Universiteit Leiden
Opleiding Informatica

Protein structure prediction

by Iterative fragmenT Assembly (PITA)

Name:	Jonathan Neuteboom
Studentnr:	s0528315
Date:	December 21, 2014
1st supervisor:	Erwin Bakker
2nd supervisor:	Michael Lew

MASTER'S THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Contents

Abstract

In this paper we introduce Protein structure prediction by Iterative fragmenT Assembly (PITA), a protein structure predictor using a dihedral angle database to iteratively build the 3D structure of the protein. Two methods of construction have been proposed: Iterative fragment assembly favouring conformations with the most probable dihedral angles and Iterative fragment assembly with energy calculation, favouring conformations with the lowest energy. Using steric hindrance and Minimal Bounding Sphere (MBS) pruning, both search techniques are capable of predicting conformationally correct structures. Using the scoring function Local Global Alignment (LGA) and a scoring function based on the secondary structure (CSS), predictions up to LGA = 0.937939 and CSS = 80.4% are reported, comparable to state of the art protein structure predictions, showing promising results for PITA. Finally, a novel Energy Adaptation Function (EAF) is introduced to overcome the many local minima of the energy landscape, resulting in a significantly higher probability that PITA will predict a fully built protein (100% for our test set).

1 Introduction

In 1972, Christian B. Anfinsen denaturated Bovine nuclease A, a 124 residue protein, by adding urea to the solution. When removing the denaturant, the protein was again fully functional, showing some proteins are able to successfully refold to their minimal energy state without Chaperones, since these proteins were not present in the solution[1]. Although the physics of proteins seams fairly straight forward in this experiment, protein structure prediction is still one of the mayor challenges in Bioinformatics today. The prime objective of protein structure prediction is predicting a proteins three dimensional structure from its amino acid sequence and has been called the *holy grail of molecular biology* and is considered equivalent to deciphering *the second half of the genetic code*[2]. In medicine, protein structure prediction can become vital for diseases linked to different folds of the protein and their origin, where an different fold is a consequence of an alteration in the gene. Diseases such as Creutzfeldt-Jakob's, Alzheimer's, Huntington's and Parkinson's [3][4] are all connected to the malformations of proteins. In structure-based drug design, protein structure prediction can be used to predict binding affinities of different substrates on the binding site of compounds. In this way, the search for promising drugs can be narrowed down to the more promising compounds[5]. Over the last 5 decades, particularly since the human genome project, the field of protein structure prediction has received enormous interest among molecular biologists and computational scientists. This is because of the potential impact on many areas, since the three dimensional structure is essential to understanding the function of a protein and the interaction with other proteins. Furthermore, since a great amount of data that is retrieved from 3rd generation DNA sequencing platforms and the cost of high-resolution NMR and very high resolution X-ray in both time and money, a need for protein structure prediction rises. Many prediction techniques have been suggested based on Hidden Markov Models (HHPRED[6]), Monte Carlo (Rosetta [7][8], SimFold[9]) and Molecular Dynamics (NAMD[10], ASTROFOLD[11], GROMACS[12][13]) and will be covered in Section 4.

2 Some Basic Concepts

2.1 Protein Biosynthesis

All the information to create and maintain a cell is stored in DNA, two biopolymers coiled around each other to form a double helix. This double helix consist only of 4 simpler units called the nucleotides: Adenine, Cytosine, Thymine, Guanine, denoted by A, C, T and G, respectively. In the DNA, both strands are aligned and for every nucleotide in one strand there is an aligned counter nucleotide in the other bound through hydrogen bonding, creating base pairs: A will bind with T and C with G, as can be seen in Figure 1. With this, two structures are created, mimicking two sentences consisting of only four letters, one read from right to left and the other sentence in reverse. In these sentences all the information is stored for the cell to survive and procreate. The creation of proteins in a living cell is known as the process of *Protein Biosynthesis*. It consist of two parts: Transcription and Translation.

2.1.1 Transcription

The nucleotides in DNA roughly encode two things: Genes and binding sites. A gene is a sequence of consecutive nucleotides in the DNA which carries the information for one (domain of the) protein. In the human DNA, only 2% is coding DNA, e.g. DNA encoding proteins. The remaining 98% is thought to be junk-DNA (non-coding), but it has been known for decades that many non-coding sequences are functional. The process of copying the information stored in the gene to the intermediary product

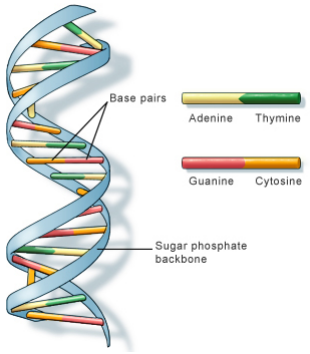


Figure 1: Abstraction of DNA: two strands consisting of nucleotides which are bonded covalently and creating base pairs, bonded through hydrogen bonds.

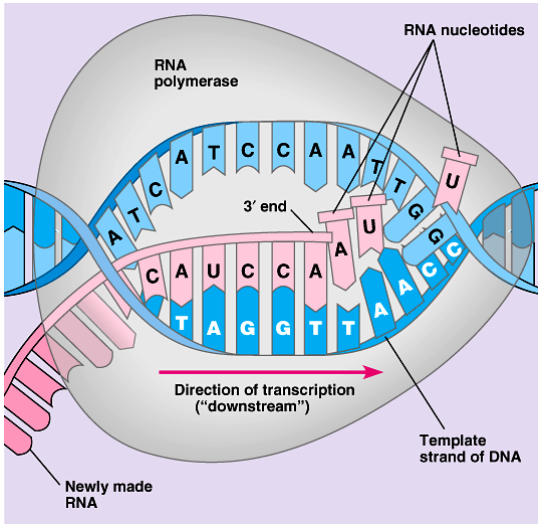


Figure 2: The process of transcription, depicting the creation of mRNA from RNA nucleotides, using the DNA strain as template. Copyright ©Pearson Education, Inc., publishing as Benjamin Cummings.

mRNA (messenger RNA) is called transcription. Binding sites around the gene are used to attach the copying machinery to the DNA strand: *RNA polymerase*. The site where the *RNA polymerase* binds to the gene is known as the promotor. The gene consists of the coding strand, the strand having the information for the protein, and the template strand, its complementary strand. The promotor is next to the template strand. Starting from there, it moves over the template strand, making a complementary strand: a copy of the coding strand, the mRNA. The mRNA is identical to the coding strand, except T is substituted for U (Uracil). The process is shown in Figure 2.

2.1.2 Translation

The second step of the protein biosynthesis is *translation*. In this process, the intermediary biopolymer mRNA, will be used to create the actual protein. In this process a sequence of 4 different letters (A, C, T, G), is changed to a sequence with 20 different letters. Living cells have solved this using the genetic code. This set of rules translates the mRNA directly and deterministically into proteins: every nucleotide triplets in the mRNA, called a codon, specifies one amino acid, as shown in Figure 3. For a string of $3n$ mRNA letters, a string of n codons can be made, and thus a protein of n amino acids. Using this machinery, shown in Figure 4, all proteins in all living (prokaryotic) cells are made.

2.2 Amino Acids

Proteins, as is stated above, are linear chains of its building blocks, called amino acids, that adopt a unique 3D structure in their native surroundings. A proteins structure is split into four different

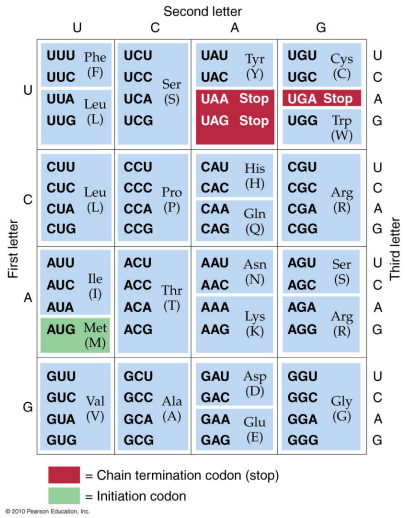


Figure 3: The genetic code, showing all possible codons and their matching amino acid. AUG is the starting codon (in green) and UAA, AUG and UGA are stop codons (in red).

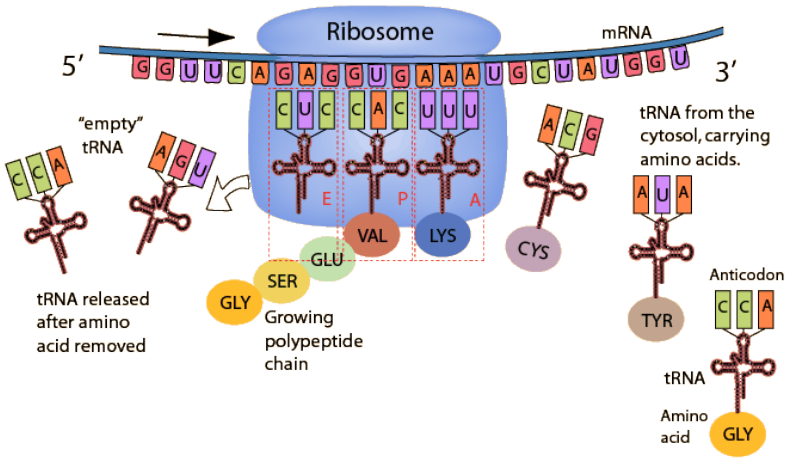


Figure 4: The translation process incorporates the 20 standard amino acids, depicted by their 3-letter-notation, in the precise sequence, dictated by the three-base codons of the mRNA. The tRNA aligns with mRNA using its anti codon. The ribosome has three sites; the A-site, P-site and E-site. The A-site, being the acceptor site, is where the new tRNA is positioned. The P-site is occupied by peptidyl-tRNA, i.e. the tRNA carrying the growing peptide chain. The E-site is the site were the empty tRNA leaves the ribosome.

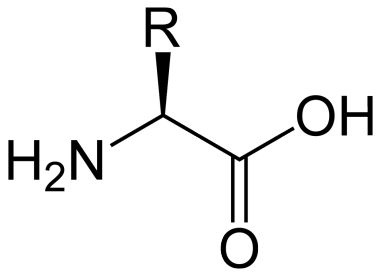


Figure 5: General overview of an amino acid.

levels of structure. Since the building blocks are connected sequentially, the first level, the *primary structure*, is the sequence of amino acids of a protein. In Figure 5, the structure of an amino acid is given, showing at the left side the amino group (H_2N), the carboxyl group at the right (COOH) and the side chain R. The primary structure of the protein is the amino acid sequence. The primary structure of a protein is defined by the 1-dimensional sequence S:

$$S = r_1 \cdots r_N, r_i \in R, R = \{A, C, D, \ldots, X\} \text{ where } 1 \leq i \leq N \tag{1}$$

where R is the set of all the residues, depicted by their one-letter-notation, r_i is a residue at position i of the sequence and N is the length of the sequence. All the amino acids in the protein are connected through a peptide bond: the C−N bond between two neighbouring amino acids. This peptide bond is the result of an amino group reacting with an acidic group. The reaction between two amino acids, is shown in Figure 6. Starting only with two amino acids, a dipeptide is created. The dipeptide can react with another amino acid to form a tripeptide and so on. The result is a polypeptide consisting of a main chain of repetitive NCC fragment, called the backbone and the R-groups connected to this backbone, one for each amino acid. There are 23 proteinogenic amino acids. Of the 23, selenocysteine and pyrrolysine are incorporated into proteins post-translationaly, and N-formylmethionine is often the initial amino acid of proteins in bacteria, mitochondria, and chloroplasts, but is often removed post-translationally. This leaves us with 20 standard amino acids, all with a unique side chain. These residues can be split into three different groups: charged residues, polar residues and hydrophobic residues, shown in Table 3. The column *Chemical structure* shows, for every amino acid, the amino group, the carboxyl group and its side chain.

Charged residues contain an electron donor or acceptor and the charge of their side chain can change at different pH levels, by donating or accepting a hydrogen atom (H^+) on the side chain. Polar residues are able to make hydrogen bonds, because of the electronegativity in its structure. Electronegativity is the ability of an atom to attract electrons to itself and is effected by the number of protons in the core of the atom and the distance from the outer electron to the nucleus. Since oxygen has a higher electronegativity than, for example, hydrogen, the oxygen atom attracts more electrons than the hydrogen atom and the electron cloud is then slightly moved towards the oxygen atom. This creates a net negative charge on the oxygen atom and a net positive charge on the hydrogen atom. The order of electronegativity for atoms in proteins is the following:

$$\mathrm{O} > \mathrm{N} > \mathrm{S} > \mathrm{C} > \mathrm{H} \tag{2}$$

where oxygen has an electronegativity of 3.44 and hydrogen 2.20, both on the Pauling scale, a relative scale running from 0.7 (Francium) to 3.98 (Fluor).

The latter group, hydrophobic residues, do not have a charge in their side chain and are not able to make hydrogen bonds. The forces that act on these side chains are van der Waals forces. Charged and polar residues also interact through van der Waals forces. In Table 1, the different intermolecular forces are shown. Ionic lattices, such as salts (NaCl, MgO_2, Al_2O_3), have the most energy stored in their bonds. For covalent bonds, such as C−H and C−C, the energy in the bond is about 100 kcal/mol. Hydrogen bonds, one of the driving forces in protein stability, is third in Table 1 and the least strongest bond is the van der Waals bond, with less than (or equal to) 0.2 kcal/mol.

2.3 Zwitter ion

In nature, only a small part of all the reactions are a one way step. Consider the dissociation of acetic acid into acetate and hydronium: $CH_3COOH + H_2O \rightleftharpoons H_2NCH_2COO^- + H_3O^+$. This notation depicts that the chemical processes occur in both directions, however, with their own (possible

Table 1: The different intermolecular forces and their approximated values. For example table salt (NaCl) has a ionic lattice energy of 787 kJ/mol. The most common bond in protein, C−N, has an energy of 305 kJ/mol. All values depend on the participating atoms and context of those atoms within the molecule.

Force	Energy range (kcal/mol)
Ionic Lattice	$500 - 15000$
Covalent Bond	$35 - 500$
Hydrogen Bonds (dipole-dipole)	$1.9 - 6.9$
Van der Waals forces	≤ 0.2

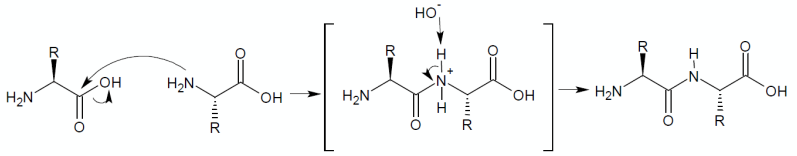


Figure 6: The formation of a peptide bond, starting with two amino acids. The free electrons of the amino group react with the carbon of the carboxylic acid, since it's negatively charged, because of the two electronegative oxygen atoms that pull away the electrons from the carbon atom. A hydroxide ion OH^- is separated because the double bonded oxygen binds more strongly to the carbon atom. The hydroxide ion then takes the proton bonded to the nitrogen, resulting in a dipeptide and 1 water molecule (not displayed).

different) speed. At equilibrium, both the reactions happen with the same speed and the acid dissociation constant, pK_a, can be measured: the equilibrium constant for a chemical reaction known as dissociation in the context of acid-base reactions and is written as:

$$pK_a = \log K_a = \log \frac{[A^-][H^+]}{[AH]} \tag{3}$$

where $[A^-]$ is the concentration of the base in M (molair = mol/litre), $[H^+]$ is the concentration of the hydrogen ions in M and $[AH]$ is the concentration of the acid in M. Functional groups also have a pK_a value, showing the dissociations of protons at certain positions in a molecule. If we look at Alanine, two functional groups are able to donate or accept an H^+: amino group (NH_2) and carboxyl group (COOH). For the NH_2 the pKA is 9.9. Using the HendersonHasselbalch equation

$$pH = pK_a + \log \frac{[A^-]}{[AH]}, \tag{4}$$

we can see that at $pH = 9.9$, the concentration of both the acid and base are the same, since $\log \frac{X}{X} = 0$, thus we can conclude that for pH values lower than 9.9 the majority of the amino groups in the molecule is present as NH_2. For the carboxyl group, the pKa in alanine is 2.3. With the same reasoning we can conclude that the majority of the carboxyl groups in the molecules will be protonated at $pH < 2.3$. In the regime between the two pKa values, one can see that the amino-group will most likely be protonated and carboxyl group will most likely be deprotonated, creating a zwitterion, where the head and tail of a (poly)peptide are charged, but the net charge is 0, as shown in Figure 7. Zwitterions are present in aqueous solution but sometimes also in solid state. In the cell, an aqueous solution, proteins form a zwitterion. For the protein, only the first and last residue need to be in the charged state.

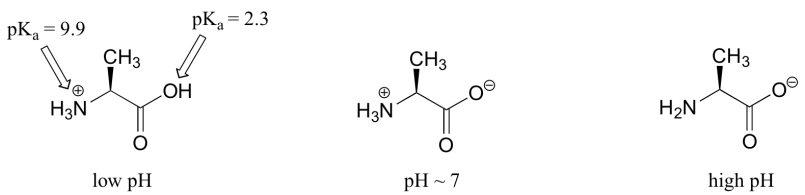


Figure 7: Three different configurations of alanine, showing the zwitterion in the middle.

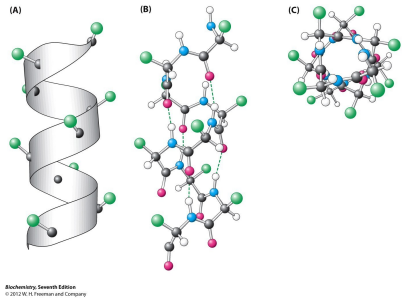


Figure 8: Abstractions of an alpha helix. (A) shows the ribbon diagram of the helix, (B) shows the hydrogen bonds between the carboxylic oxygen and the hydrogen on the nitrogen. The top view can be seen in (C).

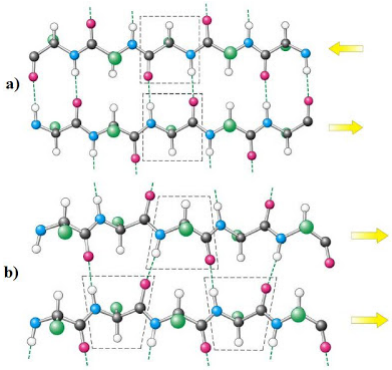


Figure 9: Abstractions of an beta sheet showing an anti-parallel beta sheet in (a) and a parallel beta sheet in (b) with the dotted lines depicting the hydrogen bonds.

2.4 Secondary structure

The next level of structure is the *secondary structure*. Amino acids that are close together in the sequence can interact with each other to form local structures. This secondary structures are thermodynamically very stable and are primarily formed due to hydrogen bonding (electrostatic forces), but also the weaker van der Waals forces. Two structures mainly occur: the alpha helix and beta sheet. The alpha helix is a helical structure where the oxygen of the the nth residue forms a hydrogen bond with the hydrogen of the $n + X$th residue, where X can be 3, 4, or 5, as shown in Figure 8. Notice that this is a very stable structure for the number of hydrogen bonds is large compared to other structures. The beta sheet is a structure where 'sheets' of residues are positioned next to each other, either parallel of anti-parallel, as shown in Figure 9. Again, the number of hydrogen bonds is high, with two hydrogen bonds per residue, creating a stable molecule.

2.4.1 DSSP

Following the DSSP[14] program (Determine Secondary Structure Program) 8 different secondary structures can been identified:

- 3-Turn helix (G)
- 4-Turn helix (H)
- 5-Turn helix (I)
- Hydrogen bonded turn (T)
- Extended strand in parallel and/or anti-parallel β-sheet conformation (E)
- Residue in isolated β-bridge (single pair β-sheet hydrogen bond formation) (B)
- Bend (the only non-hydrogen-bond based assignment) (S)
- Coil (residues which are not in any of the above conformations) (C)

These eight types are usually grouped into three larger classes: helix (G, H and I), strand (E and B) and coil (all others). The DSSP program calculates the most likely secondary structure assignment,

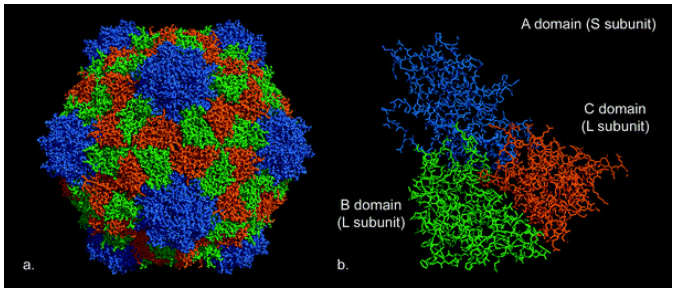


Figure 10: The structure of the Cowpea mosaic virus (CPMV) capsid. CPMV capsid (a) and the asymmetric unit (b). The capsid of CPMV is comprised of the small (S) and large (L) subunit. The A domain is shown in blue, the B domain in green and the C domain in orange.[16]

given the 3D structure of a protein. This is done by reading the position of the atoms in a protein (the ATOM records in a PDB-file) followed by calculation of the H-bond energy between all atoms. The algorithm will discard any hydrogen atoms present in the input structure and calculates the optimal hydrogen positions by placing them at 1.000 Å from the backbone nitrogen in the opposite direction from the backbone C=O bond. Subsequently, the strength of all possible hydrogen bonds are determined, by calculating the electro-statical forces for every possible hydrogen bond: ON, CH, OH and CN. By placing a charge on all the C=O pairs ($+q_1$ on C, $-q_1$ on O) and NH pairs ($-q_2$ on N, $+q_2$ on H) the electro-statical force can be calculated using Coulomb's law for biochemical units[14]:

$$E(r) = \frac{q_1 q_2}{r} \cdot f \tag{5}$$

$$E = q_1 q_2 \left(\frac{1}{ON} + \frac{1}{CH} - \frac{1}{OH} - \frac{1}{CN} \right) \cdot f \tag{6}$$

$$= 0.084 \left(\frac{1}{ON} + \frac{1}{CH} - \frac{1}{OH} - \frac{1}{CN} \right) \cdot 332 \tag{7}$$

where $q_1 = 0.42$, $q_2 = 0.2$, f is the dimensional factor and E is in kcal/mol. The best two hydrogen bonds for each atom are then used to determine the most likely class of secondary structure for each residue in the protein.

2.5 Tertiary & Quaternary Structure

The *tertiary structure* is the folding of the total chain by combining the structures of the secondary structure linked by turns and loops. The secondary structures are kept in place by the hydrogen bonding and van der Waals force as well as covalent bonding caused by disulphide bonds. These disulphide bonds are due to the fact that two cysteine residues that are close together in the tertiary structure are able to interact with one another. The side group of cysteine, CH_2SH, interacts with the nearby cysteine removing the two protons. The result are two connected residues through the two sulphur atoms: $B{-}CH_2S{-}SCH_2{-}B$, where B is the backbone of the protein. The result is a fully folded structure of amino acids that are connected to each other through covalent bonding. Such a protein sub-unit can be part of a multi sub-unit complex, where multiple domains sub-units are put together to form the *quaternary structure*, the organisation of sub-units into a complex. For example the oligomeric complex consist of a multiple of the same sub-unit and oligomeric complex consisting of 4 parts is called a tetramer. But also differing sub-units will coexist in these complexes, for example the well studied ribosome, which consist of the 21 sub-units [15]. A capsid layer, a membrane layer around a virus, is also complex quaternary structure, where a few monomers are docked together, to form an entire wall, to protect the virus, as can be seen in Figure 10.

2.6 PDB

The Protein Data Bank (PDB) is a repository for the experimentally-determined three-dimensional structural data of large biological molecules, such as proteins, nucleic acids, and complex assemblies with 103199 structures as of 12 September 2014. Since PDB is a member of the wwPDB, the RCSB PDB curates and annotates PDB data according to agreed upon standards, resulting in high quality information. The database, which is filled by biologists and biochemists from all over the world, typically contains X-ray crystallography and NMR spectroscopy data. Although NMR has the ability

to detect all the atoms, instead of only non-hydrogen by X-ray, and NMR can include spatial constraints, X-ray is often the method of choice for its high resolution and thus better accuracy of the coordinates of individual atoms. PDB supports 3 different formats: PDB, mmCIF and XML. Every structure, identified by 4 alphanumerical characters, is stored in these three formats. As a standard in the field of bioinformatics, 1-dimensional sequences are stored in the fasta-format.

2.6.1 PDB-format

The PDB-format is a textual format storing the information of a biological molecule and has been the standard file format for the PDB since its establishment in 1973. Being a very human readable format, it consist of lines of 80 characters. Every line starts with a key word and the following characters until the $80th$ are fixed where a regular expression can be used ideally to fetch different fields. However, a good C++ parser for PDB is not known. The PDB file format will be phased out in 2016.

2.6.2 mmCIF-format

The PDB also uses the macromolecular Crystallographic Information File (mmCIF) format to describe the information content of PDB entries. mmCIF is a flexible and extensible tag-value format for representing biomolecules. The set of mmCIF tags, which determine the classes of information present in a given mmCIF format file, are defined in an mmCIF dictionary. The dictionary consolidates content from a variety of crystallographic dictionaries including: the IUCr Core, mmCIF, Image and symmetry dictionaries. Since the information is described by the dictionary, a very structured dataset arises. CIFPARSE-OBJ C++ Parser is a C++ based parser, able to read any mmCIF-formatted file. PDBx/mmCIF has become the standard PDB archive format from 2014.

2.6.3 Fasta-format

The fasta-format is de standard format for representing nucleotide (DNA/RNA) and peptide sequences (proteins). It is a text-based format and has the following format:

```
;The fasta format encodes three different things
;Every line starting with a semicolon is a comment
>The sequence identifier
THE-ACTUAL-SEQUENCE
;Multiple sequences can be placed within a fasta file
;Both RNA and DNA, using the 1-letter-notation for both
>AB000263 |acc=AB000263|descr=Homo sapiens mRNA for prepro cortistatin like peptide, complete cds.|len=368
ACAAGATGCCATTGTCCCCCGGCCTCCTGCTGCTGCTGCTCTCCGGGGCCACGGCCACCGCTGCCCTGCC
CCTGGAGGGTGGCCCCACCGGCCGAGACAGCGAGCATATGCAGGAAGCGGCAGGAATAAGGAAAAGCAGC
CTCCTGACTTTCCTCGCTTGGTGGTTTGAGTGGACCTCCCAGGCCAGTGCCGGGCCCCTCATAGGAGAGG
AAGCTCGGGAGGTGGCCAGGCGGCAGGAAGGCGCACCCCCCCAGCAATCCGCGCGCCGGGACAGAATGCC
CTGCAGGAACTTCTTCTGGAAGACCTTCTCCTCCTGCAAATAAAACCTCACCCATGAATGCTCACGCAAG
TTTAATTACAGACCTGAA
>DROME_HH_Q02936
MRHIAHTQRCLSRLTSLVALLLIVLPMVFSPAHSCGPGRGLGRHRARNLY
PLVLKQTIPNLSEYTNSASGPLEGVIRRDSPKFKDLVPNYNRDILFRDEE
GTGADRLMSKRCKEKLNVLAYSVMNEWPGIRLLVTESWDEDYHHGQESLH
YEGRAVTIATSDRDQSKYGMLARLAVEAGFDWVSYVSRRHIYCSVKSDSS
ISSHVHGCFTPESTALLESGVRKPLGELSIGDRVLSMTANGQAVYSEVIL
FMDRNLEQMQNFVQLHTDGGAVLTVTPAHLVSVWQPESQKLTFVFADRIE
EKNQVLVRDVETGELRPQRVVKVGSVRSKGVVAPLTREGTIVVNSVAASC
YAVINSQSLAHWGLAPMRLLSTLEAWLPAKEQLHSSPKVVSSAQQQNGIH
WYANALYKVKDYVLPQSWRHD
```

In the above fasta-file, three sequences are stored: 'The sequence identifier', 'AB000263', an mRNA sequence from the species Homo sapiens and 'DROME_HH_Q02936', a protein sequence of Protein hedgehog of the species Drosophila melanogaster, also known as the fruit fly.

2.7 Dihedral Angles

There are multiple ways to describe the structure of a protein and with this, the location of all the atoms in the protein. An easy way is to use the X, Y and Z coordinates of all the atoms. With this method no constraints are implicitly enforced, since it's a general one. A more specific method is the description through dihedral angles, also torsion angles. With dihedral angles, the structure of a

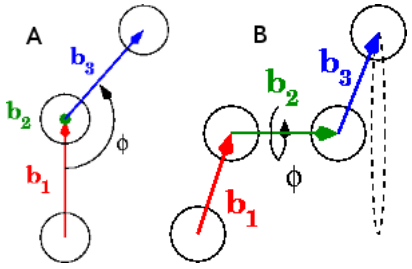


Figure 11: The dihedral angle, the angle ϕ between plane $P_{b_1 b_2}$ defined by vectors b_1 and b_2 and the plane $P_{b_2 b_3}$ defined by vectors b_2 and b_3. A) shows the front of the 4 atoms, B) shows the same situation, but from the side.

molecule can be defined between three successive chemical (covalent) bond vectors, shown in Figure 11. If we look at Figure 11A, we have three covalent bonds: b_1, b_2 and b_3. Now, take planes $P_1 = P_{b_1 b_2}$ and $P_2 = P_{b_2 b_3}$. The dihedral angle ϕ for b_1, b_2 and b_3 is the angle between P_1 and P_2, showing the range of the dihedral angle is $-180^\circ \leq \phi \leq 180^\circ$, where $\phi = -180^\circ \iff \phi = 180^\circ$. The backbone of a protein consist of three different torsion angles: C^{n-1}–N^{n}–C_α^{n}–C^{n} called ϕ, N^{n}–C_α^{n}–C^{n}–N^{n+1} called ψ and C_α^{n}–C^{n}–N^{n+1}–C_α^{n+1} called ω, where n is the position of the residue in the primary sequence. Since the PDB-format only gives us coordinates, the following scheme is used to calculate the torsion angle: Let b_1 be the vector from C^{n-1} to N^{n}, b_2 the vector from N^{n} to C_α^{n} and b_3 the vector from C_α^{n} to C^{n}. Torsion angle ϕ can be calculated using Equation 8.

$$\phi = \operatorname{atan2}\left((b_1 \times b_2) \times (b_2 \times b_3) \cdot \frac{b_2}{|b_2|}, (b_1 \times b_2) \cdot (b_2 \times b_3)\right) \tag{8}$$

$$\operatorname{atan2}(y,x) = \begin{cases} \arctan\frac{y}{x} & x > 0 \\ \arctan\frac{y}{x} + \pi & y \geq 0, x < 0 \\ \arctan\frac{y}{x} - \pi & y < 0, x < 0 \\ +\frac{\pi}{2} & y > 0, x = 0 \\ -\frac{\pi}{2} & y < 0, x = 0 \\ \text{undefined} & y = 0, x = 0 \end{cases} \tag{9}$$

The covalent bond between C and N is able to become a double bond leaving the carboxyl oxygen single bonded, as shown in Figure 14. With a single covalent C−N bond, typically the torsion is freely rotatable and all torsion angles are possible. With a double bond, all atoms are forced into one plane, making the torsion angle practically 180°, which is also the case for the ω torsion angle. Because of this double bond potential, the ω-bond will most likely be in a superstate between the two states, which forces the atoms, to a certain degree, into a plain, resulting in torsion angles around the 180°.

2.7.1 Conservation of dihedral angles

Apart from a good representation, dihedral angles carry extra information. This can be shown in two ways. The first is the Ramachandran plot, which is a visualization of the dihedral angles of the backbone in proteins. The 2-dimensional plot shows the ϕ-ψ pair of the amino acid residues, shown in Figure 12. It can be seen that the allowed values are condensed, showing allowed regions of ϕ-ψ pairs. Note that dihedral angle values are circular and 180° is the same as −180°, the edges of the Ramachandran plot wrap right-to-left and bottom-to-top. The second way dihedral angles contain extra information is the conservation of dihedral angles in polypeptide fragments. Shown in Figure 13, The possible dihedral angle values for 4 fragments are displayed. For the fragment `ALA`, possible value range from −180° to +180°. However, the probability that a ϕ−angle is in the range from −70° to −30° is more probable than from 20° to 120°, showing a tendency for fragments with the length of 1. For bigger fragments, not all possible values are registered and the tendency towards certain values is bigger. For all the fragments with pattern `THR-GLY-ALA-ASP-ASP`, 74, 1% of the ϕ angles is within the range 120° − 130°. `PITA` implements this by favouring dihedral angles of fragments with larger neighbourhoods over smaller ones.

2.8 Energy Calculation

In nature, all molecules have an internal energy. Although this energy cannot be measured directly, there are techniques to estimate these values by energy functions. All energy functions consider

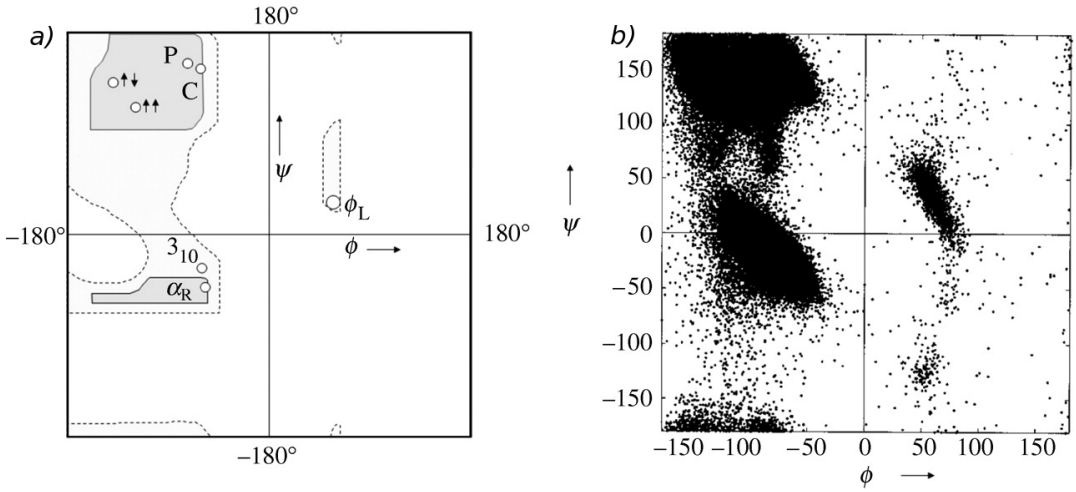


Figure 12: (a)Ramachandran map showing sterically favoured regions for L-alanine. The regions corresponding to the important secondary structures are marked α_R (right-handed α-helix), ϕ_L (left-handed α-helix), 3_{10} (3_{10}-helix), two upward arrows (parallel β-sheet), left up, right down arrows (anti-parallel β-sheet), P(polyproline II), C(collagen triple helix). (b) Scatter plot of 104718 non-Gly residues in protein crystal structures (less than or equal to 2.0Å) showing the distribution of the backbone dihedral angles ϕ and ψ[17]. The group of data points around $\phi = 50$ and $\psi = 120$ represents the nucleophilic elbow.

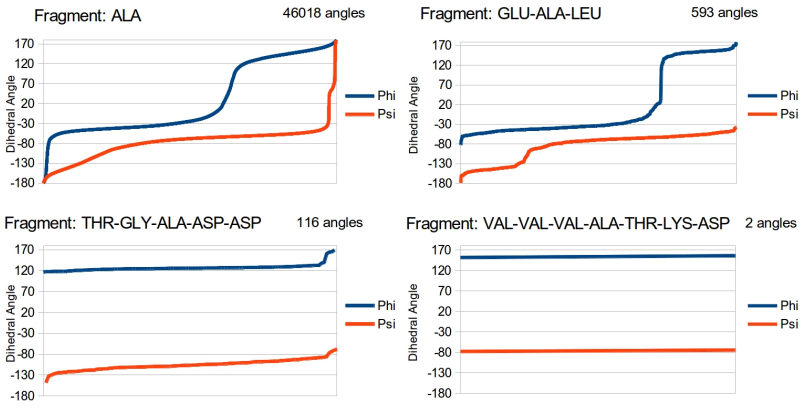


Figure 13: Possible phi (ϕ) and psi(ψ) dihedral angles for 4 different fragment sizes, with sizes $1, 3, 5$ and 7, sorted, from a data set of 104 different structures. The phi and psi curves are not correlated and all data points are sorted for their own set. For example, about 60% of the 46018 psi angles of ALA fragments are between $-80°$ and approximately $-40°$. Of the 46018 phi angles of ALA fragments are between $-80°$ and $-30°$.

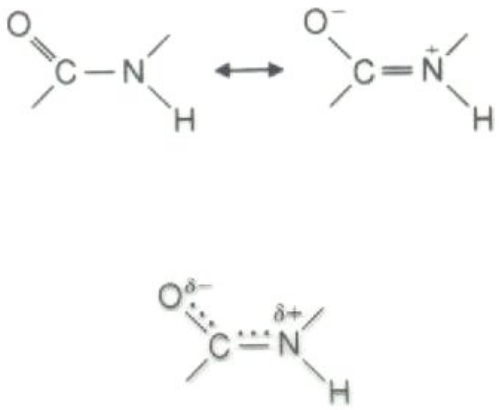


Figure 14: The double bond characteristic of the ω torsion angle, showing the normal state at the left, its charged state at the right and the transition state at the bottom. Since both (left and right) states occur, the transition state can be seen as the average state the bond.

a molecule as a collection of particles, held together by some sort of elastic forces. Many energy functions are based on (part of) the following principle[10][12].

$$E_{Total} = E_{bond} + E_{angle} + E_{dihedral} + E_{vdW} + E_{Coulomb}. \tag{10}$$

The total energy, E_{Total}, is a summation of 5 different energies:

- E_{bond}, is the term which represents the energy stored in the covalent bond. It is often compared as spring. The energy stored in this bond is estimated by:

$$E_{bond} = \sum_{\text{bond } i} k_i^{\text{bond}} (r_i - r_{0i})^2,$$

where k_i^{bond} is the bond energy, r_i the distance between the two atoms and r_{0i} the ideal distance between the two atoms.

- E_{angle}, is the energy caused by bending of the angles between each pair of covalent bonds sharing a single atom at the vertex. The modelling of the energy for this term is similar to the covalent bond energy:

$$E_{angle} = \sum_{\text{angle } i} k_i^{\text{angle}} (\theta_i - \theta_{0i})^2, \tag{11}$$

where k_i^{angle} is the angle energy, r_i the angle between the two covalent bonds and r_{0i} the average angle for those covalent bonds.

- $E_{dihedral}$, can be split into two parts: *Proper energy* and *Improper energy*. The proper energy is the energy from the torsion angle. It describes the force created by atom pairs separated by exactly three covalent bonds with the central bond, subject to the torsion angle ϕ, shown in Figure 15C. Energy from the out-of-plane bending is modelled by the improper angle. The angle is defined for the four atoms a, b, c, d, for which the central atom a is bonded to b, c, d, as the angle between the planes P_{abc} and P_{bcd}, shown in 15. The dihedral energy is formulated as:

$$E_{dihedral} = \sum_{\text{dihedral } i} \sum_{n_i} \begin{cases} k_i^{\text{prop}} [1 + cos(n\phi - \phi_0))] & n_i \neq 0 \\ k_i^{\text{impr}} (\alpha_i - \alpha_0)^2 & n_i = 0 \end{cases} \tag{12}$$

where n is the periodicity of the rotational barrier and k_i^{prop} is the associated barrier height. k_i^{impr} determines the "stiffness" of the potential in, α_i is the angle the two planes as is shown in Figure 15D and α_0 is the equilibrium value.

- E_{vdW} is the energy as a result of the van der Waals force. This force is a sum of the attractive and repulsive force between molecules and internal atoms other then covalently bonded atoms, electrostatic interactions and ionic bonds. This force is estimated by the Lennard-Jones potential:

$$E_{vdW} = \sum_i \sum_j 4\varepsilon \left[\left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^{6} \right] \qquad 1 \leq i < j \leq N, \tag{13}$$

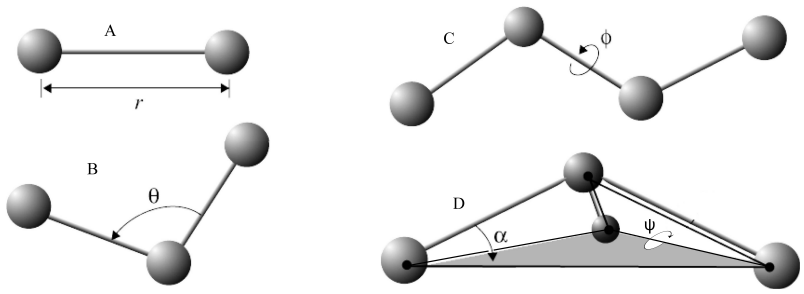


Figure 15: Representation of 4 of the 5 forces used for energy estimation. A depicts the covalent bond, represented as a stretching string. B shows the energy stored in the angles between covalent bonds. C shows the dihedral energy and D the improper energy.

where ε is the depth of the potential well, σ is the distance at which $E_{vdw}(r) = -\varepsilon$, r is the distance between the two atoms, i and j are atoms and N is the number of atoms in the structure.

- $E_{Coulumb}$, accounts for the static electrical energy. It is modelled based on Coulomb's law:

$$\sum_i \sum_j \frac{q_i q_j}{4\pi\epsilon_0 r_{ij}} \qquad 1 \leq i < j \leq N, \tag{14}$$

where q_x is the electric charge on atom x, ϵ_0 the permittivity of free space in $C^2 m^{-2} N^{-1}$.

For every particle in a given context of bonds (e.g. covalent, hydrogen, etc.), the parameters for the interactions given in Equations 11-14 are laid out in force field parameter files. The determination of these parameters is a significant undertaking generally accomplished through a combination of empirical techniques and quantum mechanical calculations. Two well established force fields are CHARMM and AMBER.

2.8.1 Steric Hindrance

The energy calculation has multiple objectives. One of the reasons is to exclude (intermediary) structures from the search. If the energy of a molecule is relatively higher, the probability that a molecule will exist in that form in a living cell will drop. For very high energies, the structures will be too improbable and these candidates can be excluded from the list of potential molecules. On the other hand, if the energy of an (intermediary) structure is low, it could lead to a final conformation with a overall low energy. The energy of a molecule will increase considerably, if two non-covalently bonded atoms are very close, since each atom (or molecule) occupies a certain amount of space. This effect is known as steric effects, of steric hindrance: the overlapping of electron clouds causing a repulsive force. Since the energy calculation is a time consuming effort, a pre-check for steric hindrance is favourable. This is done by looking for an overlap of the electron clouds for all the atom pairs, given their radii [18][19]. If an overlap of the clouds is detected, the molecule is will most probably not exist in that state in a living cell and will be eliminated. To introduce some tolerance, the radii can be multiplied by a factor x. In this way high energy intermediaries can still fold to low energy proteins.

2.9 Minimal Bounding Sphere

The structure of a protein is often very compact and proteins do not take up much space. In other words, if an intermediary structure is not compact, the probability that such a structure exists in a living cell is small. This is the basis for the pruning technique based on a bounding sphere. The bounding sphere is the 3 dimensional equivalent of the smallest circle problem. Thus, what is the smallest sphere that contains all of a given set of points in the Euclidean space. An easy way would be to search for the longest coordinates pair distance and use the middle of these coordinates as the center and the distance as the diameter. This algorithm, however, has complexity $\mathcal{O}(n^2)$, which, for large sets of coordinates, will take too much computational time. Therefore, 3 methods will be discussed: Ritter's bounding sphere, Bouncing bubble and an own implementation of the Bouncing bubble, all with better complexity than $\mathcal{O}(n^2)$, which will be explained in the next 3 sections.

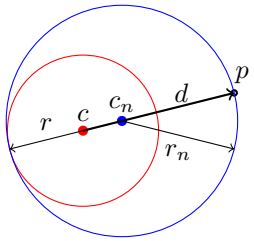


Figure 16: The construction of an enclosing circle. Starting point is circle C (red) with center c and radius r, and point P, not enclosed by C. Using Equations 15-17 the new circle C_n (blue) can be constructed with center c_n and radius r_n.

2.9.1 Ritter's bounding sphere

Known for its simplicity, Ritter's bounding sphere is widely used in various applications. The algorithm starts with the initialization of the circle. Starting with a set of points P, this is done in two steps:

1. Pick a point $x \in P$, search a point $y \in P$, which has the largest distance from x ($\mathcal{O}(n)$)
2. Search a point $z \in P$, which has the largest distance from y ($\mathcal{O}(n)$)

The center of the starting sphere is the middle between y and z ($\frac{y+z}{2}$), the radius is half the distance from y to z ($|\frac{y-z}{2}|$). From this starting point, we propagate through all points checking whether the point is within our sphere or not. If the point is within the sphere, the sphere still encloses all propagated points and nothing is done. If it's not, a new sphere is constructed using the following approach:

$$d = |\overrightarrow{p} - \overrightarrow{c}| \tag{15}$$

$$r_n = \frac{d+r}{2} \tag{16}$$

$$\overrightarrow{c_n} = \overrightarrow{c} + \frac{r_n}{d} \cdot (\overrightarrow{c} - \overrightarrow{p}), \tag{17}$$

where $\overrightarrow{p}$ is the unenclosed point, $\overrightarrow{c}$ the center of the circle with radius r, $\overrightarrow{c_n}$ the center of the new enclosing circle and its radius r_n. This creating the smallest sphere that encloses the unenclosed point and the old sphere. The construction is shown in Figure 16 for the 2 dimensional problem. To calculate the complexity of this algorithm, we add the the complexity of the different steps: the search for point $y \in P$, the search for point $z \in P$ and the propagation through all the points ($\mathcal{O}(n)$), resulting in complexity $\mathcal{O}(3n) \approx \mathcal{O}(n)$. Note that this algorithm does not create the smallest enclosing sphere, only an approximation.

2.9.2 Bouncing bubble

The approximations of Ritter's algorithm are dependent on the initialization: a different initialization will result in a different enclosing sphere. *Bouncing bubble* tries to find a better enclosing sphere by starting with a better initialization than Ritter's initialization. This is done by randomly picking two points constructing the smallest enclosing sphere for those points and propagating through all the points, checking if the point is within the sphere. Only if the point is outside the sphere, a new (not enclosing) sphere is constructed using the following method:

$$\alpha = \frac{d}{r} \tag{18}$$

$$r_n = 0.5r \cdot \frac{1+\alpha}{\alpha} \tag{19}$$

$$\overrightarrow{c_n} = 0.5 \cdot ((1 + \frac{1}{\alpha^2}) \cdot \overrightarrow{c} + (1 - \frac{1}{\alpha^2}) \cdot \overrightarrow{p}), \tag{20}$$

where $\overrightarrow{p}$ is the unenclosed point, $\overrightarrow{c}$ the center of the circle with radius r, $\overrightarrow{c_n}$ the center of the new enclosing circle and its radius r_n. This initial procedure is done twice. After the initialization, one propagation through all the points using the same method as Ritter's algorithm is performed. Since the initialization propagates twice through all the points and Equations 15-17 add a third propagation, the complexity of this algorithm is $\mathcal{O}(3n)$, the same as Ritter's algorithm.

2.9.3 Bouncing bubble with center of mass

As a third algorithm to find an approximation of the enclosing circle we created our own algorithm, based on the bounding bubble algorithm and the *geometric center*. As stated, Ritter's algorithm's initialization influences the outcome of the algorithm. In addition to the bouncing bubble initialization, the geometric center is introduced, which is defined as the arithmetic mean of all the points in P, in Euclidean space:

$$p_{gc} = \frac{1}{N}\sum_{i}^{N} p_i \qquad\qquad p_i \in P \tag{21}$$

where p_{gc} is the position of the geometric mean and N is the number of points. Point p_{gc} is the center of the sphere with the radius being the distance to a randomly picked point $p \in P$. The sphere is used as a starting sphere for one propagation of the bouncing bubble initialization. Subsequently, Ritter's propagation is used to enclose all points. The calculation of the geometric mean ($\mathcal{O}(n)$), 1 bouncing bubble initialization propagation and 1 Ritter's propagation, results in a complexity of $\mathcal{O}(3n)$.

2.10 Distance functions

In order to evaluate a prediction method, the result of the prediction can be compared with the test set. Comparing two structures can be done on multiple ways. Here 3 ways will be described: *RMSD*, *LGA* and *DSSP*.

2.10.1 RMSD

One of the most used methods for comparing two structures is the Root Mean Squared Distance or RMSD. In this method, the average distance between the atoms is calculated between the N pairs of equivalent atoms in both structures, where N is the number of atoms in the biopolymer. Since this measure in dependent on the rotation of the structures, a superposition which minimizes the RMSD is first performed. Given two structures $\mathbf{a}$ and $\mathbf{b}$, the RMSD is defined as:

$$\mathrm{RMSD}(\mathbf{a},\mathbf{b}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\|a_i - b_i\|^2} \tag{22}$$

$$= \sqrt{\frac{1}{N}\sum_{i=1}^{N}((a_{ix}-b_{ix})^2 + (a_{iy}-b_{iy})^2 + (a_{iz}-b_{iz})^2)}. \tag{23}$$

The RMSD can in principle be applied on all atoms, but the RMSD of the C_α only is used very often. Although the RMSD is known as a good and straightforward method to compare structures, it is very sensitive to small changes. A small perturbation in just one part of the protein (e.g. in a hinge joining two domains) can create a large RMSD and it would seem that the two structures are very different overall. Thus, it is desirable to also consider local regions of the proteins in assessing their similarity. In essence, the smaller such deviant regions, the more similar the two structures are.

2.10.2 LGA

LGA (Local-Global Alignment) is a method designed to facilitate the comparison of protein structures or fragments of protein structures in sequence dependent and sequence independent modes[20]. It incorporates both local and global similarity of the structures, by using two different distance functions: Longest Continuous Segment (LCS) and Global Distance Test (GDT). The LCS procedure is able to localize and superimpose the longest segments of residues. The GDT algorithm is designed to complement evaluations made with LCS searching for the largest (not necessary continuous) set of 'equivalent' residues that deviate by no more than a specified distance cut-off. The LCS is the longest

Table 2: Example of data generated by LCS and GDT analyses

Column #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
Cut-offs:					1Å	2Å	5Å	0.5Å	1.0Å	1.5Å	2.0Å	2.5Å	3.0Å	3.5Å	4.0Å	4.5Å	...
LCS_GDT	Molecule 1		Molecule 2		Length of the												
LCS_GDT	Residue		Residue		continuous												
LCS_GDT	Name	Number	Name	Number	Segment			Global distance test data									
LCS_GDT	V	40	A	29	23	26	90	10	18	22	23	24	24	27	33	49	...
LCS_GDT	A	41	Q	30	23	26	90	10	18	22	23	24	25	27	42	55	...
LCS_GDT	L	42	L	31	23	26	90	4	7	20	23	24	25	36	46	55	...
LCS_GDT	E	43	E	32	8	26	90	4	7	15	23	24	25	35	46	55	...
LCS_GDT	Q	44	V	33	8	26	90	4	6	9	18	24	26	37	46	55	...
LCS_GDT	T	45	T	34	8	26	90	4	7	9	13	22	25	36	46	55	...
LCS_GDT	G	46	G	35	8	14	90	3	7	9	12	17	22	35	46	55	...

continuous segment (in terms of C_α atoms) which can be found under a selected RMSD, shown in Equation 24:

$$\mathrm{LCS}(\mathbf{a},\mathbf{b},x,r_c) = \max\left(\{1+j-i \mid \mathrm{RMSD}(\mathbf{a}_{i..j},\mathbf{b}_{i..j}) < r_c \wedge i \le x \le j \quad\}\right), 1 \le i < j \le N \tag{24}$$

where $\mathbf{a}$ and $\mathbf{b}$ are the input structures, x is the residue for which the LCS is determined, r_c is the cut-off RMSD and N is the sequence length. For all n C_α atoms is checked to which longest continuous segment (possible continuous segments is $\frac{n^2+n}{2}$) it belongs. This is done for RMSDs 1Å, 2Å and 5Å, as shown in Table 2.

For example, residue L-31 from Molecule 2 is a member of a 23-residue long continuous segment that can be superimposed with corresponding residues from Molecule 1 under a 1Å RMSD cut-off, but residue E-32 is an element of a segment consisting of just 8 residues at an RMSD cut-off of 1Å.

The GDT is used to collect the global similarity between the two structures. This is done by looking, for every C_α atom, for the biggest set of C_α atoms it belongs to, provided is can be fit under a selected RMSD. Given two structures $\mathbf{a}$ and $\mathbf{b}$, position of the C_α in the structure, x, and the cut-off RMSD, r_c, GDT is defined as:

$$\mathrm{GDT}(\mathbf{a},\mathbf{b},x,r_c) = \max(\{|u| \mid \mathrm{mRMSD}(u,v) < r_c \wedge\ u \subseteq \mathbf{a} \wedge \mathbf{a}_x \in u \wedge v = \mathrm{Eq}(\mathbf{b},u)\}), u \supset \mathbf{a}, \tag{25}$$

where $\mathrm{Eq}(\mathbf{b},u)$ returns the equivalent set of C_α atoms of u in $\mathbf{b}$ and $\mathrm{mRMSD}(u,v)$ returns the minimum RMSD of all the equivalent C_α-pairs in u and v, after the RMSD superposition. Since it is infeasible to calculate the GDT for all the possible subsets of $\mathbf{a}$ (the number of subsets is 2^n, where n is the sequence length), the search for the optimal superposition needs to be approximated. This is done using an iterative method. For each three-residue long segments (not per se continuous) the RMSD and superposition is calculated and the superpositions of all the LCS are combined to get the set of initial superpositions. Each calculated superposition is used as a starting point to give an initial list of equivalent residues (C_α atom pairs from $\mathbf{a}$ and $\mathbf{b}$). The list of such equivalences is iteratively extended to collect the largest set of residues that can fit under a given distance cut-off. To extend the set, the following procedure is applied on all the initial superpositions:

1. Obtain the transform
2. Apply the transform
3. Identify all atom pairs for which the distance is larger than the threshold
4. Re-obtain the transform, excluding those atoms
5. Repeat steps 2 to 4, until the set of atoms used in calculations is the same for two cycles running.

The cut-off distances for the GDT are $0.5, 1.0, 1.5, \ldots, 10$, Using this method the GDT-table, as shown in Table 2, can be filled. After retrieving the LCS and GDT, de LGA can be calculated. The LGA for two structures **a** and **b** is defined as:

$$\overline{LCS}(\mathbf{a},\mathbf{b},x) = \frac{1}{N^2}\sum_{i=1}^{N} LCS(\mathbf{a},\mathbf{b},i,x), \tag{26}$$

$$\overline{GDT}(\mathbf{a},\mathbf{b},x) = \frac{1}{N^2}\sum_{i=1}^{N} GDT(\mathbf{a},\mathbf{b},i,x), \tag{27}$$

$$LCS_T(\mathbf{a},\mathbf{b}) = 2\cdot\frac{\sum_{i=1}^{k}(k-i+1)\cdot\overline{LCS}(\mathbf{a},\mathbf{b},r_i^1)}{k(k+1)}, k=|r^1|, \tag{28}$$

$$GDT_T(\mathbf{a},\mathbf{b}) = 2\cdot\frac{\sum_{i=1}^{k}(k-i+1)\cdot\overline{GDT}(\mathbf{a},\mathbf{b},r_i^2)}{k(k+1)}, k=|r^2|, \tag{29}$$

$$LGA(\mathbf{a},\mathbf{b}) = w\cdot GDT_T(\mathbf{a},\mathbf{b}) + (1-w)\cdot LCS_T(\mathbf{a},\mathbf{b}), \tag{30}$$

$$r^1 = \{1,2,5\}, \tag{31}$$

$$r^2 = \{0.1, 1.0, 1.5, \ldots, 10.0\}, \tag{32}$$

where $\overline{LCS}(\mathbf{a},\mathbf{b},x)$ is the average percent of residues in target that can fit under an RMSD cut-off x is the average LCS for structures **a** and **b**, N the number of residues, $\overline{GDT}(\mathbf{a},\mathbf{b},x)$ is an estimation of the average percent of residues of structures **a** and **b**, that can fit under the distance cut-off x, LCS_T is the weighted average of $\overline{LCS}$ and GDT_T is the weighted average of $\overline{GDT}$. r^1 and r^2, the radii for the LCS and GDT, are by default filled with the values presented above, but can be changed.

The LGA method combines these two functions resulting in a 'best' superposition and locating structural similarity on a more local level, where one can see the structural similarity on a 2D manner.

2.10.3 CSS

Although local structure similarity can be measured with the LCS-function, the secondary structure of the protein is also a good identifier for comparison of two structures and thus evaluation of a protein structure predictor. Using the DSSP-algorithm, the secondary structure of both the target and prediction can be determined, in order to asses the predictor. The result is two one-dimensional sequences indicating the secondary structure for every residue. The score CSS (CompareSS) can be calculated using the following formula:

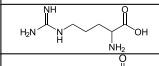
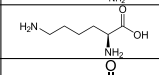
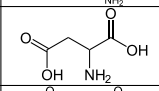
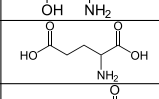
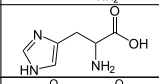
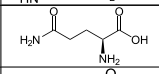
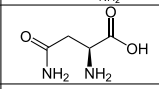
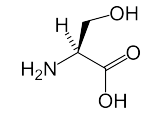
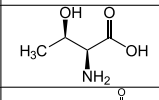
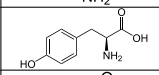
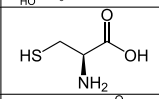
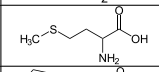
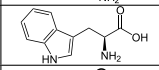
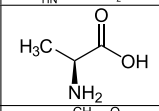
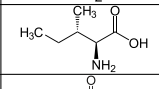
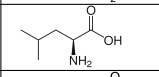
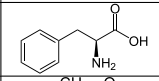
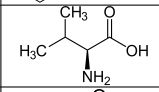
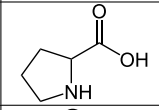
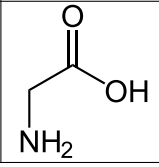
$$CSS(s^1,s^2) = \frac{1}{N}\sum_{i=1}^{N}\begin{cases} 1 & \text{if } s_i^1 = s_i^2 \wedge s_i^1 \neq \emptyset \wedge s_i^2 \neq \emptyset \\ 0 & \text{otherwise} \end{cases}, \tag{33}$$

where s^1 and s^2 are the secondary structure sequence of target and the prediction models, respectively and N is the number of residues with secondary structure assigned to them, in the first string. If no secondary structure is assigned by the DSSP-program, ($\emptyset$, coil), no comparison can be done and the residue is skipped. Note that two identical strings of no secondary structure has a score of 0%. This distance function is not per se symmetric ($CSS(\mathbf{a},\mathbf{b}) = x \not\Longrightarrow CSS(\mathbf{b},\mathbf{a}) = x$) and the target should always be the first parameter. The CSS is not used in `PITA` and is only used to asses it.

3 Computational Protein Prediction: Problem description

The basic idea of computational protein folding is to find a minimal free-energy of the three dimensional structure of the protein following the rules of nature, given a one dimensional sequence of amino acids, by searching through the immensely large search conformational space of the protein. While this still is the best way to get the actual structure of proteins, the complexity and enormity of the search space do not allow us to do so: current computing resources are insufficient and current forcefield descriptors are inadequate. As an alternative, the problem can be approximated by other techniques. Each of these techniques will be explained in the next section.

Table 3: The 20 standard amino acids, divided into three groups, showing their entire name, 3-letter notation, 1-letter notation and chemical structure.

Category	Amino acid name	three-letter notation	1-letter notation	Chemical structure
Charged	Arginine	Arg	R	
	Lysine	Lys	K	
	Aspartic acid	Asp	D	
	Glutamic acid	Glu	E	
	Histidine	His	H	
Polar	Glutamine	Gln	Q	
	Asparagine	Asn	N	
	Serine	Ser	S	
	Threonine	Thr	T	
	Tyrosine	Tyr	Y	
	Cysteine	Cys	C	
	Methionine	Met	M	
	Tryptophan	Trp	W	
Hydrophobic	Alanine	Ala	A	
	Isoleucine	Ile	I	
	Leucine	Leu	L	
	Phenylalanine	Phe	F	
	Valine	Val	V	
	Proline	Pro	P	
	Glycine	Gly	G	

4 Current State of the Art

Protein structure prediction can be classified into 4 categories: (1) Comparative Modelling, (2) Fold Recognition, (3) First principles methods with database information, and (4) First principles methods without database information[21][22]. In this section, the state of the art in these categories will be explained.

4.1 Comparative modelling

Comparative modelling, also known as homology modelling, consists of the five stages[23]:

1. Identification of related sequences of known structures (using sequence and profile similarity)
2. Aligning of the target sequence to the template structures
3. Modelling of structurally conserved regions using the known templates
4. Modelling side chains and loops which are different than the templates
5. refining and evaluating the quality of the model through conformational sampling

For comparative modelling, as a rule of thumb, if the sequence similarity is above 50%, the predictions are of very good to high quality, whereas $30 - 50\%$ sequence similarity will have 80% of the target within 3.5Å of its true position, while the "twilight zone" (between 10% and 30% similarity) and the "midnight zone" ($< 10\%$ similarity) will have significant errors. The best way to search for homology in the first regime is using pairwise sequence alignment methods, whereas profile-sequence and profile-profile alignment tools are the best for the second regimes. For the last regimes ("twilight zone" and "midnight zone"), sequence-structure threading methods get the best result. For the model building (steps 3 and 4), there are three general groups[24]:

1. Modelling by assembly of rigid bodies
2. Modelling by segment matching or coordinate reconstruction
3. Modelling by satisfaction of spatial restraints, Homology-derived restraints, stereochemical restraints, optional restraints derived from experimental data

One of the well known modellers in this category is the Rosetta Modelling Suite which is a software suite usable for multiple goals. Rosetta is one of the leading predictors on CASP events. The basis has been created in 2004. Rosetta uses a torsion space representation, using perfect bond lengths and stretch angles. Although, it was initially developed for de novo protein structure prediction, multiple other programs have been created to guide other protein related problems, such as ligand docking.

Rosetta is based on fragment insertion: It uses 9 AAs fragments and 3 AAs fragments to fold the target. For every three and nine residue long fragment, a library of fragments defining the conformational space is selected, by comparison with known protein structures from a non-redundant database of X-ray structures. Using PSIBLAST on each window, a sequence similarity score is calculated. Together with the similar secondary structure, a ranked list is constructed for all the three and nine residue fragments, of length 200 for every window.

The monte carlo search begins with a fully extended conformation of the query and uses to following steps:

1. Random 9-residue windows are selected and a fragments from the top 25 of that window is picked. The torsion angles are copied and if the energy has decreased, the conformation is retained. If the energy increases, the move is picked based upon the Metropolis criterion. Each simulation uses 28000 9-residue insertions. The first iterations, only Van der Waals forces are included and stops until all torsion angles have been replaced.
2. The next 2000 iterations, secondary structure (SS) and all terms except those rewarding compactness are included, with strand pairing at 0.3 of its final weight.
3. The next 20000 iterations, the SS score increases to its final weight. During this stage, the SS potential alternates by evaluating interactions only 10 residues apart.
4. For the last 4000 iterations, the high resolution energy function is used, consisting of the sum of the Lennard-Jones potential together with a solvation approximation, the secondary structure potential, a knowledge-based electrostatics term and a knowledge based conformation dependent amino acid interval free energy term.
5. After this, 8000 3-residue Gunn-insertions are attempted

Since the initial fragment insertion can be too perturbing, 5 basic operators have been introduced if finer sampling is required:

1. Random angle perturbation is the alteration of either one angle-couple or one angle and a compensating rotation in de opposite direction of the preceding angle, creating a shear-motion
2. Selection of globally non-perturbing fragments, is the insertion of a fragment that is similar to the existing fragment and therefore will cause not too much global perturbation.
3. Torsion angle variation to offset global perturbation, is the insertion of a fragment and subsequently alter residue angles to wear off the global perturbation
4. Monte Carlo with minimization, is the greedy gradient descent minimization along the backbone torsion
5. Rapid side-chain optimization applies a simulated annealing techniques with a rotamer library. The number of proteins fold (approximately 20000-50000) is then reduced by clustering the RMSDs and refinement is used on the resulting conformations. The best models with the lowest energy will get atom refinement.

Still major challenges need to be tackled, for instance new folds, model selection and large protein prediction. Like Rosetta, PITA uses fragments, with the difference that Rosetta uses the entire structure of the neighbourhood, where PITA only uses the atoms of the middle residue.

4.2 Fold Recognition

In protein prediction, for now, comparative modelling search is the best way to predict a protein. However, if a protein does not have structural neighbours, this technique is not feasible for it cannot use structural equivalence to calculate the 3 dimensional structure. Fold recognition and threading come in handy when little comparison is found between the target and already known structures. The concept is based on the idea that the shapes of fragments are more limited rather than the sequence possibilities. Therefore only a few structures need to be modelled instead of a wide spread of all the theoretically possible folds. Two examples of threading algorithms are TASSER[25] and I-TASSER[26] (Iterative TASSER). TASSER (Threading ASSEmbler Refinement) is a threading program described in this article. It uses the thread database creator PROSPECTOR_3. The alignment with the target is done with Needleman-Wunsch.

4.3 First principles methods with database information

The algorithm PITA belongs to this category, taking dihedral angles from a constructed database and using an energy potential to distinct bad from good configurations. For this category of algorithms, many types of database information can be used, such as sequence dependent local interactions, common structures, centroids and secondary structures. Baker [27] studied the distributions of local structures based on short sequence segments of up to 10 residues based on the protein database, and developed effective approaches that compare fragments of a target to fragments of known structures. Once appropriate fragments have been identified, they are assembled to a structure, often with the aid of scoring functions and optimization algorithms. The course grained structure of the protein is created with this information, followed by refinement by first principle forces and potentials.

4.4 First principles methods without database information

Ab initio protein folding comes in handy when no sequence similarity is known, but it is seen as the most difficult of methods, since it uses purely physics to understand the folding pathway. For the potential physics-based and knowledge-based energy functions can be used. Although MD simulations with only physics based potentials are slow, MC simulations and genetic algorithms can be used instead, to reduce the number of computations. To reduce the number of potential solutions, conformational search methods are used to check for so called decoy structures. Because there is an abundant number of these structures, smart ways to tackle this problem must be used. Using the correct way to search the space and to rule out potential candidates, the number of calculations can be held down to a minimum. In 2013, a pure MD simulation was started for the deduction of the structure of the capsid layer of HIV-1. Using tomographic images and cryo-electron microscopy, the authors deduced the structure of a monomer of the capsid layer [28]. Using this information they built a model consisting of 71 hexamers (64 million atoms) into a predefined capsid reconstruction. Using NAMD2.9 as the MD package, they simulated 100ns at 2 fs time steps. Since all the atoms in the reconstruction stayed at their place, it was concluded that the reconstruction was at equilibrium and the deduced structure was correct.

4.5 Hybrid Algorithms

As stated earlier, Protein structure prediction presents a vast search space and since the analysis of each protein structure requires a significant amount of computing time, it is necessary to take advantage of high-performance parallel computing platforms as well as to define efficient search procedures to search the space of possible protein conformations. Hybrid techniques have been proposed to tackle these two problems. In [29], two known protein structure predictions algorithms are parallelized and compared to one another. Both algorithms use a cost function consisting of 2 or 3 terms: Bonded, non-bonded and the rest of non-bonded. The 2 algorithms are PAES (Pareto Archived Evolution Strategy) and NSGA-2 (Non dominated Sorting Genetic Algorithm 2). NSGA-2 is parallelized by calculating the cost for every individual separate. PAES, which uses only one candidate for its solution and iteratively created a new individual and chooses whether to keep the current parent as its solution, is parallelized by creating multiple children at every iteration, thus generating better solutions earlier. Although they both produce proteins with similar quality, PAES is less efficient with the parallelization.

As another example for hybrid algorithms Multi-objective Evolutionary Algorithm with Many Tables (MEAMT) are introduced [30], where an ab initio technique with database information is combined with evolutionary algorithms. It uses many tables to optimize the energy of a protein. The energy is described by an equation using 4 terms (Van der Waals, electrostatic charge, solvent energy and hydrogen bonds), thus 4 terms need to be optimized at the same time. MEAMT is an extension to MEAT (with only one table) which is an extension to MT (without the evolutionary algorithm). MEAMT (and MEAT) use a population of non-dominated (ND) individuals: Each individual is part of the Pareto front, described by only the dihedral angles of the backbone and the side-chain angles. Using multiple tables (tables for the four individual and part of that power set) the search space is searched efficiently. This results in good prediction compared to GAPFCG I-PAES and Quark (although Quark seems to be superior on average).

On Lattice prediction methods, reduce their search space to quantified coordinates in space [31]. This article uses a FCC cubic space, where 12 direction of motion are allowed. First a tabu search is started, reducing the search space. The algorithm uses the possible 10 (12 two adjacent amino acids) as the neighbourhood of each amino acid. Constraints are:

1. Self avoiding walk
2. Every amino acid is on the lattice of the FCC
3. Two adjacent amino acids are bonding, only if the distance is 1 unit length

The objective function is to maximize the number of hydrophobic connections and hydrophilic connections. The second part of the algorithm uses constraint programming, and Large neighbourhood search to narrow the search. The initial constraints hold except for the objective function, which now minimizes randomly one of four objectives:

1. Minimize the distance to the origin
2. Minimize the distance to the center of mass
3. Maximize the density
4. Maximize the hydrophobic density

During this search, subsequences are subject to optimization. Although the results are not comparable to state-of-the-art techniques, it shows a good framework next to Monte Carlo simulations. Objective (2) and (3) are implemented in PITA by the BMS.

4.6 General Implementation Decisions

The different algorithms performance factors for protein structure prediction often have similar treats. Five of these are[32]:

1. Candidate representation
2. Dihedral angle space
3. Energy function
4. Folding strategy
5. Test set

Comparing these performance factors, might give more inside into good algorithms and choices. Picking a good representation (1) might significantly decrease computational power, but could be harmful to its accuracy. The candidate representation of PITA is based on dihedral angles, which is

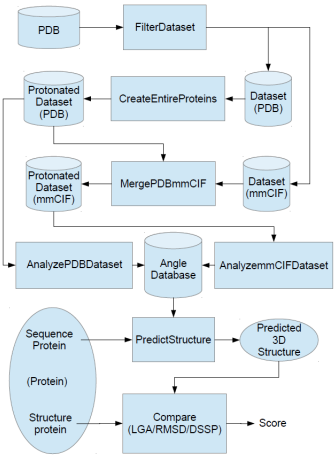


Figure 17: The entire framework PITA, showing the flow-chart of data and programs. Cylinders represent data sets, boxes represent programs

not harmful to its accuracy and reduces the search space significantly, since atoms must be close to each other (∼1.3Å) instead of all possible places in a 3-dimensional grid. This is implicitly enforced by dihedral angles, stretch angles and bond lengths. Type (2) handles the choice of angle space and atom radii. Because an infinite number of angles can be adopted, reduction of this search space is needed. Rotamer libraries, modelling the side chains as a oval appendix to the backbone, can be used or lattice space, where the number of angles significantly drops at a risk of inaccurate description. Energy functions have a great influence on the structure and therefore used wisely, since it will greatly influence the structure and computation time. Basically two groups can be specified: Physics-based and Statistics-based potentials. Although based on different concepts, both exhibit comparable results. The actual folding technique can be very different from other strategies, but two things should be noted: Using fragments may not be wise, since it reduces the potential long-range interaction and increase rigidity. Furthermore, these systems are bounded to known structures, since they use extra information of known structures. Finally, Type (5), algorithms use a test set which are different from other techniques, depending on size, protein size and protein type. Therefore no good comparison can be made between these techniques, since they might favour one very specific property. In the end, simplified protein representation together with fragment based methods using physics-based potential (slightly better) shows the best results. Specifically, I-TASSER, is the best and fastest.

5 Implementation of PITA

The approach we take is based on the first principles methods with database information. First a set of proteins is picked as being the source of information of dihedral angles (the database), as explained in Section 5.1. A database is filled with the dihedral angle information of this set of proteins, as explained in Section 5.3. The database is then used to build the protein structures. Our prediction program uses a 1-dimensional residue sequence (fasta-format file) as input and delivers as output a 3D structure (PDB-format file), described in Section 5.4. An overview of the entire framework is given in Figure 17(small) and Appendix 9(large). All the programs mentioned in this section are explained in Appendix 8.

5.1 Data set creation

The starting data set is a set of structures described in PDB format and mmCIF format (XXXX.pdb and XXXX.cif). This set can be selected by the user, the entire PDB dataset or any specific subset of the PDB/mmCIF dataset. We use two ways for selecting our starting dataset: Using the program `FilterDataset` and using the web interface of de pdb-website (*www.pdb.org*), which are two unrelated protocols. The program `FilterDataset` is provided, to filter for specific properties, which are described in the filter file. The CIFPARSE-OBJ C++ Parser was the basis for the program `FilterDataset` and was used to read and handle the initial set. The program filters using the following rule: If the input structure has all of the following qualities, it is put in the output data set:

- At least one of the substructures contains a structure constructed of only the 20 standard amino acids.
- At least one of the substructures follows all the substructure specific properties specified in the filter file, such as experimental method, number of entities, etc.
- All the experimental specific properties specified in the filter file are present in the experimental specific properties of the mmCIF-file, such as species, length, etc.

The resulting data set now contains the PDB files and associated mmCIF files following all the rules in the filter files. The program is explained in detail in Appendix 8.

The website has an alternative filtering system, with less functionality (`FilterDataset` can search on any criterion, where the website interface uses more general criteria), but the interface is very user friendly and much faster than our program.

5.2 Data set preprocessing

The data set now contains the proteins that we want to have, however the structures are not complete: apart from the hydrogen atoms missing in X-ray structures, some structures are missing atoms (e.g. C, N and H). The VMD (Visual Molecular Dynamics) package, a package created as a visual display of NAMD, the location of the missing atoms can be determined. Using the feature 'guesscoord' of the VMD package, all the coordinates of missing atoms are guessed and the structures are converted to zwitterions. The program `CreateEntireProteins` was created for this purpose. This way, a clean data set of structures with only fully described structures with only the 20 standard amino acids remains.

5.3 Database of Dihedral Angle Information

The database that is constructed, solely consist of fragments. A fragments contains the information of one residue:

1. The name of the residue
2. The context, the information of its neighbourhood, restricted to only residue names
3. The size of the neighbourhood, with 4 possible values: 1, 3, 5, and 7
4. The dihedral angles of the backbone
5. The location of the atoms in the side chain, relative to the atoms of that residue in the backbone

An example of a fragment is given in Figure 18. The fragment in Figure 18 is of a valine residue and is the middle of the sequence ALA-GLY-VAL-VAL-THR and the dihedral angles of the valine residue are $\phi = -80.30°, \psi = -35.97°, \omega = -177.84°$. Using dihedral angle ω of the previous residue, angle $\angle \mathrm{CNC}_\alpha$ and distance r_{NC_α}, the coordinates of the C_α atom can be calculated. This holds for all the atoms in the valine residue. This means that for a set of N proteins with each of length $L_i, 1 \leq i \leq N$, the number of fragments, F, would be the following:

$$F_i = |\mathrm{NS} = 1| + |\mathrm{NS} = 3| + |\mathrm{NS} = 5| + |\mathrm{NS} = 7| - 2 \tag{34}$$

$$F_i = 100 + 98 + 96 + 94 - 2, \text{ for } L_i = 100 \tag{35}$$

$$F_i = 2 \cdot (100 + 94) - 2 \tag{36}$$

$$F_i = 4 \cdot L_i - 14 \tag{37}$$

$$F = \sum_{i=1}^{N} 4 \cdot L_i - 14 \tag{38}$$

where F_i is the number of fragments for protein i and NS is the neighbourhood size. Since the ϕ dihedral angle is dependent of the previous residue, and the ω dihedral angle is dependent on the next residue, not all the torsion angles of the first and last residue (for neighbourhood size 1) could be calculated and these two fragments are excluded from the database, hence the -2 in Equation 34.

```
// Residue name
VAL

// Neighborhood
ALA-GLY-VAL-VAL-THR

// Neighborhood size
5

// Dihedral angles
phi;   -80.296074;
psi;   -35.970062;
omega; -177.835959;

// Backbone information
CNCA;119.532002; NCA;1.504769;
NCAC;111.319719; CAC;1.508882;
CACN;116.171776; CN;1.343093;

// Side chain information
T-O;149.813750;     A-O;118.661440;   D-O;1.268095;
T-CB;-123.369818;   A-CB;33.179050;   D-CB;2.542942;
T-CG1;-147.256595;  A-CG1;59.500016;  D-CG1;2.928356;
T-CG2;-130.902157;  A-CG2;20.518189;  D-CG2;3.906366;
T-HN;-27.351240;    A-HN;39.661258;   D-HN;3.050576;
T-HA;117.218974;    A-HA;29.395151;   D-HA;2.104857;
T-HB;-94.351199;    A-HB;46.742579;   D-HB;2.875899;
T-HG11;-140.551548; A-HG11;60.065964; D-HG11;3.984706;
T-HG12;-141.237050; A-HG12;81.007733; D-HG12;2.637971;
T-HG13;-170.766273; A-HG13;60.430221; D-HG13;3.191112;
T-HG21;-128.511759; A-HG21;30.474863; D-HG21;4.726406;
T-HG22;-176.672218; A-HG22;19.645031; D-HG22;4.172364;
T-HG23;-84.680083;  A-HG23;14.327177; D-HG23;4.283762
```

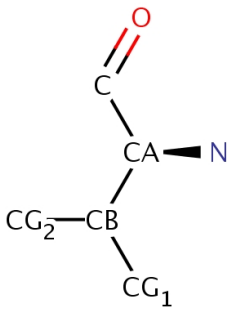


Figure 18: Representation of the information of a fragment, containing the information of the backbone and the side chain.The structure of a Valine residue is shown on the left for clarity. Using the torsion angles (`phi`, `psi`, `omega` and `T-*`), the stretch angles (`CNCA`, `NCAC`, `CACN` and `A-*`) and the distances (`NCA`, `CAC`, `CN` and `D-*`) all the coordinates of the valine residue can be reconstructed.

5.4 Prediction of the Structure

At this point we have enough information to construct a 3D structure based on the sequence information. Using the program `PITA`, the input file (fasta-format) is read and the internal sequence structure S is created. For every residue in S, the database is searched for similar fragments. A fragment is similar if the following statements are true:

1. The name of the residue S_i is equal to the residue name of the fragment.
2. The neighbourhood of S_i $(S_{i-x}, \ldots, S_i, \ldots, S_{i+x})$ is the same as the neighbourhood of the fragment, where $2x + 1$ is the neighbourhood size of the fragment.

The fragments are ordered: First all the fragments with size 7 are selected, then size 5, 3 and as last size 1. Using this method we get sequence S with all the possible fragments attached to their accompanying residues in an organized way, descending in neighbourhood size. This is shown in Figure 19. Notice that fragments #87, #77, #99 are all fragments with residue name ALA and neighbourhood size 1 and thus reappear at 3 positions in this sequence. #26 has a bigger neighbourhood size and is therefore placed before the fragments with neighbourhood size 1. Two methods have been implemented to construct the 3D structure: *Random search* and *Energy based search*.

5.4.1 Random Search

To find a structure based on the fragment database a depth first search with pruning is started. The starting point is the most common folding of the backbone ($NC_\alpha C$)[33], together with the first fragment of the first residue, recreated in its zwitterionic state. Iteratively, residues are added to the intermediary structure based on the fragment list. The fragment list per residue exists of first the fragments with the longest neighbourhood followed by the smaller neighbourhood. We do not discriminate between fragments with the same neighbourhood and this procedure is thus seen as a random search, still using the most probable fragments first. The resulting structure, Q, is checked for steric hindrance to avoid invalid protein foldings. This is done by calculating the distances between

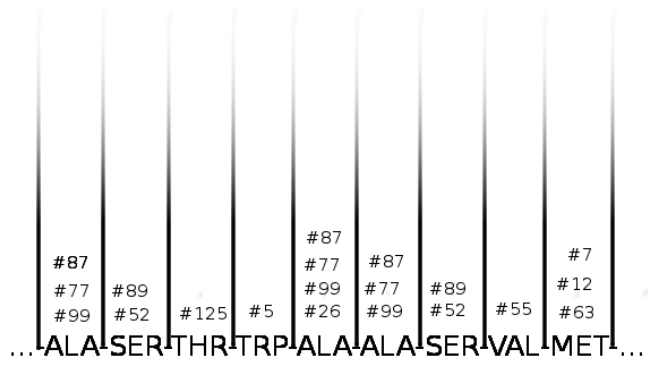


Figure 19: The one dimensional sequence and the fragments which are similar to the sequence in respect to residue name and neighbourhood, with the lowest having the largest neighbourhood.

all the atoms in the previous structure S and the newly added atoms, N:

$$S \cap N = \emptyset \tag{39}$$

$$S \cup N = Q \tag{40}$$

$$\text{StericHindrance} = \begin{cases} \text{false} & \text{if } r_{min}(s) + r_{min}(n) < d(s,n) \qquad \forall s \in S, \forall n \in N \\ \text{true} & \text{otherwise} \end{cases} \tag{41}$$

where function $d(x, y)$ returns the Euclidean distance between atom x and y and $r_{min}(x)$ return the minimum radius of atom x, based on [18]. If there is no steric hindrance, we move on to the first fragment of the next residue. If there is steric hindrance, the last attached residue of the intermediary structure is removed and the next fragment in line is added. If the end of the fragments of a residue is reached, we go back to the previous residue and add the fragment next in line. The structure of the program is described in Algorithm 1.

5.4.2 Energy based search

The fragment list for the energy based search is built as the fragment list in the random search. The starting point is created with the backbone of the ideal folding and the side chain of the fragment list and placed in the queue. From this point on, the first n intermediary structures in the queue are selected. For these n intermediary structures, the first m fragments from the fragment list are used to construct the new $n \times m$ (intermediary) structures. For all the new structures, the steric hindrance is checked and the energy is calculated explained in 5.4.3. The energies of these $n \times m$ structures are calculated in parallel. The structures with (internal) steric hindrance or a to high energy are removed the rest is added to the queue. When a new structure is added to the queue, it is inserted at the position where the previous structure has less energy and the next structure has more more. This way, a sorted queue is maintained. For the n structures that served as a base for the new structures, the last used fragment is registered. Once all fragments are used the structure will be deleted. If the first of the queue is a fully built protein, its structure is saved (in PDB-format) only if the energy is lower than the energy of currently best structure. The structure is then removed from the queue. To avoid a memory overflow, every X cycles, the queue is filtered and the best 10000 structures are kept. The pseudo code of the algorithm is displayed in Algorithm 2.

5.4.3 Energy Calculation

For the energy calculation the package *MDenergy* is used. MDenergy is a program that calculate energies based on its 3D structure. It is originally derived from Mindy, a simplified version of the NAMD engine[10]. It can calculate the different bonded and non-bonded energies for each atom based on the information of the structure and the coordinates as specified in Section 2.8. The structure, which describes which atoms are covalently bonded and what type of atoms are within the structure, is given by the PSF-file (Protein Structure File). The coordinates are given by the PDB-file. The total energy values are identical to the NAMD output. The parameter file that comes with the installation of NAMD was used (July, 2003). MDenergy's source code was edited to avoid disk activity. The structure files are loaded at the begin of the program and are stored as PSF-objects in memory. The

Algorithm 1: *PITA* with Random Search

Data: Fasta-formatted file in 1D, with name *sequence.fasta*, target T in pdb-format
Result: Predicted structures in 3D, in a pdb formatted file, *resultX.pdb*, where $X = 1, 2, \ldots, max\_structures$

$Sizes \leftarrow \{7, 5, 3, 1\}$
begin
  S = ReadInputSequence(sequence.fasta) `/* Read sequence and store it in array` S `*/`
  N = GetSize(S) `/* Get the size of the sequence */`
  foreach *size in Sizes* **do** `/* Get all the possible fragments for al the sizes */`
    for $i \leftarrow \frac{size-1}{2}$ **to** $N - \frac{size-1}{2}$ **do**
      F_i += ReturnSimilarFragments($S, i, size$)
  for $i \leftarrow 1$ **to** N **do**
    $num\_fragments_i \leftarrow$ GetNumberOfFragments(F_i)
  $next\_fragment \leftarrow \{1, 1, \ldots, 1\}$ `/* Fill array with` N `times 1 */`
  $max\_structures \leftarrow 100$
  SetInitialStructure($X, F_{1,1}$) `/* Initialize intermediary structure` X `*/`
  $i \leftarrow 1$ `/* Index of the sequence */`
  while $time\_passed < max\_time$ **do** `/* Search for a period of time */`
    if $i \neq N$ **then**
      $j \leftarrow next\_fragment_i$
      if $j \leq num\_fragments_i$ **then** `/* Check if there are possible fragments */`
        AddResidue($X, F_{i,j}$)
        if StericHindrance(X) **then** `/* Check for steric hindrance */`
          RemoveResidue(X, i)
          $next\_fragment_i = next\_fragment_i + 1$
        else
          $i = i + 1$
      else
        $next\_fragment_i = 1$
        $i = i - 1$
    else
      $rmsd$ = CalculateRMSD(X, T)
      if $rmsd < min\_rmsd$ **then**
        $min\_rmsd \leftarrow rmsd$
        WriteToFile(X)
        RemoveResidue(X, i)
        $next\_fragment_i = next\_fragment_i + 1$
end

Algorithm 2: *PITA* based on energy.

```
Data: Fasta formatted file in 1D, with name sequence.fasta
Result: Predicted structures in 3D, in a pdb formatted file, resultX.pdb, where
        X = 1, 2, ..., max_structures
Sizes ← {7, 5, 3, 1}
begin
    S = ReadInputSequence(sequence.fasta)   /* Read sequence and store it in array S */
    N = GetSize(S)                          /* Get the size of the sequence */
    foreach size in Sizes do                /* Get all the possible fragments for al the sizes */
        for i ← (size−1)/2 to N − (size−1)/2 do
            F_i += ReturnSimilarFragments(S, i, size)
    for i ← 1 to N do
        num_fragments_i ← GetNumberOfFragments(F_i)
    max_structures ← 100
    Q = SetInitialStructure(X, F_{1,1})   /* Initialize queue with most common folding X */
    qi ← 1                                                  /* queue index qi */
    while time_passed < max_time do                 /* Search for a period of time */
        X = Q_qi
        if X.size() = target_length then
            if X.energy < min_energy then
                X.energy ← min_energy
                WriteBackToFile(X)                          /* Store final structure */
                Q.delete(X)                                 /* Delete from queue */
                continue                                /* else, continue the while-loop */
        else
            for i ← 1 to num_threads do
                P_i = X
                if P_i.next_fragment = ∅ then /* All possible fragments have been tried */
                    Q.delete(X)                             /* so, delete from queue */
                    qi ← qi + 1
                    X = Q_qi
                    num_threads = num_threads − 1                   /* Reset thread */
                    continue                            /* And continue with for-loop */
                P_i.appendResidue(P_i.next_fragment)        /* Append structure with */
                                                            /* next fragment in line */
                thread_i.CalculateEnergy(P)               /* And calculate its energy */
                if !StericHindrance(P_i) ∧ P_i.energy < 10000 then
                    AddToQueue(P_i)             /* If structure is valid, add to queue */
                if i%fragments_per_cycle = 0 then
                    qi ← qi + 1
                    X = Q_qi
end
```

PDB-objects are created at run time, every time a residue is added. No other disk access then loading of structure files, loading parameter files and writing valid structures was necessary.

5.5 Optimizations

The search space quickly becomes too large to fully explore it. A few optimization are proposed to prune the tree, so the search space becomes small enough to traverse.

5.5.1 K-pruning

A very easy method to reduce the search space is too reduce the number of possible fragments in the fragment list per residue to $K \in \mathbb{N}$,. With this, the fragments with the largest neighbourhood remain. For example, if we take the fragment list in Figure 19 and let $K = 2$, the resulting fragment list will not include fragments #87(1st ALA), #87 and #77 (2nd ALA), #87 (3rd ALA) and #7 (1st MET).

5.5.2 MBS-pruning

The minimal bounding sphere can also be used in the pruning process. Assuming that a protein adopts a very packed structure, the backbone will probably not stretch far from the center. If this does happen, the minimal bounding sphere will increase. To a certain extend, this must be allowed, since the backbone does extend the minimal bounding sphere from time to time, so with a threshold based on the current (approximation of the) bounding sphere the tree can be pruned. The minimal size for every structure size is kept, as a reference. If the radius of the minimal bounding sphere of the structure is larger than the radius of the smallest minimal bounding sphere that is found, r, multiplied by factor α, this structure can be pruned. The approximation of the radius of the minimal bounding sphere is calculated by taking the minimum of these three approximations: Ritter's Algorithm, Bounding Bubble and Bounding bubble with center of gravity. The overview of the algorithm, is shown in Algorithm 3.

Algorithm 3: *MBS-pruning* based on energy.

Data: (Intermediary) Protein structure X in PDB-format and array S with the smallest minimum bounding sphere found for that specific sequence length
Result: Confirmation whether this structure can be pruned.
begin
  $r_R \leftarrow \text{RittersAlgorithm}(X)$
  $r_{BB} \leftarrow \text{BouncingBubble}(X)$
  $r_{BBCG} \leftarrow \text{BouncingBubbleWithCenterOfGravity}(X)$
  $r_{min} = \min(r_R, r_{BB}, r_{BBCG})$ `/* Get minimum of all 3 radii */`
  $l \leftarrow \mathit{GetSequenceLength}(X)$
  if $r \leq S_l$ **then**
    if $r \leq \alpha \cdot S_l$ **then** `/* If a new minimum is found for a certain length */`
      $S_l \leftarrow r_{min}$ `/* the minimum for length` l `needs to be altered */`
    return false
  else
    return true
end

5.5.3 Sequence length and energy combined

The energy function is a good measure for probability of the structure and property for pruning. However, this search technique is looking for the lowest energy. This results in a search between a breadth first search and a depth first search and therefore it takes much time to find at least one solution. In order to get results quicker, we introduce the length of the sequence (number of residues) as part of the energy function. Since newly created structures are placed in front of the queue, the program is favouring intermediary structures with low energy but not long sequences. This might be a problem for some sequences as they might have an intermediary structure with relative high energy which must be passed to get to the final low energy configuration. By favouring both energetically stable sequences and long sequences, the computation time to actually get a full structure might possibly decrease. We introduce two different Energy Adaptation Functions (EAF) using length l and energy E as its input and the new energy E_n as its output. Using the Boltzmann distribution (Equation 42) it can be seen that the probability of a structure, $F(state)$, is proportionate to $e^{-\frac{E}{k_B T}}$, where E is the energy in joules, k_B is the Boltzmann constant and T the temperature. Now, let there be 2 structures: X_1 and X_2, identical in shape, where both structures have energies E_1 and E_2, which have the same energy: $E_1 = E_2$. Using the Boltzmann ratio (Equation 43) the ratio of them in a mixture would be $1:1$.

$$F(\text{state}) \propto e^{-\frac{E}{k_B T}} \tag{42}$$

$$\frac{F(\text{state1})}{F(\text{state2})} = e^{\frac{E_2 - E_1}{k_B T}} \tag{43}$$

However, we would like the probability of a longer structure to be higher. To achieve this, we set the ratio to $1:x$, e.g. the structure X_2 is x times more likely to occur, where x is a scalar dependent on l (Equation 44). Since the energy output of *MDenergy* is in kcal/mol we have to rewrite the equation, resulting in Equation 46, where R is the gas constant, T the temperature in Kelvin and N_a the number of Avogadro. For all the experiments, $T = 310.15\text{K}$ (37°C) was used.

$$\frac{1}{x} = e^{\frac{E_2 - E_1}{k_B T}} \tag{44}$$

$$\frac{1}{x} = e^{\frac{E_2 \cdot \frac{4{,}184 \cdot 10^3}{N_a} - E_1 \cdot \frac{4{,}184 \cdot 10^3}{N_a}}{\frac{RT}{N_a}}} \qquad k_B = \frac{R}{N_A} \tag{45}$$

$$\frac{1}{x} = e^{\rho(E_2 - E_1)} \qquad \rho = \frac{4{,}184 \cdot 10^3}{RT} = \frac{4{,}184 \cdot 10^3}{8.314 \cdot 310.15} = 1.622 \tag{46}$$

1. Assume we would like to have ratio $\frac{F(\text{state1})}{F(\text{state2})} = \frac{1}{l^\alpha}$. In order to get this ratio, the energies should be changed. Using Equations 47-50, the energy and length of the sequence can be combined, resulting in the new energy function $f(E) = E - \frac{\alpha}{\rho} \ln l$. This will be called Energy Adaptation Function 1 (AEF1)

$$\frac{1}{l^\alpha} = e^{\rho(E_2 - E_1)} \tag{47}$$

$$\ln l^{-\alpha} = \rho(E_2 - E_1) \tag{48}$$

$$-\frac{\alpha \ln l}{\rho} = E_2 - E_1 \tag{49}$$

$$E_2 = E_1 - \frac{\alpha}{\rho} \ln l \tag{50}$$

2. The second function is based on the ratio $1:\alpha^l$, where α is some scalar. Equation 51-54 show the new energy function $f(E) = E - \frac{l}{\rho} \ln \alpha$. This will be called Energy Adaptation Function 2 (AEF2)

$$\frac{1}{\alpha^l} = e^{\rho(E_2 - E_1)} \tag{51}$$

$$\ln \alpha^{-l} = \rho(E_2 - E_1) \tag{52}$$

$$-l\frac{\ln \alpha}{\rho} = E_2 - E_1 \tag{53}$$

$$E_2 = E_1 - \frac{l}{\rho} \ln \alpha \tag{54}$$

5.5.4 Other optimizations

To speed up the program PITA, energy calculations were performed in parallel. 16 threads were used to speed up the energy predictions. Also, the package MDenergy was slightly adjusted to avoid disk activity and all the temporary structures were stored in main memory.

5.6 Experimental Setup

To test PITA, three datasets were created: *AlphaFragments*, *BetaFragments* and *E.coliFragments*. The focus is on whether PITA can reproduce the secondary structure of the alpha helix and the beta sheet and also if the structure of proteins of E. coli, the most widely studied prokaryotic model organism, can be reproduced. The construction of these data sets was done using the following protocol:

1. *AlphaFragments*
   A set of proteins was taken from the PDB-site, using the filter functionality of the PDB-site. With this functionality one can specify a filter and it returns all the proteins from the PDB that satisfy all the criteria in the filter. To reproduce alpha helices, all the proteins that contain alpha helices are taken from the PDB. To guarantee quality structures, the dataset contains all the structures with: (1) ScopTree Search for All alpha proteins, (2) Experimental Method is X-RAY (3) Resolution is between 0.0 and 2.0, (4) Chain Type: there is a Protein chain but not any DNA or RNA or Hybrid. Criterion (1) returns all proteins containing only one or more alpha helices. All the structures of the resulting *AllAlphaProtein*-set are checked for their residues. If a structure contains a residue other than the 20 most occurring residues, it is excluded from the set. From this set, 15 sequences were excluded, with lengths ranging from 0-150, which will be the test set. Since X-ray structure do not have the information of hydrogen atoms, the locations of all the missing hydrogen atoms are estimated using our program *CreateEntireProteins*, which is based on the *guesscoord* functionality of *VMD*. It estimates the location of the hydrogen atom based on a template of the residue. *CreateEntireProteins* creates the data set *EntireAlphaProteins*. This data set was an input for the program *AnalyzePDBDataset* which traverses all the structure, calculating the angles and distances for all possible neighbourhoods up to neighbourhood size 7, as shown in Section 5.3. The output of this set is the data set *AlphaFragments*. The test set is checked for continuity, entirety and whether the residues are correct. If not, the sequence is discarded.
2. *BetaFragments*
   The same method to create the *AlphaFragments* was used to create *BetaFragments* with the one exception that (1) is equal to "ScopTree Search for All beta proteins", being all proteins containing only one or more beta sheets.
3. *EcoliFragments*
   The entire PDB data set was downloaded and the program *FilterDataset* was used to filter the data set. *FilterDataset* uses a dataset of mmCIF files and a filter file as input and returns a data set excluding all the structures not satisfying all the criteria in the filter file.The filter stated the following specifics: Experimental Method is 'X-RAY DIFFRACTION', the structure is of type 'polypeptide(L)' (Protein containing only Levorotatory enantiomers) and the scientific name of the organism of the source gene is 'Escherichia coli', see Appendix 8 for the specific filter. The result is checked for validity, hydrogen atoms are added and the test set is excluded like the other data sets.

The result is 3 fragment sets and 3 test sets. The test sets are displayed in Table 4.

Table 4: Test sets of the 3 data sets. Data set *AlphaFragments* has 8 test sequences, *BetaFragments* 5 and *EcoliFragments* has 6 test sequences. For each sequence, the length is displayed and distribution, being the number of residues with maximum neighbourhood size $X \in \{1, 3, 5, 7\}$. For example, sequence 1NPL has 2 residues with a fragments list with maximum size 1 (first and last residue), 45 residues with maximum neighbourhood 3, 26 residues with maximum neighbourhood 5 and 36 residues with maximum neighbourhood size 7, adding it all up to 109 residues, its sequence length. The distribution gives an idea of the overlap of the sequence and the sequences in the data set. *Sequence 2BRF was missing one residue in its coordinate file. Since the database had many 5-size fragments of this sequence, the distribution of the fragments made the protein interesting enough to keep it in the test set. The missing backbone atoms were estimated using the average value of the surrounding backbone atoms. 3 atoms were added in this manner.

Sequence type	Sequence name	Length	Distribution Size (1—3—5—7)
Alpha	1AIE	31	(2 — 23 — 6 — 0)
Alpha	1C26	32	(2 — 24 — 6 — 0)
Alpha	1ROP	56	(2 — 8 — 4 — 42)
Alpha	1R69	63	(2 — 50 — 11 — 0)
Alpha	2G7O	68	(2 — 60 — 6 — 0)
Alpha	2E1F	94	(2 — 82 — 10 — 0)
Alpha	1L3P	102	(2 — 81 — 19 — 0)
Alpha	1S3P	109	(2 — 9 — 6 — 92)
Beta	1JO8	58	(2 — 49 — 7 — 0)
Beta	1MHN	59	(2 — 48 — 9 — 0)
Beta	1PWT	61	(2 — 3 — 3 — 53)
Beta	2BRF*	101	(2 — 69 — 29 — 1)
Beta	1NPL	109	(2 — 45 — 26 — 36)
E.coli	1JCD	52	(2 — 47 — 3 — 0)
E.coli	2YLC	66	(2 — 64 — 0 — 0)
E.coli	1THO	109	(2 — 9 — 4 — 94)
E.coli	3A7L	128	(3 — 118 — 7 — 0)
E.coli	1GMU	138	(3 — 127 — 8 — 0)
E.coli	1I9D	138	(3 — 123 — 12 — 0)

To see the efficiency of the three MBS algorithms, a protein was folded using `PITA` and 1000 times the MBS was calculated. The results are in Figure 20.

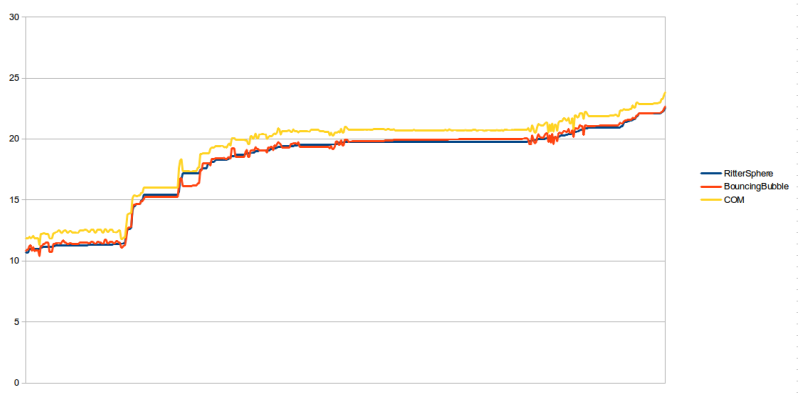


Figure 20: The calculated MBS in Å for Ritter's algorithm (RitterSphere), Bouncing bubble (BouncingBubble) and Bouncing bubble with center of gravity (COM). The data points are correlated: every RitterSphere BMS-value has a correlating bouncingBubble BMS-value and COM BMS-value on the same vertical line. The data points of RitterSphere are sorted ascending.

Table 5: The calculated LGA scores for all the sequences for K-pruning, showing the LGA scores for different values of K, using a time out of 10 minutes for every prediction without energy calculation. A '-' means no prediction was given in time.

	2	4	6	8	10	12	14	16
1GMU	-	-	-	0.123654	0.121618	0.121618	0.11225	0.11225
1I9D	-	-	-	-	-	-	-	-
1JCD	-	0.331685	0.331685	0.331685	0.327701	0.302335	0.302335	0.302335
1THO	-	0.439712	0.548711	0.561992	0.561927	0.457536	0.457864	0.457492
2YLC	-	-	0.180736	0.180736	0.182395	0.184632	0.184632	0.184632
3A7L	-	-	-	-	-	-	0.12619	0.12619
1AIE	-	-	-	-	0.463441	0.452765	0.474347	0.487558
1C26	-	-	-	-	0.448065	0.448065	0.448065	0.448065
1L3P	-	-	-	0.167624	0.167367	0.167367	0.171405	0.453548
1MHN	-	0.212107	0.212107	0.212107	0.212107	0.212107	0.212107	0.212107
1PWT	0.924005	0.924005	0.924005	0.924005	0.924005	0.924005	0.924005	0.924005
1R69	-	0.226909	0.234769	0.237339	0.238851	0.233409	0.237302	0.237302
1ROP	-	0.563393	0.563393	0.563393	0.563393	0.563903	0.563223	0.563776
1S3P	-	0.505985	0.502927	0.497335	0.497357	0.497357	0.49727	0.494124
2G70	-	-	-	0.290966	0.296814	0.296814	0.296814	0.296814
2E1F	-	-	0.501418	0.497822	0.492579	0.204813	0.204813	0.501824
1JO8	-	-	0.227778	0.230576	0.231078	0.22523	0.236466	0.238053
1NPL	-	0.209917	0.209917	0.212538	0.211817	0.211162	0.210026	0.209873
2BRF	-	0.149269	0.150896	0.144884	0.148916	0.148208	0.14314	0.131589

If we were to predict a structure using PITA and gathered all the possible fragments for the all the residues, every residue would have an average of 1000 potential fragments. Since this is infeasible for us to use, the number of fragments must be reduced. Using K-pruning, the right K must be uses, since a K too high would make the problem infeasible again but K too low could mean the correct fold might not appear because of the steric hindrance pruning. In order to estimate a good K, for all sequences the structure was predicted with different K, starting from 2 to 16, with step size 2. For all the best solutions, the LGA score was calculated. Results are displayed in Table 5. From this we can say that $K = 14$ will significantly decrease the possibilities and still gives the enough options to create a solution.

Table 6: LGA values of the best structures for different values of α. PITA was run without energy function, $K = 12$ and a time limit of 1 hour per sequence. No value, in the table, means no experiment was run, '-' means the PITA was not able to find a solution within the time limit.

α	1AIE (31)	1L3P (102)
0	0.451920	0.171405
1.01	0.451920	-
1.05	0.451920	-
1.06		-
1.07		-
1.08		0.171405
1.09		0.171405
1.10	0.451920	0.171405
1.15	0.451920	0.171405
1.20	0.451920	0.171405
1.25	0.451920	0.171405

To reduce the search space, minimal bounding sphere (MBS) was proposed. As shown in Algorithm 3, three functions determined the BMS. Since the Bouncing Bubble with center of gravity was significantly worse than the other two, only Ritter's algorithm and Bouncing Bubble were used. Bouncing Bubble performed slightly better overall, but the results were very sequence specific. To see the effect of the parameter α in the BMS-algorithm, two sequences were used: 1 short and 1 long sequence. The results are displayed in Table 6. A second run of experiments was executed with all the sequences and $\alpha = 1.01$. For 17 of the 19 no difference was seen and the remaining 2 were worse than the structures create without the BMS-pruning (not shown). From this and Table 6 no improvement was seen and in the next experiments no BMS-pruning will be used.

Table 7: The length of the longest predicted structure for different values of α using Energy Adaptation Function 1, trying to predict sequence 1L3P. $K = 12$ and time limit was set to 30 minutes.

α	Maximum length
1	18
10	22
250	24
500	42
1000	61
2000	75
3000	101

Table 8: The length of the longest predicted structure for different values of α using Energy Adaptation Function 2, trying to predict sequence 1L3P. $K = 12$ and time limit was set to 30 minutes.

α	Maximum length
1	18
10	22
250	22
500	23
1000	61
2000	75
3000	101

PITA gets stuck in local minima when trying to find the 3D structure, especially with long sequences. To tackle this problem, in addition to MBS-pruning and steric hindrance, Energy Adaptation Functions were implemented. The functionality was tested using the long sequence 1L3P. PITA itera-

Table 9: Final protein predictions for all the test sequences. Both the LGA and DSSP were calculated for the structures with the highest RMSD. the no-energy run was calculated with $K = 14$, $BS = 0$ and a time out after 1 hour. The energy calculations were performed using $K = 14$, $BS = 0$, and a time out after 1 hour. If a energy adaptation function was used to get the final structure, the EAF column depicts which and the variable α. If the EAF is empty, no EAF was used to get the final result.

Sequence name	LGA (no-energy)	LGA (energy)	DSSP (no-energy)	DSSP (energy)	EAF
1GMU	0.111801	**0.123792**	**5.7971%**	2.1898%	$F_2(\alpha = 5000)$
1I9D	-	**0.104658**	-	**10.2190%**	$F_2(\alpha = 3000)$
1JCD	0.236466	**0.354899**	9.6154%	**23.5294%**	
1THO	**0.332176**	0.195325	**41.2844%**	22.2222%	$F_2(\alpha = 3000)$
2YLC	**0.210317**	0.199784	**15.1515%**	9.2308%	$F_2(\alpha = 3000)$
3A7L	**0.141406**	0.119141	**12.5%**	11.0236%	$F_2(\alpha = 10000)$
1AIE	**0.498771**	0.378648	**25.8065%**	16.6667%	
1C26	**0.477083**	0.364881	**25%**	19.3548%	
1L3P	0.171405	**0.182049**	17.6471%	**19.8020%**	$F_2(\alpha = 3500)$
1MHN	**0.226352**	0.205408	**13.5593%**	5.1724%	$F_1(\alpha = 3000)$
1PWT	**0.937939**	0.266237	**62.2951%**	15%	$F_1(\alpha = 1000)$
1R69	**0.240703**	0.239342	**23.8095%**	22.5806%	$F_2(\alpha = 500)$
1ROP	**0.570111**	0.250255	**80.3571%**	18.1818%	
1S3P	**0.498493**	0.180756	**57.7982%**	30.5556%	$F_1(\alpha = 3000)$
2G70	0.284909	**0.303221**	**30.8824%**	13.4328%	$F_2(\alpha = 3000)$
2E1F	**0.204813**	0.159549	**27.6596%**	9.6774%	$F_2(\alpha = 3000)$
1JO8	**0.318223**	0.194779	**8.7719%**	5.3571%	$F_2(\alpha = 3000)$
1NPL	**0.209961**	0.149891	**17.4312%**	3.7037%	$F_2(\alpha = 3000)$
2BRF	**0.142763**	0.139250	**10.8911%**	3%	$F_2(\alpha = 3000)$

tively builds the structures and records the first predicted structures for all lengths. Thus we can see the process of predicting an entire structure for specific values of α. For Energy Adaptation Function 1, the results are shown in Table 7, for Energy Adaptation Function 2, result are shown in Table 8.

Based on these experiments a final run of experiments was performed. For the no-energy calculation, all 19 sequence were predicted with one specific setting. For the prediction based on the energy, no EAF was used initially, but if no structure was found, the following sequence of EAF's was used until a structure was found:

1. EAF 2, $\alpha = 100$
2. EAF 2, $\alpha = 500$
3. EAF 2, $\alpha = 1000$
4. EAF 1, $\alpha = 1000$
5. EAF 1, $\alpha = 3000$
6. EAF 1, $\alpha = 3500$
7. EAF 1, $\alpha = 10000$

This order was used, because EAF 2 adds less "length-energy" to the system with the same l and α, and consequently giving more value to the actual energy.

All the experiments were run on the 32-core `huisuil03` supercomputer, provided by the LIACS (Leiden Institute for Advanced Computer Science).

6 Discussion

One of the challenging aspects of protein structure prediction is the dynamic of the entire protein. An example of this dynamic behaviour is the disulphide bond that can occur when two cysteine residues are close to each other. This behaviour is not modelled in PITA. With the no-energy prediction of PITA, steric hindrance will prevent the two cysteines from being located next to each other and the energy function will interpret the two residues as two repelling forces, and thus a (non-favourable) high energy. For future work, these disulphide bonds could be modelled.

Looking at Table 9, the most remarkable prediction are 1PWT and 1ROP. It must be said, that the overlap of those sequences with the fragment database was more than expected: as is shown in Table 4, many fragments with neighbourhood size 7 were found. This is also the case for sequences 1S3P, 1NPL and 1THO. In the case of sequence 1ROP, the publication date is from 1992, but after this (1995-2013), 15 other sequence with 90%+ sequence similarity were published. With 1PWT, a alpha-spectrin protein, many structures of the protein alpha-spectrin were measured for many different species, resulting in the abundant description of that protein and thus multiple sequence in de PDB data set. Since we only excluded one, the remainder can be used by PITA to predict the structure.

The EAF was introduced because the energy model calculation became stuck in a local minimum. Lets assume a function F(x) that represents the energy of all the intermediary structures of protein X, with length x. This function F will be function with many local minima and maxima and PITA must overcome all the local maxima to get to the other side, being the entire protein. Table 7 and 8 nicely depict this concept, since the lengths of the longest structure for those runs have almost the same lengths: there is a local maxima near length 23, 61 and 75. The actual values of α in the EAFs are somewhat misleading as the number represents a relation of $1{:}\alpha^l$ (and $1{:}l^\alpha$), which in the most extreme example is 1000^{128} (sequence 3A7L, Table 9). Mostly long sequences are harder to predict. Future work could be to find another way to overcome the local optima. Also, the energy that was calculated was the energy of a structure in a vacuum. This is not bad practice, but in vivo folding occurs with an excess of water molecules around, which can stabilize some structures by forming hydrogen bonds especially when polar residues are on the outside of the chain. This is not a new concept and can be implemented in future versions of PITA.

The energy calculation of all structures was greatly dependent on the correct estimation of all the hydrogen atoms by *VMD*, since these were not present in X-ray structures. The estimation of the hydrogen is based on local descriptors and when applied to a tightly packed protein, some hydrogen atoms will create a repulsive force, because of steric hindrance. Ultimately this will result in a high energy for that protein. Because NMR techniques do observe hydrogen atoms, future research could indicate what the results are when a dataset of NMR structures is used instead of accurate X-ray structures.

PITA generates a lot of possible structures. It was decided to only save the predicted structure if the RMSD value is better than previous best prediction for the random search. Consequently, that research serves as an upper bound of the potential of PITA, based on the RMSD.

7 Conclusion

The protein structure predictor PITA has been created. The predictor was able to create existing (true) structures by using steric hindrance pruning or by using the energy as a guide for probable structures. Using the PDB and the package VMD, three clean datasets were created to form a basis for the prediction. Using the predictor PITA the structure 1PWT was predicted with a LGA score of 0.937939. Also, the secondary structure could be predicted with a score of 80.3571% for 1ROP. Since all the predictions of the 19 sequences have $> 0\%$ score, we can conclude PITA is able to create secondary structure, for both alpha helices as beta sheets, for all the test sets. The search of PITA based on energy showed an improvement of a few predictions, but the search without energy calculation performs better overall. Using a bounding sphere as a way to prune the search tree had little to no effect. Problematic were longer sequences. Using the EAF, a prediction could be enforced, but not for the prediction without energy. Therefore 1I9D could not be predicted. For future work, the condition to prune or not can be dependent on BMS $< \alpha \sqrt[3]{r}$, modelled to the radius-volume relation of proteins.

References

[1] Christian B Anfinsen. Principles that govern the folding of protein chains. *Science*, 181(4096):223–230, 1973.

[2] Mohammed J Zaki and Christopher Bystroff. *Protein structure prediction*. Springer, 2008.

[3] Fabrizio Chiti and Christopher M Dobson. Protein misfolding, functional amyloid, and human disease. *Annu. Rev. Biochem.*, 75:333–366, 2006.

[4] Dennis J Selkoe. Folding proteins in fatal ways. *Nature*, 426(6968):900–904, 2003.

[5] Johan Åqvist, Carmen Medina, and Jan-Erik Samuelsson. A new method for predicting binding affinity in computer-aided drug design. *Protein engineering*, 7(3):385–391, 1994.

[6] Johannes Söding, Andreas Biegert, and Andrei N Lupas. The hhpred interactive server for protein homology detection and structure prediction. *Nucleic acids research*, 33(suppl 2):W244–W248, 2005.

[7] Kristian W Kaufmann, Gordon H Lemmon, Samuel L DeLuca, Jonathan H Sheehan, and Jens Meiler. Practically useful: what the rosetta protein modeling suite can do for you. *Biochemistry*, 49(14):2987–2998, 2010.

[8] Carol A Rohl, Charlie EM Strauss, Kira MS Misura, and David Baker. Protein structure prediction using rosetta. *Methods in enzymology*, 383:66–93, 2004.

[9] Yoshimi Fujitsuka, George Chikenji, and Shoji Takada. Simfold energy function for de novo protein structure prediction: consensus with rosetta. *Proteins: Structure, Function, and Bioinformatics*, 62(2):381–398, 2006.

[10] James C Phillips, Rosemary Braun, Wei Wang, James Gumbart, Emad Tajkhorshid, Elizabeth Villa, Christophe Chipot, Robert D Skeel, Laxmikant Kale, and Klaus Schulten. Scalable molecular dynamics with namd. *Journal of computational chemistry*, 26(16):1781–1802, 2005.

[11] JL Klepeis and CA Floudas. Astro-fold: a combinatorial and global optimization framework for ab initio prediction of three-dimensional structures of proteins from the amino acid sequence. *Biophysical Journal*, 85(4):2119–2146, 2003.

[12] David Van Der Spoel, Erik Lindahl, Berk Hess, Gerrit Groenhof, Alan E Mark, and Herman JC Berendsen. Gromacs: fast, flexible, and free. *Journal of computational chemistry*, 26(16):1701–1718, 2005.

[13] Sander Pronk, Szilárd Páll, Roland Schulz, Per Larsson, Pär Bjelkmar, Rossen Apostolov, Michael R Shirts, Jeremy C Smith, Peter M Kasson, David van der Spoel, et al. Gromacs 4.5: a high-throughput and highly parallel open source molecular simulation toolkit. *Bioinformatics*, page btt055, 2013.

[14] Wolfgang Kabsch and Christian Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22(12):2577–2637, 1983.

[15] Alexander S Spirin, Igor N Serdyuk, Joseph L Shpungin, and Victor D Vasiliev. Quaternary structure of the ribosomal 30s subunit: model and its experimental testing. *Proceedings of the National Academy of Sciences*, 76(10):4867–4871, 1979.

[16] David J Evans. The bionanoscience of plant viruses: templates and synthons for new materials. *Journal of Materials Chemistry*, 18(32):3746–3754, 2008.

[17] Sunanda Chatterjee, Rituparna Sinha Roy, and P Balaram. Expanding the polypeptide backbone: hydrogen-bonded conformations in hybrid polypeptides containing the higher homologues of α-amino acids. *Journal of The Royal Society Interface*, 4(15):587–606, 2007.

[18] Ai-Jun Li and Ruth Nussinov. A set of van der waals and coulombic radii of protein atoms for molecular and solvent-accessible surface calculation, packing evaluation, and docking. *Proteins: Structure, Function, and Bioinformatics*, 32(1):111–127, 1998.

[19] Daniel Seeliger and Bert L de Groot. Atomic contacts in protein structures. a detailed analysis of atomic radii, packing, and overlaps. *Proteins: Structure, Function, and Bioinformatics*, 68(3):595–601, 2007.

[20] Adam Zemla. Lga: a method for finding 3d similarities in protein structures. *Nucleic acids research*, 31(13):3370–3374, 2003.

[21] CA Floudas. Computational methods in protein structure prediction. *Biotechnology and bioengineering*, 97(2):207–213, 2007.

[22] Jooyoung Lee, Sitao Wu, and Yang Zhang. Ab initio protein structure prediction. In *From protein structure to function with bioinformatics*, pages 3–25. Springer, 2009.

[23] Marc A Martí-Renom, Ashley C Stuart, András Fiser, Roberto Sánchez, Francisco Melo, and Andrej Šali. Comparative protein structure modeling of genes and genomes. *Annual review of biophysics and biomolecular structure*, 29(1):291–325, 2000.

[24] Narayanan Eswar, Ben Webb, Marc A Marti-Renom, MS Madhusudhan, David Eramian, Min-yi Shen, Ursula Pieper, and Andrej Sali. Comparative protein structure modeling using modeller. *Current protocols in bioinformatics*, pages 5–6, 2006.

[25] Yang Zhang and Jeffrey Skolnick. Automated structure prediction of weakly homologous proteins on a genomic scale. *Proceedings of the National Academy of Sciences of the United States of America*, 101(20):7594–7599, 2004.

[26] Ambrish Roy, Alper Kucukural, and Yang Zhang. I-tasser: a unified platform for automated protein structure and function prediction. *Nature protocols*, 5(4):725–738, 2010.

[27] David Baker. Prediction and design of macromolecular structures and interactions. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 361(1467):459–463, 2006.

[28] Gongpu Zhao, Juan R Perilla, Ernest L Yufenyuy, Xin Meng, Bo Chen, Jiying Ning, Jinwoo Ahn, Angela M Gronenborn, Klaus Schulten, Christopher Aiken, et al. Mature hiv-1 capsid structure by cryo-electron microscopy and all-atom molecular dynamics. *Nature*, 497(7451):643–646, 2013.

[29] José C Calvo, Julio Ortega, and Mancia Anguita. Comparison of parallel multi-objective approaches to protein structure prediction. *The Journal of Supercomputing*, 58(2):253–260, 2011.

[30] Christiane Regina Soares Brasil, Alexandre Claudio Botazzo Delbem, and Fernando Luís Barroso da Silva. Multiobjective evolutionary algorithm with many tables for purely ab initio protein structure prediction. *Journal of computational chemistry*, 34(20):1719–1734, 2013.

[31] Ivan Dotu, Manuel Cebrian, Pascal Van Hentenryck, and Peter Clote. On lattice protein structure prediction revisited. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, 8(6):1620–1632, 2011.

[32] Glennie Helles. A comparative study of the reported performance of ab initio protein structure prediction algorithms. *Journal of the Royal Society Interface*, 5(21):387–396, 2008.

[33] Zukang Feng, Li Chen, Himabindu Maddula, Ozgur Akcan, Rose Oughtred, Helen M Berman, and John Westbrook. Ligand depot: a data warehouse for ligands bound to macromolecules. *Bioinformatics*, 20(13):2153–2155, 2004.

8 Appendix: Created Programs

During our research a lot of programs have been written in order to get a good data set. In our final results only a subset of these programs were used, since a few of the new programs made the old ones obsolete. Here we display the entire framework and explain its functionality and how to use them. Every program is listed and all its parameters are explained. All the programs were written for Linux (Ubuntu) and must be executed using a terminal. In Figure 9 the entire framework is presented.

AnalyzeDataset

`Usage:`
`./AnalyzeDataset <mmCIF-file directory> <output directory>`

From a directory filled with mmCIF-files, all the dihedral information is distilled from the structures and put in a the output directory.

AnalyzePDBDataset

`Usage:`
`./AnalyzePDBDataset <mmCIF-file directory> <output directory>`

The same as `AnalyzeDataset` but instead of a a directory filled with mmCIF-files the files are PDB-formatted. All the dihedral information is distilled from the structures and put in a the output directory.

CompareSS

`Usage:`
`CompareSS <original-file> <prediction>`

Compares the secondary structure of two 3D structures with one another. The first parameter is the target file, the second parameter should be the structure of the prediction.

`Usage:`
`./CreateEntireProteins.sh <PDB-dir> <output-dir>`

With some molecules, hydrogen atoms are missing. This program predicts the coordinates of all the missing atoms. Given the directory where all the pdb-formatted structures reside, the output directory will be filled with the structure without any missing atom.

FilterDataset

`Usage:`
`FilterDataset <mmCIF-filedir> <PDB-filedir> <output directory> <filter-file>`

Using two input directories were structures are stored in mmCIF format (mmCIF-filedir) and PDB format (PDB-filedir), this program traverses through all the structure and checks whether one of the structures in the mmCIF file matches all the criteria in the filter file. If so, both the mmCIF-file and PDB-file of the same structure are copied to the output directory. An example of a filter file:

```
File 1, Entity 2
'exptl' 'method' 'X-RAY DIFFRACTION'
'entity_poly' 'type' 'polypeptide(L)'
'entity_src_gen' 'pdbx_gene_src_scientific_name' 'Escherichia coli'
```

In the fist line, the number of file-specific criteria and entity specific criteria are stated. A criterion is formatted in the following way:
`'table_name' 'column_name' 'value'`
using the structure of a mmCIF file. A criterion is matched if in table `table_name` the column `column_name` has the value `value`. First all the file specific criteria are listed, than the entity specific.

LGA

`Usage:`
`LGA <first-pdb-file> <second-pdb-file> <start> <length>`

Calculates the LGA-score for two proteins of equal length. The file names of the two structures can be used for parameter 1 and 2. The LGA can also be calculated for a continuous subset of the proteins. The 3rd parameter states where to start the and the length parameter can be used to specify the length of the sequence. Note that the begin of a sequence is 0.

MergePDBmmCIF

```
Usage:
MergePDBmmCIF <mmCIF-file directory> <ProtonatedPDB-files_dir> <output directory> <hydrogen/all>
```

Merges the protonated information which is stored in a PDB formatted files with the same structure without the hydrogen atoms which is in a mmCIF-formatted files and the result is store in an output directory. Whether only missing hydrogen atoms or all missing atoms must be merged, can be set with the last parameter. This program has been become obsolete.

PITA

```
Usage:
PITA <fragment-database-directory> <PSF-dir> <fasta-file> <no-energy/energy>
                  <function-number> <E-exp> <pruning-K> <BoundingSphere>
                  <ideal/fragment> <PDB model> <output-dir> <max-seconds>
```

This program predicts the three dimensional structure of the 1-dimensional sequence given by *fasta-file*. It uses the fragments in the directory *fragment-database-directory*. The 4th parameter should be *energy* for the energy search, *no-energy* for the search only based on the fragments and steric hindrance. For the energy search, the directory with all the PSF files should be given (created with `CreatePSFs`). With *function-number*, the EAF can be selected, 0 for none, 1 for EAF1 and 2 for EAF2. The α for the EAF can be set using the E-exp. The pruning-K parameter can be set with *pruning-k*, α of the BMS-algorithm can be set with *BoundingSphere*, if 0, this is turned off. The first residue is created either based on the ideal shape of that residue or based on a fragment from the database. To compare the final prediction, the (3D) target model *PDB model* must be given to the program. All the predicted structure are saved in *output-dir* and after *max-seconds*, the program will stop.

9 Appendix: Framework Overview

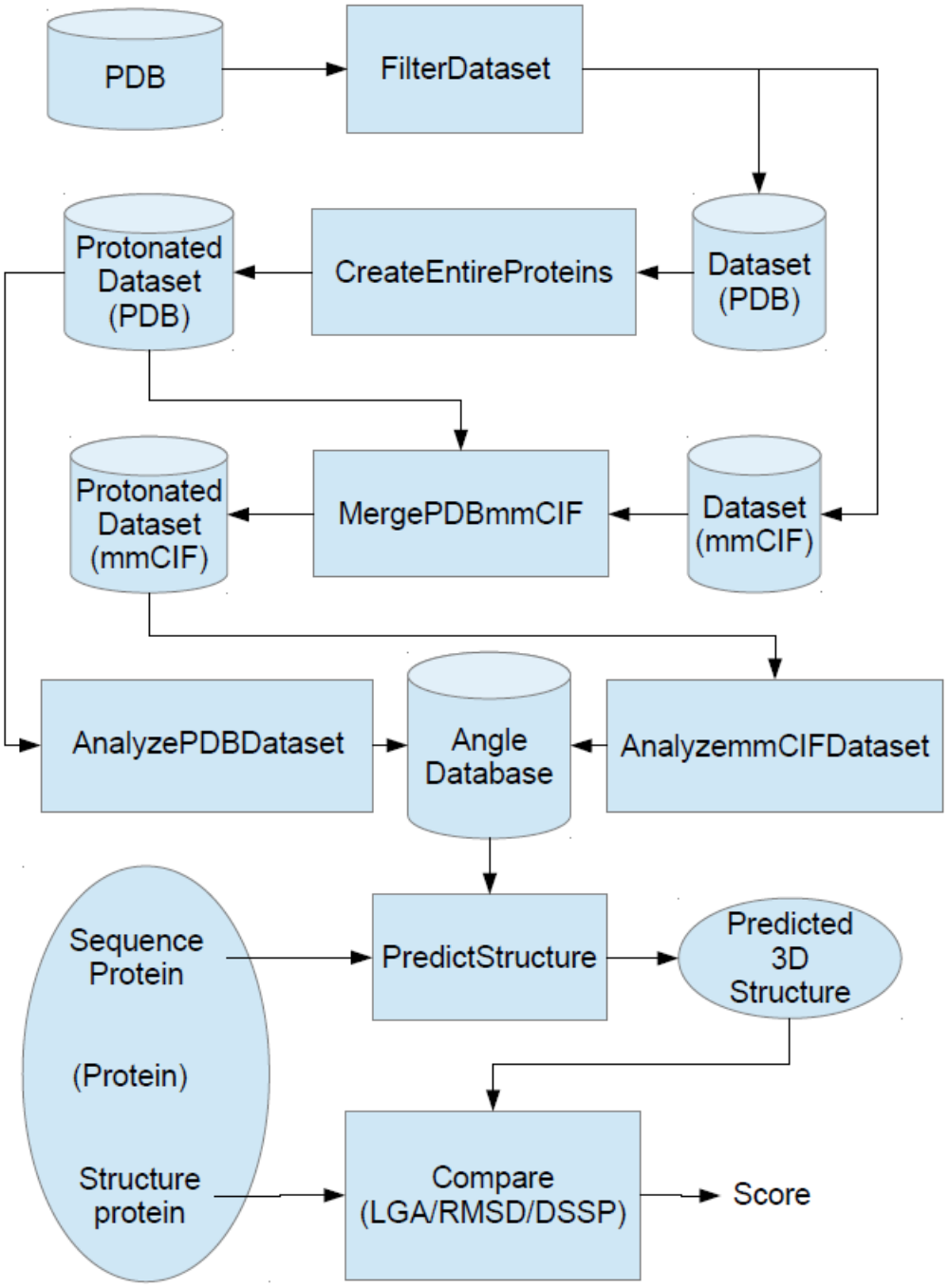


Figure 21: The entire framework PITA, showing the flow-chart of data and programs. Cylinders represent data sets, boxes represent programs