AGH UNIVERSITY OF SCIENCE AND TECHNOLOGY
Faculty of Physics and Applied Computer Science

Master's thesis

Zbigniew Baster

major: **Medical physics**

specialization: **Dosimetry and Electronics in Medicine**

The identification and the elimination of clashes in the structure of an Early-Stage intermediate in the protein folding process.

Supervisor: **Prof. Irena Roterman-Konieczna, PhD, DSc**

Cracow, September 2013

Aware of criminal liability for making untrue statements, I declare that the following thesis was written personally by myself and that I did not use any sources but the ones mentioned in the dissertation itself.

Cracow, September 2013

The subject of the master thesis and the internship by Zbigniew Baster, student of 2nd year of 2nd cycle studies, major in medical physics, specialization in dosimetry and electronics in medicine.

The subject of the master thesis: **The identification and the elimination of clashes in the structure of an Early-Stage intermediate in the protein folding process.**

Supervisor: Prof. Irena Roterman-Konieczna, PhD, DSc

Reviewer:

A place of the internship: Department of Bioinformatics and Telemedicine UJCM, Cracow

Program of the master thesis and the internship

1. Discussion with the supervisor on realization of the thesis.
2. Collecting and studying the references relevant to the thesis topic.
3. The internship:
   - getting to know the idea of the protein folding model use by Department of Bioinformatics and Telemedicine UJCM,
   - getting to know the idea of other protein folding models and their use in the industry,
   - discussion with the supervisor focused on the thesis goals,
   - work on the solution and the program,
   - preparation of the internship report.
4. Continuation of working on the thesis subject.
5. Ordering and first analysis of the calculation results.
6. Final analysis of the results obtained, conclusions – discussion with and final approval by the thesis supervisor.
7. Typesetting the thesis.

Dean's office delivery deadline: September 2013

…….. (signature of the department head)

…….. (signature of the supervisor)

Supervisor’s review

Reviewer’s review

ACKNOWLEDGMENT

I would like to express my sincere gratitude to
Department of Bioinformatics and Telemedicine UJCM,
especially to my supervisor Prof. Irena Roterman-Konieczna
for advising me during my studies,
and to Małgorzata Tomanek
from Academic Computer Centre Cyfronet AGH,
for sharing her results and opinions on my program.

My sincere thanks also go to my Canadian friend
and English teacher – Russell, and my family: Jan
and Anna for helping me editing this dissertation.

And I also want to thank my parents – because they are.

Someone told me that each equation I included in the book would halve the sales.
Stephen William Hawking

ABSTRACT

For many years, scientists have been trying to unravel the protein folding process. This paper presents a proposition of an improvement to one of models trying to describe it, the elliptical model, developed by the Department of Bioinformatics and Telemedicine UJ-CM [22]. The model assumes a division of a protein folding process into two stages: the Early-Stage (ES) and the Late-Stage (LS). After the first stage, the second one sometimes occurs unable to perform, because of accidentally created clashes between atoms.

This work demonstrates several possible solutions to remove clashes before proceeding to the LS.

Additionally, one of presented solutions describes mathematically the precession phenomenon, what might be useful in other than protein folding field of studies such as medical imaging, quantum physics or astronomy.

STRESZCZENIE

Na przestrzeni lat naukowcy starali się rozwikłać proces fałdowania białek. Poniższa praca prezentuje propozycję ulepszenia jednego z modeli starającego się to opisać, modelu eliptycznego, opracowanego przez Zakład Bioinformatyki i Telemedycyny CM-UJ [22]. Model zakłada podział fałdowania białek na dwa etapy: *wczesny etap* oraz *późny etap*. Czasami, po ukończeniu pierwszego etapu, drugi z nich jest niemożliwy do przeprowadzenia, z powodu powstałych między atomami przypadkowych kolizji.

Praca ta przedstawia kilka możliwych rozwiązań mających na celu usunięcie kolizji przed przystąpieniem do *późnego etapu*.

Dodatkowo, jedno z rozwiązań zawiera matematyczny opis zjawiska precesji, co może zostać wykorzystane w innych dziedzina nauki niż fałdowanie białek, takich jak obrazowanie medyczne, fizyka kwantowa czy astronomia.

TABLE OF CONTENTS

LIST OF PARAMETERS AND VARIABLES

R [Å]	– a radius of the curvature of a heptapeptide [6, 7]
V [°]	– a value of the dihedral angle between adjacent peptide bond planes in a heptapeptide [6, 7]
t [°]	– a parameter of the ellipse from the elliptic model
$\Phi(t)$ [°]	– a value of one of dihedral backbone angles [1]
$\Psi(t)$ [°]	– a value of one of dihedral backbone angles [1]
d_0 [Å]	– the distance of the reference
$d(a,b,c)$ [Å]	– a distance between atoms
x_1, y_1, z_1 [Å]	– coordinates of the first atom on the protein strand used to calculate a distance
x_2, y_2, z_2 [Å]	– coordinates of the second atom on the protein strand used to calculate a distance
$a(t)$ [Å]	– a function that represents a x-coordinate of the second clashed atom
$b(t)$ [Å]	– a function that represents a y-coordinate of the second clashed atom
$c(t)$ [Å]	– a function that represents a z-coordinate of the second clashed atom
r_1 [Å]	– a radius that corresponds with the shortest distance between the Φ-angle axis and the center of the rotation of the second clashed atom, with respect to the Ψ-angle axis
r_2 [Å]	– a radius that corresponds with a distance between the second clashed atom and the center of the rotation, with respect to the Ψ-angle axis
$P_{C\alpha}$ $(x_{C\alpha},y_{C\alpha},z_{C\alpha})$ ([Å],[Å],[Å])	– the point that represents C_α atom of the re-rotated residue in the Euclidean space
P_N (x_N,y_N,z_N) ([Å],[Å],[Å])	– the point that represents N atom of the re-rotated residue in the Euclidean space
P_C (x_C,y_C,z_C) ([Å],[Å],[Å])	– the point that represents C atom of the re-rotated residue in the Euclidean space
P_2 (x_2,y_2,z_2) ([Å],[Å],[Å])	– the point that represents the second clashed atom in the Euclidean space
P_o (x_o,y_o,z_o) ([Å],[Å],[Å])	– the point that represents the center of the rotation of point P_2, with respect to the Ψ-angle axis
$z_r(x)$ [Å]	– an ordinate of the rotation axis' linear equation
a_r	– a slope of the rotation axis' linear equation
$b_\perp$ [Å]	– a constant term of the normal line's linear equation
$z_\perp(x)$ [Å]	– an ordinate of the normal line's linear equation
$\Phi'(t)$ [°]	– a value of the Φ angle in the precession model

Φ_0 [°] – a value of the Φ dihedral angle at the beginning of a clash-remove process

$\Psi'(t)$ [°] – a value of the Ψ angle in the precession model

Ψ_0 [°] – a value of the Ψ dihedral angle at the beginning of a clash-remove process

Ψ'' [°] – a value additionally added to the value of the Ψ angle calculated with use of the precession model

r_z [Å] – a *z*-component of the r_2 radius for the Ψ' angle equal to zero

α [°] – an angle of inclination of the rotation axis

r_x [Å] – a semi-minor axis of an ellipse

r_y [Å] – a semi-major axis of an ellipse

$M(\Phi')$ – the rotation matrix in two dimensions

$\Psi_P'(t)$ [°] – a value of the Ψ angle for the arrangement, with proline as the re-rotated residue

LIST OF ACRONYMS

ES – Early-Stage

LS – Late-Stage

AADream – Amino Acid Dream

PDB – Protein Data Bank

PC – Personal Computer

PRNG – pseudorandom number generator

LIST OF FIGURES

LIST OF TABLES

1. Introduction

1.1. Biochemical basics

1.1.1. Protein structure

A protein is a large biomolecule, structured with hundreds and even thousands of amino acid residues linked linearly by peptide bonds. Its structure is responsible for its biological functions. The structure of a protein is described in four levels of organization:

- **Primary structure –** which is defined by an order of amino acid residues in the protein strand. Each protein can be determined by its primary structure.
- **Secondary structure –** so-called three-dimensional regular structures, stabilized by hydrogen bonds. The most common examples of secondary structures are α-helices and β-sheets (FIGURE 1).
- **Tertiary structure –** corresponds with a three-dimensional structure of a single protein strand and organisation of secondary structures, one related to the other.
- **Quaternary structure –** refers to subunits organization in proteins built of two or more of them. [1]

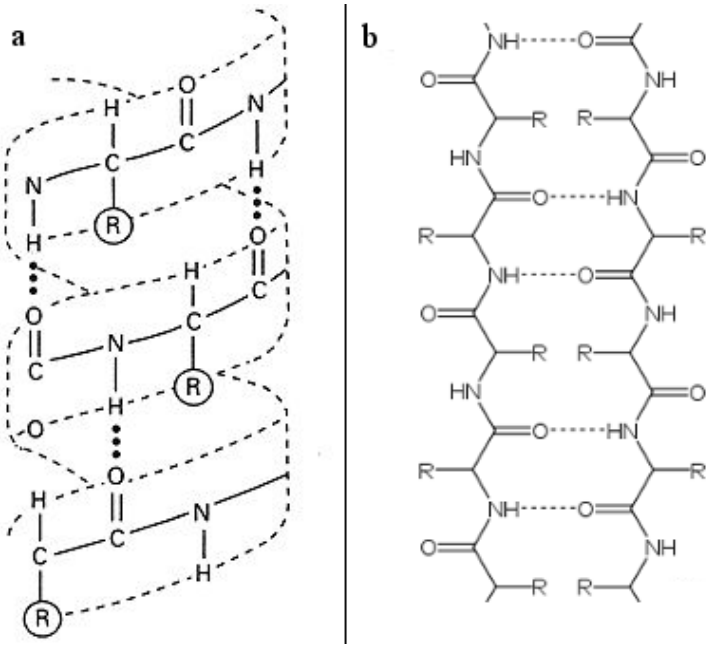


FIGURE 1. Protein secondary structures: a) an α-helix [2], b) an antiparallel β-sheet [3].

1.1.2. Representation of a protein structure with dihedral angles.

A protein secondary and tertiary structure can be approximately described with the backbone's dihedral angles (FIGURE 2). Usually, the usage of Φ and Ψ angles is sufficient; an ω angle in proteins usually takes the value of 180°, because of the mesomeric character of a peptide bond. [1]

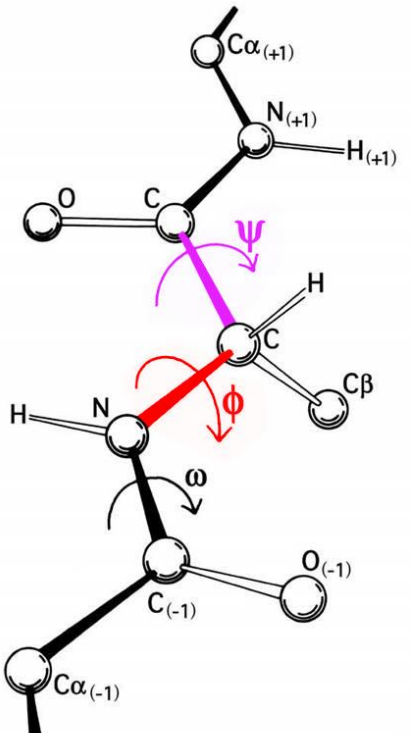


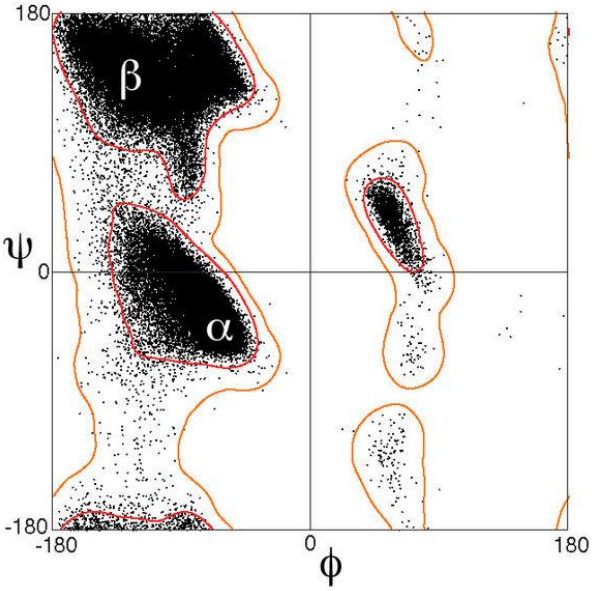
FIGURE 2. Dihedral angles in proteins. Based on [4].

1.1.3. Ramachandran plot

One of the ways to represent dihedral angles in proteins is the use of a Ramachandran plot; plotting Φ angle values against Ψ angle values of amino acid residues shows a distribution of dihedral angles in a protein.

As shown in FIGURE 3, not all combinations of dihedral angles values are possible in proteins. That phenomenon is caused by steric and torsional strains between atoms of a protein backbone. [1]

FIGURE 3. The Ramachandran plot for the general case. In the figure, regions marked with red frames contain the most common secondary structures. Therefore, it could be said that the structure of a protein is directly dependent on values of dihedral angles; the secondary protein structure can be predicted based on its dihedral angles. [5]



1.2. Bioinformatical basics

1.2.1. The elliptical model

Each point in a Ramachandran plot can be described with the use of two geometric parameters:

- ***R* [Å] –** a radius of the curvature of the construct, [6, 7]
- ***V* [^{o}] –** a value of the dihedral angle between adjacent peptide bond planes. The exact way to obtain this value was described in [6, 7].

In order to obtain these parameters, a peptide that contains seven residues of alanine must be created. All residues must have the same dihedral angles as the considered point in the plot. Based on that construct, it is possible to calculate desired values.

Arranging these parameters to low-energy structures, and plotting ln(*R*) in a dependence of *V* gives a square function shape (FIGURE 4), which can be approximated as:

$$\ln(R) = 0.00034\,V^2 - 0.02009V + 0.848. \quad \textbf{(1.1)}$$

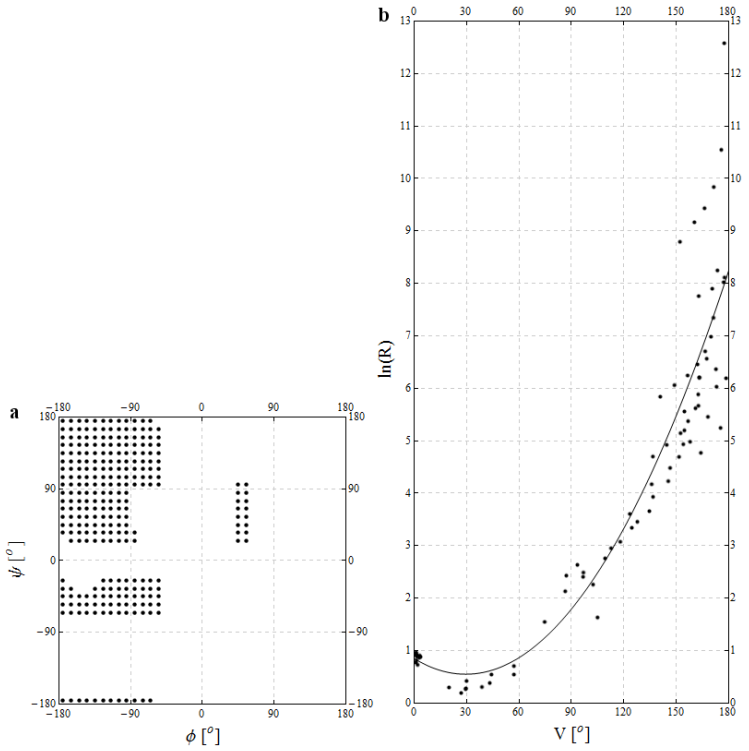


FIGURE 4. a) A Ramachandran map with a low-energy area on a 10^o grid; b) a ln(*R*) versus *V* angle value plot for points presented in (a). Based on [7].

An extension of the condition of satisfying equation (1.1) to all structures, with a dispersion tolerance at level of ± 0.2, gives another Ramachandran plot shown in FIGURE 5.a. An ellipse can be superimposed on its central points (FIGURE 5.b); these points can also be used as the start points for a structure optimization.

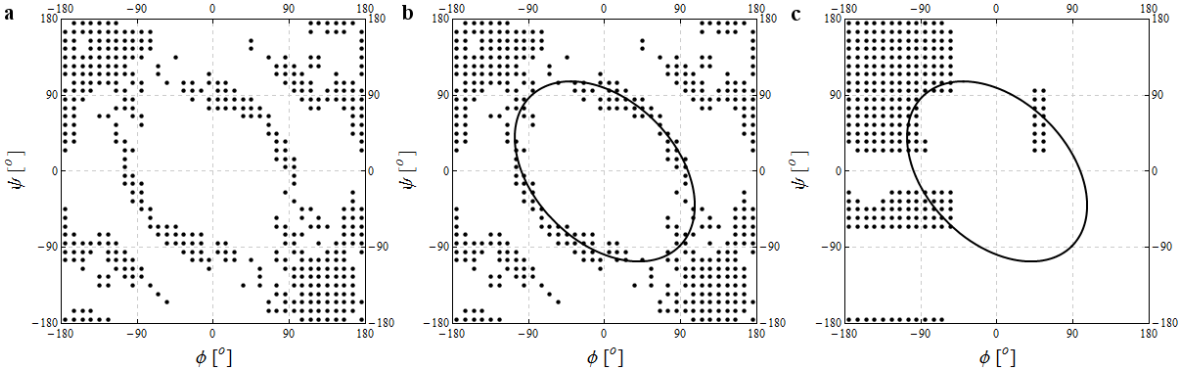


FIGURE 5. a) The result of an extension of equation (1.1) to all structures in a Ramachandran plot; b) the ellipse superimposed on results shown in (a); c) the ellipse linked with the low-energy area shown in FIGURE 4.a. Based on [7].

The parametric equations of the received ellipse-approximation in the function of values of dihedral angles are as follows [7,8]:

$$\Phi(t) = 125\ \cos(45^o)\cos(t) - 84\ \sin(45^o)\sin(t), \quad \textbf{(1.2)}$$

$$\Psi(t) = -125\ \sin(45^o)\cos(t) - 84\ \cos(45^o)\sin(t). \quad \textbf{(1.3)}$$

In a result, each pair of dihedral angles can be described with the use of only one parameter - *t*. More information about this theory is contained in [7].

1.2.2. Conformation of an Early-Stage (ES) intermediate of a protein

The elliptical model assumes that an *in silico* protein folding procedure can be started from dihedral conformations of amino acid residues that are included within the ellipse. It also assumes that an approximate start point for each residue can be determined from the primary structure of a protein. The structure received this way is called an Early-Stage intermediate of a protein, or just an Early-Stage structure.

In order to determine regularities between dihedral angles and an amino acid sequence, the Ramachandran map for the regular case was transformed into the elliptic path, by choosing the shortest way to reach it (FIGURE 6).

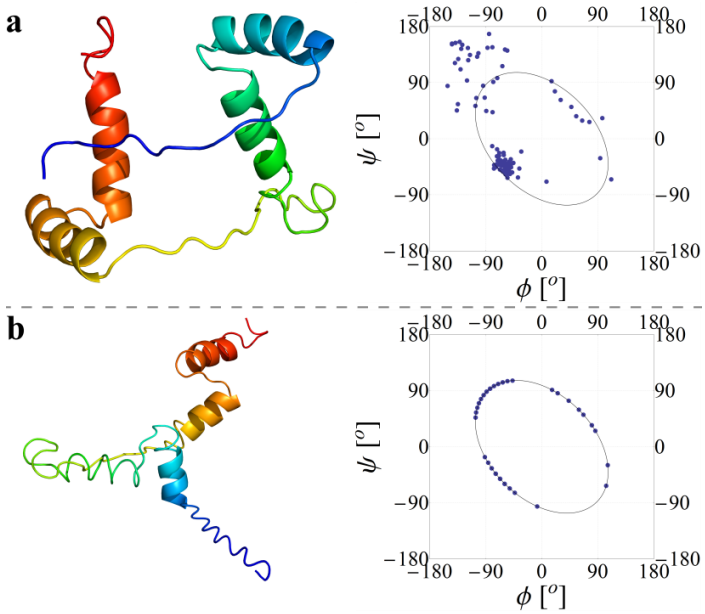


FIGURE 6. An example of a transformation of a protein into the elliptic path: a) the native structure of a protein with its amino acid residues placed in a Ramachandran plot; a) the structure and the Ramachandran plot of the protein after its approximation to the ellipse. Based on [9].

The transformation results in the probability distribution of a *t*- parameter. Furthermore, based on received maxima, the whole ellipse can be divided into seven conformational subspaces (FIGURE 7).

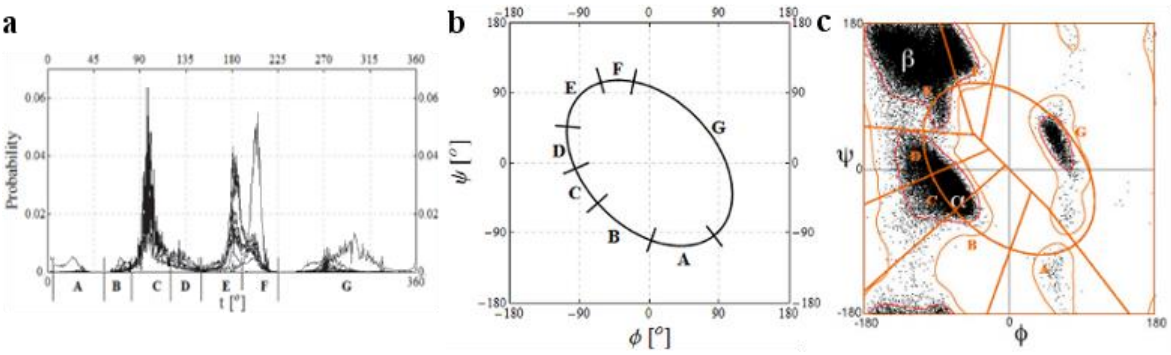


FIGURE 7. a) The probability distribution of a *t*- parameter of residues transformed into the ellipse path for twenty amino acids; b) the division of the ellipse shown in a Ramachandran plot; c) the divided ellipse superimposed on the Ramachandran plot for the general case. Based on [5,8,10].

1.2.3. Prediction of an Early-Stage protein structure

For many years, scientists have been trying to predict the structure of a protein. In [10] a model has been presented, which describes a method of receiving an ES protein structure. The model assumes that each amino acid residue can be assigned to one of seven points in the Ramachandran plot, which are maxima of conformational subspaces, described in 1.2.1 and 1.2.2. The model postulates that a prediction of secondary and tertiary structures can be based on a sequence of four amino acid residues.

1.2.4. Late-Stage (LS) of a protein structure prediction

A received ES structure is prepared for a hydrofobity density optimization called the Late-Stage (LS). This process is intended to obtain a native structure of a protein. The LS was described in [11-13].

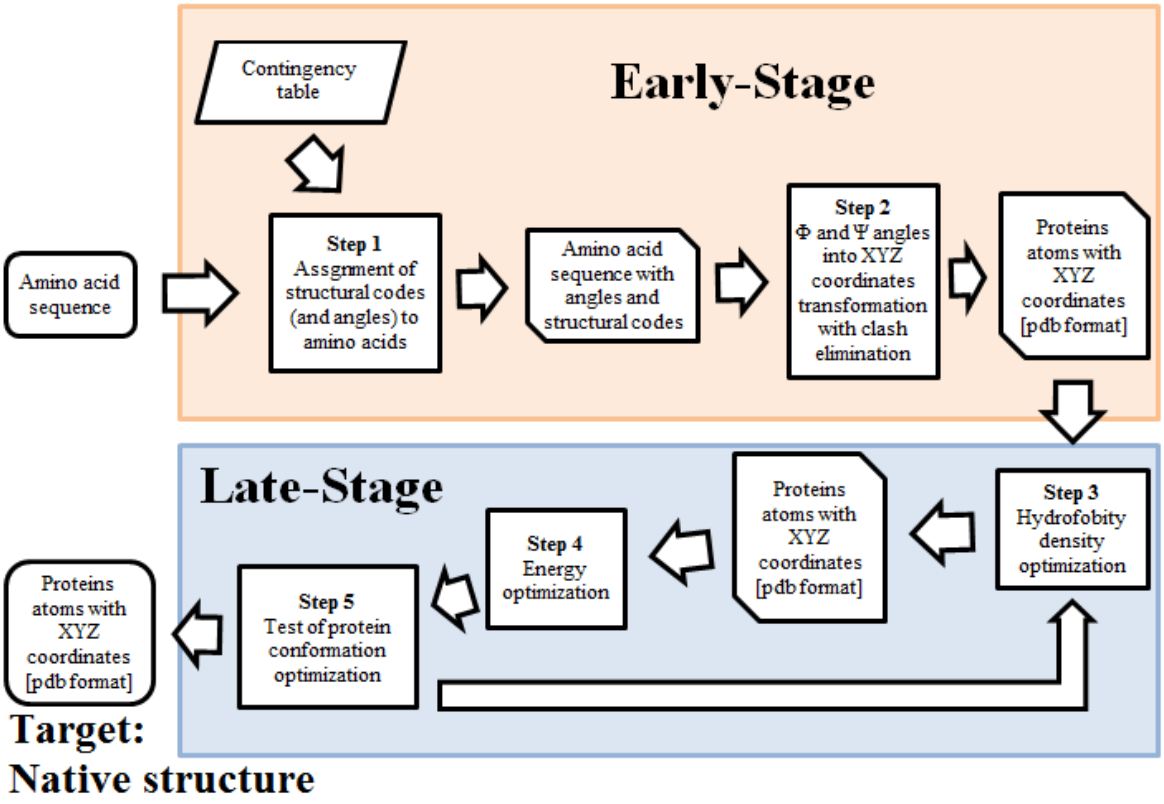


FIGURE 8. A schema of an *in silico* protein folding process. Based on [14].

2. Thesis statement

2.1. Problem statement

The possibility of a clash exists in an Early-Stage protein structure. **A clash** can be defined as a situation in which two unbounded atoms are closer than the reference distance.

This situation is possible because of the mathematical model of an ES protein structure. As described in the Introduction, the model is based only on four adjacent amino acid residues. Because of this, an eventuality that two distant atoms (in the term of their positions in the protein strand) can be found closer than it is normally possible without creating any kind of bond exists.

In some cases, clashes make impossible to conduct a hydrofobity density optimization. For these situations, clashes have to be found and removed before proceeding to the Late-Stage.

2.2. Thesis objective

The objective of this research is to find a solution that will solve, as many as it is possible, problems with eventual clashes created during the Early-Stage by changing dihedrals angles in a protein, in the most efficient way,. The solution has to fulfill all major restrictions, and should fulfill minor restrictions listed below. The solution is intended to be applied to a computer program. Most researches are performed on proteins of around two hundred residues; therefore, the solution should concentrate on clashes within small molecules.

2.3. Major restrictions

The main major restriction, for a clash removal process, is that each amino acid residue must maintain on the ellipse path, in its conformational subspace.

The second restriction implies that the change of dihedral angles, at first, should be performed in residues from zones A, B or G, and subsequently in residues from zones D or F. Residues from C and E zones should be considered at the end or not considered at all.

Results shown in [10] indicate that the biggest probability of a proper classification of amino acid residues is for zones C and E. Changing dihedral angles in residues,

moves them from their maxima in their subspaces, causing a decrease of the probability of a proper optimization in the Late-Stage. Also, these subspaces correspond with most common secondary structures on a Ramachandran plot; α-helix for zone C and β-sheet for zone E. Small dihedral angle changes might influence on the secondary and the tertiary structures. Connecting this fact with the highest probability of a proper classification, a modification of these molecules is the least desirable.

Zone D represents the so-called bridge region, the importance of which was mentioned in [15]. Zone F is associated with β-like motifs; however it is sometimes counted as a β-structure. These subspaces show an intermediate proper classification probability.

Zones A and B refer to poorly ordered structures traditionally called random coils. Part of zone G also refers to poorly ordered structures, but its central section represents a left handed helix. This is also the biggest subspace, which covers around 35% of the ellipse path (depends on an amino acid residue).

The third major restriction implies that the program must report unsolvable situations.

2.4. Minor restrictions

The first minor restriction is that the program should returns reproducible results. This restriction is dictated by the fact that the model based on nonrandom processes is easier to investigate and to improve upon.

The second minor restriction refers to the simplicity of the solution. Simple solutions are easier for eventual bug-fixes and are also easier to improve upon.

3. Materials and methods

3.1. Subspace information

Information about ranges of conformational subspaces in the ellipse and zones' maxima for each amino acid residue were provided by the supervisor of the thesis.

3.2. Base program

The received solution will be applied to a currently existing program – Amino Acid Dream 1.00 (AADream). The program was described in [16] and its code is available for download from [17]. The primary function of this program is to create Protein Data Base (PDB) format files, containing Euclidean coordinates of atoms of proteins, based on input files containing a primary structure and dihedral angles.

3.3. Programing language

Because the base program was written in JAVA, all modifications were made in this programing language.

3.4. Computer

During the work and tests, a Personal Computer (PC) was used – notebook ASUS NJ61JQ-JX096 with:

- Processor - Intel Core i7 740QM (1.73GHz)
- Memory – 4GB DDR3 RAM
- Graphic card - ATI Mobility Radeon HD 5730 with 1GB VRAM
- Operating system – Windows 7 Professional 64x SP1
- Hard drive – Seagate Momentus 5400.7 ST9640320AS 640 GB with approximately 200 GB of free space
- Other software: Wolfram Mathematica 8.0, NetBeans IDE 7.3.1

4. Modifications of the base program

4.1. Bug-fixing

During the work on modifications, some minor bugs and misspellings were found; they were fixed.

4.2. Layout modifications

Due to the application of new features, some layout modifications had to be done. Figures comparing old and new layouts can be found in Appendix A (FIGURE 25 and FIGURE 26).

4.3. New features

Features *a-r* are the same as in the base program; they were described in [16]. All new features of the program are marked with black frames, with bolded descriptions in FIGURE 26 (Appendix A):

A. Create an output file without a clash-fix. A selection of whether the program will create a standard AADream 1.00 output file. The output file's name is built of the input file's name and "*.pdb*" as a suffix.

B. Search for clashes only in a protein backbone. A selection of whether the program will search for clashes, only between atoms of a protein backbone. This option is inactive if **check-box C** is not selected.

C. A selection of weather a clash-search will be conducted. After a search the program will create an additional file in the directory typed in **field G**. The output file's name is built of the input file's name, with "*_c*" added to its end, and "*.pdb*" as a suffix.

D. A clash reference distance (expressed in ångströms)**.** If two atoms, which are not from the same or adjacent residues, were found closer than this value, the program will interpret this as a clash. The program does not accept values lower than 1 Å. This option is inactive if **check-box C** is not selected.

E. Search for clashes with hydrogen atoms. The program normally skips hydrogen atoms during a clash-search. This option is inactive if **check-box C** is not selected and **check-box B** is selected.

F. **Create a file containing a clash-fix report.** The program can additionally create a report file from a clash-fix process. A file will be created in the directory typed in **field I**. The report file's name is built of the input file's name and "*.log*" as a suffix. A report file is described in Appendix B.

G. **An output path for clash-fixed files.** A path to the directory in which files will be created after a clash-fix process. This option is inactive if **check-box C** is not selected.

H. **A selection of an output path for clash-fixed files.** Pressing this button causes an appearance of an additional dialog window, which can be used to define a new output path for clash-fixed files. Choosing a new path this way implicates a change of **field G**. This option is inactive if **check-box C** is not selected. For more information see [16], page 11.

I. **An output path for report files.** A path to the directory in which report files will be created. This option is inactive if **check-box C and F** are not selected.

J. **A selection of an output path for report files.** Pressing this button causes an appearance of an additional dialog window, which can be used to define a new output path. Choosing a new path this way implicates a change of **field I**. This option is inactive if **check-boxes C and F** are not selected. For more information see [16], page 11.

Additionally, a dialog window that informs about nonexistent output directory (described in [16], section 2.2.1.s) was changed.

FIGURE 9. A dialog window that informs about a nonexistent directory.

FIGURE 9 shows the exemplary nonexistent directory window. By the selection of the first option a new path-defining dialog window will be opened (see [16], page 11). After selecting the second option, a new directory will be created with the same path as typed in **field G, I, d or f**, dependent on the case. The last option stops calculations.

5. Global solution

In general, the solution assumes that each clash can be removed by changing only one amino acid residues' dihedral angles; in situations where changing of one residue would be insufficient to solve the clash, the program will select another residue, and follow the same procedure, separately from the first selection. If the clash still remains, the program will repeat this procedure on the remaining possible residues.

The solution can be divided into two parts: the global and the local solution. The global solution refers to the method of finding a clash and selecting a residue, which dihedral angles will be changed. The local solution, described in section 6, refers to the method of changing dihedral angles in a residue that was previously selected by the global solution.

5.1. Distance between atoms

As stated earlier, a clash is a situation where two atoms that are not bonded to each other, are closer than the reference distance (***d_0***). Because of the limited possibility of bending the backbone and a constancy of a side chain of residues in this model, the program will not search for clashes between atoms of the same residue or of adjacent ones.

A distance between two atoms can be calculated based on their coordinates in the Euclidean space:

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}, \tag{5.1}$$

where:

d [Å] – a calculated distance between atoms,

x_1, y_1, z_1 [Å] – coordinates of the first atom,

x_2, y_2, z_2 [Å] – coordinates of the second atom.

Because of a calculating speed purpose, instead of using a distance described by equation (5.1), the program uses a squared distance; this procedure allows the program to omit a root, and to speed up calculations, without losing any significant data.

The comparison of the squared reference and the squared calculated distance gives information of a clash appearance. Inequality (5.2) describes the condition in which a clash is detected:

$$d_0{}^2 > (x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2. \quad \textbf{(5.2)}$$

5.2. Determining subspace of an amino acid residue

To determine the order of amino acid residues that will be used in the clash-removal process each residue must, at first, be assigned to an appropriate subspace. Based on data included in input file about values of dihedral angles, data provided by the supervisor, and equations (1.2) and (1.3), a proper assignment can be made.

In order to assign a residue to a proper subspace, the value of its *t*-parameter must be known; it can be determined by a transformation of parametrical equations of the ellipse:

$$\Phi = 125\ \cos(45^o)\cos[\,t(\Phi,\Psi)] - 84\ \sin(45^o)\sin[\,t(\Phi,\Psi)], \quad \textbf{(5.3)}$$

$$\Psi = -125\ \sin(45^o)\cos[\,t(\Phi,\Psi)] - 84\ \cos(45^o)\sin[\,t(\Phi,\Psi)]. \quad \textbf{(5.4)}$$

The substitution of trigonometric functions with their values results in the equations:

$$\Phi = \frac{125}{\sqrt{2}}\ \cos[\,t(\Phi,\Psi)] - \frac{84}{\sqrt{2}}\sin[\,t(\Phi,\Psi)], \quad \textbf{(5.5)}$$

$$\Psi = -\frac{125}{\sqrt{2}}\cos[\,t(\Phi,\Psi)] - \frac{84}{\sqrt{2}}\sin[\,t(\Phi,\Psi)]. \quad \textbf{(5.6)}$$

Afterwards, by subtracting these equations one from the other, and transforming the result, a function describing a *t*-parameter can be received:

$$\Phi - \Psi = \frac{250}{\sqrt{2}}\ \cos[\,t(\Phi,\Psi)], \quad \textbf{(5.7)}$$

$$\cos[\,t(\Phi,\Psi)] = \frac{(\Phi - \Psi)\sqrt{2}}{250}, \quad \textbf{(5.8)}$$

$$t(\Phi,\Psi) = \arccos\left[\frac{(\Phi - \Psi)\sqrt{2}}{250}\right] \quad \textbf{(5.9)} \quad \vee \quad t(\Phi,\Psi) = -\arccos\left[\frac{(\Phi - \Psi)\sqrt{2}}{250}\right]. \quad \textbf{(5.10)}$$

Because of a cyclic property of a *t*-parameter, equation (5.10) can be written as:

$$t(\Phi,\Psi) = 360^o - \arccos\left[\frac{(\Phi-\Psi)\sqrt{2}}{250}\right]. \tag{5.11}$$

In order to determinate ranges of these functions, an analysis was performed. For each value of a *t*-parameter, between 0^o and 360^o with a step of 10^o, based on equations (1.2) and (1.3), values of the Φ and the Ψ angle were calculated; subsequently, based on received dihedral angles, both possible values of a *t*-parameter were calculated with use of equations (5.9) and (5.11). The sum of both dihedral angles was counted, in addition to other results; all results are presented in TABLE 8 in Appendix C.

With these results, it can be concluded that:

$$t(\Phi,\Psi) = \arccos\left[\frac{(\Phi-\Psi)\sqrt{2}}{250}\right] \quad for \quad \Phi(t)+\Psi(t) \le 0 \tag{5.12}$$

and:

$$t(\Phi,\Psi) = 360^o - \arccos\left[\frac{(\Phi-\Psi)\sqrt{2}}{250}\right] \quad for \quad \Phi(t)+\Psi(t) > 0. \tag{5.13}$$

5.3. The procedure of the global solution

The diagram in FIGURE 28 in Appendix D shows the global solution's procedure in general. The part marked with red shape is extended in FIGURE 29, which can also be found in Appendix D.

After creating a protein with applied dihedral angles (the process was described in [16]), the program can begin the clash-search. At first, new "black lists" (which are described in section 5.5) have to be initiated and declared. Subsequently, the loop is initiated, in which the program searches for clashes between all atoms in the protein[1] (besides atoms from the same or adjacent residues). If a clash occurs, the program will try to remove it. In situations, where the clash is possible to remove, the program will do so, and restart the loop while maintaining the number of the iteration; if the clash is unable to be remove, the

[1] Depends on options chosen during initiation of the program, not all atoms might be considered during a clash search. (see sections 4.3.B and 4.3.E)

program will either restart the loop while maintaining the number of the iteration, in situations that at least one unsuccessful attempt to receive the solution was made, or it continue the loop, in situations that no angle was changed. If, during the iteration, at least clash was found and the program made at least one attempt to remove it, or if it is from the second to the fourth iteration[1], the program will start another iteration, after this one; otherwise, the procedure will end. Additionally, the program does not search for clashes between atoms if they are in different subunits of the protein[2].

FIGURE 29 shows how the program selects a residue to solve a clash. At first, it signs residues that are in the strand between clashed atoms, to one of three groups that are corresponding with zones of the ellipse: group ABG, DF or CE. Afterwards, it sorts them in their groups by their distance from the clash[3]. At the beginning, the procedure uses the most distant residue, due to the parallax effect, which makes the clash-remove process more efficient; greater distance between the residue and the clash will result in a greater effect after the change of dihedral angles. After the segregation, the program choses the first residue to remove the clash with; at first, it takes the most distant residue from group ABG, if the attempt is unsuccessful, it removes this residue from this group, and (keeping changes made in previous step) it takes another residue from this group, and if the group is empty, it repeats this procedure on group DF. At this stage of research, the program does not use group CE during a clash-remove process, because of purposes discussed in section 2.3.

In situation, where the program is unable to solve the clash, it creates a report of this event[4]; the report file is described in Appendix B. Independently of whether the program is able to solve the problem, it also shows the result in the information window (field *n* in FIGURE 26).

5.4. The “Ping-Pong” error

One of the most problematic situations of the global solution was the possibility of the occurrence of the so-called “Ping-Pong” error. This error describes an event in which a clash-removal process causes the appearance of another clash, the removal of

[1] The purpose of this procedure is described in section 5.5.
[2] One of input types allows to define a residue’s subunit (for more information see [16], section 2.2.2)
[3] A distance between the residue and the clash is defined as a distance between backbone’s nitrogen atoms of that residue and the first residue of the clash.
[4] If such an option was chosen during the initiation of the program (see section 4.3.F).

which causes a return of the first clash; the second attempt to solve this problem causes the second clash again, and so on. In order to fix this problem, the so-called “black lists” have been used.

5.5. The “black lists”

The “black lists,” also called as “flags,” contain information about situations in which either a clash-remove process will not be conducted, or one concrete atom will not be used to solve a clash. Depends on a list and a number of the iteration, data might be erased from a list, or might not be recorded by it. There are four types of “black lists.”

5.5.1. The “clash black list”

The procedure assumes that a single clash between the same two atoms, can be considered only once during one iteration. After finding a clash, the program adds to the list sufficient information to recognize this clash in further procedures, and if the program finds it again during the same iteration, the clash will be omitted. There are two situations where the program might find the same clash again: a clash is impossible to remove, or if the solution of another clash creates the first clash again, what potentially leads to the “Ping-Pong” error.

Data is always recorded to this list, and it is cleared after each iteration.

5.5.2. The “dead-end black list”

This list contains data about clashes that cannot be solved. Therefore, the program will omit these clashes during the work for the purpose of speeding up calculations.

Data is recorded during the first, the fifth and subsequent iterations; the reason for this is described in section 5.5.5. This list is never cleared.

5.5.3. The “clash-rotation black list”

Solving the “Ping-Pong” error was the main purpose of a creation of the “black lists”. This “black list’s” main order is to try to find alternative solutions for clashes that are creating this error. This list stores information about clashed atoms, and, also, about the residue that was used to solve the clash between them. After

the occurrence of the “Ping-Pong” error, because each residue has various range of movement, and a change of its dihedral angles will probably have a different result, the program will try to use a different one to solve the problem.

Data is always collected, and it is removed after the first and the forth iteration.

5.5.4. The “rotation black list”

The “rotation black list” keeps data about residues that have been used in the clash-remove process. Its purpose is to make possibility of a proper ending of the program in problematic situations. Residues that are stored in this list will not be used in any attempt to remove a clash. It unfortunately implies that some of clashes might not be removed, but it probably will be only an occasional situation.

This list collects data after the forth iteration. It is never cleared.

5.5.5. Summary of the “black lists”

TABLE 1 contains the summary of the “black lists”:

TABLE 1. The summary of the "black lists".

A name of the “black list”	Contained data	Recording	Cleaning	Purpose
“clash”	Clashed residues	Always	After each iteration	An avoidance of checking the same clash twice during one iteration
“dead-end”	Clashed residues	During the first, the fifth and subsequent iterations	Never	An avoidance of looking for the solution in unsolvable situations
“clash-rotation”	Clashed residues and rotated residue	Always	After the first and the forth iteration	An avoidance of the “Ping-Pong” error
“rotation”	Rotated residue	Since the fifth iteration	Never	A proper ending of the program in problematic situations

The cooperation of all of the “black lists” is shown in two examples; both of them assume the existence of a clash, which removal creates a second clash, and the removal of the second clash creates the first again. To solve each of these

clashes, a change of dihedral angles in only one residue is needed. Each of them can be one of two types: a clash that might be solved with the use of one of two possible residues, and a clash that might be solved only with use of one residue, which is the same one that is used in the first type (including preference which of these two residues would be used first). Additionally, each clash must be a different type than the other one. Examples assume also that results of the removal are different for each residue that can be used to remove a clash, and the solution with the use of the more preferred residue is the one that leads to the "Ping-Pong" error, while the use of the second one dose not.

Let a clash that could be solved with the use of two residues, be called *clash 12*, and the other one *clash 1*. Also, let a more preferred residue be called *residue 1* and the second one *residue 2*. TABLE 2 shows the procedure for two situations: one, where the first clash found is *clash 12*, and the other where it is the second one.

TABLE 2. The example of the usage of the "black lists".

Iteration	**Example 1**		**Example 2**	
	First clash	**Second clash**	**First clash**	**Second clash**
	clash 12	*clash 1*	*clash 1*	*clash 12*
	Actions		**Actions**	
1	Removal with *residue 1*, creation of *clash 1*	Removal with *residue 2*, creation of *clash 12*	Removal with *residue 1*, creation of *clash 12*	Removal with *residue 2*, creation of *clash 1*
2	Removal with *residue 1*, creation of *clash 1*	Removal with *residue 2*, creation of *clash 12*	Removal with *residue 1*, creation of *clash 12*	Removal with *residue 2*, creation of *clash 1*
3	Removal with *residue 2*	Removed	Not removed	Removed
4	Removed	Removed	Not removed	Removed
5	Removed	Removed	Removal with *residue 1*, creation of *clash 12*	Removal with *residue 2*
6	Removed	Removed	Removed	Removed

In **Example 1**, in **the first iteration**, at first *clash 12* is removed with the use of *residue 1*, with the creation of *clash 1*. Afterwards, *clash 1* is removed with the use of the same residue, with the creation of *clash 12*. **The "clash-rotation black list"** is cleared.

The second iteration, in the example, looks exactly the same as the first one. This iteration was not skipped, because parameters for *clash 1* from the first and the second iteration might be slightly different. Therefore, in the second iteration, this can result in the setting *residue 2* as being more preferred to solve that clash with than the other one. The example assumes that this does not happen. Additionally, in this iteration, data written to **the "clash-rotation black list" is not cleared**, and it will not be until the fifth iteration, therefore, any of these clashes cannot be solved with the use of *residue 1* until the fifth iteration.

In **the third iteration**, because of the impossibility to remove *clash 12* with the use of *residue 1*, it is removed with *residue 2*. In this situation, the second clash is not created, though, this situation is solved.

In Example 2, the first and the second iterations look similar as in Example 1.

In **the third iteration**, because the first clash – *clash 1* cannot be removed by any other residue it stays temporary unsolved. In this situation, the program cannot do anything with the clash, so, normally, it should add it to **the "dead-end black list"**, but as it was described, it does not collect data between the second and the fourth iteration[1].

The fourth iteration is an additional iteration, similar to the third, planned for more complex situations than described in the examples above, such as ones with clashes including more possible residues to re-rotate their dihedral angles, or with more clashes involved. **The "clash-rotation black list"** is cleared after this iteration.

In **the fifth iteration**, as it was in the first and the second, *clash 1* is removed with the use of *residue 1*, and *clash 12* is created. Unlike the first iteration, now *residue 1* is added to **the "rotation black list"**, therefore, it cannot be used in any clash-remove process in the future. Subsequently, *clash 12* is

[1] In the second and the fourth iterations the probability of the occurrence of the similar situation as in the third iteration exists.

removed with the use of *residue 2*. In this situation the first clash is not created, though, this situation is solved.

The examples of the use of the "black lists" show that they are essential in solving some of the clashes. In iterations from the second to the forth some potentially solvable clashes can be marked as unsolvable. Due to this fact, the "dead-end black list" does not collect data in these iterations.

5.6. Tests

All final tests were performed with the use of the whole program, with The hybrid solution (see section 6.4) as a local solution. Tests were conducted, both, by the author on the Personal Computer (see section 3.4), and by a person not directly connected with this master's thesis, with access to the Academic Computer Centre Cyfronet AGH (for more information see [18]).

On the PC, clashes were searched for: in the whole strand, in the strand excluding hydrogen atoms, and only in the backbone. In all options, the reference distance of 4 Å was used; this distance refers to the disappearance of bonds of all kinds between atoms. Additionally, in order to determine correctness of interpreting the reference distance, tests with the distance of 2 Å and 2.5 Å were performed. On the Cyfronet's computers, the search covered only the whole strand with the reference distance of 1.8 Å; this distance is identified with the disappearance of a covalent bond.

The author prepared some basic files for tests; files containing data of short (about 10 residues) polypeptides; most of them contained at least one clash, but some did not; they were treated as a negative control. These files were used only on the PC. The rest of the files that contained information of various proteins were provided by the supervisor, and used, both, on the PC and the Cyfronet's computers.

5.7. Results

Basic tests on files containing short polypeptides showed that the program always recognizes a clash and has no false-positive recognitions. Additionally, results showed that the program properly interprets the reference distance.

Unfortunately, in test on files containing data of proteins, some clashes occurred to be unsolvable in any way described by this solution; some of them, because of too high value of the reference distance, which was higher than the biggest distance that clashed

residues were possible to be displaced while maintaining all major restrictions, some of them because of coupling with other clashes.

All major restrictions are fulfilled in this solution.

6. Local solution

As described in section 5, the local solution concentrates on the procedure of changing dihedrals angles in the residue selected by the global solution. During the work four local solutions, described below, were invented.

6.1. The solution with use of the pseudorandom number generator (PRNG)

The first solution that was tested used the PRNG. The program drew a pseudorandom value of a *t*-parameter from the chosen residue zone[1] and, based on it, calculated new values for dihedral angles, and then applied them to the protein. If the change was insufficient to solve the crash, the program drew another pseudorandom value, and followed the procedure described above, until it obtained the proper solution.

6.1.1. Tests

Tests were conducted at the initial stage of the upgrading of the program; not all features of the global solution described above were applied at this stage of the work. The global solution was just searching for clashes and choosing the residue to re-rotate; no iteration or "black list" was applied. After finding a clash and solving it, the program started a search from the beginning of the molecule.

All tests were performed on the PC, and the search covered only the backbone with the reference distance of 4 Å.

6.1.2. Results

It was impossible to get any results. The program, after finding the first unsolvable clash, was trying to solve it in an infinite loop. This situation showed that this solution not only does not report an unsolvable situation, but the program also cannot be ended properly if it found one.

[1] Because of the machine accuracy, *t*-parameter values closer than $0{,}01^{o}$ to the ellipse subspace edges are not taken under consideration.

6.1.3. Discussion

One of improvements that can be applied is a maximum number of iterations for the local solution. This solves the problem with a proper ending of the program, but still, it does not report an unsolvable situation, which is one of the major restrictions. Moreover, the program produces irreproducible data.

Additionally, this solution, without any changes applied, is using only one residue to solve a single clash.

6.2. The "walking along the ellipse" solution

The second solution that was tested based on a distribution of a distance between clashed atoms in a function of a *t*-parameter. Based on transformation matrixes and a procedure described in [16], in section 2.2.5, the program created the distribution using a thousand values of a *t*-parameter, each distant from the previous by same step, included between edges of the zone of the selected residue.

The program set up the molecule, with the method defined in [16], and subsequently, with use of transformation matrixes, it moved the second clashed atom, in the way of changing dihedral angles in the residue selected to re-rotate. After each change, the program was calculating a distance between clashed atoms and noting it into a table.

Afterwards, the program checked the most optimized distance for the clash, and, based on the *t*-parameter that was corresponding with that distance, re-rotated the whole molecule.

6.2.1. Tests

Tests that were conducted had two aims: to test the correctness of the solution and to find the most optimized distance. All tests were performed with the whole global solution applied, on the PC, with the reference distance of 4 Å, and clashes were searched for: in the whole strand, in the strand excluding hydrogen atoms, and only in the backbone.

In trials that were aimed to obtain the most optimized distance, the distance that was used by the local solution was being changed.

6.2.2. Results

Tests that were aimed to find the most optimized distance showed that the best distance was a maximal possible one. In trials that the distance used by the local solution was minimally bigger than reference distance[1], or two times bigger than the reference distance[1], despite of fact that the clash was removed, it occurred that, usually, other atoms from clashed residues were still in a clash within the same or adjacent residues atoms. In the view of these facts, it was decided to try to avoid also these additional clashes, and the optimized distance was chosen as a maximal possible distance.

Tests that were aimed to check correctness of the solution were all positive. The only flaw of the solution is its complication, such as the use of transformation matrixes, which results in a significant slowdown of calculations.

The solution fulfills all major restrictions.

6.3. The analytical solution

In the view of the fact that the optimized distance is a maximal possible distance between clashed atoms, in order to speed up the program, an attempt to find the analytical solution was made. The first part is a derivation of equations for coordinates of the second clashed atom in a function of a *t*-parameter, in the situation of the change of dihedral angles in a residue that was chosen by the global solution. The second part is the substitution of variables describing the second atom in equation (5.1) with functions obtained in the precious step:

$$d(a,b,c) = \sqrt{\left(x_1 - a(t)\right)^2 + \left(y_1 - b(t)\right)^2 + \left(z_1 - c(t)\right)^2}, \qquad \textbf{(6.1)}$$

where:

***a*(*t*) [Å]** – a function that represents a coordinate *x* of the second clashed atom,

***b*(*t*) [Å]** – a function that represents a coordinate *y* of the second clashed atom,

***c*(*t*) [Å]** – a function that represents a coordinate *z* of the second clashed atom.

[1] If possible; in other cases the program chose a maximal distance.

Next, the third, step should be an analysis of function (6.1), what implies calculating its first derivative and finding its zeroes in order to receive local extrema of the function, and its second derivative to define its local maxima.

6.3.1. The precession model

At the beginning of this analysis, it was observed that the movement of the second clashed atom during the change of dihedral angles is similar to the movement of a point in the precession phenomenon, therefore, the solution may, potentially, be obtained based on this phenomenon. Other possibility of to receive these functions is an analysis of matrix equations, which were used in the base program ([16], section 2.2.5). The choice to base on the precession phenomenon as the way of obtaining analytical solution was dictated by two facts: the procedure that was used in the base program is partially recurrent, what makes the derivation of the function more difficult; and a fully mathematically derived function of the precession phenomenon might be applied in other fields of studies such as medical imaging, quantum physics or astronomy.

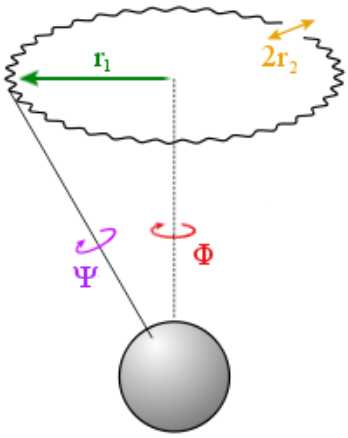


FIGURE 10. The application of the precession phenomenon into amino acid residue's dihedral angles. (Based on [19]).

As shown in FIGURE 10, amino acid residue's dihedral angles (marked with the same colors as in FIGURE 2) can be presented with the use of the precession phenomenon with independence of the rotational axis (in FIGURE 10 marked as the Ψ-angle axis) from the principal axis (marked as the Φ-angle axis). The corrugated line corresponds with a possible trajectory of the point in the precession (which corresponds with the second clashed atom). **The r_1 radius** corresponds with the

shortest distance between the principal axis and the center of the rotation of the point, with respect to the rotation axis, and **the r_2 radius** corresponds with a distance between the point and the center of the rotation, with respect to the same axis.

In order to show a clash as the precession phenomenon, some mathematical operation had to be made on the atoms arrangement. Based on functions described in [16], in section 2.2.5, some atoms of the residue that was chosen for re-rotation and both clashed atoms are transformed in the Euclidean space as listed below; all relationships between atoms maintain.

TABLE 3. The transformation of atoms in the analytical solution (unless otherwise stated, all atoms are part of the re-rotated residue).

	Coordinate			
Atom	**x**	**y**	**z**	**Position**
C_α	0	0	0	The origin of the coordinate system
N	0	0	<0 depend on distance between N and C_α	On the *z*-axis
C	>0 depend on relationships with N and C_α	0	>0 depend on relationships with N and C_α	On the *xz*-plane
Both clashed atoms	depend on relationships with other atoms	depend on relationships with other atoms	depend on relationships with other atoms	-

The situation, described in TABLE 3, is shown in FIGURE 11 and FIGURE 12. Points $\mathbf{P_{C\alpha}}$ ($x_{C\alpha},y_{C\alpha},z_{C\alpha}$), $\mathbf{P_N}$ (x_N,y_N,z_N) and $\mathbf{P_C}$ (x_C,y_C,z_C) represent corresponding atoms of the re-rotated residue, point $\mathbf{P_2}$ ($\mathbf{x_2},\mathbf{y_2},\mathbf{z_2}$) represents the second clashed atom; these point can also be called as *real* points, because they are related to atoms of the protein and their coordinates are given. The position of the first clashed atom is not included in the figures because of its marginal meaning in the description of the model. The *z*-axis overlaps with the Φ-angle axis and, also, with the principal axis of the precession; a line, marked in magenta that goes through the origin and points P_C and P_o, corresponds with the Ψ-angle axis and, also, with the rotation axis.

Point $\mathbf{P_o}$ $(\mathbf{x_o,y_o,z_o})$ represents the center of rotation with respect to the rotation axis; it can also be called as a *virtual* point, because it does not represent any atom of the protein, and its coordinates have to be calculated.

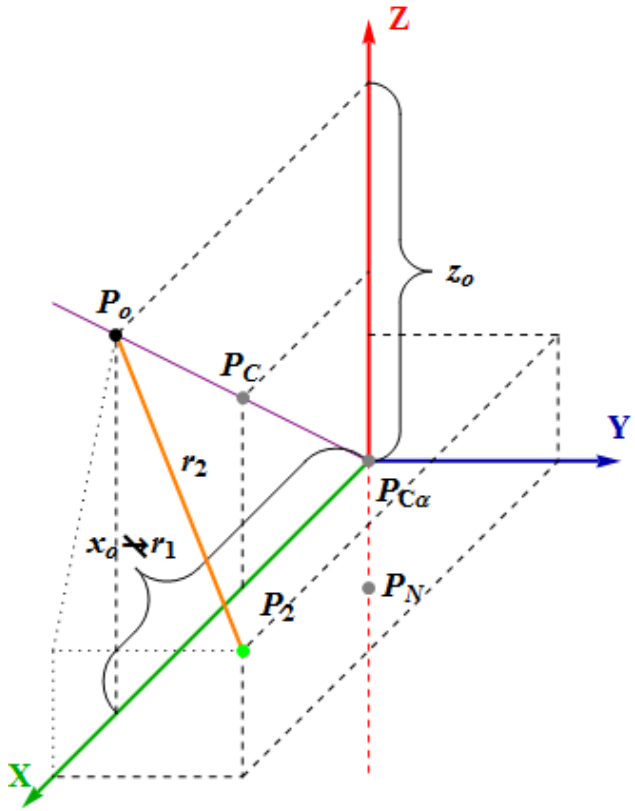


FIGURE 11. The setting of atoms in the precession model.

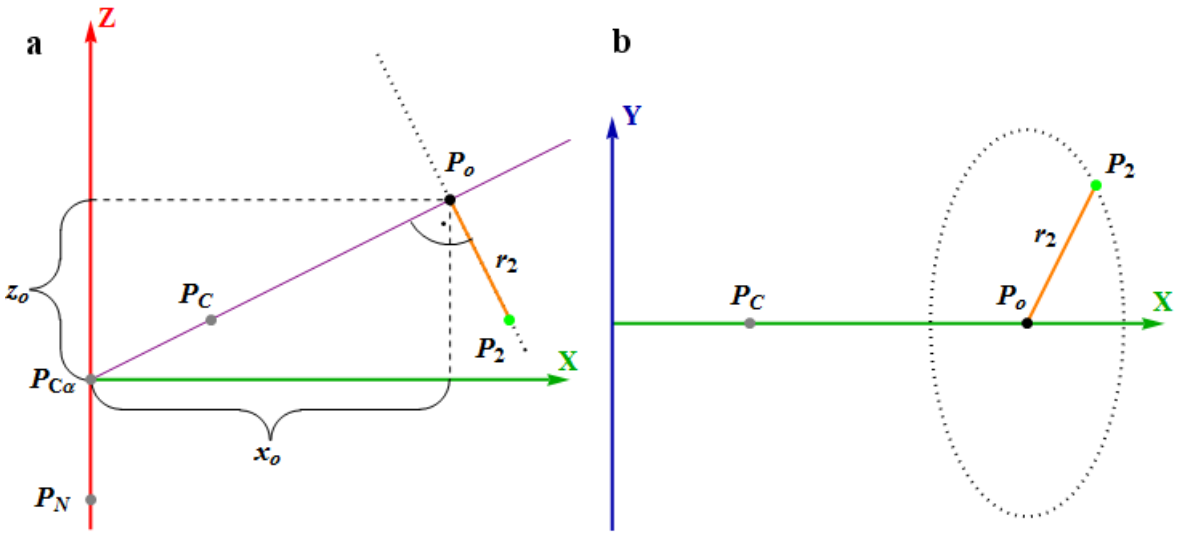


FIGURE 12. The setting of atoms in the precession model in projections: a) a projection onto the *xz*-plane; b) a projection onto the *xy*-plane. A dotted line represents the trajectory of the second clashed atom in the change of the Ψ' angle.

In FIGURE 11, line segment x_o is shown as not equal to the r_1 radius; this is caused by a possibility that the value of x_o can be negative, and it is against the mathematical rule, which states that a radius is always bigger than zero. The absolute value of x_o is equal to the length of the r_1 radius; consequently, x_o is used instead of r_1 in future equations.

6.3.2. Definitions of the Φ and the Ψ angle in the model

At the beginning, the Φ and the Ψ angle of the model (hereinafter marked as the Φ' and the Ψ' angle) have to be defined:

- **The Φ' angle** is an angle between the *x*-axis and the projection of the Ψ'-angle axis onto the *xy*-plane.
- **The Ψ' angle** is an angle between the projection of the *x*-axis onto the plane that is perpendicular to the Ψ'-angle axis and the projection of the line segment connecting P_2 and the rotation axis onto the same plane. In order to simplify this, it can also be said that a Ψ'-angle's zero value is received in situations when the lowest value of point P_2 is obtained.

6.3.3. Calculation of parameters of the model

In order to describe the model mathematically, coordinates of point P_o and the r_2 radius have to be calculated.

Point P_o lies on the rotation axis, which, before the change of dihedral angles, is a part of the *xz*-plane, therefore, its *y*-coordinate (y_o) is equal to zero. In order to receive remaining coordinates, only two points are required: P_C, which lies on the *xz*-plane; and P_2. Based on point P_2, a line that goes through this point and is normal to the Ψ'-angle axis can be constructed, and, in this situation, its projection onto the *xz*-plane is also normal. Therefore, the projection of the arrangement onto the *xz*-plane is sufficient to calculate coordinates of point P_o.

The first step is to find a linear equation of the rotation axis with the use of point P_C and equation (6.2). Because the axis goes through the origin of the coordinate system, the constant term of a linear equation can be omitted:

$$z_r(x) = a_r x, \tag{6.2}$$

where:

$\boldsymbol{a_r}$ – a slope of the rotation axis' linear equation,

$\boldsymbol{z_r(x)}$ **[Å] –** an ordinate of the rotation axis' linear equation.

The equation can be substituted with coordinates of P_C and, based on this, the value of a_r can be calculated:

$$z_C = a_r x_C, \tag{6.3}$$

$$a_r = \frac{z_C}{x_C}, \tag{6.4}$$

therefore:

$$z_r(x) = \frac{z_C}{x_C} x. \tag{6.5}$$

The next step is a calculation of a line that is perpendicular to the Ψ'-angle axis and goes through the projection of point P_2. In can be received with the use of the general equation for obtaining the normal line in the Cartesian coordinate system:

$$z_\perp(x) = -\frac{1}{a_r} x + b_\perp, \tag{6.6}$$

where:

$\boldsymbol{b_\perp}$ **[Å]** – a constant term of the normal line's linear equation,

$\boldsymbol{z_\perp(x)}$ **[Å] –** an ordinate of the normal line's linear equation.

Substituting this equation with equation (6.4) and coordinates of point P_2, the value of variable $b_\perp$ can be obtained:

$$z_2 = -\frac{x_c}{z_c} x_2 + b_\perp, \tag{6.7}$$

$$b_{\perp} = z_2 + \frac{x_c}{z_c} x_2, \tag{6.8}$$

And consequently:

$$z_{\perp} = -\frac{x_c}{z_c} x + z_2 + \frac{x_c}{z_c} x_2. \tag{6.9}$$

In this situation, a common point of lines described with equations (6.5) and (6.9) is also point P_o:

$$\begin{cases} z_o = \dfrac{z_c}{x_c} x_o \\ z_o = -\dfrac{x_c}{z_c} x_o + z_2 + \dfrac{x_c}{z_c} x_2, \end{cases} \tag{6.10}$$

$$\begin{cases} x_o = \dfrac{x_c}{z_c} z_o \\ z_o = -\dfrac{{x_c}^2}{{z_c}^2} z_o + z_2 + \dfrac{x_c}{z_c} x_2, \end{cases} \tag{6.11}$$

$$z_o {z_c}^2 = -z_o {x_c}^2 + z_2 {z_c}^2 + x_c z_c x_2, \tag{6.12}$$

$$z_o({x_c}^2 + {z_c}^2) = z_2 {z_c}^2 + x_c z_c x_2, \tag{6.13}$$

$$\begin{cases} \boldsymbol{z_o = z_c \dfrac{z_c z_2 + x_c x_2}{{z_c}^2 + {x_c}^2}} \\ \boldsymbol{x_o = x_c \dfrac{z_c z_2 + x_c x_2}{{z_c}^2 + {x_c}^2}}. \end{cases} \tag{6.14}$$

After receiving coordinates of point P_o, the value of **the r_2 radius** is possible to calculate as a distance between points P_o and P_2[1]:

$$\boldsymbol{r_2 = \sqrt{(x_2 - x_o)^2 + {y_2}^2 + (z_2 - z_o)^2}}. \tag{6.15}$$

6.3.4. Relationship between dihedral angles in a residue and angles in the model

The model describes a residue with the second pair of angles' values, which are differently defined then they are in dihedral angles, therefore, a transfer

[1] Because y_o is equal to zero, it is omitted in the equation.

function is essential to connect values of this model with values received from equations of the elliptical model (equations (1.2) and (1.3)). In both arrangements, corresponding values of angles change accordingly to each other, therefore, they are always differenced only by constants.

The difference in the value of **the Φ angle** can be eliminated with a subtraction of the value of the dihedral angle at the beginning of a clash-remove process from the value received from the elliptical model:

$$\Phi'(t) = \Phi(t) - \Phi_0, \tag{6.16}$$

where:

***Φ'(t)* [°]** – a value of the Φ angle in the precession model,

***Φ₀* [°]** – a value of the Φ angle at the beginning of a clash-remove process.

The situation with **the Ψ angle** looks similar as with the Φ angle, but, unlike the Φ angle, after the setting, the Ψ' angle, usually, will not be equal to zero; therefore, an additional constant have to be added, which has to be equal to this difference:

$$\Psi'(t) = \Psi(t) - \Psi_0 + \Psi'', \tag{6.17}$$

where:

***Ψ'(t)* [°]** – a value of the Ψ angle in the precession model,

***Ψ₀* [°]** – a value of the Ψ angle at the beginning of a clash-remove process,

***Ψ''* [°]** – a value additionally added to the value of the Ψ angle calculated from in the precession model.

In order to calculate a value of the Ψ'' angle, FIGURE 13 was prepared. In this figure, it is clearly seen that this constant can be received by using an inverse trigonometric function:

$$\Psi'' = \arcsin\frac{y_2}{r_2}. \tag{6.18}$$

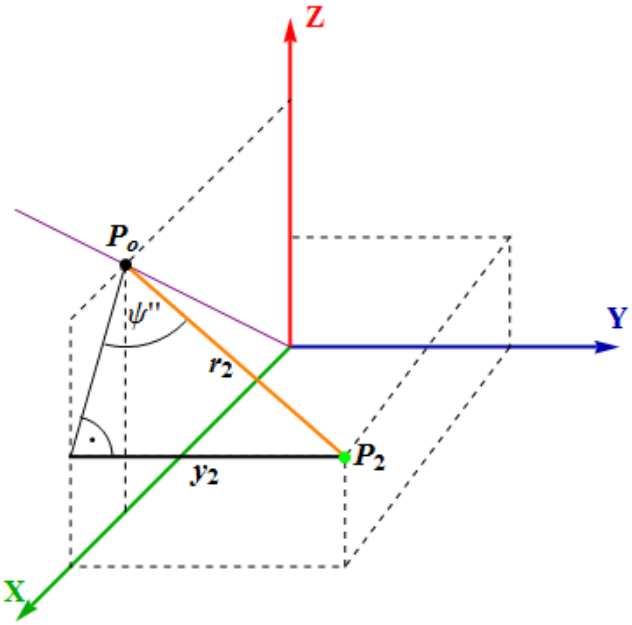


FIGURE 13. The Ψ'' angle.

Because of properties of an arcsine function, the use of equation (6.18), without any additional conditions, provides to receive incorrect data for angles between 90° and 270°; therefore, some improvements had to be done:

$$\Psi'' = \arcsin\frac{y_2}{r_2} \qquad for \qquad z_2 \leq z_o \tag{6.19}$$

and:

$$\Psi'' = 360^o - \arcsin\frac{y_2}{r_2} \qquad for \qquad z_2 > z_o. \tag{6.20}$$

6.3.5. The *z*-variable function

The determination of functions describing the coordinates of the second clashed atom in a function of dihedral angles can be divided into a two separated parts: the determination of the *z*-variable function, and the determination of the *x*-variable and *y*-variable functions. This situation, as shown in FIGURE 12, is possible, because these variables are independent from each other, and the *z*-variable function is dependent only on the Ψ' angle, unlike other functions, which are dependent on both angles.

As shown in FIGURE 14 the range of changes of the *z*-coordinate depends on the r_z radius and z_o parameter. The function of the change (*c*-function) is described by equation (6.21):

$$c(\Psi') = z_o - r_z \cos \Psi', \tag{6.21}$$

where:

$\boldsymbol{r_z}$ **[Å]** – a *z*-component of the r_2 radius for the Ψ' angle equals to zero.

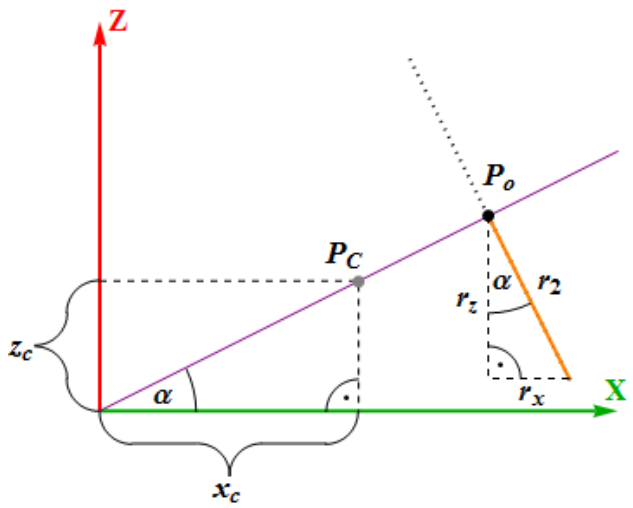


FIGURE 14. A derivation of *c*-function in a projection onto the *xz*-plane. The r_2 radius is marked for the Ψ' angle equal to zero. The dotted line shows a trajectory of point P_2 in a function of the Ψ' angle.

The r_z radius can be calculated based on a similarity of triangles and the Pythagorean theorem:

$$r_z = r_2 \cos \alpha, \tag{6.22}$$

where:

$\boldsymbol{\alpha}$ **[°]** – an angle of inclination of the rotation axis,

$$\cos \alpha = \frac{x_c}{\sqrt{{x_c}^2 + {z_c}^2}}, \tag{6.23}$$

therefore:

$$\boldsymbol{c(\Psi') = z_o - \frac{r_2 x_c}{\sqrt{{x_c}^2 + {z_c}^2}} \cos \Psi'}. \tag{6.24}$$

6.3.6. The *x*-variable and *y*-variable functions

As written in the previous section, derivations of functions describing a change of the *x*-coordinate (*a*-function) and the *y*-coordinate (*b*-function) are easiest to perform at the same time. This derivation can be divided into two parts: the first, the description of the correlation between coordinates and the Ψ' angle; and the second, the description of their correlation with the Φ' angle.

As shown in FIGURE 15, the trajectory of point P_2 in the function of the Ψ' angle is in the shape of an ellipse. Therefore, the equations describing the correlation with the Ψ' angle can be based on parametric equations of an ellipse (equation (6.25)). Additionally, in the equation describing the *x*-coordinate a shift, equal to x_o, have to be added.

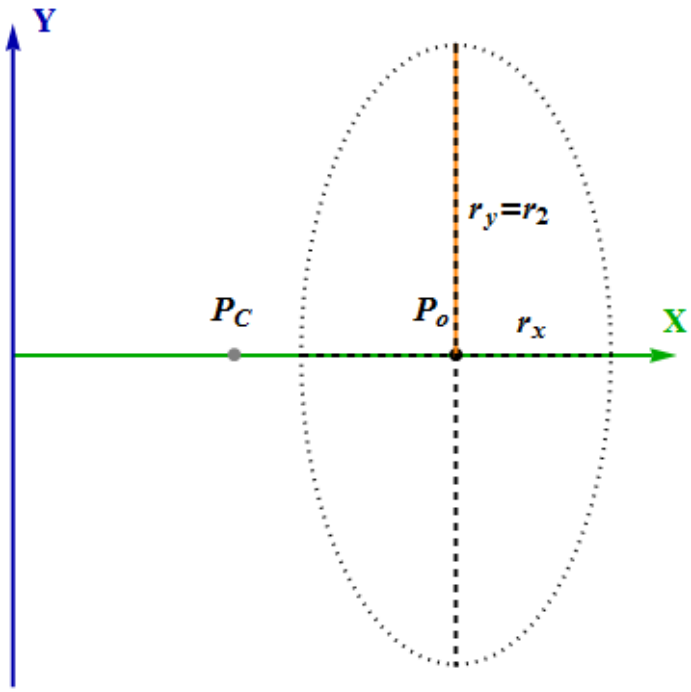


FIGURE 15. A derivation of *a*- and *b*-function in a projection onto the *xy*-plane. The r_2 radius is marked for the Φ' angle equals to zero and the Ψ' angle equals to 90^0. The dotted line shows a trajectory of point P_2 in a function of the Ψ' angle.

$$\begin{cases} a(\Psi') = r_x \cos\Psi' + x_o \\ b(\Psi') = r_y \sin\Psi' \end{cases}, \tag{6.25}$$

where:

r_x [Å] – a semi-minor axis of an ellipse,

r_y [Å] – a semi-major axis of an ellipse.

Based on FIGURE 15, it can be stated that r_y is equal to r_2. Based on FIGURE 14, r_x can be calculated in a similar way as r_z was:

$$r_x = r_2 \sin\alpha, \tag{6.26}$$

$$\sin\alpha = \frac{z_c}{\sqrt{{x_c}^2 + {z_c}^2}}, \tag{6.27}$$

therefore:

$$r_x = \frac{r_2 z_c}{\sqrt{{x_c}^2 + {z_c}^2}}, \tag{6.28}$$

and consequently:

$$\begin{cases} a(\Psi') = \dfrac{r_2 z_c}{\sqrt{{x_c}^2 + {z_c}^2}} \cos\Psi' + x_o \\ b(\Psi') = r_2 \sin\Psi' \end{cases}. \tag{6.29}$$

The easiest way to apply **the Φ' angle** into the solution is the use of the rotation matrix in two dimensions (equations (6.30) and (6.31)). This is possible, because the Φ'-angle axis overlaps with the *z*-axis, therefore, its projection onto the *xy*-plane covers the origin of the coordinate system:

$$M(\Phi') = \begin{bmatrix} \cos\Phi' & -\sin\Phi' \\ \sin\Phi' & \cos\Phi' \end{bmatrix}, \tag{6.30}$$

where:

***M*(*Φ'*)** – the rotation matrix in two dimensions,

$$\begin{bmatrix} a(\Phi', \Psi') \\ b(\Phi', \Psi') \end{bmatrix} = M(\Phi') \cdot \begin{bmatrix} a(\Psi') \\ b(\Psi') \end{bmatrix}; \tag{6.31}$$

therefore:

$$\begin{bmatrix} \boldsymbol{a(\Phi', \Psi')} \\ \boldsymbol{b(\Phi', \Psi')} \end{bmatrix} = \begin{bmatrix} \dfrac{\boldsymbol{r_2 z_c}}{\sqrt{\boldsymbol{{x_c}^2 + {z_c}^2}}} \boldsymbol{\cos\Phi' \cos\Psi' - r_2 \sin\Phi' \sin\Psi' + z_o \cos\Phi'} \\ \dfrac{\boldsymbol{r_2 z_c}}{\sqrt{\boldsymbol{{x_c}^2 + {z_c}^2}}} \boldsymbol{\sin\Phi' \cos\Psi' + r_2 \cos\Phi' \sin\Psi' + z_o \sin\Phi'} \end{bmatrix}. \tag{6.32}$$

6.3.7. Summary of the derivation of the precession model

Substituting the equations above, one with another, a function describing the correlation between the distance between clashed atoms and the *t*-parameter can be received. Unfortunately, because of the complexity of the solution, presenting it as a single equation would be highly confusing; instead of showing one complete equation, the most important equations of the solution are listed below as the summary of the derivation:

$$d(a,b,c) = \sqrt{\left(x_1 - a(\Phi',\Psi')\right)^2 + \left(y_1 - b(\Phi',\Psi')\right)^2 + \left(z_1 - c(\Phi',\Psi')\right)^2}, \quad \textbf{(6.33)}$$

$$a(\Phi',\Psi') = \frac{r_2 z_c}{\sqrt{{x_c}^2 + {z_c}^2}}\cos\Phi'(t)\cos\Psi'(t) - r_2\sin\Phi'(t)\sin\Psi'(t) + z_o\cos\Phi'(t), \quad \textbf{(6.33)}$$

$$b(\Phi',\Psi') = \frac{r_2 z_c}{\sqrt{{x_c}^2 + {z_c}^2}}\sin\Phi'(t)\cos\Psi'(t) + r_2\cos\Phi'(t)\sin\Psi'(t) + z_o\sin\Phi'(t), \quad \textbf{(6.34)}$$

$$c(\Psi') = z_o - \frac{r_2 x_c}{\sqrt{{x_c}^2 + {z_c}^2}}\cos\Psi'(\mathrm{t}), \quad \textbf{(6.35)}$$

$$\Phi'(t) = \Phi(t) - \Phi_0, \quad \textbf{(6.36)}$$

$$\Psi'(t) = \Psi(t) - \Psi_0 + \Psi'', \quad \textbf{(6.37)}$$

$$\Phi(t) = 125\cos(45^o)\cos(t) - 84\sin(45^o)\sin(t), \quad \textbf{(6.38)}$$

$$\Psi(t) = -125\sin(45^o)\cos(t) - 84\cos(45^o)\sin(t). \quad \textbf{(6.39)}$$

6.3.8. Proline

Proline, unlike the rest of amino acids, is a cyclic molecule (FIGURE 16); because of its build, the fixity of the Φ angle can be adopted and set as 75° (as stated in [16]).

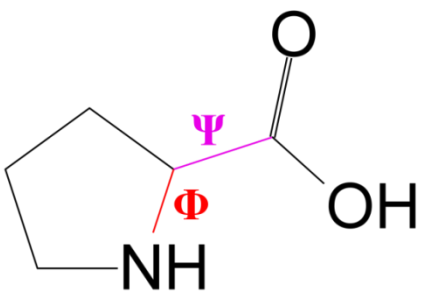


FIGURE 16. Proline with its dihedral angles marked.

Based on this assumption, for proline, some simplifications to the model can be made. At first, the residue have to be set in a different way, described in TABLE 4, and presented in FIGURE 17.

TABLE 4. The transformation of atoms in the analytical solution with proline as the re-rotated residue (unless otherwise stated, all atoms are part of the proline residue).

Atom	Coordinate			Position
	x	y	z	
C	0	0	0	The origin of the coordinate system
C_α	0	0	<0 depend on distance between C and C_α	On the *z*-axis
The second clashed atom	>0 depend on relationships with C and C_α	0	>0 depend on relationships with C and C_α	On the *xz*-plane
The first clashed atom	depend on relationships with other atoms	depend on relationships with other atoms	depend on relationships with other atoms	-

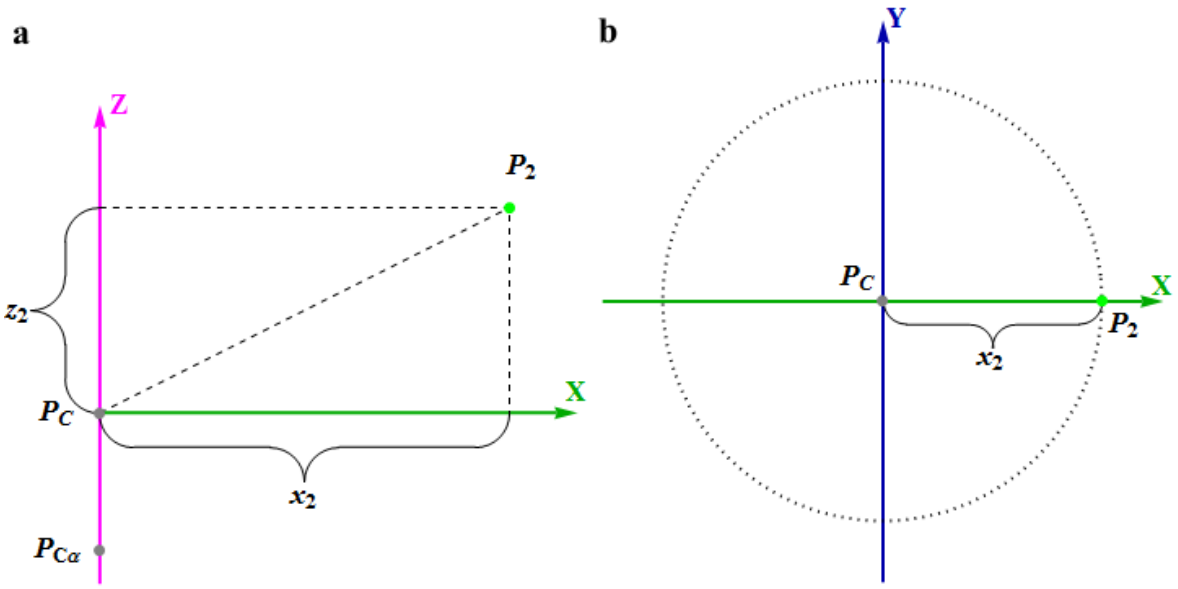


FIGURE 17. The setting of proline as the re-rotated residue in projections: a) a projection onto the *xz*-plane; b) a projection onto the *xy*-plane. The dotted line represents the trajectory of the second clashed atom in the change of the Ψ' angle.

As presented above, the Ψ'-angle axis overlaps with the *z*-axis. Moreover, the *z*-coordinate is constant for all values of the angle; therefore, the *xy*-plane projection is sufficient to describe the trajectory functions.

In this situation, the distance function is as follows:

$$d(a,b,c) = \sqrt{(x_1 - a(\Psi_P'))^2 + (y_1 - b(\Psi_P'))^2 + (z_1 - z_2)^2}, \quad \textbf{(6.40)}$$

where:

$\boldsymbol{\Psi_P'(t)}$ **[°]** – a value of the Ψ angle for the arrangement with proline as the re-rotated residue.

As shown in FIGURE 17.b, the trajectory of point P_2 is in the shape of a circle, therefore, its equation looks as below:

$$\begin{cases} a(\Psi_P') = x_2 \cos \Psi_P'(t) \\ b(\Psi_P') = x_2 \sin \Psi_P'(t). \end{cases} \quad \textbf{(6.41)}$$

As angles in the precession model, the Ψ_P' angle is not equal to the Ψ angle, but they are differed by a constant, which is equal to the value of the Ψ angle at the beginning of the clash-remove process:

$$\Psi_P'(t) = \Psi(t) - \Psi_0. \quad \textbf{(6.42)}$$

6.3.9. The analysis of the function

In order to analyze the function described by equations (6.33) – (6.39), the first derivative of it was calculated and equated to zero. These and further operation were made using Mathematica on the PC. Afterwards, an attempt to find zeros was made.

Unfortunately, after two days of working, the program was unable to find the solution, therefore, calculations were stopped.

6.3.10. Tests

Despite the fact that the solution remains unfinished, some tests, checking the correctness of the solution, were made with the use of Mathematica. Tests were

based on mathematical functions of the precession model, and were checking calculations of parameters. Results are shown in a graphic presentation of functions and points that were calculated.

The first test was checking the correctness of calculations of coordinates with respect to the primary position of point P_2.

The second test was checking the correctness of an application of dihedral angles by the model.

6.3.11. Results

All parameters, calculated in the first test, were returned as expected. The graphic presentation of the results is shown in FIGURE 18.

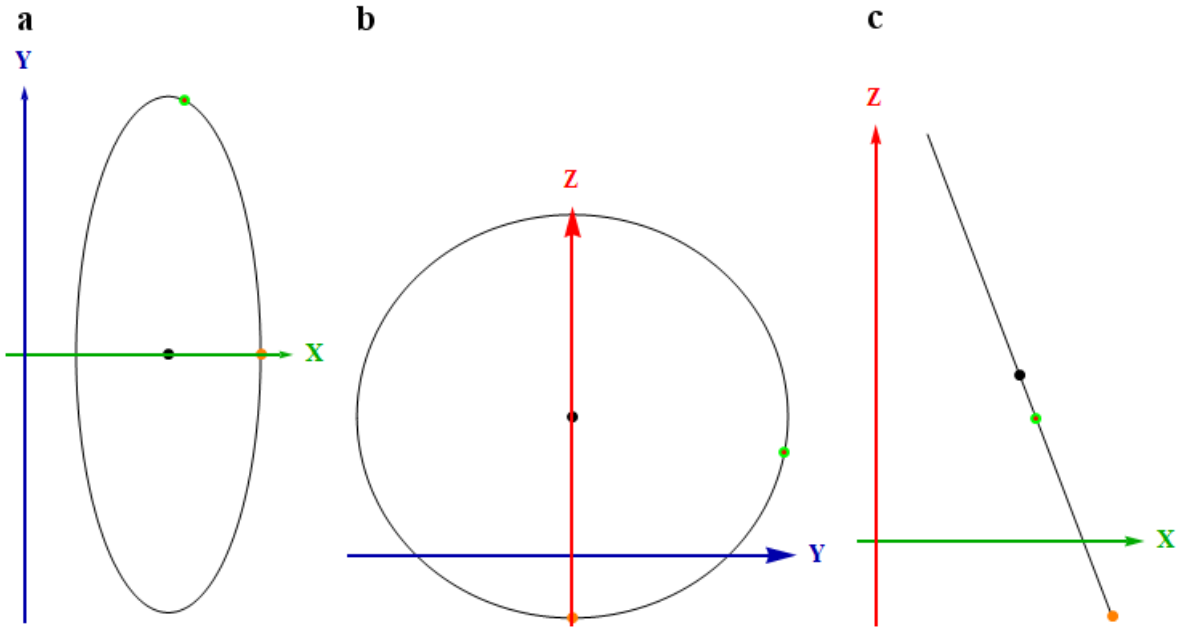


FIGURE 18. A graphic presentation of the results of the first test in projections: a) a projection onto the *xy*-plane; b) a projection onto the *yz*-plane; c) a projection onto the *xz*-plane. A green dot represents a primary position of point P_2, drawn based on given data, a small red dot represents z position of the same point, but drown based on coordinates calculated with the use of the precession model. A black dot represents a calculated position of point P_o. An orange dot represents a calculated position of point P_2 with the Φ' and the Ψ' angle equal to zero. A black line represents a calculated trajectory of point P_2 with a constant value of the Φ' angle and a changing Ψ' angle.

The results of the second test, presented in FIGURE 19, are in line with expectations.

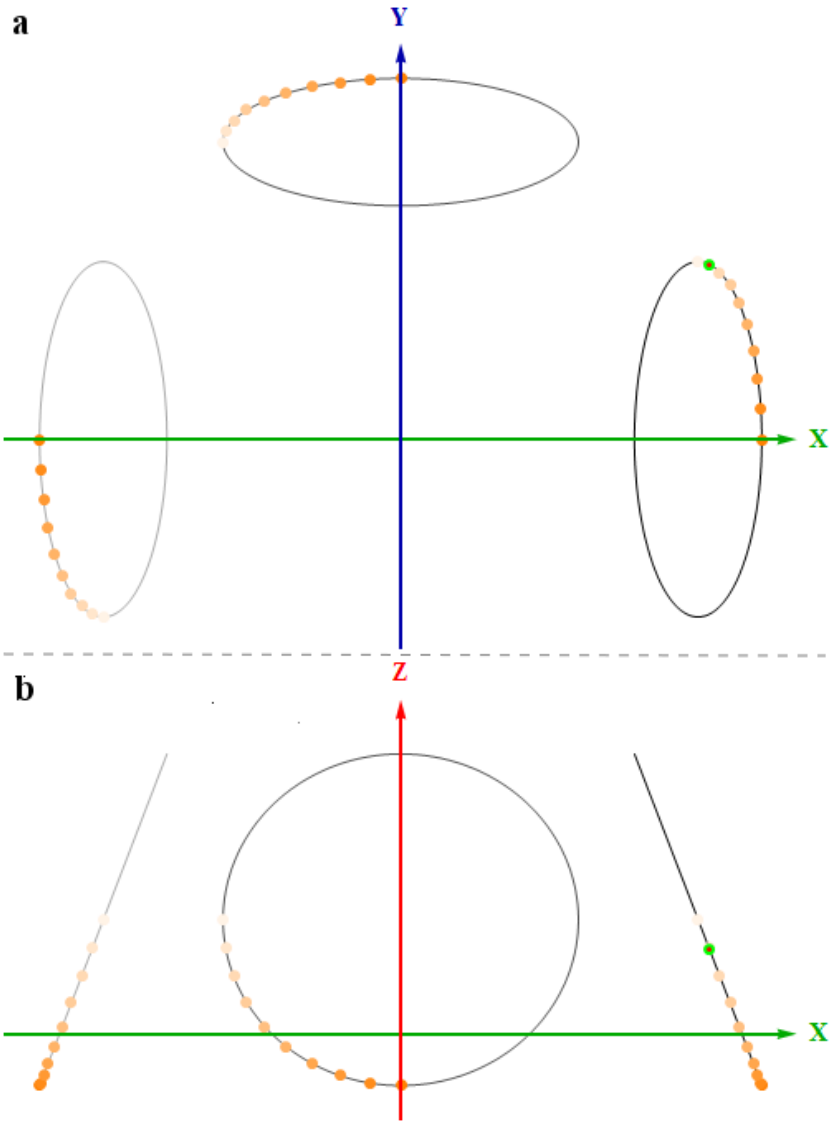


FIGURE 19. A graphic presentation of the results of the second test in projections: a) a projection onto the *xy*-plane; b) a projection onto the *xz*-plane. The green and the red dot represent the same points as in FIGURE 18. Orange dots represent an increasing value of the Φ' angle with a decreasing saturation of the color (between angles of zero and 90°, with a step of 10°); the black line corresponds with the trajectory with the Ψ' angle equal to zero, the dark gray - equal to 90°, and the light gray - equal to 180°.

6.4. The hybrid solution

The hybrid solution connects the "walking along the ellipse" and the analytical solutions. In this solution, the program counts values of the distance between clashed atoms in a function of the *t*-parameter, alike as in the "walking along the ellipse" solution, but instead of using transformation matrixes, it uses equations derived for the analytical solution. This modification allows omitting matrix multiplications, and results in speeding up the program.

6.4.1. Tests

Tests were conducted on the same files as in the “walking along the ellipse” solution’s test; they were performed with the whole global solution applied, on the PC, with the reference distance of 4 Å, and clashes were searched for: in the whole strand, in the strand excluding hydrogen atoms, and only in the backbone.

6.4.2. Results

All received results were the same as those received from the “walking along the ellipse” solution’s tests; they showed correctness of the solution. The solution is more complicated than other solutions, but avoidance of matrix multiplications results in speeding up calculations.

The solution fulfills all major restrictions.

6.5. Discussion

The hybrid solution, along with the global solution described in section 5, have been applied to the final version of the program, because of its fulfillment of all of the major restrictions and the highest speed of calculations among all proper solutions.

7. Efficiency tests

After the application of the global solution and the hybrid solution, as the local solution, to the base program, in order to measure efficiency of removing clashes, tests were made.

7.1. Tests

All tests were performed on the PC, they covered the whole molecule, and they had the reference distance equal to 1.8 Å; the program is intended to work with this settings. Input files, provided by the supervisor, contained primary structures and, appropriate for the elliptic model, dihedral angles of seven proteins[1]: 1ARR, 1B8Z, 1AC6, 1ARJ, 1AD3, 1AOZ, 1ARQ. All files were in the AAD format (see [16], section 2.2.2); therefore the division into separate strands was not applied.

Additionally, some tests were performed by a person not directly connected with this master's thesis, with an access to the Academic Computer Centre Cyfronet AGH (for more information, see [18]). Settings of the program were the same, and input files were unknown to the author, but they met the requirements of the elliptic model. Moreover, output files were directed into the Late-Stage.

7.2. Results and analysis

All results with an analysis are presented in TABLE 9 in Appendix E. Based on this table following charts were made:

[1] All of them can be found in Protein Data Bank [20].

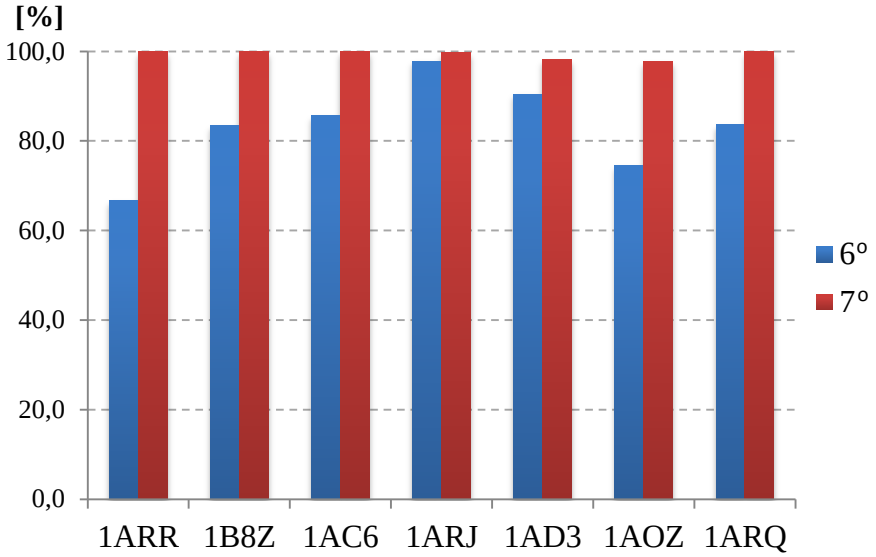


FIGURE 20. The efficiency of removing clashes in the first iteration (see the legend in TABLE 5).

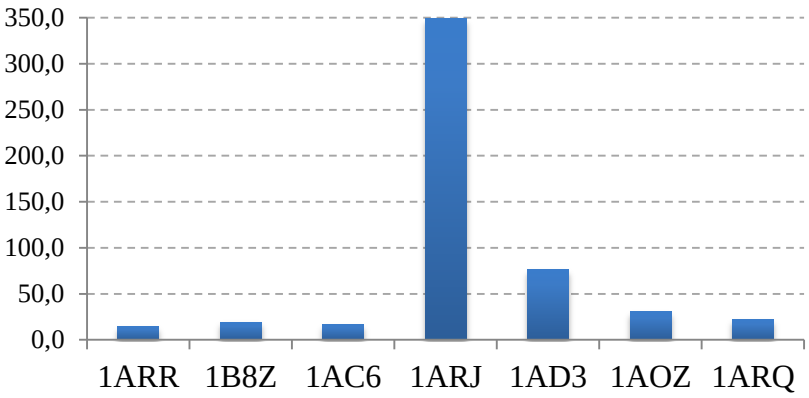


FIGURE 21. The number of clashes in the first iteration per 100 residues (8^o in the legend in TABLE 5).

TABLE 5. The legend of FIGURE 20 and FIGURE 21.

ID	Description
$\mathbf{1^o}$	The length of a protein [residues]
$\mathbf{2^o}$	The number of clashes in the first iteration
$\mathbf{3^o}$	The number of unsolved clashes in the first iteration
$\mathbf{4^o}$	The number of unsolved clashes in the first iteration with only zone-C or –E residues available to solve the clash with
$\mathbf{6^o}$	**The efficiency of removing clashes in the first iteration** $\left(\frac{3^o}{2^o} \cdot 100\%\right)$ **[%]**
$\mathbf{7^o}$	**The efficiency of removing clashes in the first iteration with zone-C or –E clashes omitted** $\left(\frac{3^o-4^o}{2^o-4^o} \cdot 100\%\right)$ **[%]**
$\mathbf{8^o}$	**The number of clashes in the first iteration per 100 residues** $\left(\frac{2^o}{1^o} \cdot 100\right)$

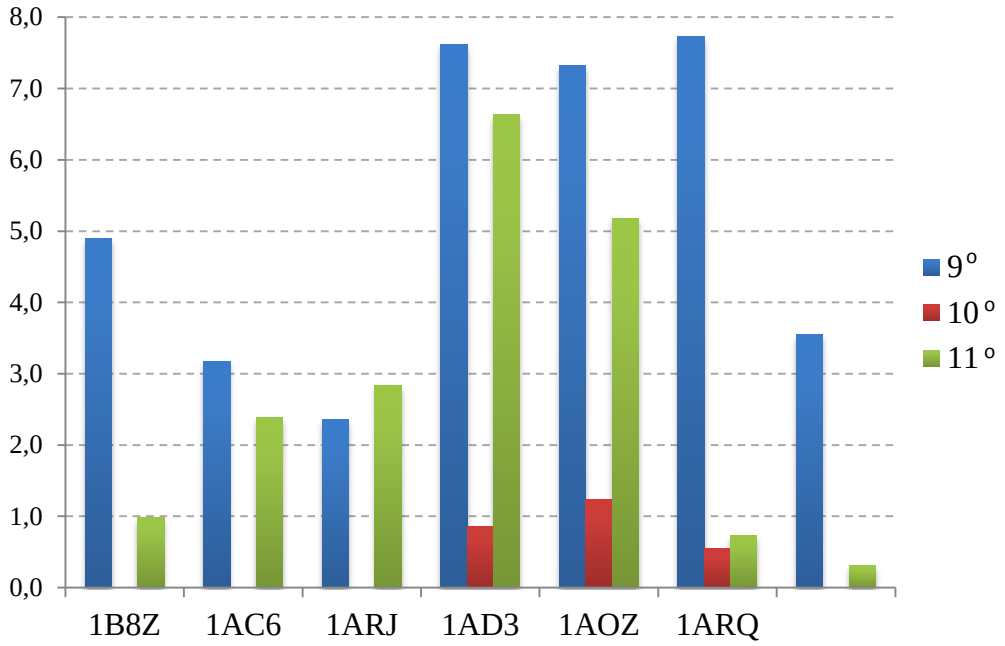


FIGURE 22. The number of unsolved clashes per 100 residues (see the legend in TABLE 6).

TABLE 6. The legend of FIGURE 22.

ID	Description
1^o	The length of a protein [residues]
3^o	The number of unsolved clashes in the first iteration
4^o	The number of unsolved clashes in the first iteration with only zone-C or –E residues available to solve the clash with
5^o	The number of unsolved clash situations[1] in the fifth or greater iterations
9^o	**The number of unsolved clashes in the first iteration per 100 residues $\left(\frac{3^o}{1^o} \cdot 100\right)$**
10^o	**The number of unsolved clashes in the first iteration with zone-C or –E clashes omitted per 100 residues $\left(\frac{3^o - 4^o}{1^o} \cdot 100\right)$**
11^o	**The number of unsolved clash situations in the fifth or greater iterations per 100 residues $\left(\frac{5^o}{1^o} \cdot 100\right)$**

[1] Usually, in fifth or greater iteration not only one atom from one reside clashes with another atom, but many atoms from two colliding residues do it with each other, and this situation, often, can be change with only one try (if that try was possible); if clashes covering the same or adjacent residues occur, they are counted as one.

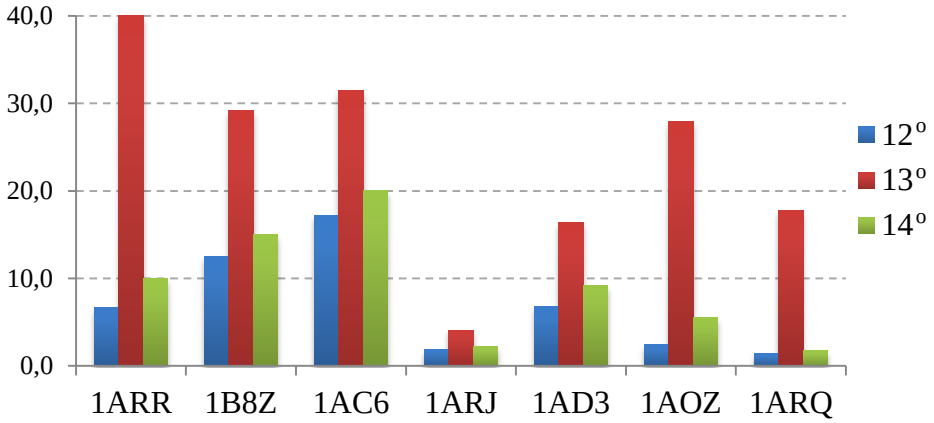


FIGURE 23. The number of unsolved clashes per 100 clashes (see the legend in TABLE 7).

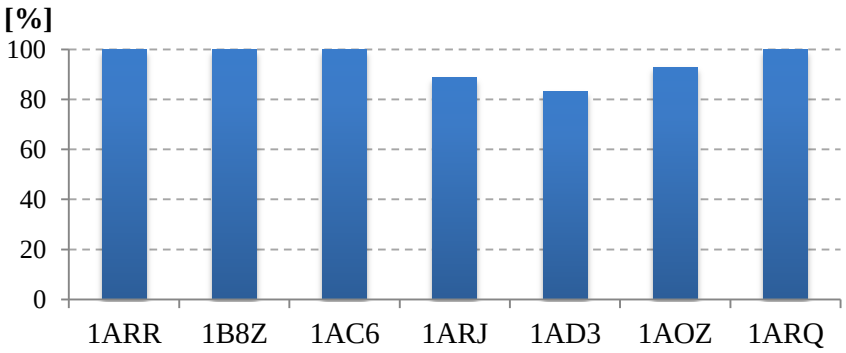


FIGURE 24. Zone-C or –E clashes to unsolved first iteration clashes ratio (15^o in the legend in TABLE 7).

TABLE 7. The legend of FIGURE 23 and FIGURE 24.

ID	Description
2^o	The number of clashes in the first iteration
3^o	The number of unsolved clashes in the first iteration
4^o	The number of unsolved clashes in the first iteration with only zone-C or –E residues available to solve the clash with
5^o	The number of unsolved clash situations in the fifth or greater iterations
12^o	**The number of unsolved clash situations in the fifth or greater iteration per 100 first iterations clashes** $\left(\frac{5^o}{2^o} \cdot 100\right)$
13^o	**The sum of the number of unsolved clashes in the first iteration and the number of unsolved clash situations in the fifth or greater iterations per 100 first iteration clashes** $\left(\frac{3^o+5^o}{2^o} \cdot 100\right)$
14^o	**The sum of the number of unsolved clashes in the first iteration and the number of unsolved clash situations in the fifth or greater iterations per 100 first iteration clashes with zone-C or –E clashes omitted** $\left(\frac{3^o+5^o-4^o}{2^o-4^o} \cdot 100\right)$
15^o	**The number of zone-C or –E clashes to the number of unsolved first iteration clashes ratio** $\left(\frac{4^o}{3^o} \cdot 100\%\right)$ **[%]**

Additionally, after tests on Cyfronet's computers, it occurred that, still, some of clashes that weren't removed in the Early-Stage, make the Late-Stage unable to be conducted.

7.3. Interpretation of the results

As shown in FIGURE 20 most first iteration clashes are removed. FIGURE 24 presents that most of them are clashes with residues between clashed atom only in zones C or E; therefore, no attempt was made to solve these clashes. FIGURE 21, FIGURE 22 and FIGURE 23 show that there is no strict relationship between the length of a protein and the number of clashes; received results are, probably, dependent on a class of a protein. There might be trend for shorter molecules, but, unfortunately, because of a small size of the sample, any correlations are uncertain.

As shown in FIGURE 22 and FIGURE 23 the "Ping-Pong" error still occurs, but it is also not very common in small molecules.

8. Discussion

8.1. Accessibility to the program

The upgraded program (AADream 2.00) and its code can be downloaded from [17].

8.2. Current application of the program

The final program is currently used in some projects carried out on PL-Grid Infrastructure [21].

8.3. Possible future improvements

The greatest defect of presented solution is, probably, its global part. The change of the order of solving the clashes might result in increasing the efficiency of solving the "Ping-Pong" errors; if shorter clashes[1] were solved earlier than longer ones, it might lead to a reduction in the depletion of re-rotable residues between clashed atoms in higher iterations.

Additionally, a permission to change dihedral angles in atoms from zones C and E will highly reduce a quantity of unsolved clashes in the first iteration, but there is no information about its influence on a prediction of protein structure. Recently, the newest version of the program, with this feature applied, was made (AADream 2.01 available at [17]), but it have not returned any significant data yet.

Probably, the biggest improvement to the precession model would be finishing its analysis with the use of a supercomputer to speed up the calculations of the zeros of the first derivative, and applying the results to the program.

8.4. Other observations

The fact that the second clashed atom during any possible change of dihedrals angles is located on the surface of a sphere with respect to point $P_{C\alpha}$ was observed. While line segments $P_{C\alpha}P_o$ and r_2 represent the catheti of a right triangle, a segment $P_{C\alpha}P_2$, which is also the radius of the sphere, represents the hypotenuse. Because values of the catheti, and the angle between them are constant, also the radius is constant, therefore the second clashed atom makes a sphere movement.

[1] The length of a clash is a number of residues between clashed atoms in the primary structure of a protein.

REFERENCES

1. Garrett R., Grisham Ch. M. (2013), *Biochemistry*. Belmont, CA: Brooks/Cole, Cengage Learning.
2. Author: jpdesousa17 from http://www.wikispaces.com, License: Creative Commons Attribution Share-Alike 3.0, Retrieved September 12rd ,2012, from http://ibbiology-guide.wikispaces.com/file/view/alpha_helix.jpg/113619625/alpha_helix.jpg
3. Wikiuser: Fvasconcellos, License: Public Domain, Retrieved July 9th ,2013, from http://upload.wikimedia.org/wikipedia/commons/thumb/b/b7/Beta_sheet_bonding_antiparallel-color.svg/179px-Beta_sheet_bonding_antiparallel-color.svg.png
4. Jane Shelby Richardson, License: Creative Commons Attribution 3.0 Unported, Retrieved July 23rd ,2012, from http://upload.wikimedia.org/wikipedia/commons/c/c0/Protein_backbone_PhiPsiOmega_drawing.jpg
5. Lovell S. C., Davis I. W., Arendall W. B., de Bakker P. I. W., Word J. M., Prisant M. G., Richardson J. S., Richardson D. C. (2003), Structure validation by Cα geometry: ϕ, ψ and Cβ deviation. *Proteins* 50: 437–450. DOI: 10.1002/prot.10286, License: Creative Commons Attribution 3.0 Unported, Retrieved July 14th, 2013, from http://upload.wikimedia.org/wikipedia/commons/9/90/Ramachandran_plot_general_100K.jpg
6. Roterman, I. (1995), The Geometrical Analysis of Peptide Backbone Structure and Its Local Deformations. *Biochimie* 77: 204-216.
7. Roterman I. (1995), Modelling the optimal simulation path in the peptide chain folding - studies based on geometry of alanine heptapeptide. *Journal of Theoretical Biolog*, 177:283–288.
8. Brylinski M., Konieczny L., Czerwonko P., Jurkowski W., Roterman I. (2005), Early-Stage Folding in Proteins (In Silico) Sequence-to-Structure Relation. *Journal of Biomedicine and Biotechnology* 2005.2:65-79. DOI: 10.1155/JBB.2005.65.
9. Jurkowski W., Brylinski M., Konieczny L., Wiíniowski Z., Roterman I. (2004), Conformational Subspace in Simulation of Early-stage Protein Folding. *Proteins: Structure, Function, and Bioinformatics* 55.1:115-127. DOI: 10.1002/prot.20002.
10. Kalinowska B., Alejster P., Sapała K., Baster Z., Roterman I. (2013), Hypothetical in silico model of the early-stage intermediate in protein folding. *Journal of Molecular Modeling*, 06/2013; DOI: 10.1007/s00894-013-1909-6.
11. Konieczny L., Bryliński M., Roterman I. (2006), Gauss-Function-Based Model of Hydrophobicity Density in Proteins. *In Silico Biology* 6, 15-22.

12. Kauzmann W. (1959), Some factors in the interpretation of protein denaturation. *Adv Protein Chem* 14:1–63.
13. Roterman I., Konieczny L., Jurkowski W., Prymula K., Banach M. (2011), Two-intermediate model to characterize the structure of fast-folding proteins. *J Theor Biol.* 283(1):60-70.
14. Roterman I., Tomanek M., Sterzel M., Szepieniec T., Kalinowska B., Baster Z., Dułak D. (2013), Managing Protein Folding Process with Intelligent Process Parameters Adjustment. In: K. Wiatr, J. Kitowski, M. Bubak (Eds) *Proceedings of the Sixth ACC Cyfronet AGH Users' Conference*, ACC CYFRONET AGH, Cracow, ISBN 978-83-61433-07-1, pp. 9-12.
15. Zhou A. Q., O'Hern C. S., Regan L. (2011), Revisiting the Ramachandran plot from a new angle. *Prot Sci* 20:1166–1171.
16. Baster, Z. (2011), *Generowanie Struktury Trójwymiarowej Białka Dla Zadanych Wartości Kątów Dwuściennych Phi, Psi.* (Bachelor's Thesis). AGH University Of Science and Technology, Cracow, Retrieved September 12th ,2013, from http://www.zbaster.com/engthesisbaster.pdf
17. www.zbaster.com, Retrieved July 14th, 2013.
18. Academic Computer Centre Cyfronet AGH, http://www.cyfronet.krakow.pl, Retrieved August 8th, 2013.
19. Wikiuser: Urutseg, License: Creative Commons Attribution-Share Alike 3.0 Unported, Retrieved July 14th ,2013, from http://upload.wikimedia.org/wikipedia/commons/b/bb/Praezession.svg
20. Protein Data Bank, http://www.rcsb.org, Retrieved July 14th, 2013.
21. PL-Grid Infrastructure, http://www.plgrid.pl/en, Retrieved July 14th, 2013.
22. Roterman I., Tomanek M., Sterzel M., Szepieniec T., Kalinowska B., Baster Z., Dułak D. (2012), Managing Protein Folding Process as Workflow Model with Wise Data Selection. In: M. Bubak, M. Turała, K. Wiatr (Eds) *CGW'12 Proceedings*, ACK CYFRONET AGH, Kraków, ISBN 978-83-61433-06-4, pp. 93-94.

Appendix A. Program layout

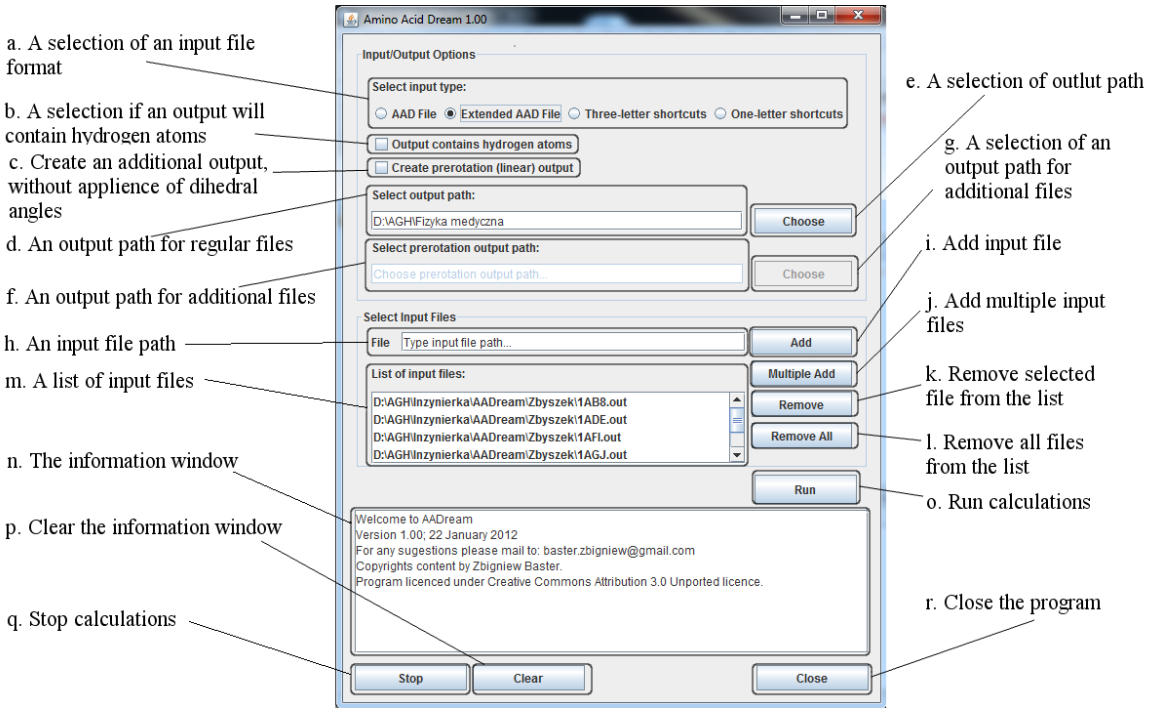


FIGURE 25. The layout of AADream 1.00. Based on [16].

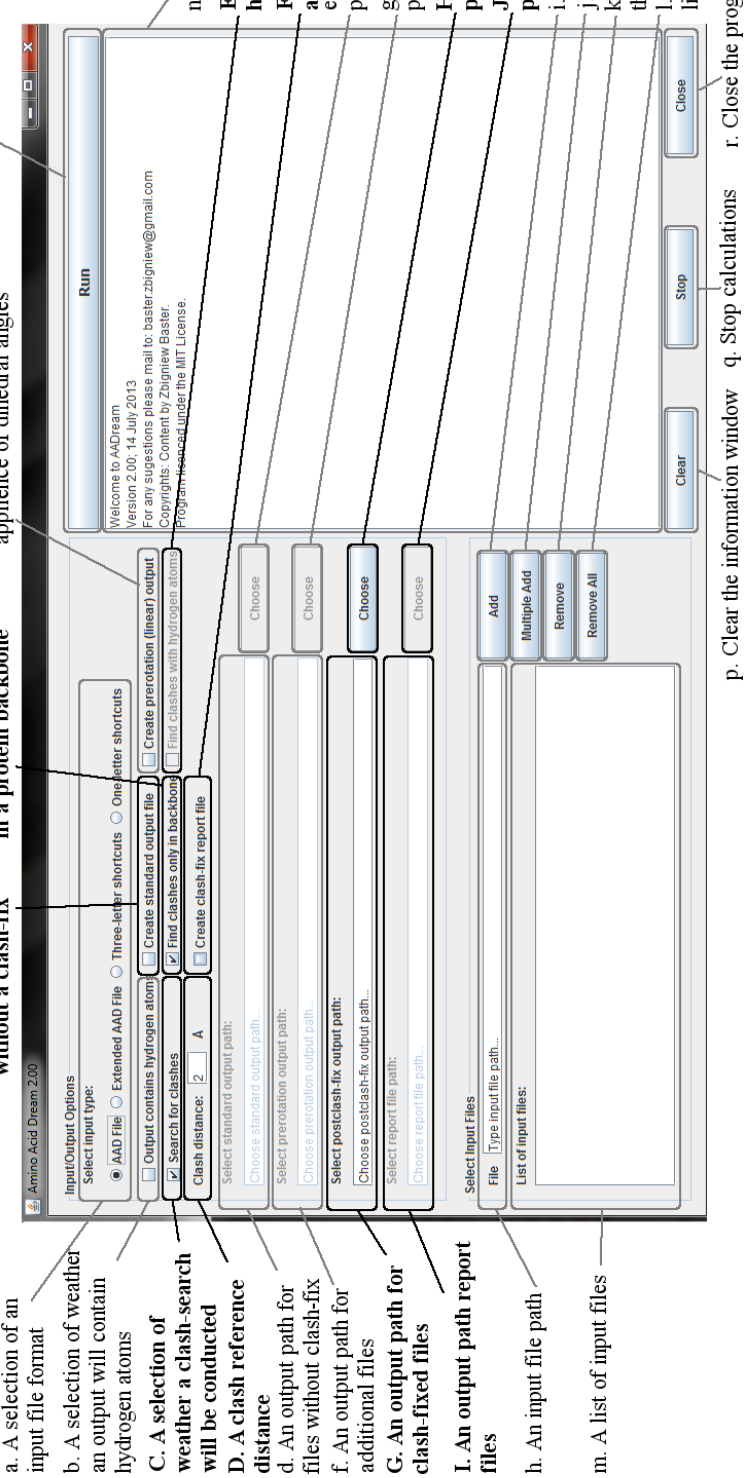


FIGURE 26. The layout of AADream 2.00. New features are marked with black frames, with bolded descriptions; old are marked with grey frames.

Appendix B. Report file

A report file contains information about conditions of the clash-remove process, unsolved clashes, and total number of clashes in each iteration. A sample report file is presented below:

```
The report of file 1ARR

Parameters:
Clash distance: 1.8 Å
Clashes searched for in the whole protein

1.1  A5 LYS O      A7 PRO HD    1.643  1.643   E   E     F
1.2  A5 LYS HG     A7 PRO HD    1.788  1.788   E   E     F
1.3  A12 ARG O     A14 PRO HD   1.338  1.338   E   E     F
1.7  A56 LYS O     A58 PRO HD   1.494  1.494   D   E     F
1.8  A63 ARG O     A65 PRO HD   1.338  1.338   E   E     F
Number of clashes found in iteration 1: 16
Number of clashes found in iteration 2: 7
Number of clashes found in iteration 3: 21
4.1  A82 SER HN    A86 GLU CA   1.735  1.735   F   ABC   B
4.2  A82 SER O     A86 GLU N    1.621  1.621   F   ABC   B
4.3  A82 SER HB    A86 GLU N    1.478  1.478   F   ABC   B
Number of clashes found in iteration 4: 4
5.6  A81 ARG HD    A85 SER N    1.530  1.530   E   FAB   C
Number of clashes found in iteration 5: 7
Number of clashes found in iteration 6: 1
```

The first line of a report file informs about an output file that it is correlated with.

The forth line of a report file informs about the reference clash distance.

The fifth line of a report file informs about the coverage of the search, possible sentences are:

- *Clashes searched for in the whole protein*
- *Clashes searched for only in the backbone*
- *Clashes searched for in the protein with hydrogen atoms excluded*

The seventh and next lines inform about unsolved clashes, or the number of clashes found in an iteration.

FIGURE 27 presents a single report line of an unsolved clash:

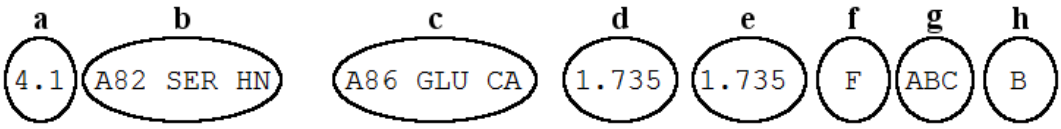


FIGURE 27. A report line of an unsolved clash.

Where:

a) An ID number of a clash, the first letter represents the number of an iteration, the second one – the number of a clash found in the iteration
b) Information about the first clashed atom: a protein strand letter, a residue number, a type of residue, and an atom of a residue.
c) Information about the second clashed atom; description similarly to the first one.
d) A distance between atoms before the clash-remove process expressed in ångströms.
e) A distance between atoms after the clash-remove process expressed in ångströms.
f) The zone of the first clashed atom's residue.
g) Zones of residues between clashed atoms, in their order in a strand; each letter represents a single residue.
h) The zone of the second clashed atom's residue.

Appendix C. Determination of a *t*-parameter.

TABLE 8. The determination of a *t*-parameter. Proper values are marked in gray.

t [^{o}]	Φ(t) [^{o}]	Ψ(t) [^{o}]	Φ(t)- Ψ(t) [^{o}]	$arccos\left[\frac{(\Phi(t)-\Psi(t))\sqrt{2}}{250}\right]$ [^{o}]	$360^o - arccos\left[\frac{(\Phi(t)-\Psi(t))\sqrt{2}}{250}\right]$ [^{o}]	Φ(t)+Ψ(t) [^{o}]
0	88,4	-88,4	176,8	0	360	0
10	76,7	-97,4	174,1	10	350	-20,6
20	62,7	-103,4	166,1	20	340	-40,6
30	46,8	-106,2	153,1	30	330	-59,4
40	29,5	-105,9	135,4	40	320	-76,4
50	11,3	-102,3	113,6	50	310	-91
60	-7,2	-95,6	88,4	60	300	-102,9
70	-25,6	-86	60,5	70	290	-111,6
80	-43,1	-73,8	30,7	80	280	-117
90	-59,4	-59,4	0	90	270	-118,8
100	-73,8	-43,1	-30,7	100	260	-117
110	-86	-25,6	-60,5	110	250	-111,6
120	-95,6	-7,2	-88,4	120	240	-102,9
130	-102,3	11,3	-113,6	130	230	-91
140	-105,9	29,5	-135,4	140	220	-76,4
150	-106,2	46,8	-153,1	150	210	-59,4
160	-103,4	62,7	-166,1	160	200	-40,6
170	-97,4	76,7	-174,1	170	190	-20,6
180	-88,4	88,4	-176,8	180	180	0
190	-76,7	97,4	-174,1	170	190	20,6
200	-62,7	103,4	-166,1	160	200	40,6
210	-46,8	106,2	-153,1	150	210	59,4
220	-29,5	105,9	-135,4	140	220	76,4
230	-11,3	102,3	-113,6	130	230	91
240	7,2	95,6	-88,4	120	240	102,9
250	25,6	86	-60,5	110	250	111,6
260	43,1	73,8	-30,7	100	260	117
270	59,4	59,4	0	90	270	118,8
280	73,8	43,1	30,7	80	280	117
290	86	25,6	60,5	70	290	111,6
300	95,6	7,2	88,4	60	300	102,9
310	102,3	-11,3	113,6	50	310	91
320	105,9	-29,5	135,4	40	320	76,4
330	106,2	-46,8	153,1	30	330	59,4
340	103,4	-62,7	166,1	20	340	40,6
350	97,4	-76,7	174,1	10	350	20,6
360	88,4	-88,4	176,8	0	360	0

Appendix D. Diagrams

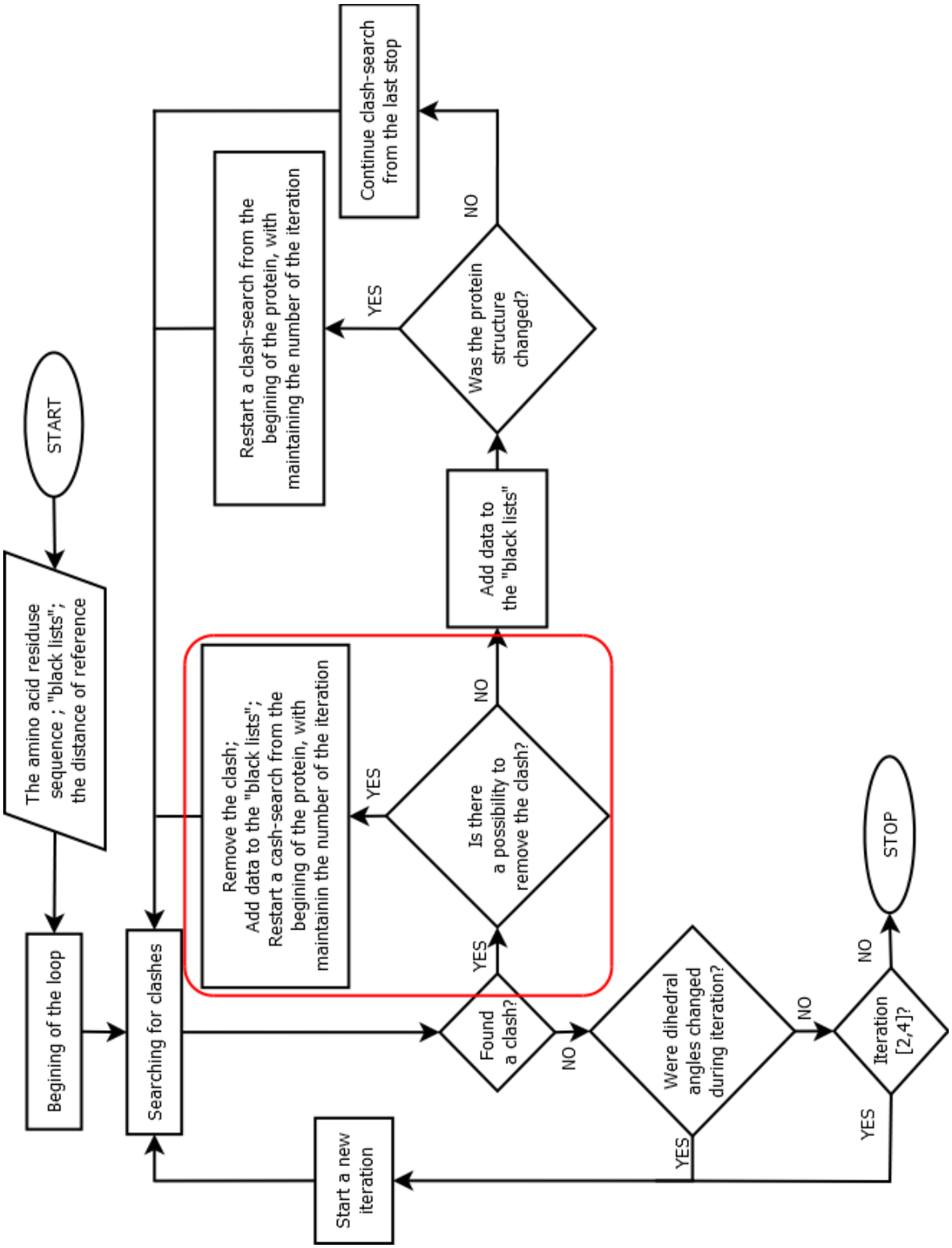


FIGURE 28. The diagram of the global solution in general.

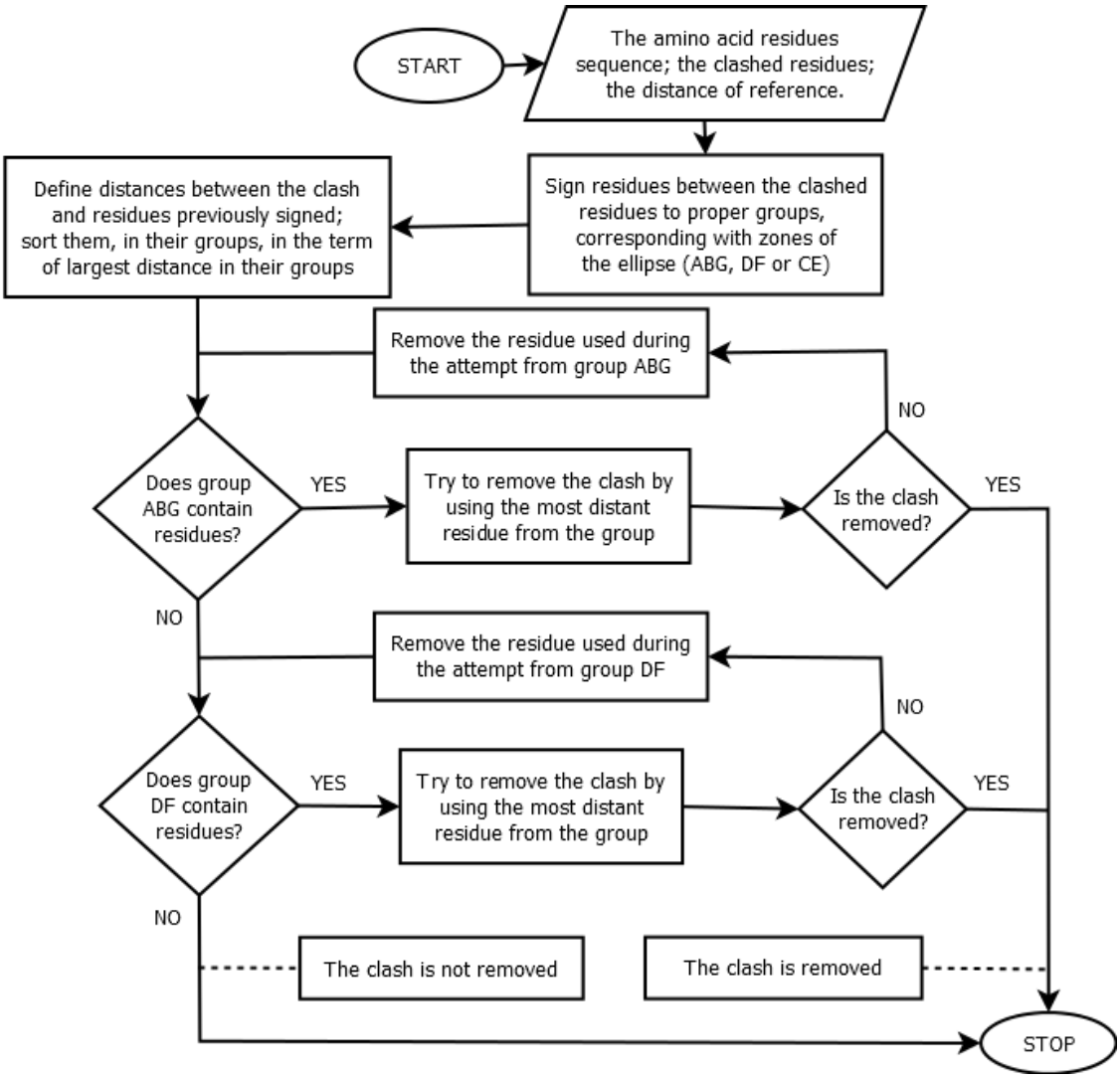


FIGURE 29. The diagram of the clash-remove sequence, after finding the clash.

Appendix E. Results table

TABLE 9. Efficiency tests results with an analysis.

ID	Protein	1ARR	1B8Z	1AC6	1ARJ	1AD3	1AOZ	1ARQ
1^o	**The length of a protein [residues]**	102	126	212	814	888	1100	1632
2^o	**The number of clashes in the first iteration**	15	24	35	2842	679	333	354
3^o	**The number of unsolved clashes in the first iteration**	5	4	5	62	65	85	58
4^o	**The number of unsolved clashes in the first iteration with only zone-C or –E residues available to solve the clash with**	5	4	5	55	54	79	58
5^o	**The number of unsolved clash situations[1] in the fifth or greater iterations**	1	3	6	54	46	8	5
6^o	**The efficiency of removing clashes in the first iteration** $\left(\frac{3^o}{2^o} \cdot 100\%\right)$ **[%]**	66,7	83,3	85,7	97,8	90,4	74,5	83,6
7^o	**The efficiency of removing clashes in the first iteration with zone-C or –E clashes omitted** $\left(\frac{3^o-4^o}{2^o-4^o} \cdot 100\%\right)$ **[%]**	100	100	100	99,7	98,2	97,6	100
8^o	**The number of clashes in the first iteration per 100 residues** $\left(\frac{2^o}{1^o} \cdot 100\right)$	14,7	19,0	16,5	349,1	76,5	30,3	21,7
9^o	**The number of unsolved clashes in the first iteration per 100 residues** $\left(\frac{3^o}{1^o} \cdot 100\right)$	4,9	3,2	2,4	7,6	7,3	7,7	3,6
10^o	**The number of unsolved clashes in the first iteration with zone-C or –E clashes omitted per 100 residues** $\left(\frac{3^o-4^o}{1^o} \cdot 100\right)$	0	0	0	0,9	1,2	0,5	0

[1] Usually, in fifth or greater iteration not only one atom from one reside clashes with another atom, but many atoms from two colliding residues do it with each other, and this situation, often, can be change with only one try (if that try was possible); if clashes covering the same or adjacent residues occur, they are counted as one.

TABLE 9. Continuation.

ID	Protein	1ARR	1B8Z	1AC6	1ARJ	1AD3	1AOZ	1ARQ
11º	**The number of unsolved clash situations in the fifth or greater iterations per 100 residues** $\left(\frac{5^o}{1^o} \cdot 100\right)$	1	2,4	2,8	6,6	5,2	0,7	0,3
12º	**The number of unsolved clash situations in the fifth or greater iterations per 100 first iteration clashes** $\left(\frac{5^o}{2^o} \cdot 100\right)$	6,7	12,5	17,1	1,9	6,8	2,4	1,4
13º	**The sum of the number of unsolved clashes in the first iteration and the number of unsolved clash situations in the fifth or greater iterations per 100 first iteration clashes** $\left(\frac{3^o+5^o}{2^o} \cdot 100\right)$	40	29,2	31,4	4,1	16,3	27,9	17,8
14º	**The sum of the number of unsolved clashes in the first iteration and the number of unsolved clash situations in the fifth or greater iterations per 100 first iteration clashes with zone-C or –E clashes omitted** $\left(\frac{3^o+5^o-4^o}{2^o-4^o} \cdot 100\right)$	10	15	20	2,2	9,1	5,5	1,7
15º	**The number of zone-C or –E clashes to the number of unsolved first iteration clashes ratio** $\left(\frac{4^o}{3^o} \cdot 100\%\right)$ **[%]**	100	100	100	88,7	83,1	92,9	100