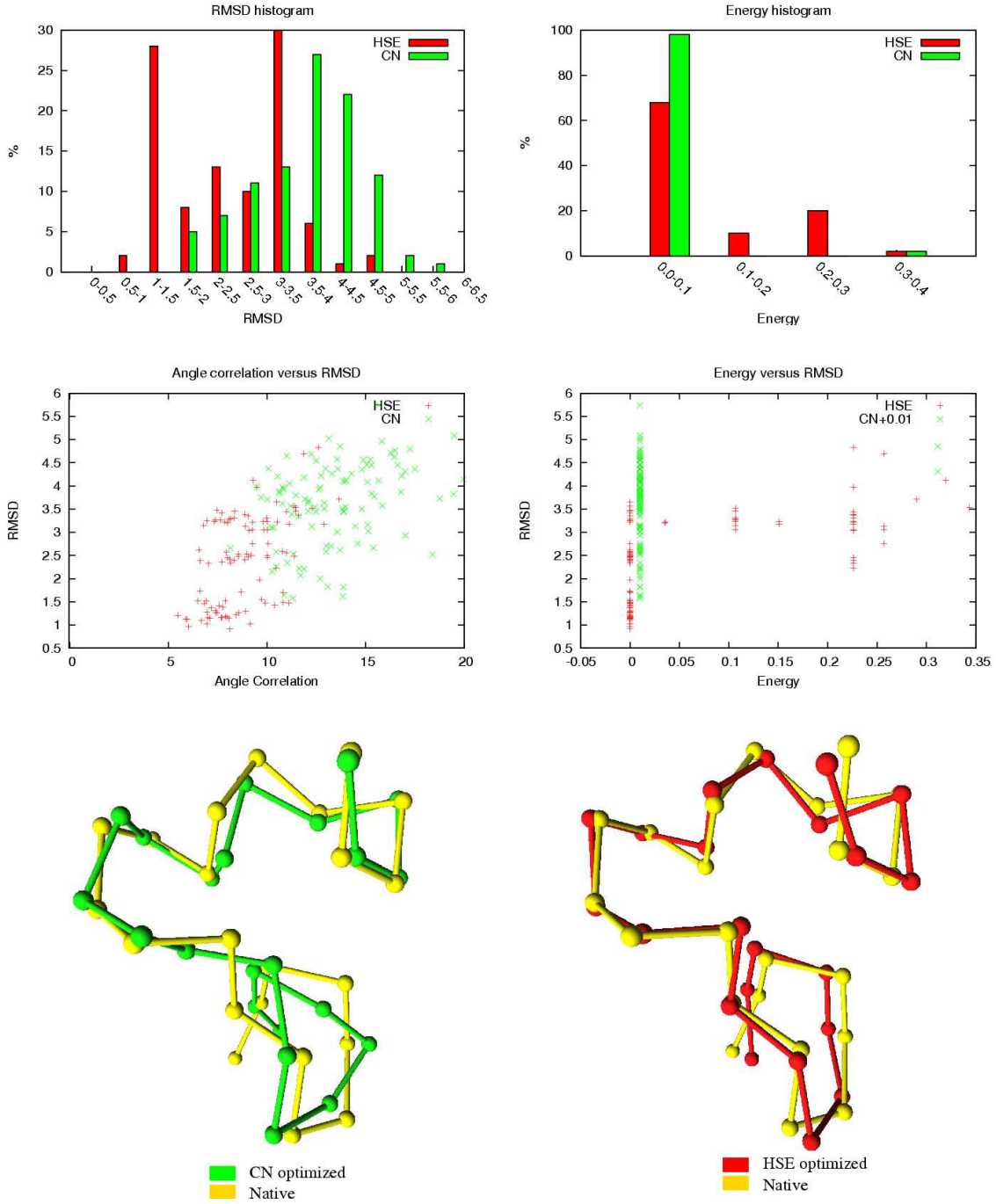


Figure 10
Mu-Conotoxin GIIA (1TCH). 23 residues. In the energy versus RMSD plot, the CN values have an offset of 0.01 for better illustration.

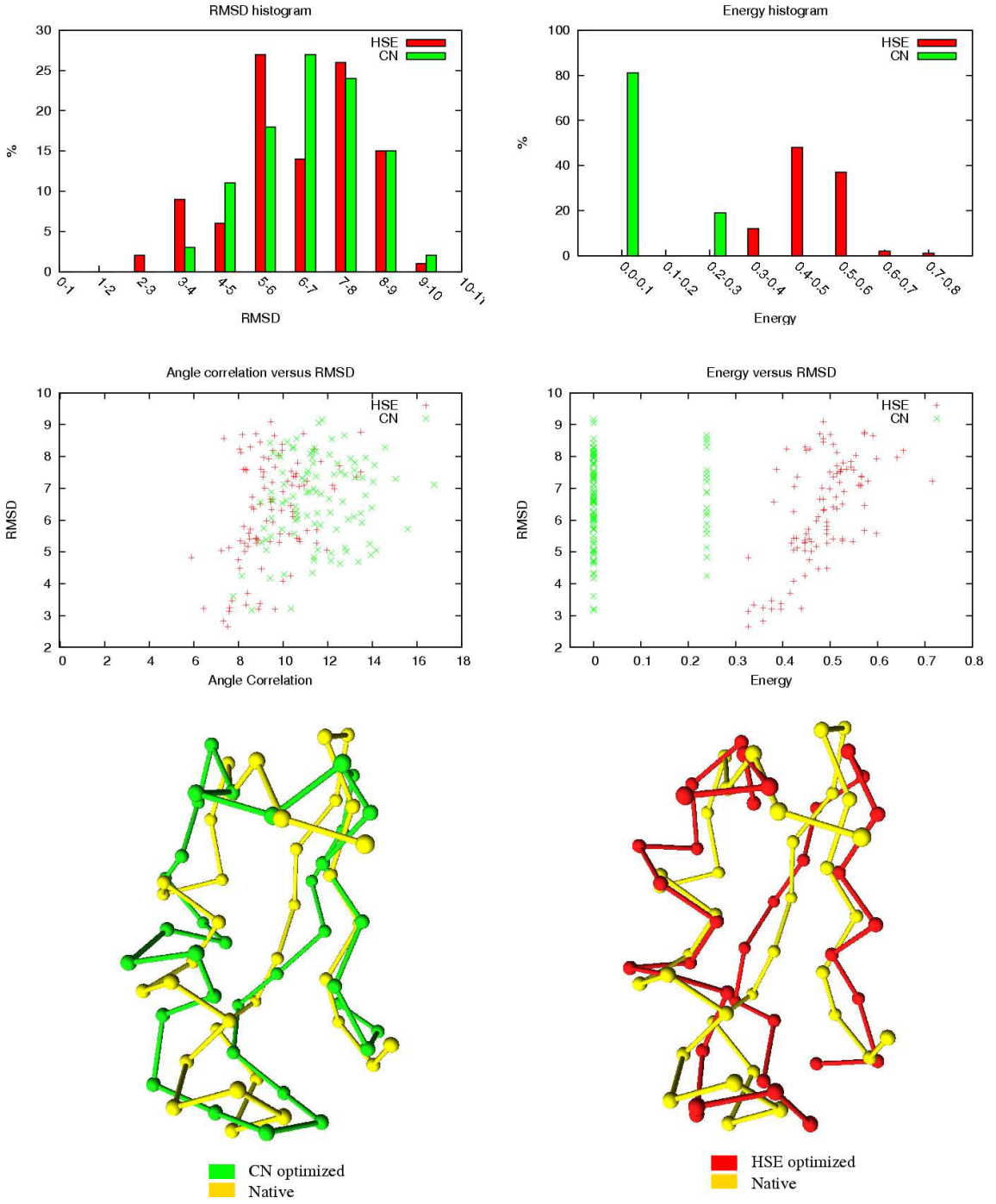


Figure 11
Pandinus Toxin (2PTA). 35 residues.

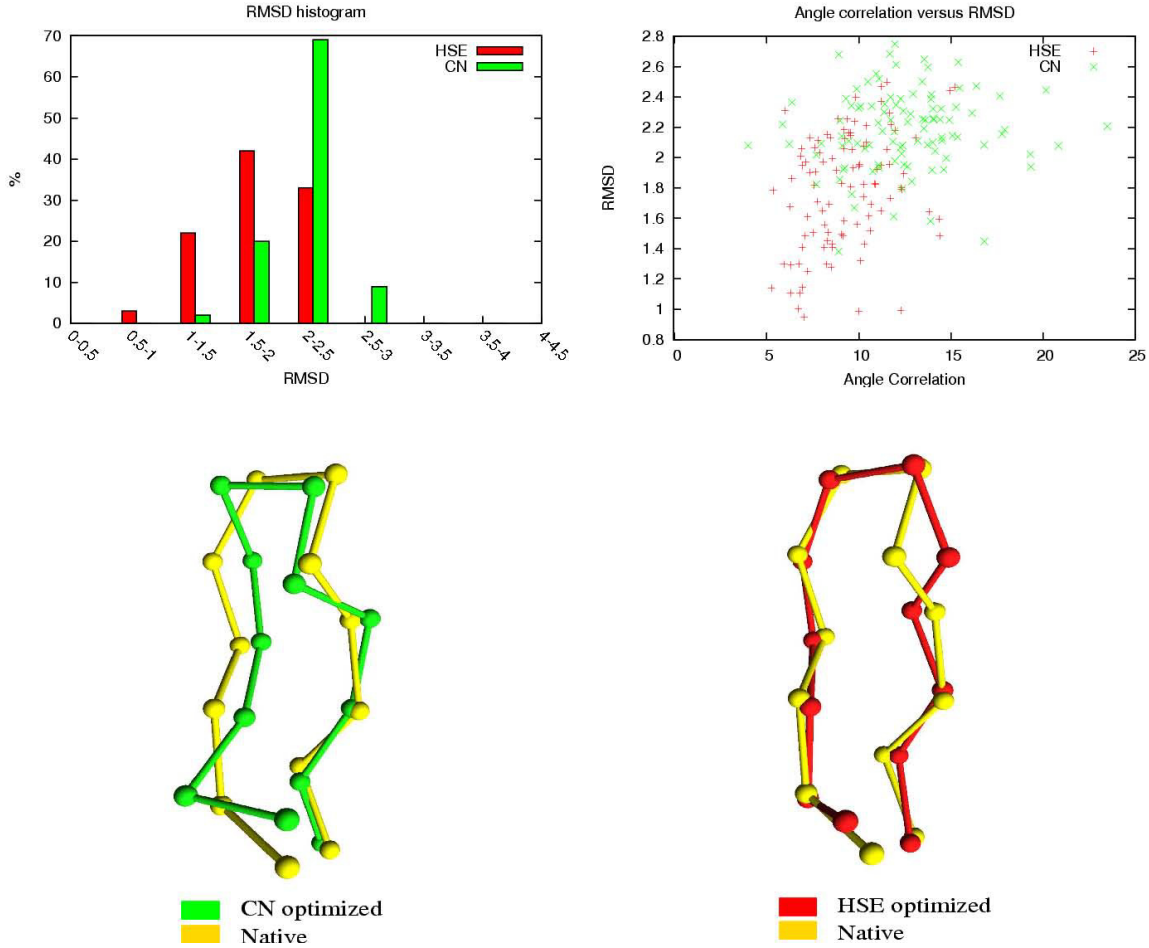


Figure 12
Tryptophan Zipper 1 (1LEO). 13 residues. All optimized structures have zero energy.

In this article, we only considered lattice models. However, off-lattice models and other conformational search heuristics such as replica exchange MCMC[30] could be considered as well.

Acknowledgements

Thomas Hamelryck is supported by a Marie Curie Intra-European Fellowship within the 6th European Community Framework Programme. Martin Paluszewski and Pawel Winter are partially supported by a grant from the Danish Research Council (51-00-0336).

References

1. Lee B, Richards F: **The Interpretation of Protein Structures: Estimation of Static Accessibility.** *J Mol Biol* 1971, **55:**379-400.
2. Greer J, Bush BL: **Macromolecular shape and surface maps by solvent exclusion.** *Proc Natl Acad Sci USA* 1978, **75:**303-7.
3. Connolly ML: **Solvent-accessible surfaces of proteins and nucleic acids.** *Science* 1983, **221(4612):**709-13.
4. Chakravarty S, Varadarajan R: **Residue depth: a novel parameter for the analysis of protein structure and stability.** *Structure* 1999, **7(7):**723-32.
5. Pintar A, Carugo O, Pongor S: **Atom depth in protein structure and function.** *Trends Biochem Sci* 2003, **28(11):**593-7.
6. Pintar A, Carugo O, Pongor S: **Atom depth as a descriptor of the protein interior.** *Biophys J* 2003, **84(4):**2553-61.
7. Pollastri G, Baldi P, Fariselli P, Casadio R: **Prediction of coordination number and relative solvent accessibility in proteins.** *Proteins* 2002, **47(2):**142-53.
8. Kinjo A, Horimoto K, Nishikawa K: **Predicting absolute contact numbers of native protein structure from amino acid sequence.** *Proteins* 2005, **58:**158-65.
9. Hamelryck T: **An amino acid has two sides: a new 2D measure provides a different view of solvent exposure.** *Proteins* 2005, **59:**38-48.
10. Simons KT, Kooperberg C, Huang E, Baker D: **Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions.** *J Mol Biol* 1997, **268:**209-25.

11. Bjarni Vilhjalmsson, Thomas Hamelryck: **Predicting a New Type of Solvent Exposure.** *ECCB Computational Biology Madrid 05 P-C35 Poster* 2005.
12. Kinjo AR, Nishikawa K: **Recoverable one-dimensional encoding of three-dimensional protein structures.** *Bioinformatics* 2005, **21(10):**2167-70.
13. Porto M, Bastolla U, Roman HE, Vendruscolo M: **Reconstruction of protein structures from a vectorial representation.** *Phys Rev Lett* 2004, **92(21):**.
14. Pardalos PM, Liu X, Xue GL: **Protein Conformation of a Lattice Model Using Tabu Search.** *Journal of Global Optimization* 1997, **11:**55-68.
15. Morales LB, Garduño-Juárez R, Aguilar-Alvarado JM, Riveros-Castro FJ: **A parallel tabu search for conformational energy optimization of oligopeptides.** *Journal of Computational Chemistry* 2000, **21(2):**147-156.
16. Oakley M, Garibaldi J, Hirst J: **Lattice models of peptide aggregation: Evaluation of conformational search algorithms.** *J Comput Chem* 2005, **26(15):**1638-46.
17. Dill KA, Bromberg S, Yue K, Fiebig KM, Yee DP, Thomas PD, Chan HS: **Principles of protein folding-a perspective from simple exact models.** *Protein Sci* 1995, **4:**561-602.
18. Thomas PD, Dill KA: **Statistical potentials extracted from protein structures: how accurate are they?** *J Mol Biol* 1996, **257:**457-69.
19. Zeldovich KB, Berezovsky IN, Shakhnovich EI: **Physical origins of protein superfamilies.** *J Mol Biol* 2006, **357(4):**1335-43.
20. Chikenji G, Fujitsuka Y, Takada S: **Shaping up the protein folding funnel by local interaction: Lesson from a structure prediction study.** *Proc Natl Acad Sci USA* 2006, **103:**3141-3146.
21. Adam Godzik, Andrzej Kolinski, Jeffrey Skolnick: **Lattice Representations of Globular Proteins: How Good Are They.** 1993 [http://www3.interscience.wifey.com/cgi-bin/abstract/109582884/ABSTRACT].
22. Park B, Levitt M: **The complexity and accuracy of discrete state models of protein structure.** *J Mol Biol* 1995, **249(2):**493-507.
23. Kihara D, Lu H, Kolinski A, Skolnick J: **TOUCHSTONE: an ab initio protein structure prediction method that uses threading-based tertiary restraints.** *Proc Natl Acad Sci USA* 2001, **98(18):**10125-30.
24. Zhang Y, Kolinski A, Skolnick J: **TOUCHSTONE II: a new approach to ab initio protein structure prediction.** *Biophys J* 2003, **85(2):**1145-64.
25. Glover F: **Tabu Search, PART I.** *ORSA J Comput* 1989, **1:**190-206.
26. Glover F: **Tabu Search, PART II.** *ORSA J Comput* 1990, **2:**4-32.
27. Erkut E: **The Discrete P-Dispersion Problem.** *European Journal of Operational Research* 1990, **46:**48-60.
28. Rackovsky S, Scheraga HA: **Hydrophobicity, hydrophilicity, and the radial and orientational distributions of residues in native proteins.** *Proc Natl Acad Sci USA* 1977, **74(12):**5248-51.
29. Yan A, Jernigan RL: **How do side chains orient globally in protein structures?** *Proteins* 2005, **61:**513-22.
30. Swendsen RH, Wang JS: **Replica Monte Carlo simulation of spin glasses.** *PHYSICAL REVIEW LETTERS* 1986, **57(21):**2607-2609.

Chapter 10

Paper: Protein Decoy Generation using Branch and Bound with Efficient Bounding

M. Paluszewski and P. Winter. Protein Decoy Generation using Branch and Bound with Efficient Bounding. *Proc. of 8th International Workshop on Algorithms in Bioinformatics, WABI*, 2008.
Status: Published

Note that an extended version (technical report) of this paper is included in Appendix A.

Protein Decoy Generation Using Branch and Bound with Efficient Bounding

Martin Paluszewski and Pawel Winter

Department of Computer Science, University of Copenhagen,
Universitetsparken 1, 2100 Copenhagen, Denmark
palu@diku.dk, pawel@diku.dk

Abstract. We propose a new discrete protein structure model (using a modified face-centered cubic lattice). A novel branch and bound algorithm for finding global minimum structures in this model is suggested. The objective energy function is very simple as it depends on the predicted half-sphere exposure numbers of C_α-atoms. Bounding and branching also exploit predicted secondary structures and expected radius of gyration. The algorithm is fast and is able to generate the decoy set in less than 48 hours on all proteins tested.

Despite the simplicity of the model and the energy function, many of the lowest energy structures, using exact measures, are near the native structures (in terms of RMSD). As expected, when using predicted measures, the fraction of good decoys decreases, but in all cases tested, we obtained structures within 6 Å RMSD in a set of low-energy decoys. To the best of our knowledge, this is the first *de novo* branch and bound algorithm for protein decoy generation that only depends on such one-dimensional predictable measures. Another important advantage of the branch and bound approach is that the algorithm searches through the entire conformational space. Contrary to search heuristics, like Monte Carlo simulation or tabu search, the problem of escaping local minima is indirectly solved by the branch and bound algorithm when good lower bounds can be obtained.

1 Background

The contact number (CN) is a very simple solvent exposure measure that only depends on the positions of C_α-atoms. Given a fixed backbone structure, the CN of a residue A_i is the number of other C_α-atoms in a sphere of radius r centered at the C_α-atom of A_i. The CN of all residues of a given structure is called the *CN-vector*. A more information rich measure is called the *half-sphere-exposure* (HSE) measure [5]. Here, the sphere is divided into an upper and a lower hemisphere as illustrated in Figure 1. The up and down numbers of a residue therefore refer to the number of other C_α-atoms in the upper and lower hemispheres respectively. For a given fixed structure, the up and down numbers for all residues is called the *HSE-vector*. CN- and HSE-vectors therefore only depend on the radius of the spheres and the coordinates of C_α-atoms, which is very convenient when working with simplified models.

K.A. Crandall and J. Lagergren (Eds.): WABI 2008, LNBI 5251, pp. 382–393, 2008.

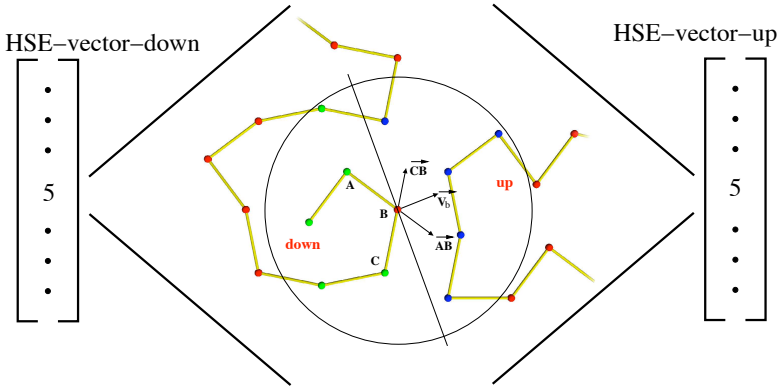


Fig. 1. Given the positions of 3 consecutive C_α-atoms (A, B, C), the approximate side-chain direction $\bar{V}_b$ can be computed as the sum of $\bar{AB}$ and $\bar{CB}$. The plane perpendicular to $\bar{V}_b$ cuts the sphere centered at B in an upper and a lower hemisphere.

Recently it was shown that it is possible to approximately reconstruct small protein structures from CN-vectors or HSE-vectors only [12]. These results showed that HSE-optimized structures tend to have better coordinate RMSD with the native structure and more accurate orientations of the side-chains compared to CN-optimized structures. This is very interesting in regards to *de novo* protein decoy generation, because CN- and HSE-vectors can be predicted with reasonable accuracy [19,14]. To use these results for *de novo* decoy generation, one could therefore first predict the HSE-vector from the amino acid sequence and then reconstruct the protein backbone from this vector. However, the results in [12] were only based on small proteins with up to 35 amino acids and it was conjectured that the reconstruction of larger proteins would require more information than what is contained in an HSE-vector [12]. Another difficulty is that HSE-based energy functions appear to have many local minima in the conformational space. This is often a problem for search heuristics like Monte Carlo simulation or tabu search, since they get trapped in these minima and must spend much time escaping them.

The problem of reconstructing protein structure from vectors of one-dimensional structural information has also been studied by Kinjo et al. [7]. They used exact vectors of secondary structure, CN and *residuewise contact order* (RWCO) together with refinement using the AMBER force field to reconstruct native-like structures. Their results indicated that secondary structure information and CN *without* the use of RWCO is *not* enough to reconstruct native-like structures. Unfortunately, RWCO is difficult to predict compared to CN, HSE and secondary structure [7] and it would therefore be difficult to use their method directly for *de novo* decoy generation.

Here we attack these problems by adding more predicted information to our model and use a thorough branch and bound algorithm for finding minimum energy structures. By adding more predicted information we expect to increase the

probability of the energy function to have global minimum near the native structure. Furthermore, using a branch and bound approach we are able to implicitly search the whole conformational space and therefore avoid getting trapped in local minima. Besides using HSE-vectors, we also use *secondary structure* (SS) and *radius of gyration* (Rg). These three measures, (HSE, SS and Rg), can all be predicted from the amino acid sequence only [19,10,16], and can therefore be used for *de novo* protein decoy generation. The energy function is simple, and we show how a good lower bound of the energy for a subset of the conformational space can be computed in polynomial time. This lower bound enables the branch and bound algorithm to efficiently bound large conformational subspaces and to find global minimum energy structures in a reasonable amount of time. Throughout the text our branch and bound algorithm is referred to as EBBA (Efficient Branch and Bound Algorithm).

The idea of using secondary structure elements in a discrete model has been suggested by others, i.e., Fain et al. [4] and Levitt et al. [8]. However, their models have a relatively small conformational space and it is therefore possible to completely enumerate all structures allowed by the model. Branch and bound algorithms and other algorithms for determining global minimum structures have been used for protein structure prediction earlier. Some of these algorithms work on very simplified models like the HP-lattice model [1]. Even though these algorithms can solve most problems to optimality, the global minimum structures are often very far from the native structure. Another branch and bound algorithms, called αBB[9] uses more detailed potential energy functions which depend on several physical terms. In [9], the αBB is shown to be successful on small molecules. In [17], the αBB was improved and was used for prediction of real protein structures. Dal Palu et al.[11] use a constraint logic programming approach for protein structure prediction. They also use secondary structure segments in a simplified model. However, in their model, all C_α-atoms must be placed in a lattice (FCC). This differs from our approach, where we only demand lattice directions of the secondary structure segments. Dal Palu et al. use a standard solver (SICStus Prolog) which makes use of standard bounding techniques, while we have developed a much more efficient bounding algorithm specialized for this particular problem. Furthermore, the results published in [11,17] are not true *de novo* - the secondary structures are all derived from the native structure of the proteins. On the contrary, the results presented here are true *de novo*. All parts of the energy function are predicted from amino acid sequences only. EBBA is, to our knowledge, the first *de novo* branch and bound algorithm that only use one-dimensional predictable measures.

We use 6 benchmark proteins for evaluating EBBA. These benchmark proteins are chosen because they are used in similar studies before [15,6] and we are therefore able to compare our method with the state-of-the-art protein conformational sampler FB5-HMM [6]. Our results show that EBBA is able to find global minimum energy structures for most of these proteins in less than 48 hours. We have evaluated EBBA using both exact values and predicted values to estimate the importance of prediction quality. The results show that predicted structures having

global minimum energy are *not always* native-like, however among the 10.000 lowest energy structures we typically find many good decoys (less than 6 Å RMSD).

2 Methods

A sequence of residues of the same secondary structure class is called a *segment*. Segments can be considered as rigid rods that describe the overall path of C_α-atoms belonging to the segment. Segments have a start coordinate and a direction, and for helices and sheets their end coordinate can also be determined because of their constrained geometry. A segment is therefore an abstract representation of a sequence of residues and it does not explicitly contain the coordinates of internal C_α-atoms. We define a *segment structure* to be the coordinates of all C_α-atoms of a segment. Note that a segment in principle allows for infinitely many different segment structures even though they are restricted to be of a specific secondary structure class. However, this model is discrete and therefore only a finite representative set of segment structures are generated. This is described in more detail in Section 2.1.

Any tertiary structure of a protein can be described in these terms; a list of segments and a segment structure for each segment. We call such a list of segments a *super structure* and a super structure with a fixed segment structure for each segment is called a *complete structure*.

To discretize and reduce the conformational space of this model, we reduce the degree of freedom for segments. Segments are therefore only allowed to have a discrete set of predefined directions between the first and last C_α-atoms. Ad-hoc experiments show that the 12 uniformly distributed directions acquired from the *face-centered cubic* (FCC) lattice is a good tradeoff between discretization and flexibility. The direction of a segment therefore has one of the following 12 direction vectors

```
[1,1,0], [1,0,1], [1,-1,0], [1,0,-1], [-1,1,0], [-1,0,1],
[-1,-1,0], [-1,0,-1], [0,1,1], [0,1,-1], [0,-1,1], [0,-1,-1]
```

To further discretize the model, we set an upper limit (u) on the number of possible segment structures allowed by a segment. Given an amino acid sequence with m segments and u possible segment structures for each segment, the total number of complete structures, N (disregarding symmetric structures), allowed by this model is

$$N = 4 \times 11^{m-2} \times u^m \tag{1}$$

2.1 Segment Structures

Here we briefly describe how the allowed segment structures of a given segment are computed. This computation depends on the secondary structure class of the segment.

Given a helix or sheet segment, we generate one segment structure having the angle properties of a right-handed helix or a beta strand. Then the other $u-1$ segment structures are generated by rotating the first structure uniformly around the axis going through the first C_α-atom and ending at the beginning of the next segment.

There are no simple geometric constraints that describe coil structures. Experiments show that short sequences with similar amino acid sequences, so-called homologous sequences, often have similar tertiary structures [3]. Given a coil segment, we therefore query PDB Select (25) with protein sequences and their known structures and find the $\sqrt{u}$ best fragment matches in terms of amino acid similarity. Each of these structures is also rotated uniformly $\sqrt{u}$ times as for helices and sheets such that a total of u structures are obtained. The fragment database does of course not contain the proteins used in the experiments. Even though we are querying PDB Select (25) for coil fragments, we still consider our algorithm to be *de novo*, because we do not explicitly make use of templates. One of the most successful structure prediction algorithms (Rosetta[15]) also makes use of fragments from proteins in PDB and is also considered to be *de novo*.

2.2 Energy

The structures allowed by the model always have the desired secondary structure (from a prediction), however the HSE-vector and radius of gyration of the structures varies. Therefore, we want to identify those structures having correct radius of gyration and HSE-vectors similar to the predicted HSE-vectors. The radius of gyration can be predicted from the number of residues n of the protein [16]:

$$R_g = 2.2n^{0.38} \tag{2}$$

This prediction is often accurate for globular proteins. We therefore assign infinite energy to structures having radius of gyration more than 5% away from the predicted R_g. We assign infinite energy to structures if their subchain of amino acids from the first amino acid to the l'th ($l < n$) amino acid is more than 5% away from the predicted R_g. A structure is said to be *clashing* if the distance between two C_α-atoms is less than 3.5 Å. We also assign infinite energy to clashing structures and conformations where two succeeding segment structures have unlikely angle properties.

Let $\mathcal{P}$ denote the conformational space of a protein with n residues $A_1, A_2, ...,$ A_n. Let $P \in \mathcal{P}$. The total energy $Q(P)$ of P is defined as the sum of the residue energy contributions $Q_P(A_i)$, i.e.,

$$Q(P) = \sum_{i=1}^{n} Q_P(A_i) \tag{3}$$

with

$$Q_P(A_i) = \begin{cases} \Delta CN(A_i)^2 & \text{if } A_i \text{ is the first residue of a segment} \\ \Delta HD(A_i)^2 + \Delta HU(A_i)^2 & \text{otherwise} \end{cases} \tag{4}$$

where

- $\Delta CN(A_i)$ is the difference between the contact number of the i-th residue A_i in P and the desired (i.e., predicted) contact number of A_i.
- $\Delta HD(A_i)$ is the difference between the down half sphere exposure number of A_i in P and the desired down half sphere exposure number of A_i.
- $\Delta HU(A_i)$ is the similar difference for the up half sphere exposure.

The reason why CN instead of HSE is used for the first residue of a segment is that the HSE value depends on the position of the two neighbour residues as illustrated in Figure 1. On the other hand, HSE can be used for the last residue of a segment, because one of the neighbours are an interior residue and the other neighbour is the end position of the segment whose coordinates are always known. The radius of the contact sphere is set to 15 Å.

2.3 Branch and Bound

An explicit evaluation of all allowed structures is only feasible for proteins with very few segments and segment structures. A standard approach for overcoming such combinatorial explosion is to use the branch and bound technique [20].

Branching. The root of the branch and bound tree represents all complete structures allowed by the model. This is done by only fixing the direction of the first segment. Every other node s represents a smaller subset of complete structures $\mathcal{P}_s$ than its parent. This is done by either fixing a segment direction or by fixing a segment structure. Therefore, when branching on a node, either 11 children with fixed segment directions are created or u children with fixed segment structures are created. A node at level $2 \times m$ has all segment directions and segment structures fixed and therefore represents a complete structure. Nodes at level $2 \times m$ cannot be branched on further and are called leaves.

Bounding. A lower bound is a value that is less than, or equal to the lowest energy of any leaf in the subtree of the node. Such a value can be used to disregard, or *bound*, the subtree of a node if the lower bound is larger than some observed energy (*an upper bound*). An upper bound of the energy can be found using some advanced heuristic or a simple depth first search as described in section 2.4. Here we present a reasonable tight lower bound that can be computed fast. The use of this lower bound makes it possible to solve large problems to optimality as shown in the results section.

Let $\mathcal{P}_S$ denote the subset of the conformational space $\mathcal{P}$ at any node of the branch and bound tree where some segments might have fixed directions while others might have fixed segment structures (i.e., fixed coordinates of all C_α-atoms)

as explained in the description of the branching strategy above. We are looking for a lower bound for $\min_{P \in \mathcal{P}_S}\{Q(P)\}$.

Consider the j-th segment S_j, $1 \leq j \leq m$, where m is the number of segments. Let

$$Q_P(S_j) = \sum_{A_i \in S_j} Q_P(A_i)$$

where $Q_P(A_i)$ is defined in Equation 4. Then the energy of a structure can be written as the sum of segment energies

$$Q(P) = \sum_{1 \leq j \leq m} Q_P(S_j)$$

Suppose that a lower bound for $\min_{P \in \mathcal{P}_S}\{Q_P(S_j)\}$ can be determined. Summing up these lower bounds for all m segments will therefore yield a lower bound for the energy of all conformations in $\mathcal{P}_S$. To compute such a lower bound for a segment S_j, the following problem is solved for all segment structures of S_j. For simplicity we only describe how a lower bound using CN-vectors can be computed, however it is straightforward to use a similar approach for HSE-vectors.

Given a segment structure for S_j, we determine for each of its C_α-atoms all possible values of CN when the super structure is fixed. This problem can clearly be solved in exponential time by complete enumeration of all possible segment structures. However, using the following dynamic programming approach, the problem can be solved much faster in polynomial time.

Let $c_{a,b}(i,r)$ where $(1 \leq i \leq m)$ and $(1 \leq r \leq u)$ be the number of contacts of residue a in segment b contributed by residues in segment i having segment structure r. Let (i,j) be an entry in the dynamic programming table and let $q_{a,b}(i,j) \in \{0,1\}$ represent whether or not residue a in segment b can have a total of j, $(0 \leq j < n)$, contacts contributed by residues in segments S_l, $(l < i)$. Then the recursive equation of the dynamic programming algorithm is:

$$q_{a,b}(i,j) = \begin{cases} 1 \text{ if } i = 1 \text{ and } c_{a,b}(1,r) = j \text{ for some r} \\ 1 \text{ if } i > 1 \text{ and } q(i-1,k) = 1 \text{ and } c_{a,b}(i,r) = j - k \text{ for some r} \\ 0 \text{ otherwise} \end{cases} \quad (5)$$

Each row can be computed in $\mathcal{O}(n \times u)$ time using the values from the previous row, so the total running time of the algorithm is $\mathcal{O}(m \times n \times u)$. The last row in the table represents all possible contact numbers for residue a in segment b. The last row can therefore easily be used to find the minimum difference between the desired CN and one of the possible CNs. The dynamic programming problem is solved for all residues of the segment and the sum of the minimum differences for each residue is the lower bound of the segment energy. For more details and examples of computing lower bounds, refer to [13].

In the above discussion, it was assumed that all C_α-atoms in S_j have their coordinates fixed in $\mathcal{P}_S$. Lower bounds can also be computed if the segment structure has not been fixed yet. The above lower bound computation is then merely repeated for each of the u possible segment structures, and the smallest one is selected as the overall lower bound of the segment.

Lower bounds can also be computed for nodes where a number of the last segment directions have not yet been fixed. Here, the input to the dynamic programming algorithm is only the first fixed segments. Then, the CN row for the last fixed segment is augmented by checking whether each C_α-atom on the free segments can possibly be in contact with the C_α-atom in question.

2.4 Searching

We search the branch and bound tree by keeping a set of nodes for which the lower bound has been computed but not bounded. Initially, the set contains only the root of the branch and bound tree. Iteratively the algorithm chooses the lowest cost node and replaces it with the children obtained by branching. When using this strategy, an optimal solution is found when the lowest cost node in the set is a leaf node. In practice the set of unbranched nodes might become very large and difficult to store in memory. We therefore combine it with a depth first search, such that when the node set contains more than 50.000 nodes we shift to depth first search until the set is less than 50.000 again.

3 Experiments

Here we predict the tertiary structures of 6 proteins. The tertiary structures of these proteins are known and we can therefore evaluate the quality of our results. These proteins have previously been used for benchmarks in the literature [15,6] and our results can therefore be directly compared with the state-of-the-art conformational sampler FB5-HMM[6].

The input to EBBA is a secondary structure assignment, HSE-vector and the radius of gyration. For each protein we obtain these values using prediction tools. Based on the amino acid sequence, we predict the secondary structure using PSIPRED [10] and we predict HSE-vectors using LAKI [19]. Note that PSIPRED and LAKI are neural networks trained on a selection of proteins from PDB. The 6 benchmark proteins used here also exist in PDB, so there is a slight chance that the training sets for PSIPRED and LAKI contain some of these proteins. However, the prediction quality of the 6 benchmark proteins is close to what should be expected from PSIPRED and LAKI. Here, the average Q_3 score of secondary structure prediction is 80.7% (compared to an average score of 80.6% on CASP targets). The average correlation of the HSE up and down values are respectively 0.74 and 0.66 (compared to the reported up and down correlations of 0.713 and 0.696 respectively). We do therefore not consider it to be a problem that the benchmark proteins exist in PDB. We predict the radius of gyration using Equation 2.

Branch and bound algorithms are typically used to find the global minimum solutions. However, we use EBBA for protein decoy generation and we therefore want to obtain a large number of structures. The 10.000 global best structures in terms of energy are therefore found and not just the global minimum. This can be done by maintaining a queue of 10.000 structures during the search. This number is still very small compared to the exponential size of the conformational space. For

Table 1. Column 2 shows the number of segments m and column 3 shows the number of segment structures u. Column 4 shows the order of helix, sheet and coil segments. Column 5 shows the size of the conformational space given by Equation 1 and column 6 shows the number of hours spent by the algorithm. Column 7 and 8 show the percentage of the 10.000 structures that fall below the given threshold. Column 9 shows the lowest RMSD of the 10.000 structures. Column 10 shows the energy of P^* which is the lowest energy structure. For each protein, there is an *exact* and a *predicted* row. Exact refers to HSE-vectors, radius of gyration and secondary structure obtained from the native structure. In the *predicted* rows, all input values are predicted from the amino acid sequence and the results can therefore be considered as *de novo*.

Type	m	u	SS	N	T	< 6 Å	< 5 Å	lowest	$Q(P^*)$
Protein A (1FC2), 43 residues									
Exact	5	8	CHCHC	1.7×10^8	0.1	18.1	7.0	**2.8**	4.34
Predicted	7	8	CHCHCHC	1.4×10^{12}	6.9	33.0	13.8	**4.5**	5.26
Homeodomain (1ENH), 54 residues									
Exact	6	8	CHCHCH	1.5×10^{10}	0.6	21.6	13.2	**3.1**	4.36
Predicted	7	8	CHCHCHC	1.4×10^{12}	6.1	4.1	0.8	**4.1**	5.70
Protein G (2GB1), 56 residues									
Exact	9	8	SCSCHCSCS	1.0×10^{16}	18.2	60.8	36.6	**3.4**	4.22
Predicted	10	8	SCSCHCSCSC	9.2×10^{17}	4.7	73.1	0.0	**5.3**	6.22
Cro repressor (2CRO), 65 residues									
Exact	11	4	CHCHCHCHCHC	4.0×10^{16}	24.1	5.7	1.4	**4.3**	6.49
Predicted	10	3	HCHCHCHCHC	5.1×10^{13}	7.4	1.5	0.0	**5.3**	5.89
Protein L7/L12 (1CTF), 68 residues									
Exact	8	8	SCHCHCHC	1.2×10^{14}	5.6	5.1	1.9	**4.6**	7.19
Predicted	11	3	SCHSHCHCHCS	1.7×10^{15}	19.2	0.1	0.0	**5.4**	5.84
Calbindin (4ICB), 76 residues									
Exact	11	2	CHCSHCHCHCH	1.9×10^{13}	3.56	4.5	0.7	**4.4**	6.18
Predicted	8	7	CHCHCHCH	4.1×10^{13}	31.4	0.5	0.0	**5.1**	6.79

comparison and evaluation of the model and prediction quality, all experiments are also done using the exact secondary structure and exact HSE-vectors obtained from the native structure of the proteins. All experiments were initially run with $u = 8$ (the number of segment structures). Some did not finish in 48 hours, and they were run with the highest value of u that could be solved in less than 48 hours. All computations were performed on a 2.4 GHz P4 with 512 RAM.

4 Results and Discussion

Table 1 shows the complexity of the models for different proteins and the running time of EBBA. The table also shows the results of running EBBA on the 6 benchmark proteins.

The maximum number of segment structures (u) that could be solved in less than 48 hours depends much on the number of segments of the protein. For the smallest proteins (1FC2 and 1ENH) the algorithm terminated in less than 48 hours using $u = 8$. Even though 2GB1 has relatively many segments the algorithm also

Table 2. Comparison between FB5-HMM and EBBA. Column 2 and column 4 show the percentage of good decoys for FB5-HMM and EBBA respectively. Column 3 and column 5 show the lowest RMSD of a structure found by FB5-HMM and EBBA respectively. Both algorithms uses predicted secondary structure information and predicted radius of gyration.

Protein	FB5-HMM		EBBA	
	< 6 Å	Min. RMSD	< 6 Å	Min. RMSD
Protein A (1FC2)	17.1	2.6	33.0	4.5
Homeodomain (1ENH)	12.2	3.8	4.1	4.1
Protein G (2GB1)	0.001	5.9	73.1	5.3
Cro repressor (2CRO)	1.0	4.1	1.5	5.3
Protein L7/L12 (1CTF)	0.3	4.1	0.1	5.4
Calbindin (4ICB)	0.4	4.5	0.5	5.1

terminated in less than 48 hours using $u = 8$. This is mainly because bounding occured early in the branch and bound tree.

The most difficult protein in terms of bounding efficiency is 4ICB (using predicted measures), where it turns out that significant bounding first occurs in level 5 of the branch and bound tree. In all instances, the conformational space is huge, and it is clear that finding global minimum structures could not have been done in reasonable time without efficient bounding.

Table 1 shows that the set of 10.000 low energy structures for all 6 proteins contains good decoys (RMSD less than 6 Å). Also, for all proteins the lowest RMSD is smallest when using exact values compared to the predicted values. This is expected since the energy landscape should have a global minimum closer to the native structure when using exact values. However, it is surprising that for two of the proteins (1FC2 and 2GB1) the fraction of good decoys (< 6 Å RMSD) is better when using predicted values compared to exact values.

The results have been compared directly with FB5-HMM [6] in Table 2. FB5-HMM is the state-of-the-art method for conformational sampling. The method is based on a Hidden Markov Model and generates a large set of structures which usually contains many good decoys when enforcing compactness. The major difference between FB5-HMM and EBBA is that FB5-HMM does not use an energy function. FB5-HMM can also benefit from the secondary structure prediction and radius of gyration prediction. The results we have shown for FB5-HMM are therefore obtained using predicted secondary structure and using a greedy collapse scheme. The results for FB5-HMM are from [6] where 100.000 structures are generated. The results show that EBBA finds a better percentage of good decoys for most of the proteins (1FC2, 2GB1, 2CRO and 4ICB). The high amount of good decoys for protein G is very interesting since protein G is known to be one of the more difficult structures in this benchmark set [15,6]. For all proteins, except 2GB1, FB5-HMM finds at least one structure with lower RMSD than EBBA. This is not surprising since FB5-HMM here generates 10 times as many decoys than EBBA and therefore has a much higher probability of hitting a low RMSD

structure. Another advantage of structures generated by EBBA, is that the geometry of the secondary structure segments is perfect because they are constructed using the correct secondary structure geometry. The running time of FB5-HMM for producing the set of 100.000 decoys is comparable to the running time of EBBA.

5 Conclusions

We have presented a branch and bound algorithm for finding the lowest energy structures in a large conformational search space. The results show that the set of low-energy structures is a very good decoy set. The energy function is based on HSE which is a simple predictable measure. This algorithm is the first *de novo* branch and bound algorithm for protein decoy generation using only one-dimensional predictable information. We have shown experimentally that good decoys always exist among the 10.000 lowest energy structures for the proteins used here. We have also shown that the algorithm is comparable in performance with the state-of-the-art conformational sampler FB5-HMM. The energy function is not accurate enough to pinpoint the lowest RMSD structure in this set. An important future research direction is therefore to examine this set of low energy structures with a more detailed energy function and to identify the native-like structures. The largest protein considered have 76 residues. There is a problem using the branch and bound algorithm on larger proteins since then only a small fraction of the conformational space can be searched in reasonable time. However, we believe that exploiting how super secondary structures [18,2] arrange in nature, might be a way to solve this problem. Better search heuristics for finding upper bounds on the energy can also be relevant since a good upper bound on the energy also improves the performance of the branch and bound algorithm. Using a more probabilistic approach might also improve the quality of the results. It might also be possible to train a Bayesian network to predict the probability of a given HSE-vector given the amino acid sequence. This would be a more detailed usage of the HSE-vector compared to the simple energy function used here.

Acknowledgements

We would like to thank Thomas Hamelryck at the Bioinformatics Centre, University of Copenhagen for valuable contributions and insights. Martin Paluszewski and Pawel Winter are partially supported by a grant from the Danish Research Council (51-00-0336).

References

1. Backofen, R., Will, S.: A constraint-based approach to fast and exact structure prediction in three-dimensional protein models. Constraints 11(1), 5–30 (2006)
2. Boutonnet, N.S., Kajava, A.V., Rooman, M.J.: Structural classification of alphabetabeta and betabetaalpha supersecondary structure units in proteins. Proteins 30(2), 193–212 (1998)

3. Chothia, C., Lesk, A.M.: The relation between the divergence of sequence and structure in proteins. The EMBO Journal 5, 823–826 (1986)
4. Fain, B., Levitt, M.: A novel method for sampling alpha-helical protein backbones. Journal of Molecular Biology 305, 191–201 (2001)
5. Hamelryck, T.: An amino acid has two sides: a new 2D measure provides a different view of solvent exposure. Proteins 59(1), 38–48 (2005)
6. Hamelryck, T., Kent, J.T., Krogh, A.: Sampling realistic protein conformations using local structural bias. PLoS Computational Biology 2(9), 1121–1133 (2006)
7. Kinjo, A.R., Nishikawa, K.: Recoverable one-dimensional encoding of three-dimensional protein structures. Bioinformatics 21(10), 2167–2170 (2005)
8. Kolodny, R., Levitt, M.: Protein decoy assembly using short fragments under geometric constraints. Biopolymers 68(3), 278–285 (2003)
9. Maranas, C.D., Floudas, C.A.: A deterministic global optimization approach for molecular structure determination. J. Chem. Phys. 100, 1247–1261 (1994)
10. McGuffin, L.J., Bryson, K., Jones, D.T.: The PSIPRED protein structure prediction server. Bioinformatics 16, 404–405 (2000)
11. Palu, A.D., Dovier, A., Fogolari, F.: Constraint logic programming approach to protein structure prediction. BMC Bioinformatics 5(186) (2004)
12. Paluszewski, M., Hamelryck, T., Winter, P.: Reconstructing protein structure from solvent exposure using tabu search. Algorithms for Molecular Biology 1 (2006)
13. Paluszewski, M., Winter, P.: EBBA: Efficient branch and bound algorithm for protein decoy generation, Department of Computer Science, Univ. of Copenhagen, vol. 08(08) (2008)
14. Pollastri, G., Baldi, P., Fariselli, P., Casadio, R.: Prediction of coordination number and relative solvent accessibility in proteins. Proteins 47(2), 142–153 (2002)
15. Simons, K.T., Kooperberg, C., Huang, E., Baker, D.: Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions. J. Mol. Biol. 268(1), 209–225 (1997)
16. Skolnick, J., Kolinski, A., Ortiz, A.R.: MONSSTER: A method for folding globular proteins with a small number of distance restraints. J. Mol. Biol. 265, 217–241 (1997)
17. Standley, D.M., Eyrich, V.A., Felts, A.K., Friesner, R.A., McDermott, A.E.: A branch and bound algorithm for protein structure refinement from sparse nmr data sets. J. Mol. Biol. 285, 1961–1710 (1999)
18. Sun, Z., Jiang, B.: Patterns and conformations of commonly occurring supersecondary structures (basic motifs) in protein data bank. J. Protein Chem. 15(7), 675–690 (1996)
19. Vilhjalmsson, B., Hamelryck, T.: Predicting a New Type of Solvent Exposure. In: ECCB, Computational Biology Madrid 2005, P-C35, Poster (2005)
20. Wolsey, L.A.: Integer Programming. Wiley-Interscience, Chichester (1998)

Chapter 11

Paper: Model Quality Assessment using Distance Constraints from Alignments

M. Paluszewski and K. Karplus. MQA using Distance Constraints from Alignments. *Proteins, Structure, Function and Bioinformatics (to appear)*, 2008.
Status: To appear

Model Quality Assessment using Distance Constraints from Alignments

Martin Paluszewski* and Kevin Karplus†

July 29, 2008

1 Abstract

Given a set of alternative models for a specific protein sequence, the *model quality assessment* (MQA) problem asks for an assignment of scores to each model in the set. A good MQA program assigns these scores such that they correlate well with real quality of the models, ideally scoring best that model which is closest to the true structure.

In this paper, we present a new approach for addressing the MQA problem. It is based on distance constraints extracted from alignments to templates of known structure, and is implemented in the Undertaker [9] program for protein structure prediction. One novel feature is that we extract non-contact constraints as well as contact constraints.

We describe how the distance constraint extraction is done and we show how they can be used to address the MQA problem. We have compared our method on CASP7 targets and the results show that our method is at least comparable with the best MQA methods that were assessed at CASP7 [7].

We also propose a new evaluation measure, Kendall's τ, that is more interpretable than conventional measures used for evaluating MQA methods (Pearson's r and Spearman's ρ).

We show clear examples where Kendall's τ agrees much more with our intuition of a correct MQA and we therefore propose that Kendall's τ be used for future CASP MQA assessments.

2 Introduction

Most search algorithms for protein structure prediction are guided by cost functions that assess how "protein-like" particular conformations of the polypeptide chain are. In theory, a perfect cost function would guide a good search algorithm to the native state of the protein, but such a cost function has yet to be discovered.

One of the obstacles is that many low-cost structures usually exist in the conformational search space and even good cost functions have trouble identifying the most native-like structure among them. For a given set of alternative models for some specific protein target, the *model quality assessment* (MQA) problem asks for an assignment of a score to each model in the set, such that the scores correlate well with the real quality of the model (that is, the similarity with the native structure). This assignment of scores is, of course, done without knowing the native structure of the protein.

A good MQA is crucial when one has to choose the best model among several different models—for example, in a metaserver for protein structure prediction. Metaservers use structure models generated by other methods and either choose one of the models using an MQA or construct a consensus model to make a predicted structure. The most successful MQA methods in the past have been either consensus methods (looking for features shared by many models in the set) or similarity to a single predicted model [7, 18].

The Lee group has been fairly successful at predicting the tertiary structure of CASP targets. Their method for MQA therefore first predicts the structure of the target and then measures the similarity between their prediction and the models to be assessed [7], a method which always predicts that their model will be the best. Our method differs from the Lee method in that we use a cost function with features derived from either multiple templates or multiple predictions. One of the strengths of our method is therefore that we do not have to come up with a consistent model from the inconsistent constraints. In fact, our method predicts one of our own server models to be best on only 16 of 91 CASP7 targets.

*University of Copenhagen; Computer Science

†University of California, Santa Cruz; Biomolecular Engineering

The Pcons method [18] uses a consensus approach, where consensus features are extracted from other predictions and used to score the models. The Pcons method therefore need the predictions from other methods and can not be used to assess the quality of a single model. Our method differs from Pcons since it does not depend on other predictions when the distance constraints are derived from templates.

Qiu et al. [15] recently proposed an MQA algorithm based on *support vector regression* (SVR). The method is trained on a large number of models (CASP5 and CASP6) to learn the weights in a complex score function. This score function is a linear combination of both consensus-based features and individual features, but relies mainly on the consensus-based features. Our method is simpler, does not rely on consensus, and does not depend much on machine-learned parameters. In a companion paper, Archie and Karplus use a different machine learning approach to extend our method to include consensus terms similar to those used by Qiu et al., improving further on our method. [3]

The most accurate methods for protein structure prediction are based on copying backbone conformations from *templates*, proteins of known structure with sequences similar to the target sequence. Proteins with similar sequences are usually the result of evolution from a common ancestral sequence and most often have very similar structures [6]. In this paper, we use techniques borrowed from template-based modeling and use them to address the MQA problem.

Different template search methods exist in literature. Among the simplest and fastest methods are BLAST [1] and FASTA [14], which are powerful when the sequence similarity between the target and templates is high. For more difficult cases, methods like SAM_T04 [10] and PSI-BLAST [2] do a better job of detecting remote homologs. In addition to identifying the actual template(s) for a target, most methods also compute one or more alignments of the target sequence to the templates. These alignments are used in many ways by different protein structure prediction algorithms: the most common is to copy the backbone from the aligned residues, also common is to use the alignment to get rigid fragments for a fragment assembly algorithm [10, 21, 20, 4], and yet another approach is to extract spatial constraints and construct a protein model that best satisfies these constraints as in MODELLER [17].

Our method is also based on alignments from templates. We use the SAM_T06 hidden Markov model protocol (a slightly improved version of the SAM_T04 protocol) to search for templates and compute alignments. Then we identify pairs of aligned residues that are in contact in some template and compute a consensus distance between these residues.

Our method then uses a combination of predicted contact probability distributions and E-values from the template search to choose a subset of high quality consensus distances. These selected distances are then used for scoring the models in the MQA problem. The steps of extracting alignments, computing consensus distances, and selecting high quality distances are described in more detail in the Methods section.

We show that the consensus distances from alignments can be treated as weighted distance constraints, where the weights are heavily correlated with their real quality. The cost functions obtained from the distance constraints are evaluated on the MQA problem from CASP7 where the participating groups were asked to evaluate the quality of server models of different targets.

At CASP7 the MQA methods were initially evaluated using Pearson's r between the predicted quality and GDT_TS and the ranking of the methods was done from the z-scores of Pearson's r [7]. Later, McGuffin noticed that "the data are not always found to be linear and normally distributed," and he therefore used Spearman's ρ for his analysis [13].

Here we propose an alternative measure, Kendall's τ, which measures the degree of correspondence between two rankings. See Section 3.2 for an explanation of why we believe it is more interpretable than Pearson's r and Spearman's ρ and for examples of quality assessments where Kendall's τ agrees more with the intuition of a good MQA than Pearson's r does.

The results show that our method is comparable to the best ranked methods at CASP7 (Pcons and Lee) without using consensus-based methods. When

the distance constraints are combined with the other Undertaker cost functions our MQA method can be improved even further as described in Archie and Karplus [3].

3 Materials and Methods

3.1 Benchmarks

At CASP7 there were a total of 95 targets assessed. The benchmarks used here consist of the 86 targets from CASP7 that had a native structure released in PDB by July 2007. For each target, we include all complete models (no missing atoms) from the tertiary structure prediction category and all models (including those with missing atoms) from server predictions. For each model we also compute a SCWRL'ed model, by running SCWRL 3.0 [5] to re-optimize the position of the sidechains. For the backbone-only models, we include only the SCWRL'ed models in the benchmark, since our distance constraints are on C_β atoms. This benchmark set is called *benchmark A* and is primarily used for testing different versions of our MQA method.

Benchmark B consists of 91 targets (it was generated later than benchmark A and consequently PDB had more targets released) but contains only complete models from server predictions, not SCRWL'ed models or models from human predictions. The server models were assessed at CASP7, so our MQA method on Benchmark B can therefore be compared directly with other methods.

When we construct the benchmarks in this way, benchmark A will eventually include benchmark B. The reason for this is, that we want to evaluate our MQA methods using as many models as possible (benchmark A) to make our results more reliable. Benchmark A generally also contains better models than benchmark B. However, only benchmark B results for the other MQA methods have are available and we therefore use benchmark B for comparisons of the different methods, even though a larger benchmark would have been more appropriate. A problem with this approach could be that training a method to give good results on benchmark A would eventually also give good results on benchmark B. Our MQA method, however, does not contain parameters that need to be trained on a specific set. The few parameters that determine the shape of the cost function have been given ad-hoc values and we therefore do not believe that the inclusion of benchmark B in benchmark A is a problem for evaluating our method.

3.2 Evaluation of MQA

There are several ways of evaluating a model-quality-assessment method depending on the application. For some applications, it suffices to determine the true quality of the best-scoring model. In other applications, it is important for the MQA function to do a proper *ranking* of the models. Correlation measures that evaluate the ranking of models are more robust than measures that examine only the quality of the best-scoring model.

In CASP7, the participating methods were ranked using Pearson's r, which measures the *linear* correspondence between the predicted quality from the MQA and a measure of true quality. The particular measure of true quality used in CASP7 was GDT (global distance test) [19] which is roughly the fraction of C_α atoms that are correctly placed. This measure ignores errors in sidechain and peptide-plane placement, but is well accepted as a measure of the quality of a C_α trace.

We favor the use of a correlation measure, but we think that it is more important to predict a good *ranking* of the models than predicting a *linear* relation between quality and GDT. We therefore propose Kendall's τ for evaluating MQA methods and suggest that it be used for ranking methods at future CASPs. Kendall's τ measures the degree of correspondence between two rankings and is defined as

$$\tau = \frac{4P}{N(N-1)} - 1$$

where N is the number of points, and P is the number of *concordant* pairs. A pair of points is said to be concordant if

$$\text{sign}(X_A - X_B) = \text{sign}(Y_A - Y_B) \ .$$

In the case of ties, if either $X_A = X_B$ or $Y_A = y_B$, we add 0.5 to P rather than 1.

In other words, if two random points (A and B) are chosen and $X_A > X_B$, then Kendall's τ is the probability that $Y_A > Y_B$. We think that Kendall's

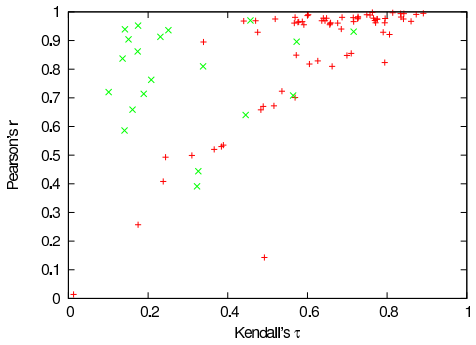


Figure 1: Each point corresponds to a target in benchmark B. The Pearson's r and Kendall's τ are computed from the Pcons MQA. The green points (x) are the 20 assessments with most ties, which inflates the values of Pearson's r.

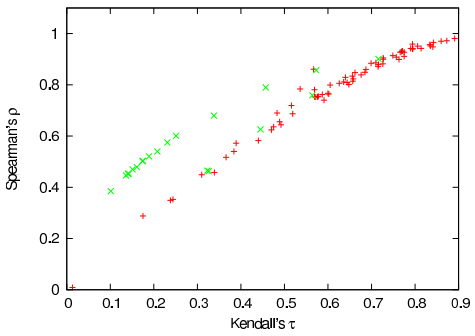


Figure 2: Each point corresponds to a target in benchmark B. The Spearman's ρ and Kendall's τ are computed from the Pcons MQA. The green points (x) are the 20 assessments with most ties, which inflates the value of Spearman's ρ.

τ is much more interpretable than either Pearson's r or Spearman's ρ, and it does a better job of ranking MQA methods than Pearson's r.

In Figure 1 a plot shows Kendall's τ vs. Pearson's r for benchmark B using the assessments from Pcons. In many cases, MQA algorithms like Pcons give equal scores to different models. This, of course, makes sense if the method can not establish a proper ranking of the different model. However, the plot in Figure 1 clearly shows that Pearson's r highly rewards the tied assessments. The plot also shows that this is not the case when using Kendall's τ. A similar problem exists with Spearman's ρ. Even though it measures ranking explicitly, it slightly favors highly tied assessments (Figure 2). Figure 3 shows two of the highly tied assessments compared with our assessments. The facts that Pearson's r measures *linear* correlation and highly favors tied ranks make it inappropriate for evaluating MQAs. Spearman's ρ is a better measure than Pearson's r because it measures the correlation of the *ranks*, however it still slightly favors tied ranking. Kendall's τ is much more interpretable than Pearson's r and Spearman's ρ and does not have the problems mentioned above, we therefore recommend Kendall's τ for evaluation of MQA methods.

Other measures, like the ability to select the best model, could also be considered when comparing MQA algorithms, though this approach, relying as it does on a single data points, is very sensitive to noise. In all cases, the individual scatter plots should be examined as the examples in Figure 3 to avoid misleading correlation coefficients.

The naive implementation of Kendall's τ, which simply considers all pairs of points, runs in $O(n^2)$. However, the more efficient algorithm by Knight [12] runs in $O(n \log n)$, which is not much more expensive than the $O(n)$ algorithms for other correlations. Statistical tools like R [16] include routines for Kendall's τ computations.

3.3 Model Quality Assessment method

Our MQA consists of the following steps which are described in details in the following sections.

1. Templates and alignments are found using SAM_T06.
2. The distances between pairs of residues in contact are extracted for each alignment.
3. For each pair of residues that are in contact in at least one alignment, a consensus distance is computed (the *desired distance*).
4. Weighted constraints are constructed from the desired distances.
5. (Optional) An optimization algorithm selects a subset of constraints using predicted contact distributions.

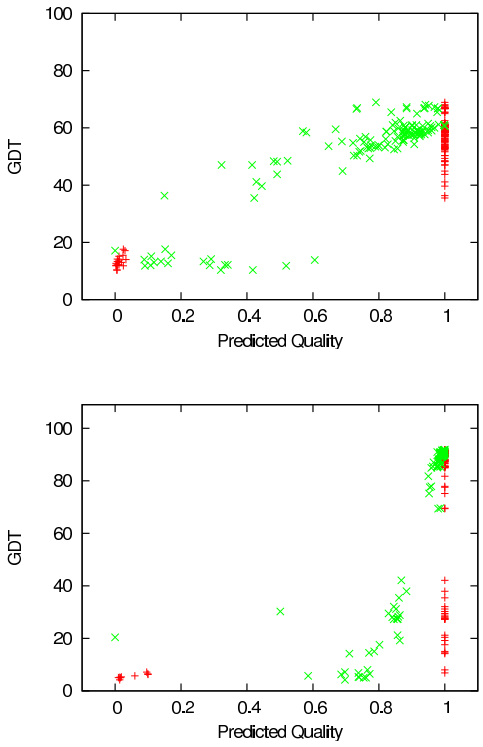


Figure 3: Different MQAs for complete server models of targets T0370 (upper) and T0334 (lower). The set of red points (+) is the MQA from the top-ranked group (634, Pcons) at CASP7 and the set of green points (X) is our MQA. The correlation values for the assessments in the Figure are: T0370 (+): r=0.94, τ=0.25, ρ=0.60. T0370 (X): r=0.89, τ=0.61, ρ=0.79. T0334 (+): r=0.59, τ=0.14, ρ=0.45. T0334 (X): r=0.54, τ=0.72, ρ=0.90. Evaluations using Pearson's r would slightly prefer the Pcons MQA even though it clearly is not what we expect of a good MQA. However, Kendall's τ and Spearman's ρ are much higher for our assessments, because they do much better *rankings* of the models.

6. Each model is scored according to the (selected) distance constraints.

3.4 Templates and Alignments

We use the fully automated SAM_T06 protocol to find templates and compute alignments. SAM_T06 is a profile HMM that excels in detecting remote homologs. The alignments used are local alignments to a three-track HMM [9, 8] using the amino-acid alphabet, the str2 backbone alphabet [10], and the near-backbone-11 burial alphabet [10], with weights 0.8, 0.6, and 0.8 respectively. This is the alignment setting that has worked best in our tests of various alignment methods for maximizing the similarity to a structural alignment—we did not optimize these settings for the MQA application. For each template, there were three such alignments, using the SAM_T2K, SAM_T04, and SAM_T06 multiple-sequence alignments as the base for the local structure predictions and the HMMs.

3.5 Distance Extraction

The next step is to extract the conserved distances of the residue pairs from the alignments. Distance is measured between the C_β-atoms of the residues (C_α-atoms for glycines). For each alignment, the distances between all C_β pairs that have a separation of more than 8 residues and a Euclidean distance ≤ 8 Å are stored. We use a chain separation of 8 residues to avoid trivial chain neighbor contacts—we have not yet experimented with different separation cutoffs. We have experimented with various values of the cutoff radius. Small cutoff radii increase the accuracy of the constraints, but fewer constraints are detected. On the other hand, larger cutoff radii generate more constraints, but their quality decreases rapidly because the larger distances are less conserved. Our ad-hoc experiments therefore suggest that a cutoff radius between 7 and 9 Å gives a good trade-off between sensitivity and accuracy.

This distance extraction therefore results in a triangular *protein length* $\times$ *protein length* table, where a table entry holds the set of all alignment distances between the corresponding pair of residues. Together with each distance, we also store a weight corresponding to the quality of the template from which the distance was extracted. The quality of a template is calculated directly from the E-values of the template. However, we normalize it such that the weight $w(E)$ of an E-value is in the range [0.1:1]

$$w(E) = 1 - 0.9 \left(\frac{E - E_{\min}}{(E_{\max} - E_{\min}) + \epsilon} \right) .$$

$w(E) = 1$ therefore corresponds to the highest-quality template (lowest E-value) and $w(E) = 0.1$

corresponds to the lowest-quality template (highest E-value). The parameter ϵ is an arbitrary very small number for avoiding division by zero.

The $E_{\max}$ value was generally around 36 for the CASP7 targets. Easy targets with many good hits ($E <\!\!\ll 1$) therefore have many hits with weights close to 1. This might be problematic, since this weighting scheme can not distinguish between excellent hits and only fairly good hits, as they are almost equally close to $E_{\min}$. We have not yet experimented with other weighting schemes, but this problem might be avoided by limiting the number of templates examined, so that targets with several good hits would have much lower $E_{\max}$ values.

3.6 Desired Distances

From the table of distances and weights, a consensus distance for each pair of residues is computed by calculating a weighted average of the observed distances. After this step, the templates and alignments are therefore reduced to a table of so-called *desired distances* between residues. Each desired distance also has an associated weight (the sum of the weights of the templates where the distances were observed). If two residues have been in contact in many alignments that scored well, the weight is therefore high. Correspondingly, if two residues have only been in contact in few alignments coming from poorly scoring templates, the desired distance will have a low weight. The weights of the desired distances can therefore be interpreted as the confidence of the distance prediction. If two residues have not been observed to be in contact, the desired distance is undefined and the associated weight is 0.

3.7 Weighted Distance Constraints

For each desired distance D_{ij} between residues i and j, we generate a weighted *distance constraint*. A distance constraint has a *minimum distance* A_{ij}, *desired distance* D_{ij}, *maximum distance* B_{ij} and a *weight* W_{ij}. For the constraints in our MQA, the minimum and maximum distances are set somewhat arbitrarily to $A_{ij} = 0.8D_{ij}$ and $B_{ij} = 1.3D_{ij}$. A distance constraint defines a cost function that is a rational function with minimum $C(D_{ij}) = -W_{ij}$, $C'(D_{ij}) = 0$, and $C(A_{ij}) = C(B_{ij}) = 0$:

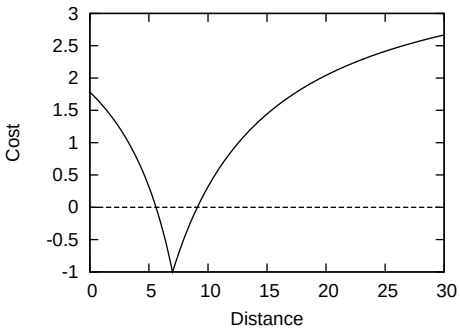


Figure 4: The cost function with parameters $D_{ij} = 7$, $\alpha = 200$, $\beta = 50$, $W_{ij} = 1$.

$$C(\delta_{ij}) = W_{ij}\frac{\alpha S_{ij}^2 + (1-\alpha)S_{ij} - 1}{\beta S_{ij}^2 + (\alpha - 1)S_{ij} + 1} \quad (1)$$

$$S_{ij} = \frac{(\delta_{ij} - D_{ij})}{(L_{ij} - D_{ij})} \quad (2)$$

$$L_{ij} = \begin{cases} B_{ij} & \text{if } \delta_{ij} \geq D_{ij} \\ A_{ij} & \text{otherwise} \end{cases} \quad (3)$$

The α and β parameters define the shape of the function (Equations 4 and 5) and are most easily interpreted in terms of the asymptote at ∞ and the slope at the maximum distance:

$$C(\infty) = \alpha/\beta \quad (4)$$

$$C'(B_{ij}) = \frac{\alpha + 1}{(\alpha + \beta)(B_{ij} - D_{ij})} \quad (5)$$

Figure 4 shows a plot of the function with typical settings. The final cost function is the weighted average of the individual costs for all constraints used.

3.8 Selection of Constraints

For the basic MQA method, the model cost function is the sum of all of the cost functions for the pairs of residues, but the method can be improved by using only a good subset of the constraints. We have evaluated several selection strategies and describe two of them here. The *selection by fraction* strategy is very simple, but improves the performance of the MQA method only marginally. The *selection using contact predictions* strategy is more complicated, but is the best selection we have tried.

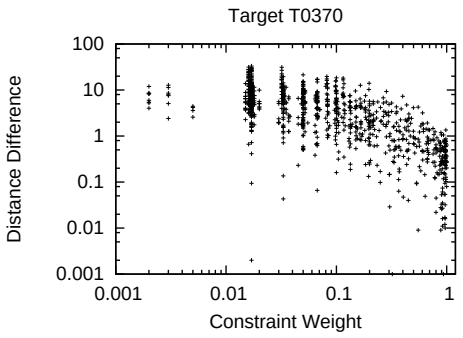


Figure 5: Each point corresponds to a constraint. The weight of the constraint is shown on the x-axis and the magnitude of the difference between the actual distance in the experimental structure and the desired distance of the constraint is shown on the y-axis. From the scatter diagram, it is easy to see that high-weight constraints tend to have low errors in distance. This property is true for almost all targets considered.

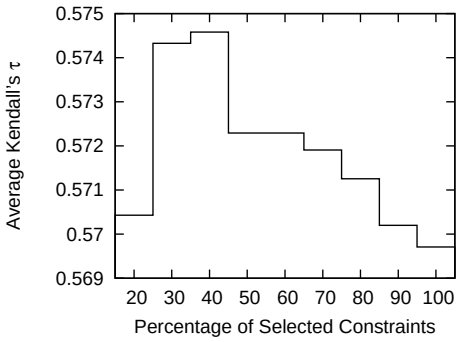


Figure 6: The average Kendall's τ is maximized when selecting approximately 30%–40% of the highest weight constraints.

3.8.1 Selection by Fraction

A plot of the error of the constraints vs. their weight is shown in Figure 5 for Target T0370. It clearly shows that high-weight constraints are generally more correct than low-weight constraints. Although we show this property only for one arbitrarily chosen target, a similar relationship holds for most of the targets, though it is strongest for targets for which good templates are available. A simple selection strategy is therefore to sort the constraints by weight and to select a fraction of the highest weight constraints for the final model cost function.

Figure 6 shows the average Kendall's τ for selecting different fractions of the high-weight constraints. The plot shows that the average Kendall's τ for Benchmark A increases from 0.570 using all constraints (100%) to 0.575 when selecting only 40% of the highest weight constraints, but that the quality of our MQA method decreases rapidly when selecting less than 30%. This decline is because we are beginning to discard many good constraints at this point. Even though the increase in average Kendall's τ is small, the result is important because it shows that a proper selection of constraints can improve the performance of our method.

3.8.2 Selection using Contact Predictions

We can predict how many contacts each residue should have using neural nets, then select constraints so that residues predicted to have more contacts have more constraints also.

We trained neural nets to predict probability $P_{i,c}$ of residue i having c contacts with separation greater than 8 residues. Residues are said to be in contact if the distance between their C_β-atoms (C_α-atoms for glycines) is less than 8 Å; the same definition we used for extracting constraints. The contact number predictions are done using the same neural network program (predict-2nd) that we use for all our local structure prediction [11].

Our main selection strategy is to select a subset of constraints that maximizes the contact number probability for each residue, but we also want to have many high-weight constraints. Two objectives must therefore be maximized: the contact number probability and the average weight of the chosen constraints. We used a simple greedy algorithm to do this optimization: Figure 7.

The asymptotic running time of the algorithm is $O(In^2)$ where I is the number of improvements and n is the number of constraints. In practice the algorithm runs in reasonable time $< 5s$ for problems with fewer than 10 000 constraints. For larger problems, the quadratic-time optimization step is

```
C ← list of constraints sorted from highest weight to lowest weight
improved←true
while improved do
  improved←false
  for i ← 1 to size(C) do
    if insertion of C_i improves total probability then
      insert C_i, improved←true
    end if
  end for
  {Here, no insertions can improve the total probability}
  for i ← size(C) to 1 do
    if removal of C_i improves total probability then
      remove C_i, improved←true
    end if
  end for
  {Here, no removals can improve total probability}
  for i ← 1 to size(C) do
    for j ← i+1 to size(C) do
      if changing insertion state for C_i and C_j improves total probability
      and average weights of constraints then
        change state of C_i and C_j, improved←true
      end if
    end for
  end for
end while
```

Figure 7: The optimization algorithm for selecting high-weight constraints based on neural-net predictions of the contact number for each residue. Note that each constraint C_i affects the probability for the contact number of two residues. When there are more than 10 000 constraints in set C, we skip the final quadratic-time step, since it offers only small improvements.

skipped, since it only contributes small improvements compared to the initial linear-time optimization. Using the optimized set of constraints the average Kendall's τ improved from 0.570 using all constraints to 0.582. This selection strategy gives the best improvement in terms of average Kendall's τ of any we have tried, and we do not have to tune a parameter that might be benchmark-dependent like the fraction parameter.

3.8.3 Prediction of non-contacts

The above selection strategies show how a reduction of the constraint set can improve the quality of the method. We have also found that the addition of so-called *non-contact constraints* also improves the method substantially. The idea is simply that if a pair of residues is not observed to be in contact in any alignment, then a non-contact constraint is added to the constraint set. This is a special constraint that only penalizes residues being in contact. This behavior can also be modeled with our standard cost function (Equation 1) by setting $D_{ij} = 8$, $A_{ij} = 7.5$, $B_{ij} = \infty$ (in practice, we use 10 000 to be effectively ∞). The non-contact cost function is illustrated in Figure 8.

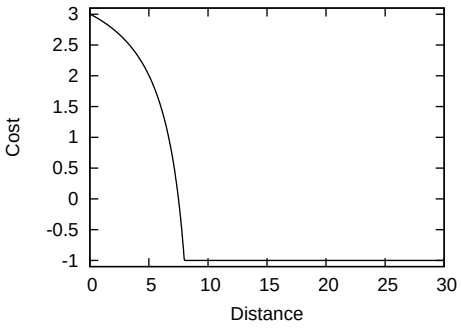


Figure 8: The non-contact cost function with parameters $D_{ij} = 8$, $A_{ij} = 7.5$, $B_{ij} = \infty$, $\alpha = 200$, $\beta = 50$.

Using the optimized set of constraints together with the non-contact constraints improves the average Kendall's τ from 0.582 using just the optimized contact constraints to 0.589.

3.8.4 Constraints from Predicted Models

The top-ranked method (Pcons) at CASP7 builds its scoring function from consensus features of the models to be assessed. This approach works very well for CASP MQA since many of the models are of high quality and the consensus features are therefore more likely to be good. Our method for constraint extraction and optimization can easily be generalized to consider the predicted models as well. However, we stress that this approach can only be successful when the model set is large enough to express correct consensus features. In the case of assessing the quality of few models (or one model in the extreme case), the constraints should be extracted from alignments.

When extracting distance constraints from the alignments, we have a clear indication of the alignment quality from the template E-value. This is usually not the case when extracting constraints from predicted models. We therefore performed one experiment where all of the models are equally weighted and another experiment where the models are weighted according to the model cost given by alignment constraints. The results of these experiments are summarized in Table 1. In both exper-

Experiment	All	Optimized
Equal Model Weights	0.591 (0.830)	0.621 (0.863)
Weighted Models	0.598 (0.839)	0.622 (0.866)

Table 1: The *all* and *optimized* columns show the average Kendall's *tau* (Pearson's r in parenthesis) for the two consensus experiments that used constraints extracted from the set of models to be evaluated. *All* corresponds to selecting all constraints. The *optimized* column corresponds to the constraints selected by the optimization algorithm described in Figure 7.

Constraint Set	$\bar{\tau}$	$\bar{r}$
All	0.570	0.825
Best fraction	0.575	0.833
Optimized	0.582	0.838
Opt+noncontacts	0.589	0.827
Opt+models	0.622	0.866

Table 2: Average Kendall's τ and Pearson's r for different versions of the MQA method using Benchmark A. Note that while Kendall's τ is improved by using non-contact constraints, Pearson's r is decreased. The inclusion of non-contacts decreases the *linearity* of the correlation, but improves the ranking of models. We have argued that we prefer Kendall's τ over Pearson's r, and so we consider the non-contacts to be beneficial to our MQA method.

iments there are significant gains when optimizing the constraint sets. However, the qualities of two optimized constraint sets are very similar, which indicates that the optimization algorithm is able to the choose good constraints also in the unweighted experiment.

The performances of the different alignment extraction algorithms and the weighted model extraction algorithm are summarized in Table 2.

4 Results

Here we evaluate our alignment constraints for MQA. This is done by splitting the constraints into three disjoint sets.

alignment constraints These are the constraints that are selected by the optimization algorithm described in Figure 7.

rejected alignment constraints These are the constraints that were not selected by the optimization algorithm in Figure 7.

non-contact constraints Constraints between pairs of residues that were not observed to be in contact in any alignment.

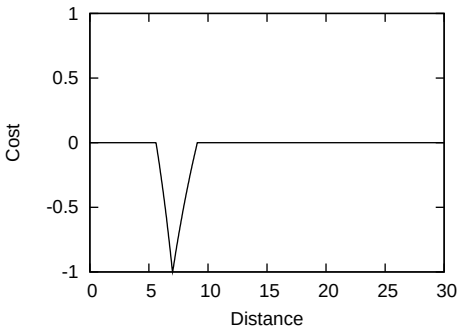


Figure 9: The *bonus* cost function with parameters $D_{ij} = 7$, $\alpha = 200$, $\beta = 50$. This type of cost function is 0 when $D_{ij} < A_{ij}$ or $D_{ij} > B_{ij}$, otherwise it behaves as described in Equation 1. The bonus cost functions are useful for low quality constraints. As the name indicates, a bonus constraint only rewards models when the constraint is satisfied.

We also consider three additional sets, which are constructed by using a bonus cost functions on the above constraint sets (Figure 9), which provides negative costs, but no positive costs (truncating the standard cost function for a constraint at 0). The total cost function is a weighted sum of costs from the 6 constraint sets. A five-fold cross-validation was done to test the weighted cost function, using the cross-validation and optimization techniques described in the companion paper by Archie and Karplus [3]. We do not report the weights for the various cost functions here, as they came out very slightly different for each train/test split.

We compare our MQA with various MQA methods including the best ranked group at CASP7. This is done using Benchmark B consisting of complete (no missing atoms) server models from CASP7. The

results are summarized in Table 3. The table is extracted from the companion paper by Archie and Karplus [3], which describes the statistics and the data used in Table 3. Optimal weights trained on all CASP7 targets are shown in Table 4. Pooled standard deviation is defined by

$$\sigma_{\text{pooled}} = \sqrt{\frac{\sum_{t\in\mathbb{T}}(n_t-1)\sigma_t^2}{\sum_{t\in\mathbb{T}}(n_t-1)}} \qquad (6)$$

where $\mathbb{T}$ is the set of targets, n_t is the number of structures for target t, and σ_t is the standard deviation of the cost function among models of target t. The pooled standard deviation of the weighted cost function component is a useful way of gauging how much the component contributes to the final cost function. It is more informative than the raw weight of the component, because it does not depend on the rather arbitrary scaling of the individual components.

Figure 10 shows a comparison between our MQA method and the two best MQA methods at CASP7. When comparing our method with Pcons (upper Figure), the plot clearly shows that our algorithm is generally performing better on the easy targets (template-based targets). When comparing our algorithm with the Lee algorithm we, surprisingly, see the opposite behavior: our method does better on the harder targets.

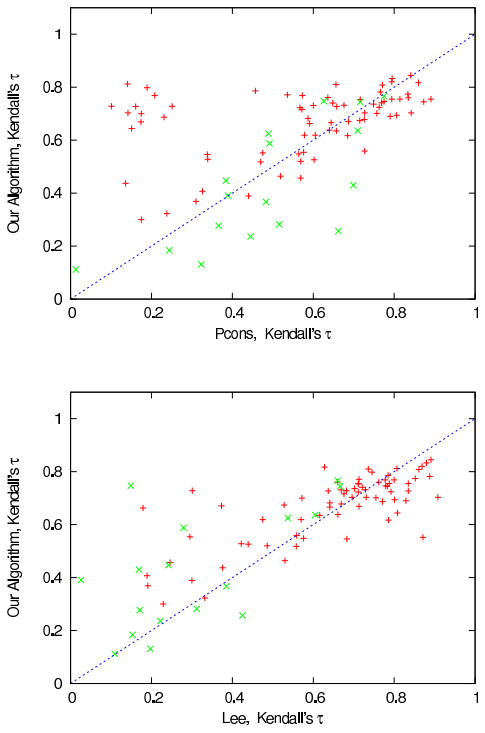


Figure 10: Each point corresponds to a target in benchmark B. Here we show average Kendall's τ using our algorithm (constraints extracted from models) vs. Pcons and the Lee algorithm. Easy targets (marked with red +) correspond to template-based targets and hard targets (marked with green x) correspond to template-free models using the CASP7 classification.

4.1 Quality of Templates is Important

Since our MQA method is based on homology modeling, the existence of good templates is crucial. It is not possible to know the real quality of a template without knowing the native structure of the target, but the E-value of the template from the search is a good indication of its quality. Figure 11 shows the relationship of the lowest E-value for the target compared to the Kendall's τ for that target. If we find a template with E-value less than 0.9, then the performance of the MQA is generally good, but if the best template E-value is more than 0.9, we can't predict the performance of the MQA based on the E-value only.

5 Discussion and Conclusion

We have presented a simple and powerful method for extracting distance constraints from alignments. We have shown how these constraints can be used as a score function for model quality assessment. Our results in Table 3 indicate that MQA using the alignment constraints is comparable in quality to the best methods at CASP7. The distance constraints from alignments are based on evolutionary information only, but are often useful even when sensitive fold-recognition methods do not reliably detect templates.

Even though we here focus on extracting distance constraints from *alignments*, our algorithm also performs very well when extracting the constraints from the models to be assessed. The models from the CASP7 MQA are generally of high quality and we

Group	$\bar{\tau}$	$\bar{r}$
Meta-weighted	**0.624**	**0.862**
Meta-unweighted	**0.624**	**0.861**
Lee	0.585	0.805
Qiu	0.581	0.853
Align-all	**0.574**	**0.832**
Align-only	**0.570**	**0.832**
Pcons	0.560	0.847
TASSER	0.538	0.633

Table 3: The table shows the average Kendall's τ and average Pearson's r using benchmark B. Correlation is computed separately for each target, then averaged. The *Align-all* row is the results of MQA with distance constraints from alignments using the 6 constraint sets described here. The *Align-only* row is the results of MQA with no noncontacts. The *Meta-weighted* and *meta-unweighted* rows are the results of extracting constraints from the models to be assessed (with weighted models and unweighted models respectively). TASSER, Lee, and Pcons are top-ranked MQA methods presented at CASP7 (groups 125, 556, and 634 respectively). Qiu is a newer MQA method described in Qiu et al. [15]. The companion paper by Archie and Karplus [3], evaluates our MQA algorithm on more measures.

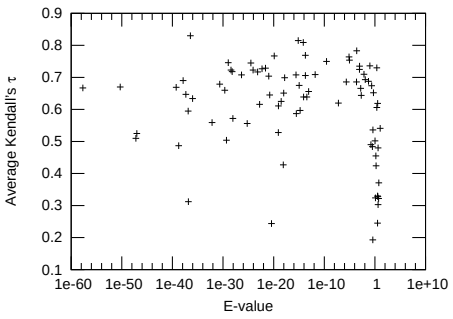


Figure 11: Each point corresponds to a target. The lowest E-value of any template for the target is shown on the x-axis and the Kendall's τ of the models is shown on the y-axis.
The two outliers are T0379 (4E-21,0.244) and T0375 (1.4E-37,0.312). Both targets had many templates and good models from many servers, so that getting a high correlation with quality requires detecting fairly small differences between models. There appear to be two sets of models for both targets (one using a good template and one using a poorer template), with high correlation between the MQA measure and GDT within each set, but without clean separation of the sets.

Cost Function	Weight	Pooled SD
align_constraint	9.95242	6.16873
noncontacts	59.6361	0.854129
noncontacts_bonus	30.4114	0.300117

Table 4: Optimized weights for alignment-based cost functions. Weights were optimized to maximize a weighted measure of correlation (τ_3, described elsewhere [3]) with GDT_TS on complete models.

therefore get a better performance when extracting constraints from the models compared to the alignments. However, in general we can not always expect to have such a large fraction of good models and extracting from alignments seems safer when the method is applied to an unknown collection of models.

When comparing our method with the two best ranked methods at CASP7, we notice that our method is generally better than Pcons on the template-based targets.

On the other hand it is quite surprising that our method performs better than the Lee method on most of the hard targets. The reason for this is that the Lee method only use one predicted base model for comparison. This, of course, works well when the predicted model is good. For the hard targets where our method is doing particularly better than the Lee method, (T0321 and T0350), the base models predicted by the Lee group were poor. Our algorithm therefore seems to be robust on both easy and hard targets.

We have also presented an alternative measure for evaluating an MQA method, the Kendall's τ, and provided several arguments why this measure should be used for future CASP MQA assessments.

6 Acknowledgments

We would like to thank all people who have worked on Undertaker and SAM. Specifically John Archie, who created the test framework and the optimization algorithm for combining cost functions; Grant Thiltgen trained the predict-2nd neural network for prediction of the distribution of contact counts.

Martin Paluszewski is partially supported by a grant from the Danish Research Council (51-00-0336). This research was also supported by NIH grant R01 GM068570.

References

[1] S. Altschul, W. Gish, W. Miller, E. Myers, and D. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410, 1990.

[2] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, 25(17):3389–3402, September 1997.

[3] John Archie and Kevin Karplus. Applying Undertaker cost functions to model quality assessment. *Proteins: Structure, Function, and Bioinformatics*, 2008. Manuscript in preparation.

[4] Philip Bradley, Lars Malmström, Bin Qian, Jack Schonbrun, Dylan Chivian, David E. Kim, Jens Meiler, Kira M.S. Misura, and David Baker. Free modeling with Rosetta in CASP6. *Proteins: Structure, Function, and Bioinformatics*, 61(S7):128–134, September 2005.

[5] A. A. Canutescu, A. A. Shelenkov, and R. L. Dunbrack. A graph-theory algorithm for rapid protein side-chain prediction. *Protein Science*, 12(9):2001–2014, September 2003.

[6] C. Chothia and A.M. Lesk. The relation between the divergence of sequence and structure in proteins. *EMBO Journal*, 5(4):823–826, April 1986.

[7] Domenico Cozzetto, Andriy Kryshtafovych, Michele Ceriani, and Anna Tramontano. Assessment of predictions in the model quality assessment category. *Proteins: Structure, Function, and Bioinformatics*, 69(S8):175–183, 6 August 2007.

[8] R. Karchin, M. Cline, Y. Mandel-Gutfreund, and K. Karplus. Hidden Markov models that use predicted local structure for fold recognition: alphabets of backbone geometry. *Proteins: Structure, Function, and Genetics*, 51:504–514, June 2003.

[9] Kevin Karplus, Rachel Karchin, Jenny Draper, Jonathan Casper, Yael Mandel-Gutfreund, Mark Diekhans, and Richard Hughey. Combining local-structure, fold-recognition, and new-fold methods for protein structure prediction. *Proteins: Structure, Function, and Genetics*, 53(S6):491–496, 15 October 2003.

[10] Kevin Karplus, Sol Katzman, George Shackelford, Martina Koeva, Jenny Draper, Bret Barnes, Marcia Soriano, and Richard Hughey. SAM-T04: what's new in protein-structure prediction for CASP6. *Proteins: Structure, Function, and Bioinformatics*, 61(S7):135–142, September 2005.

[11] Sol Katzman, Christian Barrett, Grant Thiltgen, Rachel Karchin, and Kevin Karplus. Predict-2nd: a tool for generalized local structure prediction. *Bioinformatics*, 2008. Manuscript in preparation.

[12] W. R. Knight. A computer method for calculating Kendall's tau with ungrouped data. *J. Am. Stat. Assoc.*, 61:436–439, 1966.

[13] Liam J. Mcguffin. Benchmarking consensus model quality assessment for protein fold recognition. *BMC Bioinformatics*, 8:345+, September 2007.

[14] W. R. Pearson and D. J. Lipman. Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences, USA*, 85(8):2444–2448, April 1988.

[15] Jian Qiu, Will Sheffler, David Baker, and William Stafford Noble. Ranking predicted protein structures with support vector regression. *Proteins: Structure, Function, and Bioinformatics*, 71(3):1175–1182, May 15 2008.

[16] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0.

[17] A. Sali and T. L. Blundell. Comparative protein modelling by satisfaction of spatial restraints. *Journal of Molecular Biology*, 234(3):779–815, December 1993.

[18] Björn Wallner and Arne Elofsson. Prediction of global and local model quality in CASP7 using Pcons and ProQ. *Proteins: Structure, Function, and Bioinformatics*, 69(S8):184–193, September 2007.

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

[20] Yang Zhang. Template-based modeling and free modeling by I-TASSER in CASP7. *Proteins: Structure, Function, and Bioinformatics*, 69(S8):108–117, 2007.

[21] Yang Zhang, Adrian K. K. Arakaki, and Jeffrey Skolnick. TASSER: An automated method for the prediction of protein tertiary structures in CASP6. *Proteins: Structure, Function, and Bioinformatics*, 61(S7):91–98, September 2005.

Chapter 12

Paper: Protein Structure Prediction using Bee Colony Optimization Metaheuristic (draft)

R. Fonseca, M. Paluszewski, and P. Winter. Protein Structure Prediction using Bee Colony Optimization Metaheuristic (draft).

Status: This research is work in progress. We need a few more experiments and the paper needs some fine tuning before we are ready to submit it to a journal or conference. However, I choose to include it here, because it contains important results that are relevant in this study.

Also note that the design of the algorithms and implementations were primarily done by Rasmus Fonseca.

Protein Structure Prediction Using Bee Colony Optimization

R. Fonseca, M. Paluszewski & P. Winter

September 5, 2008

Abstract

Predicting the native structure of proteins is one of the most challenging problems in computational biology. The goal is to determine the three-dimensional structure from the one-dimensional amino acid sequence. *De novo* prediction algorithms seek to do this by developing a representation of the proteins structure, an energy potential and some optimization algorithm that finds the structure with minimal energy.

Bee Colony Optimization is a new metaheuristic approach to optimization based on the foraging behaviour of bees. The method is a very simple swarm-algorithm that can easily be expanded or be used to prioritize parallel runs of local search methods. We have implemented the Bee Colony Optimization metaheuristic using hill-climbing as local search to solve the protein structure prediction problem. The results show that Bee Colony Optimization generally finds better solutions than simulated annealing in the same amount of time. The quality of the predicted structures are compared with other algorithms using a standard benchmark and two template-free proteins from CASP7.

1 Introduction

Proteins are the primary building blocks in all living organisms. They are made of amino acids bound together by peptide bonds. Depending on the sequence of amino acids, the proteins fold in three dimensions so that the Gibbs free energy is minimized. The shape determines the function of the protein. *Protein structure prediction* (PSP) is the problem of predicting this three-dimensional structure from the amino acid sequence and is considered one of the most important open problems of theoretical molecular biology. The PSP has applications in medicine within areas like drug- and enzyme design [1].

The PSP proves to be a very difficult optimization problem. Solving it exactly is only possible when using very simplified models. Use of heuristics is therefore necessary when using more detailed models and energy functions. However, even in simplified scenarios, many computational problems arise. One of these problems is the belief that free energy landscapes tend to have many local minima [2].

Lately, several optimization heuristics inspired by bee colonies has been proposed. The two main approaches are the evolutionary algorithms and the foraging algorithms. The evolutionary approach was initially proposed by [3] and was based on the mating of bee drones with a queen bee. The foraging approach was proposed simultaneously in [4] and [5] and mimics the foraging behaviour of honey bees searching for and collecting nectar in a flower field. This heuristic, like real honey-bees, performs a wide search for good solutions and has a flexible method for allocating resources to intensify the local searches. This seems like a good strategy in the PSP to avoid getting stuck in the local minima of the energy landscape. Several names have been given to the foraging algorithm but here *Bee Colony Optimization* (BCO) is chosen.

Bahamish et al. [6] previously used the *Bees Algorithm* [4] to find the native state of the 5-residue peptide 'met-enkalphin' (PDB-ID: 1PLW) using a full resolution torsion angle-based representation. In our work, we apply the BCO metaheuristic to the PSP problem using a simplified representation. Good quality solutions, in terms of the RMSD similarity measure, are generated. These decoy solutions can be used as starting solutions for more advanced methods. Since a coarser representation is used, real-sized protein structures can be attacked by the BCO metaheuristic. To our knowledge this is the first time a bee heuristic has been used to predict the structure of actual

proteins. We do not claim to solve the PSP or even compete with state-of-the-art PSP algorithms like Rosetta[7] or I-Tasser [8], however the BCO metaheuristic has appealing properties that we believe makes it suitable for the PSP.

In section 2 the model of PSP and the energy function is defined. Next our adaptation of BCO is described in section 3. Finally experiments are described in section 4 and discussed in section 5.

2 Protein Structure Prediction

The representation of proteins is important since it determines the size and structure of the search-space. In the following section three representations of decreasing complexity are described. The first and most complex is the one used in [6] and the last is the one used in this paper.

2.1 Full resolution representation of proteins

Proteins consist of a chain of amino acids. There are 20 different kinds of amino acids, each represented by a letter. The sequence of amino acids is called the primary structure of the protein. Frequent occuring local structures of amino acids, such as helices and strands, are called secondary structure and the full description of the protein (i.e. 3D coordinates of all atoms) is called the tertiary structure. The protein representation described here is able to represent the tertiary structure of proteins.

All amino acids consist of identical 'backbones' (nitrogen, carbon and carbon) and a side chain denoted R. Bonded to the backbone atoms are two hydrogen atoms and an oxygen atom.

One amino acid (glycine) only contains a single hydrogen atom in the side chain and therefore requires no parameters to represent R. Others have up to 18 atoms in the side chain and can require up to 5 *rotamer* angles (χ_{1-5}) to be fully represented.

The chemical bonds within the backbone fixate the six atoms from (including) C^{α} in one amino acid to (including) C^{α} in the next on a planar rhombus (see Figure 1). The backbone for each amino acid can therefore be represented using two angles – Φ and Ψ.

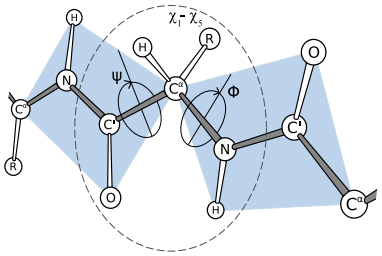


Figure 1: The atoms and side chains of an amino acid (within the dotted line). The backbone is specified by the torsion angles Φ and Ψ, and the side chains by rotamer angles χ_1 to χ_5.

2.2 Simplified C^{α} -trace representation

When trying to determine the overall structure of a protein sometimes the side chains and the atoms of the backbone are disregarded, and only the central carbon atom – C^{α} – of a protein is represented. This leads to the C^{α} -trace representation of proteins illustrated in Figure 2. An amino acid can be represented by two angles, θ and τ.

2.3 Simplified segment representation

The simplified segment representation described here, is the one used in the BCO algorithm. It was first introduced in [10].

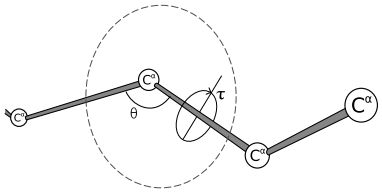


Figure 2: C_α trace of backbone. Each amino acid is here specified by two angles θ and τ. The graphics are generated by Rasmol [9].

Each amino acid of a protein can be classified as belonging to exactly one secondary structure. Here three classes of secondary structures are considered; helix, strand and coil. Helices and strands are distinguished by the unique geometrical layout of the C^α atoms in the tertiary structure (see Figure 3). Strands additionally are characterized by pairing up with strands different places in the protein. Coil is the class of all other shapes that are neither helices nor strands. C^α -atoms of a coil therefore have a large degree of freedom, compared to helices and strand, since there are few geometric constraints on the tertiary structure of a coil.

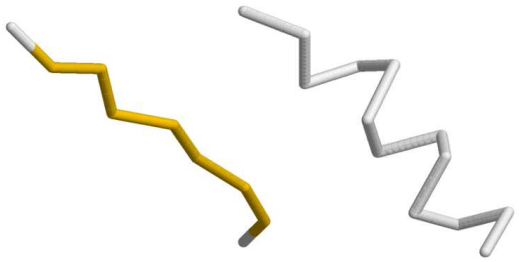


Figure 3: Typical backbone structure for a strand (left) and a helix (right)

A sequence of residues of the same secondary structure class is here called a segment. Segments can be considered as rigid rods that define the overall path of C^α -atoms belonging to the segment. Segments always have a start coordinate and a direction, and for helices and strands their end coordinate can also be determined because of their constrained geometry. A segment is therefore an abstract representation of a sequence of residues and it does not explicitly contain the coordinates of internal C^α -atoms. A segment structure is therefore defined to be the coordinates of all C^α -atoms of a segment. The list of all segment structures is called the complete structure. Figure 4 is an illustration of a complete structure in the simplified segment representation.

The tertiary structure of any protein can be described by a complete structure. However, to discretize and reduce the conformational space of this model, the degree of freedom for segments are reduced. Segments are therefore only allowed to have a discrete amount of predefined directions (d) between the first and last C^α -atoms. Obviously the chance of being able to represent a complete structure similar to the native structure of the protein increases the more directions are allowed. To further discretize the model, the number of possible segment structures allowed by a segment is limited to s. The method used to determine the structures for helix, strand and coils are described in section 2.4.

Ad-hoc experiments show that $d = 73$ uniformly distributed directions acquired by combining the face centered cubic (FCC) lattice, the simple cubic (SC) lattice and the body centered cubic (BCC) lattice is suitable for representing realistic proteins. Experiments also show that allowing $s = 16$ structures seems suitable for BCO.

Given an amino acid sequence with m segments, d possible segment directions and s possible segment structures for each segment, the total number of complete structures, N, allowed by this

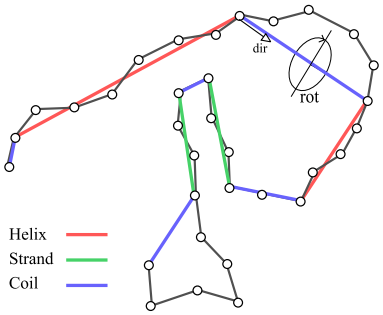


Figure 4: Segment representation of proteins. Each segment can point in 73 directions and the amino acids can assume 16 distinct rotations around the segment-line

model is limited by

$$N < d^m \cdot s^m$$

One might think that this should be $N = d^m \cdot s^m$, but because of rotational and mirror symmetry many complete structures can be disregarded. For instance the first segments direction can be fixed, and more than half the directions of the second segment results in symmetrical structures that can be ignored.

2.4 Segment Structures

In this section it is described how the s allowed segment structures of a given segment are computed. This computation depends on the secondary structure class of the segment.

2.4.1 Helix and Strand Structures

The right-handed helix is the most commonly observed secondary structure in proteins. In helices, the most observed angle pair for an amino acid is $(\theta, \tau) = (91°, 49°)$. Given a helix segment, one segment structure having these angle properties are generated. Then the other $s - 1$ segment structures are generated by rotating the first structure uniformly around the axis going through the first and last C^α -atoms.

Strand structures are constructed in the same way as helices, but with other angle values. For strands, the most observed angle pair is $(\theta, \tau) = (120°, 163)$. The angle values were found after using P-SEA [11] to compute secondary structure of 3080 proteins from PDB Select (25) [12].

2.4.2 Coil Structures

There are no simple geometric constraints that describe coil structures. However, experiments show that short sequences with similar amino acid sequences, so-called homologous sequences, often have similar tertiary structures [13]. Given a coil segment, PDB Select (25) is queried with protein sequences and their known structures and find the $\sqrt{s}$ best fragment matches in terms of amino acid similarity. Each of these structures are rotated uniformly $\sqrt{s}$ times, as for helices and strands, such that a total of s structures are obtained. The fragment database does of course not contain the proteins used in the experiments.

2.5 Energy

Determining an energy function for protein structures that is computationally fast and correlates well to the real native structure of proteins is still an open problem within bioinformatics. Some energy functions are based on quantum mechanical interactions between atoms of the protein, and although the quality of the minimum energy structures is good the computation of the energy usually takes a long time. Other energy functions – pseudo-energy-functions – are based on statistical

analysis of large sets of proteins. These are usually very fast but the quality of the minimal energy structure varies greatly.

A promising pseudo-energy-function described in [10] is based on *Half-Sphere Exposure* (HSE) [14] and *Contact Numbers* (CN). This energy requires very little computation and represents many of the important properties of protein native structures.

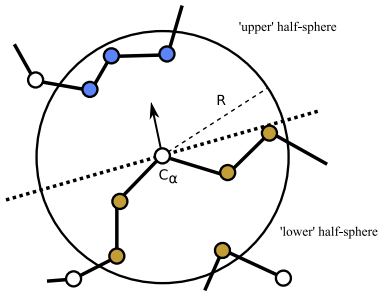


Figure 5: Half-sphere exposure for an amino acid. The up/down pair is $(3, 5)$. The contact number is 8.

For some amino acid the HSE is a pair of integers describing how many amino acids are contained in a half-sphere *above* the amino acid and how many are contained in the half-sphere *below* (See Figure 5). The up vector relative to some amino acid A_i is defined as

$$\overrightarrow{up} = \overrightarrow{A_{i-1}A_i} + \overrightarrow{A_{i+1}A_i}$$

HSE is undefined for the terminals of the protein. CN for an amino acid is the number of amino acids contained in the *entire* exposure sphere.

Let $\mathcal{P}$ denote the conformational space of a protein with n residues $A_1, A_2, ..., A_n$. Let $p \in \mathcal{P}$. The total energy $Q(p)$ is defined as the sum of the residue energy contributions $Q_p(A_i)$, i.e.,

$$Q(p) = \sum_{i=1}^{n} Q_p(A_i)$$

$$Q_p(A_i) = \begin{cases} \Delta CN(A_i)^2 & \text{if } A_i \text{ is the first residue of a segment.} \\ \Delta HD(A_i)^2 + \Delta HU(A_i)^2 & \text{otherwise} \end{cases}$$

where

- $\Delta CN(A_i)$ is the difference between the contact number of the i-th residue A_i in p and the desired (i.e., predicted) contact number of A_i .
- $\Delta HD(A_i)$ is the difference between the down half sphere exposure number of A_i in P and the desired down half sphere exposure number of A_i .
- $\Delta HU(A_i)$ is the similar difference for the up half sphere exposure.

The reason why CN instead of HSE is used for the first residue of a segment is that it was necessary for the Branch and Bound algorithm described in [10, 15]. In order to compare solutions found here with those in [10] the same energy function is preserved.

A radius of the contact sphere around 13Å is known to give a good prediction quality [16] and it seems to capture both local and non-local contacts. The optimal radius has yet to be determined, both in terms of predictability and information content.

Since many amino acids are hydrophobic, globular proteins fold into a very tight spheric conformation. An HSE based energy function is not enough to ensure this behaviour, so the radius of the surrounding sphere – the *radius of gyration* (Rg)– is introduced. Rg can be predicted from the number of residues n of the protein [17]:

$$Rg = 2.2n^{0.38} \tag{1}$$

This prediction is often accurate for globular proteins. Infinite energy is therefore assigned to structures having radius of gyration more than 20% away from the predicted Rg.

A structure is said to be clashing if the distance between two C^{α} atoms is less than 3.5. A clashing structure is also assigned infinite energy.

3 Bee Colony Optimization

In nature, a foraging bee can be said to be in one of three states: A scout bee, a worker bee or an onlooker. Scout bees fly around a flower field at random and when a flowerbed is found they return to the hive and perform a waggle dance. The dance indicates the estimated amount of nectar, direction and distance to the flowerbed. Onlooker-bees present in the hive watch different waggle dances, choose one and fly to the selected flowerbeds to collect nectar. Worker bees act like scout bees except that when they have performed the waggle dance they return to their old flowerbed to retrieve more nectar. A bee usually chooses to become a worker bee when the chosen flowerbed has a very high concentration of nectar.

In our adaptation of the BCO metaheuristic, each bee corresponds to a specific solution, and the nectar amount corresponds to an objective value in the energy landscape. Sending out scout bees corresponds to finding a random feasible solution and sending out onlookers corresponds to finding a neighbourhood solution. The onlookers choose sites for neighbourhood search based on the objective value of scouts and workers in previous iterations. This method is largely the *Bees Algorithm* proposed in [4]. In a non-changing solution space a solution does not deplete in the same way a real life flowerbed depletes of nectar. Exhaustion is therefore forced when a solution cannot be improved. This idea is somewhat similar to the idea of pruning parts of the search space as described in [18]. The process of exhausting a local search is proposed as part of the *Artificial Bee Colony* algorithm described in [5]. Our adaptation of the BCO metaheuristic is a synthesis of these approaches.

Algorithm 1: Bee-Colony-Optimization

input : S, W, O, $Exhaust$, OS, NS, SS
output: The best solution
1 Initialize population with $S + W$ random solutions using SS
2 Evaluate cost of the population
3 **while** *Stopping criterion is not met* **do**
4 Recruit O onlooker-bees and assign each to a member of the population according to OS
5 **for** *Each onlooker assigned to some member n of the population* **do**
6 Perform an iteration of the local search algorithm NS on n
7 **end**
8 Evaluate cost of the O neighbourhood solutions
9 If a member of the population has not improved for $Exhaust$ iterations, save the solution and replace it with a random solution
10 Find S random solutions using SS and replace the S members of the population that has the worst costs
11 **end**
12 **return** The best solution – either from the population or from the saved solutions

Here S, W and O is the amount of scout, worker and onlooker bees respectively. OS is the strategy for assigning onlookers, NS is the neighbourhood strategy for performing a local search and SS is the method for generating a random solution.

3.1 Bee Colony Optimization applied to PSP

The above pseudocode can be used for any optimization problem where OS, NS and SS can be defined. So to utilize BCO for PSP these three methods have to be defined.

3.1.1 Scout Search Strategy (SS)

To find a random feasible solution a depth first search is used to determine the direction d_i and structure s_i of each segment i. At each level in the depth first search a random ordering of direction and structure is tried so the same solution is not generated every time.

3.1.2 Onlooker Choosing Strategy (OS)

The onlookers choose a member n of the population based on the members energy function. If the member has a low energy then it is more likely to be chosen. This is implemented by letting each onlooker choose the member with highest estimated fitness:

$$fitness_n = \textsc{RandomNumberBetween}(0,1) \cdot \frac{1}{\textsc{Energy}(n)}$$

3.1.3 Onlookers Local Search (NS)

Any local search could be utilized as neighbourhood strategy so a simple hill-climbing strategy is chosen. Each iteration finds a random neighbour to the existing solution and replaces the existing solution if the energy is improved.

4 Experiments and results

The tertiary structures of 8 proteins is predicted. 5 proteins have previously been used for benchmarks in the literature [19, 10, 20]. The remaining 2 are somewhat bigger and were chosen from the targets of CASP7. We have intentionally chosen a pair that proved to be hard to predict by CASP7 participants. Most succesful CASP7 methods where homology-based. Since our algorithm is not using homology modelling, it should be compared with other methods by applying it to proteins with no good templates in PDB. The tertiary structures of the proteins are known and the quality of the results can therefore be evaluated using GDT [21].

The input to BCO is a secondary structure assignment, HSE-vector and the radius of gyration. For each protein these values are obtained using prediction tools. Based on the amino acid sequence, the secondary structure is predicted using PSIPRED [22] and HSE-vectors using LAKI [16] and HSEpred [23]. For better comparison of energy levels the HSE predictions from in [10], which were done using LAKI [16], were used. For the CASP proteins the newer and more accurate HSE prediction server HSEpred [23] were used. Note that PSIPRED, LAKI and HSEpred are neural networks trained on a selection of proteins from PDB. The 8 benchmark proteins used here also exist in PDB, so there is a slight chance that the training sets for PSIPRED, LAKI and HSEpred contain some of these proteins. However, the prediction quality of the 8 benchmark proteins is close to what should be expected. We therefore do not consider it to be a problem that the benchmark proteins exist in PDB. The radius of gyration is predicted using Equation 1.

For comparison and evaluation of the model and prediction quality, all experiments are also done using the exact secondary structures and exact HSE-vectors obtained from the native structures of the proteins. These structures cannot be considered solved *de novo*. All computations were performed on a 3.4GHz Intel Xeon with 2GB RAM.

By ad-hoc experiments an appropriate configuration for BCO was determined. $S = 10$ scouts, $W = 10$ workers and $O = 100$ onlookers were used, $Exhaust$ was set to 5 and the algorithm was set to stop when it had run for 48 hours. Since the purpose of the BCO algorithm is to find many good decoys the best 1000 unique solutions are registered.

To evaluate BCO as an optimization metaheuristic it is compared to simulated annealing (SA) by running 10 parallel instances of SA in 48 hours in total on every protein. The SA algorithm also stores 1000 unique registered decoy solutions with minimal energy. A solution is registered if it is encountered at some point in one of the 10 searches. The results from EBBA [10] are also presented here for comparison. Even though the representation in [10] is the same as here, some parameters diverge, namely the amount of segment directions d (12 in [10], 73 for BCO) and rotations r (2 to

8 in [10], 16 for BCO). Also the tolerated divergence from the predicted radius of gyration differs (5% in [10], 20% here).

Table 1 summarizes the results of the runs from BCO, SA, EBBA and CASP7. p^* is the protein structure encountered during a search for which the energy function $Q(p)$ is lowest. For BCO, SA and EBBA this energy function is identical. $p^\dagger$ is the protein structure – among the 1000 saved decoys – for which GDT(p) is highest.

PDB	Size	SS &	BCO				SA			EBBA		CASP7
id		energy	$Q(p^*)$	RMSD(p^*)	GDT(p^*)	GDT($p^\dagger$)	$Q(p^*)$	GDT(p^*)	GDT($p^\dagger$)	$Q(p^*)$	RMSD(p^*)	GDT(p^*)
1FC2	43	pred.			%	%	3.76	47.67%	58.14%	5.26	8.4	-
		exact	1.94	1.65	83.72%	84.30%	2.62	66.28%	79.07%	4.34	6.6	-
1ENH	54	pred.	4.67	6.99	40.28%	50.93%	4.91	40.28%	50.46%	5.70	10.2	-
		exact	2.91	2.28	71.30%	73.61%	3.56	54.63%	67.13%	4.36	3.5	-
2GB1	56	pred.	5.41	8.86	30.80%	41.96%	5.50	29.46%	42.41%	6.22	7.8	-
		exact	5.52	9.18	31.70%	47.32%	5.03	27.68%	49.11%	4.22	4.3	-
2CRO	65	pred.			%	%	4.44	35.38%	39.62%	5.89	9.4	-
		exact	6.10	7.61	35.38%	47.69%	6.13	41.54%	51.54%	6.49	9.2	-
1CTF	68	pred.	5.43	9.01	36.03%	38.97%	5.74	33.46%	37.87%	5.84	11.3	-
		exact	5.67	7.50	38.60%	44.12%	5.83	25.74%	49.63%	7.19	11.0	-
4ICB	76	pred.	4.77	9.02	32.57%	38.49%	5.32	29.28%	44.08%	6.79	6.4	-
		exact	5.38	10.38	28.29%	44.41%	5.45	28.95%	42.11%	6.18	7.4	-
2HG6	106	pred.	6.14	16.26	14.89%	22.17%	6.61	17.69%	27.59%	-	-	30.34%
		exact	4.70	14.49	20.05%	24.29%	5.19	19.81%	30.19%	-	-	-
2J6A	136	pred.	6.79	14.34	14.34%	19.30%	6.79	17.10%	20.59%	-	-	27.78%
		exact	6.20	16.31	18.38%	22.98%	7.25	17.46%	21.88%	-	-	-

Table 1: Results from Bee Colony Optimization (BCO), Simulated Annealing (SA), Efficient Branch and Bound Algorithm (EBBA) and CASP7. At CASP7 the proteins 2HG6 and 2J6A had target numbers T0314 and T0319 respectively. The GDT similarity measure is calculated as the largest set of C^α positions within a defined distance cutoff of their position in the target structure. Large values of GDT are therefore preferrable whereas low values of RMSD are preferred. Since structure prediction seeks to minimize the energy, $Q(p)$ should be as low as possible. p^* is the structure, encountered during search, with lowest energy and $p^\dagger$ is the one with highest GDT. The same combinatorial protein representation is used for BCO and SA. An identical representation is used for EBBA but some parameters diverge. Note that not all runs were completed when printing this paper draft.

5 Discussion and conclusion

The results of BCO, SA compared to those achieved at CASP7 is shown for the proteins 2HG6 and 2J6A in Table 1. It can be seen that the HSE energy function does not identify the best structure since GDT(p^*) is relatively low for BCO and SA. Assuming, however, that a more advanced energy function can identify $p^\dagger$, this would rank the structures obtained by BCO as 17^{th} of 132 for 2J6A and 30^{th} out of 132 for 2HG6 at CASP7.

When comparing BCO to SA, the focus should be on the values of $Q(p)$ since both algorithms optimize the energy. For all the problems, except 2GB1 exact, BCO achieves a lower value of $Q(p)$ which indicates that BCO is superior to SA on this type of problems. It is worth noting that SA usually is the algorithm of choice when choosing a metaheuristic for PSP.

EBBA is an exact algorithm that guarantees to find the structure with minimal energy, yet $Q(p)$ is higher than the energy BCO finds because more segment directions and rotations are allowed in BCO.

When looking at the results for 1FC2 (exact) and 1ENH (exact) it is clear that they differ from the other rows. The lowest energy observed is less than 3 for both runs wich is considerably lower than for the other runs. It is remarkable that the corresponding very low energy structures are native-like. This supports the hypothesis that HSE, secondary structure and Rg contains enough information to identify the native structure of the protein. There are two possible reasons for why we do not find these very low energy structures for the other proteins. One reason is that native-like structures cannot be represented accurately enough in our model. The other possibility is that our search algorithms require much more time. This is a subject for further investigation.

References

[1] T.S Mayuko, T. Daisuke, C. Chieko, T. Hirokazu, and U. Hideaki. Protein structure prediction in structure based drug design. *Current Medicinal Chemistry*, 11(5):551–558, 2004.

[2] Z. Li and H. A. Scheraga. Monte Carlo-minimization approach to the multiple-minima problem in protein folding. *Proc. Natl. Acad. Sci.*, 84(19):6611–6615, October 1987.

[3] H. A. Abbass. MBO: Marriage in honey bees optimization - A haplometrosis polygynous swarming approach. In *Proceedings of the 2001 Congress on Evolutionary Computation CEC2001*, pages 207–214, COEX, World Trade Center, 159 Samseong-dong, Gangnam-gu, Seoul, Korea, 27-30 2001. IEEE Press.

[4] D.T. Pham, A. Ghanbarzadeh, E. Koc, S. Otri, S. Rahim, and M. Zaidi. The bees algorithm. Technical report, Manufacturing Engineering Centre, Cardiff University, UK, 2005.

[5] D. Karaboga. An idea based on honey bee swarm for numerical optimization technical report-TR06. Technical report, Erciyes University, Engineering Faculty, Computer Engineering Department, November 2005.

[6] H. A. A. Bahamish, R. Abdullah, and R. A. Salam. Protein conformational search using bees algorithm. In *Asia International Conference on Modelling and Simulation*, pages 911–916. IEEE Computer Society, 2008.

[7] C. A. Rohl, C. E. Strauss, K. M. Misura, and D. Baker. Protein structure prediction using Rosetta. *Methods Enzymol*, 383:66–93, 2004.

[8] Yang Zhang. I-tasser server for protein 3d structure prediction. *BMC Bioinformatics*, 9:40+, January 2008.

[9] R. Sayle. RasMol v2.5 a molecular visualisation program, biomolecular structure glaxo research and development greenford, 1994. Roger Sayle and Biomolecular Structure.

[10] M. Paluszewski and P. Winter. Protein decoy generation using branch and bound with efficient bounding. *Lecture Notes in Bioinformatics, WABI, (to appear)*, 2008.

[11] G. Labesse, N. Colloc'h, J. Pothier, and J.-P. Mornon. P-SEA: A new efficient assignment of secondary structure from calpha trace of proteins. *Bioinformatics*, 13:291–295, 1997.

[12] U. Hobohm and C. Sander. Enlarged representative set of protein structures. *Protein Sci*, 3(3):522–524, 1994.

[13] C. Chothia and A. M. Lesk. The relation between the divergence of sequence and structure in proteins. *The EMBO Journal*, 5:823–826, 1986.

[14] T. Hamelryck. An amino acid has two sides: A new 2D measure provides a different view of solvent exposure. *J Proteins: Structure, Function, and Bioinformatics*, 59(1):38–48, 2005.

[15] M. Paluszewski and P. Winter. EBBA: Efficient branch and bound algorithm for protein decoy generation. *Technical report. Department of Computer Science, Univ. of Copenhagen*, 08(08), 2008.

[16] B. Vilhjalmsson and T. Hamelryck. Predicting a new type of solvent exposure. ECCB, Computational Biology Madrid 05, P-C35, Poster, 2005.

[17] J. Skolnick, A. Kolinski, and A. R. Ortiz. MONSSTER: A method for folding globular proteins with a small number of distance restraints. *J. Mol. Biol.*, 265:217–241, 1997.

[18] M. Paluszewski, T. Hamelryck, and P. Winter. Reconstructing protein structure from solvent exposure using tabu search. *Algorithms For Molecular Biology (ALMOB)*, 2006.

[19] T. Hamelryck, J. T. Kent, and A. Krogh. Sampling realistic protein conformations using local structural bias. *PLOS Computational Biology*, 2, 2006.

[20] K. T. Simons, C. Kooperberg, E. Huang, and D. Baker. Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions. *J Mol Biol*, 268(1):209–25, 1997.

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

[22] L. J. McGuffin, K. Bryson, and D. T. Jones. The PSIPRED protein structure prediction server. *Bioinformatics*, 16:404–405, 2000.

[23] J. Song, K. Takemoto, and T. Akutsu. HSEpred: Predict half-sphere exposure from protein sequences. *Bioinformatics*, 24:1489–1497, 2008.

Appendix A

Paper: EBBA: Efficient Branch and Bound Algorithm for Protein Decoy Generation

M. Paluszewski and P. Winter. EBBA: Efficient Branch and Bound Algorithm for Protein Decoy Generation. *Technical Report, Department of Computer Science, Univ. of Copenhagen*, 08(08), 2008.

EBBA: Efficient Branch and Bound Algorithm for Protein Decoy Generation

Martin Paluszewski and Pawel Winter

Department of Computer Science, University of Copenhagen, Universitetsparken 1, 2100 Copenhagen, Denmark

Abstract. We are faced with three major challenges when dealing with the problem of de novo protein structure prediction. One is to determine a suitable energy function having a global minimum near the native structure of the protein. The second challenge is to sample the conformational space such that some of the sampled decoys are near the native structure. The third challenge is to identify the native-like structures among the sampled decoys. Here we present a novel method for decoy generation and therefore attack the second of these challenges.
We propose a new discrete protein structure model (using a modified face-centered cubic lattice). A novel branch and bound algorithm for finding global minimum structures in this model is suggested. The objective energy function is very simple as it depends on the predicted half-sphere exposure numbers of C_α-atoms. Bounding and branching also exploit predicted secondary structures and expected radius of gyration. The algorithm is fast and is able to generate the decoy set in less than 48 hours on all proteins tested.
Despite the simplicity of the model and the energy function, many of the lowest energy structures, using exact measures, are near the native structures (in terms of RMSD). As expected, when using predicted measures, the fraction of good decoys decreases, but in all cases tested, we obtained structures within 6 Å RMSD in a set of low-energy decoys. To the best of our knowledge, this is the first *de novo* branch and bound algorithm for protein decoy generation that only depends on such one-dimensional predictable measures. Another important advantage of the branch and bound approach is that the algorithm searches through the entire conformational space. Contrary to search heuristics, like Monte Carlo simulation or tabu search, the problem of escaping local minima is indirectly solved by the branch and bound algorithm when good lower bounds can be obtained.

1 Background

Here we present our approach for protein decoy generation using the branch and bound paradigm. A shorter version of this paper appeared in [1]. The contact number (CN) is a very simple solvent exposure measure that only depends on the positions of C_α-atoms. Given a fixed backbone structure, the CN of a residue A_i is the number of other C_α-atoms in a sphere of radius r centered at the C_α-atom

of A_i. The CN of all residues of a given structure is called the *CN-vector*. A more information rich measure is called the *half-sphere-exposure* (HSE) measure [2]. Here, the sphere is divided into an upper and a lower hemisphere as illustrated in Figure 1. The up and down numbers of a residue therefore refer to the number of other C_α-atoms in the upper and lower hemispheres respectively. For a given fixed structure, the up and down numbers for all residues is called the *HSE-vector*. CN- and HSE-vectors therefore only depend on the radius of the spheres and the coordinates of C_α-atoms, which is very convenient when using simplified models.

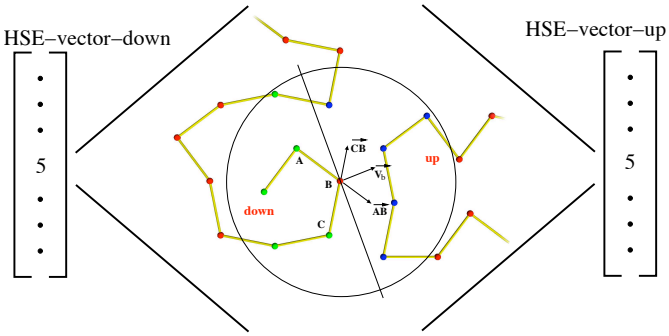


Fig. 1. Given the positions of 3 consecutive C_α-atoms (A, B, C), the approximate side-chain direction $\vec{V}_b$ can be computed as the sum of $\vec{AB}$ and $\vec{CB}$. The plane perpendicular to $\vec{V}_b$ cuts the sphere centered at B in an upper and a lower hemisphere.

Recently it was shown that it is possible to approximately reconstruct small protein structures from CN-vectors or HSE-vectors only [3]. These results showed that HSE-optimized structures in general have better coordinate RMSD with the native structure and more accurate orientations of the side-chains compared to CN-optimized structures. This is very interesting in regards to *de novo* protein structure prediction, because CN- and HSE-vectors can be predicted with reasonable accuracy [4, 5]. To use these results for *de novo* structure prediction, one could therefore first predict the HSE-vector from the amino acid sequence and then reconstruct the protein backbone from this vector. However, the results in [3] were only based on small proteins with up to 35 amino acids and it was conjectured that the reconstruction of larger proteins would require more information than what is contained in an HSE-vector [3]. Another difficulty is that HSE-based energy functions appear to have many local minima in the conformational space. This is often a problem for search heuristics like Monte Carlo simulation or tabu search, since they get trapped in these minima and must spend much time escaping them.

The problem of reconstructing protein structure from vectors of one-dimensional structural information has also been studied by Kinjo et al. [6]. They used exact vectors of secondary structure, CN and *residuewise contact order* (RWCO) together with refinement using the AMBER force field to reconstruct native-like structures. Their results indicated that secondary structure information and CN without the use of RWCO is *not* enough to reconstruct native-like structures. Unfortunately, RWCO is difficult to predict compared to CN, HSE and secondary structure [6] and it would therefore be difficult to use their method directly for de novo structure prediction.

Here we attack these problems by adding more predicted information to our model and use a thorough branch and bound algorithm for finding minimum energy structures. By adding more predicted information we expect to increase the probability of the energy function to have global minimum near the native structure. Furthermore, using a branch and bound approach we are able to implicitly search the whole conformational space and therefore avoid getting trapped in local minima. Besides using HSE vectors, we also use *secondary structure* (SS) and *radius of gyration* (Rg). These three measures, (HSE, SS and Rg), can all be predicted from the amino acid sequence only [4, 7, 8], and can therefore be used for *de novo* protein structure prediction. The energy function is simple, and we show how a good lower bound of the energy for a subset of the conformational space can be computed in polynomial time. This lower bound enables the branch and bound algorithm to bound large conformational subspaces and to find global minimum energy structures in a reasonable amount of time. Throughout the text our branch and bound algorithm is referred to as EBBA (Efficient Branch and Bound Algorithm).

The idea of using secondary structure elements in a discrete model has been suggested by others, i.e., Fain et al. [9] and Levitt et al. [10]. However, their models have a relatively small conformational space and it is therefore possible to completely enumerate all structures allowed by the model. Branch and bound algorithms and other algorithms for determining global minimum structures have been used for protein structure prediction earlier. Some of these algorithms work on very simplified models like the HP-lattice model [11, 12]. Even though these algorithms can solve most problems to optimality, the global minimum structures are often very far from the native structure. Another branch and bound algorithms, called αBB[13] uses more detailed potential energy functions which depend on several physical terms. In [13], the αBB is shown to be successful on small molecules. In [14], the αBB was improved and was used for prediction of real protein structures. Dal Palu et al.[15] use a constraint logic programming approach for protein structure prediction. They also use secondary structure segments in a simplified model. However, in their model, all C_α-atoms must be placed in a lattice (FCC). This differs from our approach, where we only demand lattice directions of the secondary structure segments. Dal Palu et al. use a standard solver (SICStus Prolog) which makes use of standard bounding techniques, while we have developed a much more efficient bounding algorithm specialized for this particular problem. Furthermore, the results published in [15, 14] are not

true *de novo* - the secondary structures are all derived from the native structure of the proteins. On the contrary, the results presented here are true *de novo*. All parts of the energy function are predicted from amino acid sequences only. EBBA is, to our knowledge, the first *de novo* branch and bound algorithm that only use one-dimensional predictable measures.

We use 6 benchmark proteins for evaluating EBBA. Our results show that EBBA is able to find global minimum energy structures for most of these proteins in less than 48 hours. We have evalutated EBBA using both exact values and predicted values to estimate the importance of prediction quality. The results show that predicted structures having global minimum energy are *not always* native-like, however among the 10.000 lowest energy structures we typically find many good decoys (less than 6 Å RMSD). Our algorithm therefore reduces the protein structure prediction problem to the problem of identifying a near-native structure in a relatively small set of decoys.

2 Methods

Each amino acid of a protein can be classified as belonging to a unique secondary structure. Here we consider three classes of secondary structures; helix, sheet and coil. Helices and sheets are distinguished by the unique geometric shape of the C_α atoms in their tertiary structure. Coil is the class of all other shapes that are neither helices nor sheets. C_α-atoms of a coil therefore have a large degree of freedom, compared to helices and sheets, since there are few geometric constraints on the tertiary structure of a coil.

A sequence of residues of the same secondary structure class is called a *segment*. Segments can be considered as rigid rods that describe the overall path of C_α-atoms belonging to the segment. Segments always have a start coordinate and a direction, and for helices and sheets their end coordinate can also be determined because of their constrained geometry. A segment is therefore an abstract representation of a sequence of residues and it does not explicitly contain the coordinates of internal C_α-atoms. We therefore define a *segment structure* to be the coordinates of all C_α-atoms of a segment. Note that a segment in principle allows for infinitely many different segment structures even though they are restricted to be of a specific secondary structure class. However, this model is discrete and therefore only a finite representative set of segment structures are generated. This is described in detail in Section *Segment structures*.

Any tertiary structure of a protein can be described in these terms; a list of segments and a segment structure for each segment. We call such a list of segments a *super structure* and a super structure with a segment structure for each segment is called a *complete structure*.

The tertiary structure of any protein can always be described by a complete structure. However, to discretize and reduce the conformational space of this model, we reduce the degree of freedom for segments. Segments are therefore only allowed to have a discrete set of predefined directions between the first and last C_α-atoms. Obviously, the more directions allowed, the more super structures can

be described by the model. This of course also increases the chance of describing a super structure similar to the native structure. Therefore, there is a trade-off between the number of directions allowed and the computational feasibility of the model. Ad-hoc experiments show that the 12 uniformly distributed directions acquired from the *face-centered cubic* (FCC) lattice is a good tradeoff (see the results section for further discussion). The direction of a segment therefore has one of the following 12 direction vectors: [1,1,0], [1,0,1], [1,-1,0], [1,0,-1], [-1,1,0], [-1,0,1], [-1,-1,0], [-1,0,-1], [0,1,1], [0,1,-1], [0,-1,1], [0,-1,-1]. Figure 2 shows an example of a super structure and a corresponding complete structure.

To further discretize the model, we set an upper limit (u) on the number of possible segment structures allowed by a segment. Given an amino acid sequence with m segments and u possible segment structures for each segment, the total number of complete structures, N, allowed by this model is

$$N = 4 \times 11^{m-2} \times u^m \tag{1}$$

One might think that this should be $N = 12^m \times u^m$ (a segment has 12 possible directions and u possible segment structures), but because of rotational symmetry of the energy function, many complete structures can be disregarded and therefore the first segment direction can be fixed. Also, the angle between two FCC vectors is $0°$, $60°$, $90°$ or $120°$. Therefore, only 4 directions of the second segment need to be considered. The factors $(4 \times 11^{m-2})$ therefore describe the possible directions of segments in the super structure. Note that a segment only has 11 (not 12) possible directions, since a segment is not allowed to clash with the previous segment.

2.1 Segment Structures

Here we describe how the allowed segment structures of a given segment are computed. This computation depends on the secondary structure class of the segment.

Helix and Sheet Structures The right-handed helix is the most commonly observed secondary structure in proteins. In helices, the most observed angle between three consecutive C_α-atoms is $\phi \simeq 91°$ and the most observed dihedral angle of four consecutive C_α-atoms is $\tau \simeq 49°$. Given a helix segment, we generate one segment structure having these angle properties. Then the other $u - 1$ segment structures are generated by rotating the first structure uniformly around the axis going through the first and last C_α-atoms (Figure 3).

Sheet structures are constructed in the same way as helices, but with other angle values. For sheets, the most observed angle between three consecutive C_α-atoms is $\phi \simeq 120°$ and the dihedral angle $\tau \simeq 163°$. The angle values were found by using P-SEA [16] to compute secondary structure of 3080 proteins from PDB Select (25) [17].

6

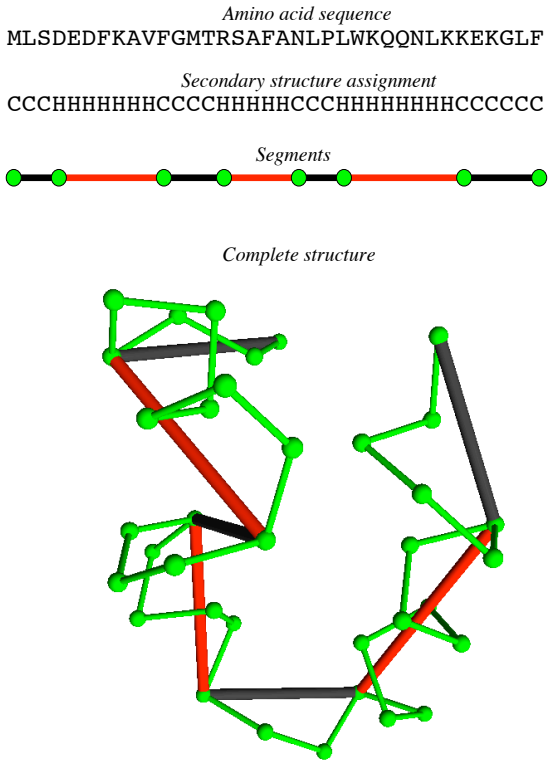


Fig. 2. The Figure shows an example of how an amino acid sequence (from Villin headpiece) can be described as a list of segments based on the secondary structure (H: helix, C: coil). The Figure also shows an example of a super structure and a corresponding complete structure (coordinates of internal C_α-atoms).

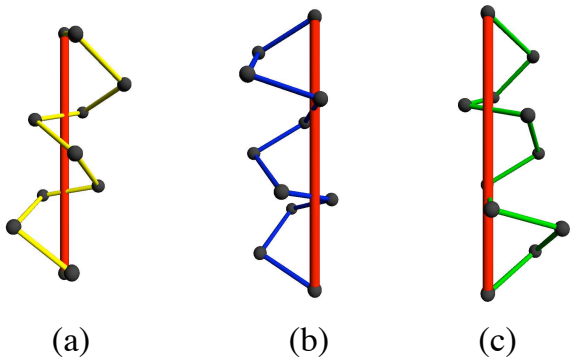


Fig. 3. (a) The first helix with angles $\phi \simeq 91°$ and $\tau \simeq 49°$. (b) and (c) Two other helices are generated (when $u = 3$) by uniformly rotating the first helix around the axis of the segment.

Coil Structures There are no simple geometric constraints that describe coil structures. However, experiments show that short sequences with similar amino acid sequences, so-called homologous sequences, often have similar tertiary structures [18]. Given a coil segment, we therefore query PDB Select (25) with protein sequences and their known structures and find the $\sqrt{u}$ best fragment matches in terms of amino acid similarity. Each of these structures is also rotated uniformly $\sqrt{u}$ times as for helices and sheets such that a total of u structures are obtained. The fragment database does of course not contain the proteins used in the experiments.

2.2 Energy

The structures allowed by the model always have the desired secondary structure (from a prediction), however the HSE-vector and radius of gyration of the structures varies. Therefore, we want to identify those structures having correct radius of gyration and HSE-vectors similar to the predicted HSE-vectors. The radius of gyration can be predicted from the number of residues n of the protein [8]:

$$R_g = 2.2n^{0.38} \tag{2}$$

This prediction is often accurate for globular proteins. We therefore assign infinite energy to structures having radius of gyration more than 5% away from the predicted R_g. We assign infinite energy to structures if their subchain of amino acids from the first amino acid to the l'th ($l < n$) amino acid is more than 5% away from the predicted R_g.

A structure is said to be *clashing* if the distance between two C_α-atoms is less than 3.5 Å. We also assign infinite energy to clashing structures.

Let $\mathcal{P}$ denote the conformational space of a protein with n residues $A_1, A_2, ..., A_n$. Let $P \in \mathcal{P}$. The total energy $Q(P)$ of P is defined as the sum of the residue energy contributions $Q_P(A_i)$, i.e.,

$$Q(P) = \sum_{i=1}^{n} Q_P(A_i) \tag{3}$$

with

$$Q_P(A_i) = \begin{cases} \Delta CN(A_i)^2 & \text{if } A_i \text{ is the first residue of a segment.} \\ \Delta HD(A_i)^2 + \Delta HU(A_i)^2 & \text{otherwise.} \end{cases} \tag{4}$$

where

- $\Delta CN(A_i)$ is the difference between the contact number of the i-th residue A_i in P and the desired (i.e., predicted) contact number of A_i.
- $\Delta HD(A_i)$ is the difference between the down half sphere exposure number of A_i in P and the desired down half sphere exposure number of A_i.
- $\Delta HU(A_i)$ is the similar difference for the up half sphere exposure.

The reason why CN instead of HSE is used for the first residue of a segment is that the HSE value depends on the position of the two neighbour residues as illustrated in Figure 1. For all residues of a segment structure except the first residue, the neighbour positions are always fixed and the upper and lower hemispheres can be computed. In the branch and bound algorithm we want to evaluate the energy of structures where not all segment structures are fixed which is described in detail in the next section. Instead of using HSE for these residues, we use CN which ultimately gives tighter bounds.

The radius of the contact sphere is set to 15 Å. This is known to give a good prediction quality [4] and it seems to capture both local and non-local contacts. The optimal radius has yet to be determined, both in terms of predictability and information content.

2.3 Branch and Bound

Searching for a structure with minimum global energy can be done by evaluating all structures allowed by the model. However, the number of allowed complete structures grows exponentially in terms of the number of segments m and the number of segment structures u (Equation 1). An explicit evaluation of all allowed structures is therefore only feasible for proteins with very few segments and segment structures. A standard approach for overcoming such combinatorial explosion is to use the branch and bound technique [19].

Branching The root of the branch and bound tree represents all complete structures allowed by the model. This is done by only fixing the direction of the first segment. Every other node s represents a smaller subset of complete

structures $\mathcal{P}_s$ than its parent. This is done by either fixing a segment direction or by fixing a segment structure. Therefore, when branching on a node, either 11 children with fixed segment directions are created or u children with fixed segment structures are created. A node at level $2 \times m$ has all segment directions and segment structures fixed and therefore represents a complete super structure. Nodes at level $2 \times m$ cannot be branched on further and are called leaves.

We branch the directions of segments in the order they occur in the protein. Experiments show that the total running time of the algorithm depends much on the order of how the segment directions and segment structures are fixed. The best performance is when the segment directions are fixed as early as possible and the segment structures are fixed as late as possible. The ideal case would therefore be to fix the directions in the first m levels and the segment structures in the next m levels. However, if a protein contains coil segments, it is not possible to fix all segment directions in the first m levels. This is because the end point of a coil segment depends on which coil structure is eventually chosen from the fragment database. Note that this is only a problem for coils, since all helix and sheet structures of a segment share the same end point once the start point and direction are fixed. An example of a branch and bound tree is shown in Figure 4. In the first two levels, the helix and coil segment directions are fixed. In the third level, the structure of the coil segment is fixed. This decision cannot be postponed, because the positions of the following segments depend on the chosen coil structure. At level 4 the direction of the last helix is fixed and at levels 5 and 6 the segment structures of the helices are fixed. In level 6 all directions and segment structures are fixed and the leaves therefore represent complete structures.

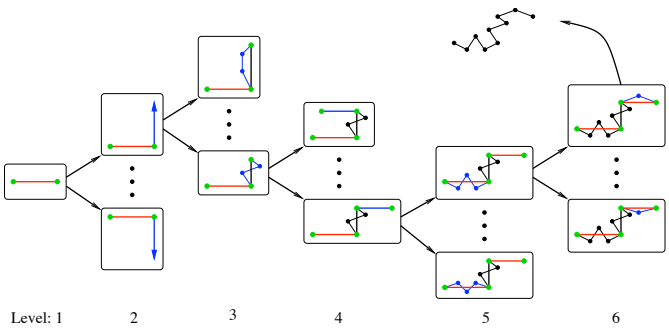


Fig. 4. The super structure consists of three segments: *helix, coil, helix*. For simplicity, in each level, only two nodes are shown and only one node is branched on.

Bounding In theory, one could simply construct the full tree, evaluate the energy function on all leaves and return the lowest energy structure. Unfortunately, because of the exponential number of leaves, this approach is computationally infeasible. Instead, we describe here a method for computing a lower bound of a non-leaf node. A lower bound is a value that is less than, or equal to the lowest energy of any leaf in the subtree of the node. Such a value can be used to disregard, or *bound*, the subtree of a node if the lower bound is larger than some observed energy (*an upper bound*). An upper bound of the energy can be found using some advanced heuristic or a simple depth first search as described in section *Searching*. Here we present a reasonable tight lower bound that can be computed fast. The use of this lower bound makes it possible to solve large problems as described in the results section.

Let $\mathcal{P}_S$ denote the subset of the conformational space $\mathcal{P}$ at any node of the branch and bound tree where some segments have fixed directions while others might have fixed segment structures (i.e., fixed coordinates of all C_α-atoms) as explained in the description of the branching strategy above. We are looking for a lower bound for $\min_{P \in \mathcal{P}_S}\{Q(P)\}$.

Consider the j-th segment S_j, $1 \leq j \leq m$, where m is the number of segments. Let

$$Q_P(S_j) = \sum_{A_i \in S_j} Q_P(A_i)$$

where $Q_P(A_i)$ is defined in Equation 4. Then the energy of a structure can be written as

$$Q(P) = \sum_{1 \leq j \leq m} Q_P(S_j)$$

Suppose that a lower bound for $\min_{P \in \mathcal{P}_S}\{Q_P(S_j)\}$ can be determined. Summing up these lower bounds for all m segments will therefore yield a lower bound for the energy of all conformations in $\mathcal{P}_S$. To compute such a lower bound for a segment S_j, the following problem is solved for all segment structures of S_j. For simplicity we only describe how a lower bound using CN vectors can be computed, however it is straightforward to use a similar approach for HSE vectors.

Given a segment structure for S_j, we determine for each of it's C_α-atom all possible values of CN when the super structure is fixed. This problem can clearly be solved in exponential time by complete enumeration (see Figure 5). However, using the following dynamic programming approach, this problem can be solved in polynomial time. The input to the dynamic programming algorithm is the table constructed as described in Figure 5(c). This table is in the following called $c_{a,b}$.

Let $c_{a,b}(i, r)$ where $(1 \leq i \leq m)$ and $(1 \leq r \leq u)$ be the number of contacts of residue a in segment b contributed by residues in segment i having segment structure r. Let (i, j) be an entry in the dynamic programming table and let $q_{a,b}(i, j) \in \{0, 1\}$ represent whether or not residue a in segment b can have a total of j contacts contributed by residues in segments S_l, $(l < i)$. Then the

recursive equation of the dynamic programming algorithm is:

$$q_{a,b}(i,j) = \begin{cases} 1 \text{ if } i = 1 \text{ and } c_{a,b}(1,r) = j \text{ for some r} \\ 1 \text{ if } i > 1 \text{ and } q(i-1,k) = 1 \text{ and } c_{a,b}(i,r) = j - k \text{ for some r} \\ 0 \text{ otherwise} \end{cases} \tag{5}$$

Each row can be computed in $\mathcal{O}(n \times u)$ time using the values from the previous row, so the total running time of the algorithm is $\mathcal{O}(m \times n \times u)$. The last row in the table represents all possible contact numbers for residue a in segment b. The last row can therefore easily be used to find the minimum difference between the desired CN and one of the possible CNs. The dynamic programming problem is solved for all residues of the segment and the sum of the minimum differences for each residue is the lower bound of the segment energy.

In the above discussion, it was assumed that all C_α-atoms in S_j have their coordinates fixed in $\mathcal{P}_S$. Lower bounds can also be computed if only the segment structure has not been fixed. The above lower bound computation is then merely repeated for each of the u possible segment structures, and the smallest one is selected as the overall lower bound of the segment.

Lower bounds can also be computed for nodes where a number of the last segment directions have not yet been fixed. Here, the input to the dynamic programming algorithm is only the first fixed segments. Then, the CN row for the last fixed segment is augmented by checking whether each C_α-atom on the free segments can possibly be in contact with the C_α-atom in question.

We also bound structures where two succeding segment structures have unlikely angle properties. Figure 7 shows a plot of (θ, τ) pairs from proteins in PDB. The regular angle between 3 consecutive C_α positions is θ and τ is the dihedral angle between 4 consecute C_α positions as illustrated in Figure 8. The plot shows that some regions in the (θ, τ)-plane are much more likely than others. We have marked what we think is a reasonable seperation between likely and unlikely points. Therefore structures with one ore more (θ, τ) points in the unlikely region are bounded.

2.4 Searching

Searching the branch and bound tree is done using a combination of cost first and depth first search. The cost of a non-leaf node is the lower bound of the energy and the cost of a leaf node is the energy of the corresponding structure. We search the branch and bound tree by keeping a set of nodes for which the lower bound has been computed but not bounded. Initially the set contains only the root of the branch and bound tree. Iteratively the algorithm chooses the lowest cost node and replaces it with the children obtained by branching. When using this strategy, an optimal solution is found when the lowest cost node in the set is a leaf node. In practice the set of unbranched nodes becomes very large and difficult to store in memory. We therefore combine it with a depth first search, such that when the node set contains more than 50.000 nodes we shift to depth first search until the set is less than 50.000 again. This approach gives

12

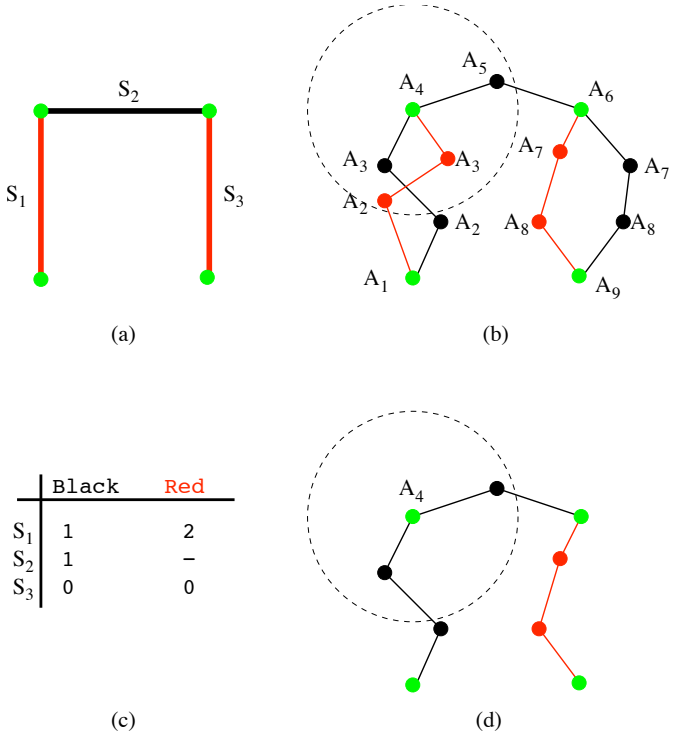


Fig. 5. (a) shows the directions of three segments (a super structure). In this example we want to compute all possible CN values for residue A_4 which is the first residue of segment S_2. The contact radius of residue A_4 is illustrated by the circle in (b). S_1 and S_3 both have two choices of segment structures (red and black), so $u = 2$. The table in Figure (c) shows the contribution of contacts to residue A_4 if either red or black segment structure is chosen. If the black structure of S_1 is chosen, S_1 only contributes with 1 contact to A_4 and if the red structure is chosen, S_1 contributes with 2 contacts. Computing all possible CN values for A_4 can be done by considering all combinations of segment structures for the other segments which is exponential. (d) shows one of these combinations which gives a CN value of 2 for A_4.

	Black	Red
S_1	1	2
S_2	1	–
S_3	0	0

	0	1	2	3	4	5	6	7	8
S_1		X	X						
S_2			X	X					
S_3			X	X					

Fig. 6. (a) shows the input to the dynamic programming algorithm as constructed in Figure 5. (b) shows Table $q_{a,b}$ where empty entries correspond to 0 and x correspond to 1. In the first row, only 1 or 2 contacts can be contributed to residue A_4 if either black or red structure of segment S_1 is chosen. Segment S_2 has a fixed segment structure and therefore always contributes with one contact as shown in row 2 and finally row 3 shows that segment S_3 does not contribute with any contacts to A_4. The last row is also the solution to the problem. It shows that from all combinations of segment structures, the CN value of residue A_4 can only be 2 or 3.

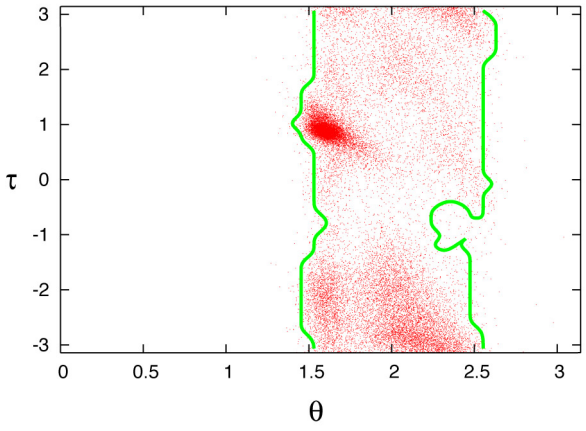


Fig. 7. A plot of (θ, τ) pairs from PDB

14

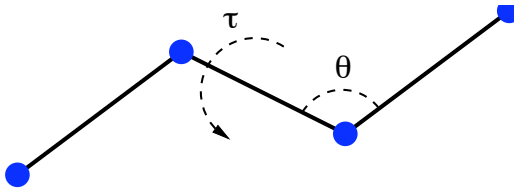


Fig. 8. θ is the normal angle between 3 consecutive C_α positions and τ is the dihedral angle between 4 consecute C_α positions.

a more memory efficient algorithm, but we might end up computing more lower bounds than in a pure cost first search.

3 Experiments

Here we predict the tertiary structures of 6 proteins. The tertiary structures of the proteins are known and we can therefore evaluate the quality of our results. These proteins have previously been used for benchmarks in the literature [20, 21].

The input to EBBA is a secondary structure assignment, HSE-vector and the radius of gyration. For each protein we obtain these values using prediction tools. Based on the amino acid sequence, we predict the secondary structure using PSIPRED [7] and we predict HSE-vectors using LAKI [4]. Note that PSIPRED and LAKI are neural networks trained on a selection of proteins from PDB. The 6 benchmark proteins used here also exist in PDB, so there is a slight chance that the training sets for PSIPRED and LAKI contain some of these proteins. However, the prediction quality of the 6 benchmark proteins is close to what should be expected from PSIPRED and LAKI. Here, the average Q_3 score of secondary structure prediction is 80.7% (compared to an average score of 80.6% on CASP targets). The average correlation of the HSE up and down values are respectively 0.74 and 0.66 (compared to the reported up and down correlations of 0.713 and 0.696 respectively). We do therefore not consider it to be a problem that the benchmark proteins exist in PDB. We predict the radius of gyration using Equation 2.

Branch and bound algorithms are typically used to find the global minimum solutions. However, our experiments show that the global minimum solutions in our models are not always native-like. Therefore, EBBA is modified such that the 10.000 best structures in terms of energy are found and not just the global minimum. This can be done by maintaining a queue fo 10.000 structures during the search. This number is still very small compared to the exponential size of the conformational space. For comparison and evaluation of the model and prediction quality, all experiments are also done using the exact secondary structure

and exact HSE-vectors obtained from the native structure of the proteins. All experiments were initially run with $u = 8$ (the number of segment structures). Some did not finish in 48 hours, and they were run with the highest value of u that could be solved in less than 48 hours. All computations were performed on a 2.4 GHz P4 with 512 RAM.

4 Results and Discussion

Table 1 shows the complexity of the model for different proteins and the running time of EBBA. Table 1 also shows the results of running EBBA on the 6 benchmark proteins. Figures 9 and 10 show 2D histograms of the energy vs. RMSD distribution for the 10.000 structures. For better comparison of the energies for the different proteins, the root-mean of the energies are reported in this section.

The maximum number of segment structures (u) that could be solved in less than 48 hours depend much on the number of segments of the protein. For the smallest proteins (1FC2 and 1ENH) the algorithm terminated in less than 48 hours using $u = 8$. Even though 2GB1 has relatively many segments the algorithm also terminated in less than 48 hours using $u = 8$. This is because of the efficiency of the bounding algorithm. In Figure 11 it is shown that for 2GB1 a large fraction of the search space can be bounded early. The most difficult protein in terms of bounding efficiency is 4ICB (predict), where it turns out that significant bounding first occurs in level 5 of the branch and bound tree. In all instances the conformational space is huge, and it clear that finding global minimum structures could not have been done in reasonable time without efficient bounding.

Figures 9 and 10 show that the exact energy vs. RMSD is well correlated for the three smallest proteins while this is not the case for the larger proteins. The larger proteins have a higher degree of freedom, and it therefore seems that secondary structure, radius of gyration and HSE do not contain enough information to identify the native structure of proteins with more than ~ 60 residues. However, among the 10.000 best structures, structures close to the native structure exists for the longer proteins also.

Table 4 shows that the set of 10.000 low energy structures for all 6 proteins contains good decoys (RMSD less than 6 Å). Also, for all proteins the lowest RMSD is smallest when using exact values compared to the predicted values. This is expected since the energy landscape should have a global minimum closer to the native structure when using exact values. However, it is surprising that for two of the proteins (1FC2 and 2GB1) the fraction of good decoys (< 6 Å RMSD) is better when using predicted values compared to exact values. The plots in Figures 10 show that for these two proteins, the structures are much more clustered when using the predicted values. This indicates that the energy landscapes described using the predicted values have fewer local minima and for 1FC2 and 2GB1 they are clustered closer to the native structure.

In Table 2 the energy span of the 10.000 structures is shown. The table also shows the energy of the native structure of the protein using the predicted energy

16

Type	m segments	u SS	N	T hours	< 6 Å RMSD	< 5 Å RMSD	< 4 Å RMSD	lowest RMSD	$Q(P^*)$	P^* RMSD
			Protein A (1FC2), 43 residues							
Exact	5	8 CHCHC	1.7×10^8	0.1	18.1	7.0	0.7	**2.8**	4.34	6.6
Predicted	7	8 CHCHCHC	1.4×10^{12}	6.9	33.0	13.8	0.0	**4.5**	5.26	8.4
			Homeodomain (1ENH), 54 residues							
Exact	6	8 CHCHCH	1.5×10^{10}	0.6	21.6	13.2	1.8	**3.1**	4.36	3.5
Predicted	7	8 CHCHCHC	1.4×10^{12}	6.1	4.1	0.8	0.0	**4.1**	5.70	10.2
			Protein G (2GB1), 56 residues							
Exact	9	8 SCSCHCSCS	1.0×10^{16}	18.2	60.8	36.6	13.7	**3.4**	4.22	4.3
Predicted	10	8 SCSCHCSCSC	9.2×10^{17}	4.7	73.1	0.0	0.0	**5.3**	6.22	7.8
			Cro repressor (2CRO), 65 residues							
Exact	11	4 CHCHCHCHCHC	4.0×10^{16}	24.1	5.7	1.4	0.0	**4.3**	6.49	9.2
Predicted	10	3 HCHCHCHCHC	5.1×10^{13}	7.4	1.5	0.0	0.0	**5.3**	5.89	9.4
			Protein L7/L12 (1CTF), 68 residues							
Exact	8	8 SCHCHCHC	1.2×10^{14}	5.6	5.1	1.9	0.0	**4.6**	7.19	11.0
Predicted	11	3 SCHSHCHCHCS	1.7×10^{15}	19.2	0.1	0.0	0.0	**5.4**	5.84	11.3
			Calbindin (4ICB), 76 residues							
Exact	11	2 CHCSHCHCHCH	1.9×10^{13}	3.56	4.5	0.7	0.0	**4.4**	6.18	7.4
Predicted	8	7 CHCHCHCH	4.1×10^{13}	31.4	0.5	0.0	0.0	**5.1**	6.79	6.4

Table 1. Column 2 shows the number of segments m and column 3 shows the number of segment structures u. Column 4 shows the order of helix, sheet and coil segments. Column 5 shows the size of the conformational space given by Equation 1 and column 6 shows the number of hours spent by the algorithm. Column 7 to 9 show the percentage of the 10.000 structures that fall below the given threshold. Column 10 shows the lowest RMSD of the 10.000 structures. Column 11 shows the energy of P^* which is the lowest energy structure. The last column shows the coordinate RMSD between the native structure and P^*. For each protein, there is an *exact* and a *predicted* row. Exact refers to HSE-vectors, radius of gyration and secondary structure obtained from the native structure. In the *predicted* rows, all input values are predicted from the amino acid sequence and the results can therefore be considered as *de novo*.

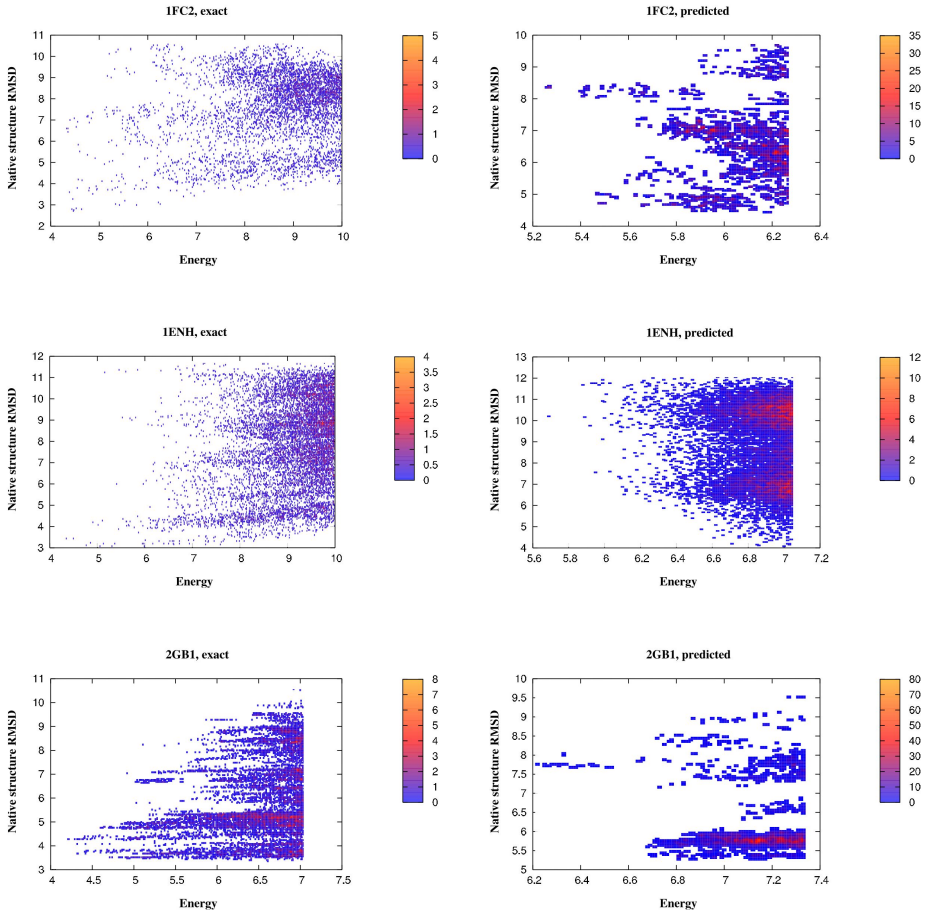


Fig. 9. Energy vs. RMSD histograms of 1FC2, 1ENH and 2GB1.

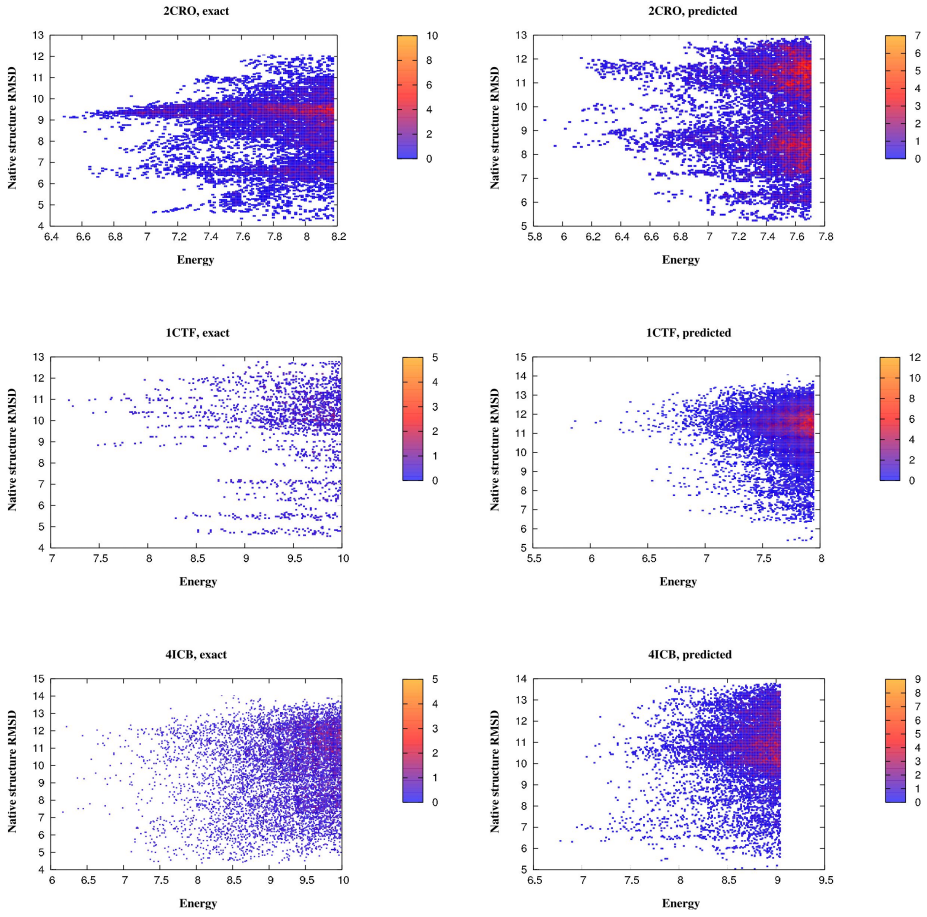


Fig. 10. Energy vs. RMSD histograms of 2CRO, 1CTF and 4ICB.

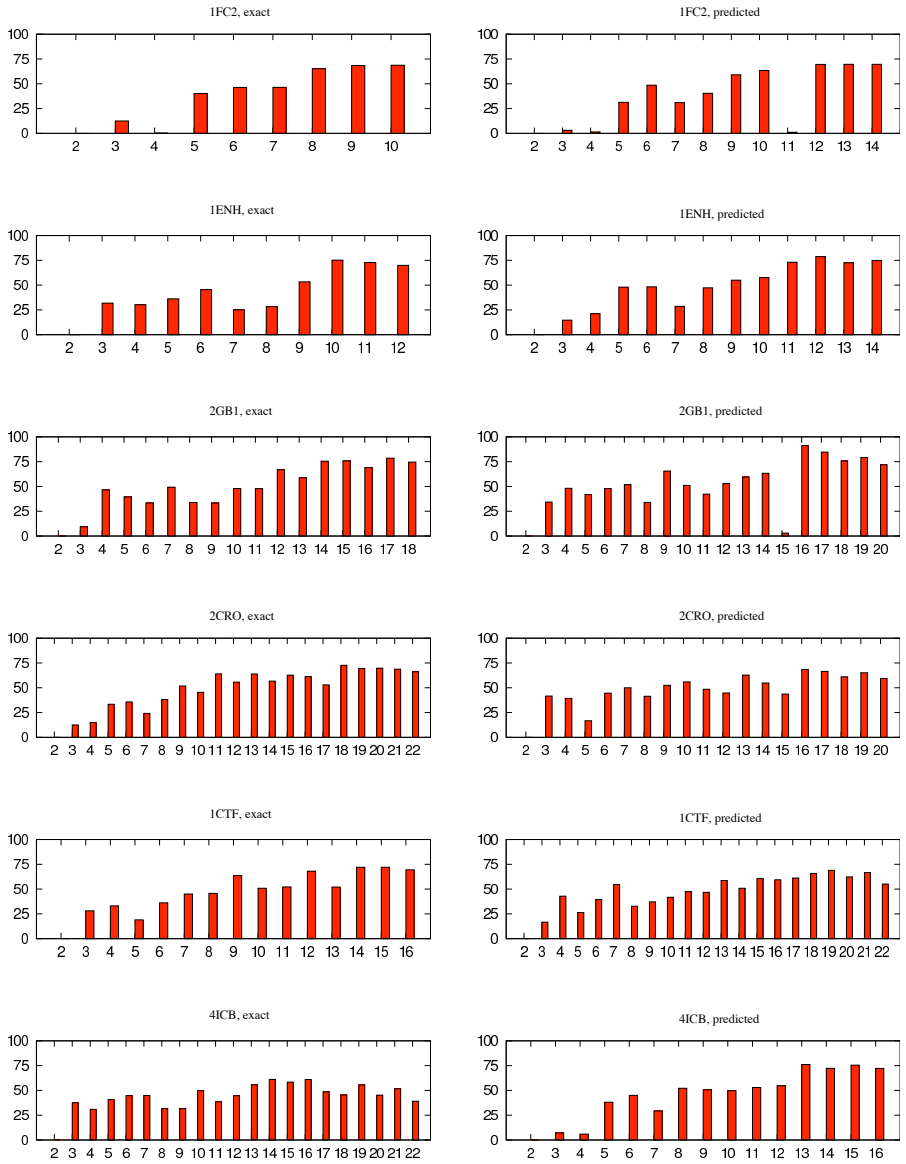


Fig. 11. The histograms show the bounding efficiency for each of the 12 runs of EBBA. The bars show the percentage of nodes in each level that was bounded. Level 1 is omitted, since the node in level 1 is never bounded (this would cause the whole search space to be bounded)

function. The values indicate that for most of the proteins (except 4ICB), the model is able to represent structures with lower energy than the native structure. Adding more degree of freedom in terms of segment directions and using more segment structures could consequently lower the energy of the 10.000 structures. However, since the energies of these structures are already comparable to the energy of the native structure, it should *not* be expected that more degree of freedom would improve the RMSD of the structures. Instead, improvements should come from adding more predictable information to the model or energy function or using more accurate predictions of HSE and secondary structure.

PDB	$Q(P^*)$	$Q(P^{10.000})$	Q^{native}
1FC2	5.26	6.28	6.46
1ENH	5.70	7.06	6.63
2GB1	6.22	7.34	7.53
2CRO	5.89	7.71	8.40
1CTF	5.84	7.96	7.58
4ICB	6.79	9.05	6.67

Table 2. For each protein the lowest energy of the 10.000 structures is $Q(P^*)$. The highest energy of the 10.000 structures is $Q(P^{10.000})$ and the energy of the native structure is Q^{native}.

The results have been compared directly with FB5-HMM [21] in Table 3. FB5-HMM is a successful method for conformational sampling. The method is based on a Hidden Markov Model and generates a large set of structures which usually contains many good decoys (< 6 Å RMSD) when enforcing compactness. The major difference between FB5-HMM and EBBA is that FB5-HMM does not use an energy function. FB5-HMM can also benefit from the secondary structure prediction and radius of gyration prediction. The results we have shown for FB5-HMM are therefore obtained using predicted secondary structure and using a greedy collapse scheme. The results for FB5-HMM are from [21] where 100.000 structures are generated. For all proteins, except 2GB1, FB5-HMM finds at least one structure with lower RMSD than EBBA. However, EBBA finds a better percentage of good decoys for most of the proteins (1FC2, 2GB1, 2CRO and 4ICB). Another advantage of the EBBA generated structures, is that the geometry of the secondary structure segments is perfect because they are constructed using the correct secondary structure geometry.

5 Conclusions

We have presented a branch and bound algorithm for finding the lowest energy structures in a large conformational search space. The energy function is based on HSE which is a simple predictable measure. This algorithm is the first ab initio

Protein	FB5-HMM		EBBA	
	< 6 Å	Min. RMSD	< 6 Å	Min. RMSD
Protein A (1FC2)	17.1	2.6	33.0	4.5
Homeodomain (1ENH)	12.2	3.8	4.1	4.1
Protein G (2GB1)	0.001	5.9	73.1	5.3
Cro repressor (2CRO)	1.0	4.1	1.5	5.3
Protein L7/L12 (1CTF)	0.3	4.1	0.1	5.4
Calbindin (4ICB)	0.4	4.5	0.5	5.1

Table 3. Comparison between FB5-HMM and EBBA. Column 2 and column 4 show the percentage of good decoys for FB5-HMM and EBBA respectively. Column 3 and column 5 show the lowest RMSD of a structure found by FB5-HMM and EBBA respectively. Both algorithms uses predicted secondary structure information and predicted radius of gyration.

branch and bound algorithm for prediction of protein structure using only one-dimensional predictable information. We have shown experimentally that good decoys always exist among the 10.000 lowest energy structures for the proteins used here. However, the energy function is not accurate enough to pinpoint the lowest RSMD structure in this set. An important future research direction is therefore to examine this set of low energy structures with a more detailed energy function and to identify the native-like structures. The largest protein considered have 76 residues. There is a problem using the branch and bound algorithm on larger proteins since then only a small fraction of the conformational space can be searched in reasonable time. However, we believe that exploiting how super secondary structures [22, 23] arrange in nature, might be a way to solve this problem. Better search heuristics for finding upper bounds on the energy can also be relevant since a good upper bound on the energy also improves the performance of the branch and bound algorithm. Using a more probabilistic approach might also improve the quality of the results. One idea is to compute probabilities from the (ϕ, ψ)-plot in Figure 7 instead of a simple threshold bound used here. It might also be possible to train a Bayesian network to predict the probability of a given HSE-vector given the amino acid sequence. This would be a more detailed usage of the HSE-vector compared to the simple energy function used here.

6 Acknowledgements

We would like to thank Thomas Hamelryck at the Bioinformatics Centre, University of Copenhagen for valuable contributions and insights. Martin Paluszewski and Pawel Winter are partially supported by a grant from the Danish Research Council (51-00-0336).

References

1. Paluszewski, M., Winter, P.: Protein decoy generation using branch and bound with efficient bounding. Lecture Notes in Bioinformatics, (to appear) (2008)
2. Hamelryck, T.: An amino acid has two sides: a new 2D measure provides a different view of solvent exposure. Proteins **59**(1) (2005) 38–48
3. Paluszewski, M., Hamelryck, T., Winter, P.: Reconstructing protein structure from solvent exposure using tabu search. Algorithms for Molecular Biology **1** (2006)
4. Vilhjalmsson, B., Hamelryck, T.: Predicting a New Type of Solvent Exposure. ECCB, Computational Biology Madrid 05, P-C35, Poster (2005)
5. Pollastri, G., Baldi, P., Fariselli, P., Casadio, R.: Prediction of coordination number and relative solvent accessibility in proteins. Proteins **47**(2) (2002) 142–53
6. Kinjo, A. R., Nishikawa, K.: Recoverable one-dimensional encoding of three-dimensional protein structures. Bioinformatics **21**(10) (2005) 2167–70
7. McGuffin, L. J., Bryson, K., Jones, D. T.: The PSIPRED protein structure prediction server. Bioinformatics **16** (2000) 404–405
8. Skolnick, J., Kolinski, A., Ortiz, A. R.: MONSSTER: A method for folding globular proteins with a small number of distance restraints. J. Mol. Biol. **265** (1997) 217–241
9. Fain, B., Levitt, M.: A novel method for sampling alpha-helical protein backbones. Journal of Molecular Biology **305** (2001) 191–201
10. Kolodny, R., Levitt, M.: Protein decoy assembly using short fragments under geometric constraints. Biopolymers **68**(3) (March 2003) 278–285
11. Backofen, R.: The protein structure prediction problem: A constraint optimization approach using a new lower bound. Constraints **6** (2004) 223–255
12. Backofen, R., Will, S.: A constraint-based approach to fast and exact structure prediction in three-dimensional protein models. Constraints **11**(1) (January 2006) 5–30
13. Maranas, C. D., Floudas, C. A.: A deterministic global optimization approach for molecular structure determination. J. Chem. Phys. **100** (1994) 1247–1261
14. Standley, D. M., Eyrich, V. A., Felts, A. K., Friesner, R. A., McDermott, A. E.: A branch and bound algorithm for protein structure refinement from sparse nmr data sets. J. Mol. Biol. **285** (1999) 1961–1710
15. Palu, A. D., Dovier, A., Fogolari, F.: Constraint logic programming approach to protein structure prediction. BMC Bioinformatics **5**(186) (November 2004)
16. Labesse, G., Colloc'h, N., Pothier, J., Mornon, J.-P.: P-SEA: a new efficient assignment of secondary structure from calpha trace of proteins. Bioinformatics **13** (1997) 291–295
17. Hobohm, U., Sander, C.: Enlarged representative set of protein structures. Protein Science **3** (1994) 522–524
18. Chothia, C., Lesk, A. M.: The relation between the divergence of sequence and structure in proteins. The EMBO Journal **5** (1986) 823–826
19. Wolsey, L. A.: Integer Programming. Wiley-Interscience (1998)
20. Simons, K. T., Kooperberg, C., Huang, E., Baker, D.: Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions. J Mol Biol **268**(1) (1997) 209–25
21. Hamelryck, T., Kent, J. T., Krogh, A.: Sampling realistic protein conformations using local structural bias. PLoS Computational Biology **2**(9) (September 2006) 1121–1133

22. Sun, Z., Jiang, B.: Patterns and conformations of commonly occurring supersecondary structures (basic motifs) in protein data bank. J. Protein Chem. **15**(7) (October 1996) 675–690
23. Boutonnet, N. S., Kajava, A. V., Rooman, M. J.: Structural classification of alphabetabeta and betabetaalpha supersecondary structure units in proteins. Proteins **30**(2) (February 1998) 193–212

Appendix B

Extended Abstract: Protein Structure Prediction using Bee Colony Optimization Metaheuristic

R. Fonseca, M. Paluszewski, and P. Winter. Protein Structure Prediction using Bee Colony Optimization Metaheuristic. *Meta'08, Extended Abstract.*
Status: To appear
This is the extended abstract of the paper in Chapter 12. The abstract was accepted for the Meta'08 conference.

Protein Structure Prediction Using Bee Colony Optimization Metaheuristic: Extended Abstract

R. Fonseca[1], M. Paluszewski, and P. Winter[1]

Dept. of Computer Science, Uni. of Copenhagen (DIKU) Universitetsparken 1, 2100 Copenhagen Ø
{ hite, palu, pawel }@diku.dk

1 Introduction

Proteins are the primary building blocks in all living organisms. They are made of amino acid chains bound together by peptide bonds. Depending on the sequence of amino acids, the proteins fold in three dimensions so that the Gibbs free energy is minimized. The shape determines the function of the protein. *Protein structure prediction* (PSP) is the problem of predicting this three-dimensional structure from the amino acid sequence and is considered one of the most important open problems of theoretical molecular biology. The PSP has applications in medicine within areas like drug- and enzyme design.

The PSP proves to be a very difficult optimization problem. Solving it exactly is still far from realistic. Use of heuristics and less complex models proves to be an absolute necessity. However, even in simplified scenarios, many computational problems arise. One of these problems is the belief that free energy landscapes tend to have many local minima [1]

The *Bee Colony Optimization* (BCO) metaheuristic is a relatively new approach based on swarm-intelligence for solving complex optimization problems. It mimics the foraging behavior of honey-bees searching for nectar in a flower field. The algorithm, like real honey-bees, performs a wide search for good solutions and has a flexible method for allocating resources to intensify the local searches. This seems like a good strategy in the PSP to avoid getting stuck in the local minima of the energy landscape.

Hesham et al. [2] previously used the *Bees Algorithm* [3] to find the native state of the 5-residue peptide 'met-enkalphin' (PDB-ID: 1PLW) using a full resolution torsion angle-based representation. In our work, we apply the BCO metaheuristic to the PSP problem using a simplified representation and generate good quality solutions in terms of the RMSD similarity measure. These decoy solutions can be used as starting solutions for more advanced methods (protein structure refinement algorithms). Since we use a coarser representation, real-sized protein structures can be attacked by our BCO metaheuristic. To our knowledge this is the first time a bee heuristic has been used to predict the structure of proteins. We do not claim to solve the PSP or even compete with state-of-the-art PSP algorithms like Rosetta[4] or I-Tasser [5], however the BCO metaheuristic has nice properties that we believe makes it suitable for the PSP.

2 Model

Proteins usually consist of thousands of atoms, and their full description must contain the coordinates of all atoms. By considering the geometry of the backbone, this representation can be simplified to an average of 5 degrees of freedom per amino acid. However, even for small proteins, this conformational space is still very large and difficult to search. Here, we therefore apply predictions of secondary structure to reduce the degrees of freedom even further by regarding a protein as a sequence of connected segments.

3 Algorithm

In nature, a foraging bee can be said to be in one of three states: A scout bee, a worker bee or an onlooker. Scout bees fly around a flowerfield at random and when a flowerbed is found they return

[1] Partially supported by a grant from the Danish Research Counsil (51-00-0336)

to the hive and perform a waggle dance. The dance indicates the estimated amount of nectar, direction and distance to the flowerbed. Onlooker-bees present in the hive watch different waggle dances, choose one and fly to the selected flowerbeds to collect nectar. Worker bees act like scout bees except that when they have performed the waggle dance they return to their old flowerbed to retrieve more nectar.

In our adaptation of the BCO metaheuristic, each bee corresponds to a solution, and the nectar amount corresponds to an objective value in the energy landscape. Sending out scout bees corresponds to finding a random feasible solution and sending out onlookers corresponds to finding a neighborhood solution. The onlookers choose sites for neighborhood search based on the objective value of scouts and workers in previous iterations. This method is largely the *Bees Algorithm* proposed in [3]. In a non-changing solution space a solution does not deplete in the same way a real life flowerbed depletes of nectar. Exhaustion is therefore forced when a solution cannot be improved. This idea is somewhat similar to the idea of pruning parts of the searchspace as described in [6]. The process of exhausting a local search is proposed as part of the *Artificial Bee Colony* algorithm described in [7]. Our adaptation of the BCO metaheuristic is a synthesis of these approaches.

4 Dataset

To test our BCO metaheuristic, we try both simple and complex proteins with respect to both residue-length and the number of secondary structure segments. Six proteins are from [9] and all have less than 12 segments and from 54 to 76 residues. Six different proteins are chosen from CASP7 [10] which all have more than 76 residues and more than 12 segments.

5 Results and perspective

Simulated Annealing (SA) and Monte Carlo are often used in the PSP [8], so for comparison both BCO and SA were used to minimize the energy of the 12 selected proteins. Despite the fact that SA is so frequently used for the PSP, BCO outperforms SA by finding lower energy structures for the 6 smaller proteins. Partial results show promising predictions for the 6 larger ones as well.

BCO seems to differ from SA in its wide search and good prioritizing of local searches. Furthermore, the algorithm seems extremely flexible. The local search performed by onlookers and the random solutions found by scouts can be implemented using any of the well-known algorithms. SA, Monte Carlo or hill-climbing can for instance be used for local search and genetic algorithms for generating random solutions. Different strategies can even be combined.

References

1. Li, Z. and Scheraga, H. A. Monte carlo-minimization approach to the multiple-minima problem in protein folding. Proceedings of the NationalAcademy of Sciences 84(19):6611-6615,October, 1987.
2. Hesham Awadh Abdallah Bahamish, Rosni Abdullah, Rosalina Abdul Salam, "Protein Conformational Search Using Bees Algorithm," ams , pp. 911-916, 2008.
3. Pham DT, Ghanbarzadeh A, Koc E, Otri S, Rahim S and Zaidi M. The Bees Algorithm. Technical Note, Manufacturing Engineering Centre, Cardiff University, UK, 2005
4. Rohl, C. A., Strauss, C. E. M., Misura, K. M. S., and Baker, D. Protein structure prediction using Rosetta. Methods in Enzymology 383:66-93, 2004.
5. Y Zhang: I-TASSER server for protein 3D structure prediction. BMC Bioinformatics **9**:40 (Jan 2008)
6. M. Paluszewski, T. Hamelryck, P. Winter. Reconstructing protein structure from solvent exposure using tabu search, Algorithms for Molecular Biology, 2006.
7. D. Karaboga, An Idea Based On Honey Bee Swarm for Numerical Optimization, Technical Report-TR06,Erciyes University, Engineering Faculty, Computer Engineering Department 2005.
8. G. Helles. A comparative study of the reported performance of ab initio protein structure prediction algorithms. J R Soc Interface. 2008 Apr 6;5(21):387-96.
9. Hamelryck, T., Kent, J. T., Krogh, A.: Sampling realistic protein conformations using local structural bias. PLoS Computational Biology **2**(9) (September 2006) e131
10. Critical Assessment of Techniques for Protein Structure Prediction, Asilomar Conference Center, Pacific Grove, CA November 26-30, 2006. http://predictioncenter.org/casp7/Casp7.html

Appendix C

Poster: Protein Structure Prediction Using Tabu Search and Half-Sphere-Exposure Measure

M. Paluszewski, T. Hamelryck, and P. Winter. Protein Structure Prediction using Tabu Search and Half-Sphere-Exposure Measure. *RECOMB (poster)*, 2006.

Status: Abstract published

Protein Structure Prediction Using Tabu Search and Half-Sphere Exposure Measure

Martin Paluszewski[1], Thomas Hamelryck[2], Pawel Winter[3]

[1]Department of Computer Science, University of Copenhagen. Email: **palu@diku.dk**
[2]Bioinformatics Center, University of Copenhagen. Email: **thamelry@binf.ku.dk**
[3]Department of Computer Science, University of Copenhagen. Email: **pawel@diku.dk**

Introduction

The extent to which an amino acid is accessible to the surrounding solvent is highly determined by the type of amino acid. In general, hydrophilic amino acids tend to be near the solvent accessible surface and hydrophobic amino acids tend to be packed in the core of the protein. To measure this effect, several solvent exposure measures have been proposed,[1–3,6,9–11] and one of these is the *contact number measure* (CN).[11] The CN of a residue is the number of other C_α atoms in a sphere centered at the C_α atom of the residue. The CN of all residues is called the CN vector.

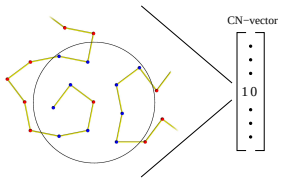


Figure 1: The contact number (CN) of a residue.

Recently, a new promising solvent exposure measure, called *half-sphere-exposure* (HSE), has been proposed.[4] While the CN measure uses a single sphere centered at C_α atoms, the HSE measure considers two hemispheres. Two values, an *up* and a *down* value, are therefore associated with each residue corresponding to the upper and lower hemisphere. Hamelryck[4] showed that the up and down values are almost uncorrelated and well conserved and could therefore be considered independently.

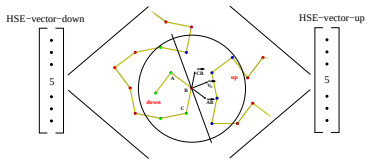


Figure 2: Given the positions of 3 consecutive C_α atoms (A, B, C), the approximate side-chain direction $\vec{V_b}$ can be computed as the sum of $\vec{AB}$ and $\vec{CB}$. The plane perpendicular to $\vec{V_b}$ cuts the sphere centered at B in an upper and a lower hemisphere.

Here, we study the reconstruction of a protein's C_α trace solely from a CN vector or a pair of up/down vectors (HSE vector). This problem could become important for de novo structure prediction when the vectors are predicted.
It is straightforward to compute the CN-/HSE vector of a given structure. However, it is difficult to compute a structure realizing a given CN-/HSE vector. If the CN-/HSE vector is determined by a prediction method, there might not be a structure that exactly realizes the vector. In that case, we are interested in finding a structure with a CN-/HSE vector similar to the predicted vector. We define potential functions based on the HSE- or CN-vectors and minimize them using two metaheuristics: Monte Carlo simulation (MCS) and tabu search (TS). MCS has been widely used for protein structure prediction, and TS has been applied with great success to many optimization problems, but has rarely been used for protein structure prediction.[5,7,8]

HSE Potential Function

Given two HSE vectors, A and B of length N, their similarity can be described using the RMS deviation. If B is the *reference* HSE vector, then the HSE potential of a structure S with HSE vector A is defined as:

$$\mathrm{HSE}(S_A) = \sqrt{\frac{\sum_{i=1}^{N}((A_{u_i} - B_{u_i})^2 + (A_{d_i} - B_{d_i})^2)}{2N}},$$

where X_{u_i} and X_{d_i} are the up and down values of the i'th residue. These definitions are easily specialized to the CN measure.

Protein Model

To reduce and discretize the protein conformation space, we require the C_α atoms of the chain to be positioned in a 3D lattice. A lattice can be defined by a set of basis vectors corresponding to the directions to the neighbouring nodes. The basis vectors of the *simple cubic lattice* (SCC) are the cyclic permutations of $[\pm 1, 0, 0]$ and the basis vectors of the *face centered cubic lattice* (FCC) are the cyclic permutations of $[\pm 1, \pm 1, 0]$. The length of an edge between two neighbouring nodes is 3.8 Å which is also the average distance between two consecutive C_α atoms in proteins.
A *high coordination lattice* has an underlying cubic lattice with unit length less than $3.8/N$ Å for some integer $N > 1$. Cubic lattice points are connected in the high coordination lattice if their Euclidean distance is between $3.8 \pm \epsilon$ for some ϵ. The high coordination lattice used in the experiments has parameters $N = 8$, $\epsilon = 0.2$, which gives 390 basis vectors.

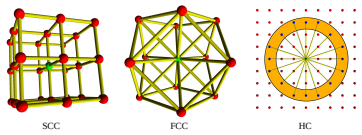


Figure 3: Interior nodes of the SCC and FCC lattices are connected to respectively 6 and 12 neighbouring nodes. Nodes of high coordination lattices have many neighbours because of variable edge size.

Conformational Search Heuristics

TS is basically a local improvement heuristic where the best structure in a neighbourhood is repeatedly selected. Previous TS implementations[7] inserted visited structures in a *tabu list* and only considered new structures that were not in the tabu list. We have found that extending the tabu definition improves the performance considerably.
Here, we still keep a list of previously visited structures called an *explicit tabu list*. Each structure in the explicit tabu list defines a set of *implicit tabu structures*. Given a structure E in the explicit tabu list, a structure I is said to be implicit tabu if the distance-RMSD (dRMSD) between E and I is less than α and the potential of I is greater than or equal to the potential of E. The adjustable parameter α is called the *tabu difference*.

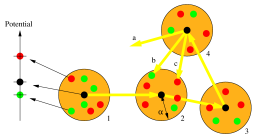


Figure 4: The figure illustrates a sequence of visited structures (black points) in a solution space. The visited structures are inserted in the explicit tabu list. The additional green and red points correspond to structures within α dRMSD of the explicit tabu structure. Green points are structures with lower potential and red points are structures with higher potential than the explicit tabu structure. When choosing a new solution in the neighbourhood three things can happen. a) A solution is more than α dRMSD away from any explicit tabu structure. b) the solution is within α dRMSD, and the potential is *lower* than the explicit tabu structure. c) the solution is within α dRMSD, and the potential is *higher* than the explicit tabu structure. Structures that comply with case c are said to be tabu and cannot be visited.

MCS heuristics are stochastic and therefore differ from TS heuristics by being nondeterministic. An MCS iteration consists of randomly choosing a protein conformation in the neighbourhood of a current conformation. For a fixed temperature T, the new protein conformation is accepted with the probability

$$p = e^{-\Delta E/T},$$

where ΔE is the difference between the potential of the current conformation and the new conformation.

Experimental Results: Comparison of Heuristics

For both TS and MCS, 20 searches starting at random conformations are optimized in 20 minutes using the HSE potential for Protegrin 1 (1PG1, length 18 amino acids).

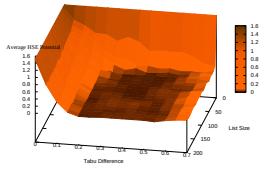


Figure 5: TS has two adjustable parameters - tabu difference and tabu list length. The figure shows the average potential of the 20 runs with different combinations of parameters.

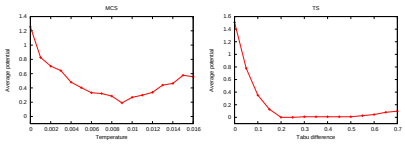


Figure 6: MCS has one adjustable parameter which is the temperature. The figure shows the average potential of the 20 runs with different temperatures. For comparison the results for TS with a fixed tabu list size of 200 is also shown.

The analysis shows that on average TS performs better than MCS for this problem. TS also seems to be less sensitive to the value of the parameter (temperature or tabu difference) and therefore easier to control.

Experimental Results: Comparison of CN and HSE

In the following experiments, TS with tabu difference of 0.3 and tabu list length of 200 are applied to compare the CN-/HSE measures. We use two measures to determine the quality of an optimized structure; the coordinate RMSD and *angle correlation*. Angle correlation is a new measure with the following definition.
For each C_α, let $\vec{V_\alpha}$ be the vector pointing in the side chain direction (see Figure 2). Let $\vec{V_{amc}}$ be the vector pointing in the direction of the mass center, and let θ_α be the angle between $\vec{V_\alpha}$ and $\vec{V_{amc}}$. The angle correlation measure is the average of the differences in θ_α between the optimized structure and the native structure. Zero angle correlation is perfect correlation, 90 is random correlation and 180 is 'anti'-correlation.

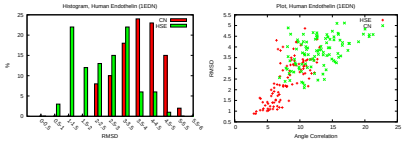


Figure 7: 100 random structures are optimized using TS. Each structure is optimized using the CN-potential and the HSE-potential. The RMSD between the optimized (low potential) structure and the native structure is computed. The histogram shows the distribution of structures in terms of RMSD with the native structure (human endothelin 1EDN, length 21 amino acids). More results are included in the handouts.

Conclusion

We have developed a framework for minimizing the CN-/HSE potential. The search heuristic is based on TS with a novel tabu definition and it performs significantly better than MCS for this problem. The results of CN and HSE comparisons show that structures found using the HSE potential are generally much closer to the native structure than structures found using the CN potential. These results are found using short proteins (the largest protein has 36 amino acids). When using larger proteins, it becomes very difficult to find near optimal solutions. Future research could therefore consider a more detailed potential function using secondary structure information.

Acknowledgments Thomas Hamelryck is supported by a Marie Curie Intra-European Fellowship within the 6th European Community Framework Programme. Martin Paluszewski and Pawel Winter are partially supported by a grant from the Danish Research Council (51-00-0336).

References

[1] S. Chakravarty and R. Varadarajan. Residue depth: a novel parameter for the analysis of protein structure and stability. *Structure*, 7(7):723–32, 1999.

[2] M. L. Connolly. Solvent-accessible surfaces of proteins and nucleic acids. *Science*, 221(4612):709–13, 1983.

[3] J. Greer and B. L. Bush. Macromolecular shape and surface maps by solvent exclusion. *Proc Natl Acad Sci U S A*, 75(1):303–7, 1978.

[4] T. Hamelryck. An amino acid has two sides: a new 2D measure provides a different view of solvent exposure. *Proteins*, 59(1):38–48, 2005.

[5] L. B. Morales, R. Garduño-Juárez, J. M. Aguilar-Alvarado, and F. J. Riveros-Castro. A parallel tabu search for conformational energy optimization of oligopeptides. *Journal of Computational Chemistry*, 21(2):147–156, 2000.

[6] B. Lee and F. Richards. The Interpretation of Protein Structures: Estimation of Static Accessibility. *J Mol Biol*, 55:379–400, 1971.

[7] M.T. Oakley, J.M. Garibaldi, and J.D. Hirst. Lattice models of peptide aggregation: Evaluation of conformational search algorithms. *J Comput Chem*, 26(15):1638–46, 2005.

[8] P. M. Pardalos, X. Liu, and G. L. Xue. Protein Conformation of a Lattice Model Using Tabu Search. *Journal of Global Optimization*, 11:55–68, 1997.

[9] A. Pintar, O. Carugo, and S. Pongor. Atom depth as a descriptor of the protein interior. *Biophys J*, 84(4):2553–61, 2003.

[10] A. Pintar, O. Carugo, and S. Pongor. Atom depth in protein structure and function. *Trends Biochem Sci*, 28(11):593–7, 2003.

[11] G. Pollastri, P. Baldi, P. Fariselli, and R. Casadio. Prediction of coordination number and relative solvent accessibility in proteins. *Proteins*, 47(2):142–53, 2002.

Appendix D

Poster: Branch and Bound Algorithm for Protein Structure Prediction using Efficient Bounding

M. Paluszewski and P. Winter. Branch and Bound Algorithm for Protein Structure Prediction using Efficient Bounding. *RECOMB (poster)*, 2007.
Status: Abstract published

Branch and Bound Algorithm for Protein Structure Prediction Using Efficient Bounding

Martin Paluszewski[1], Pawel Winter[2]

Department of Computer Science, University of Copenhagen. Email: **palu@diku.dk**[1], **pawel@diku.dk** [2]

Introduction

We are faced with two major challenges when dealing with the problem of *de novo* protein structure prediction. One is to determine a suitable energy function having a global minimum near the native structure of the protein. The other challenge is to *identify* the global minimum structures in a huge conformational space. Here we attack the latter of these challenges.

We propose a new discrete model which makes use of secondary structure information and propose a novel branch and bound algorithm for finding global minimum structures. The energy function is very simple while structures obtained are of very high quality compared to the native structure of the protein. The model only depends on the position of C_α-atoms and is based on predictable contact and half-sphere-exposure numbers [2, 4, 1]. The success of the branch and bound algorithm comes from the simplicity of the energy function which allows for efficient lower bound calculations of the energy. Despite the simplicity of the model, we show experimentally, that many of the lowest energy structures in the model are near the native structure.

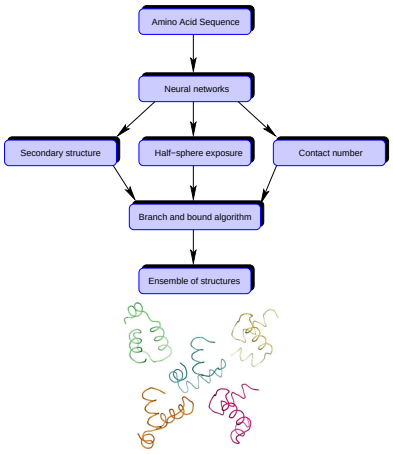


Figure 1: From amino acid sequence, three 1-dimensional vectors are predicted using neural networks [3, 1]. These predictions are used as input for the branch and bound algorithm to generate an ensemble of good structures.

Branch and Bound Algorithm

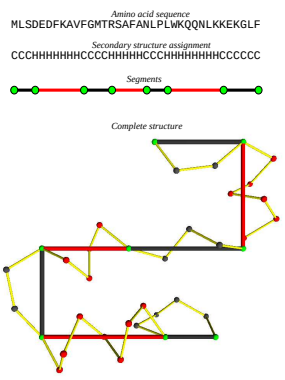


Figure 2: An example of how an amino acid sequence (from Villin headpiece) can be described as a chain of segments based on the secondary structure (H: helix, C: coil). It also shows a super structure and a corresponding complete structure (coordinates of internal C_α-atoms). For illustrationary purposes the super structure in this example is 2-dimensional, in general the super structure is of course 3-dimensional.

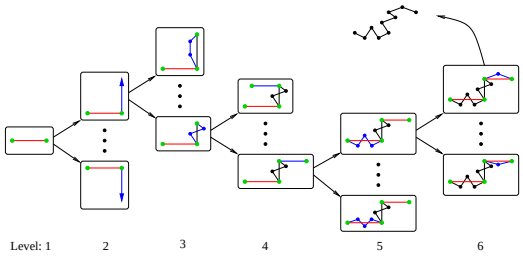


Figure 3: An example of a branch and bound tree. The super structure consists of three segments: *helix, coil, helix*. For simplicity, in each level, only two nodes are shown and only one node is branched on.

Energy and Bounding

All structures found by the branch and bound algorithm have their secondary structure fixed according to the neural network prediction. However, we want to identify structures having contact and half-sphere exposure numbers close to the predicted values. This is done using an energy function having $E = 0$ when half-sphere exposure and contact numbers are similar to the predicted values. The energy becomes larger the more the predicted measures differ.

It is possible to efficiently compute lower bounds of the energy for internal nodes of the branch and bound tree. This enables the algorithm to bound subspaces and solve large problems to optimality. A much more detailed description of energy and bounding is given in [5].

Results

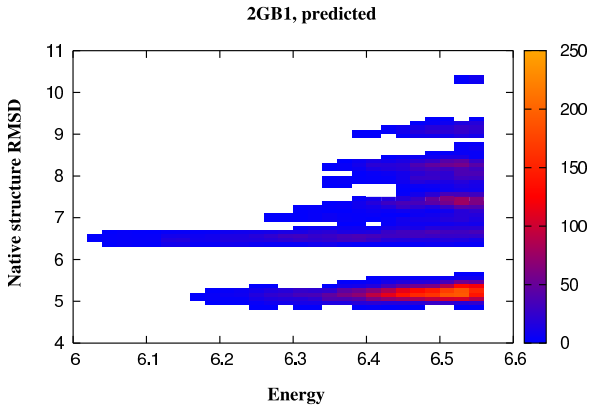


Figure 4: The branch and bound algorithm identifies 10.000 lowest energy structures. A 2D histogram of energy vs. RMSD of these structures is plotted. For most proteins, the algorithm finds candidate structures having RMSD less than 6 Å from the native structure. See [5] for more results.

References

[1] Bjarni Vilhjalmsson and Thomas Hamelryck. Predicting a New Type of Solvent Exposure. ECCB, Computational Biology Madrid 05, P-C35, Poster, 2005.

[2] T. Hamelryck. An amino acid has two sides: a new 2D measure provides a different view of solvent exposure. *Proteins*, 59(1):38–48, 2005.

[3] McGuffin, L. J., Bryson, K., and Jones, D. T. The PSIPRED protein structure prediction server. *Bioinformatics*, 16:404–405, 2000.

[4] Paluszewski, M., Hamelryck, T., and Winter, P. Reconstructing protein structure from solvent exposure using tabu search. *Algorithms for Molecular Biology*, 1, 2006.

[5] Paluszewski, M. and Winter, P. Branch and bound algorithm for protein structure prediction using efficient bounding. *Submitted*, 2007.