Bachelor Thesis

Hybrid Genetic Algorithm for Protein Folding Simulations in the 2D HP model

Katherina von Carlowitz
kvoncarl@uos.de

August 25, 2008

First supervisor:	Prof. Dr. Volker Sperschneider
Second supervisor:	Jun. Prof. Dr. Sigrid Knust

Abstract

The prediction of a protein's structure from its amino-acid sequence is one of the most important problems in computational molecular biology. In this thesis, we demonstrate a hybrid genetic algorithm that simulates protein folding under the widely studied 2-dimensional hydrophobic- hydrophilic (HP) lattice model. The protein folding problem in the HP model is to find a lowest energy conformation, which is known to be a NP-hard combinatorial problem. In comparison to similar algorithms, our algorithms performed well on standard benchmark instances. In addition, we present a graphical version of the genetic algorithm that uses secondary structures in protein folding.

Contents

List of Tables

List of Figures

1. Introduction

Proteins play a variety of important roles in biological systems. Their functions are quite diverse, for example myosin and actine are involved in muscle contraction, haemoglobin is responsible for the oxygen transport in the blood, structural proteins determine the structure of cells, other proteins help in the control of brain signals, and so forth.

The amino acid sequence determines the three-dimensional shape of a protein. Any protein can spontaneously fold into a stable unique native conformation which is influenced by cellular environment surrounding the polypeptide chain. Proteins are only active if they have completed the folding and are in their native state. Thus, a protein's function depends mainly on its tertiary structure, and the tertiary structure in turn depends on the amino acid sequence.

Mistakes in the folding process can lead to a loss of protein function, which can cause several sporadic and genetic diseases. These include diseases such as Alzheimer's and Parkinson's [4]. A better understanding of protein folding would certainly lead to an improved treatment of these diseases.

The protein folding problem, that is predicting the three dimensional structure of a protein from knowledge of its amino acid is a problem of long standing interest in molecular biology, computational biology, biochemistry and physics. Even under simplified models, such as the HP model, this problem is proved to be NP-hard.

According to the *thermodynamic hypothesis* the native state of a protein is that conformation which has the lowest free energy. Though this hypothesis has not yet been proven, many computer simulations are based on predicting the lowest energy conformation. Therefore, an energy function is required that allows to determine the native state of a protein. In addition, they have to use a suitable model that captures the essence of the important components of protein folding. To find minimal energy conformations, an appropriate search algorithm has to be found. Thereby, the systematic evaluation of the free energy of every possible conformation is not practical using.

1.1. Previous Work

A number of search methods have been proposed in the literature to solve the protein folding problem for both the two-dimensional and three-dimensional HP model, including Monte Carlo algorithms and evolutionary algorithms.

In 1993, Unger and Moult presented a genetic algorithm for protein folding simulations [27], an early application of genetic algorithms. Thereby, characteristics of the familiar Monte Carlo methods were incorporated in the general search technique.

Other approaches that incorporated the idea of genetic algorithms, combined and optimized it with other techniques, include e.g. the genetic algorithm combined with Tabu search introduced by Jianga et al., 2003 [14], where Tabu search is applied to the crossover operator. Liang and Wong presented an Evolutionary Monte Carlo Algorithm, 2001 [20], that incorporates the genetic algorithm and simulated tempering. In addition they proposed the use of secondary structures. An idea that was further refined by Bui and Sundarraj in their Protein Folding Genetic Algorithm, 2005 [2].

Other approaches for solving the protein folding problem include the dynamic Monte Carlo Algorithm by Ramakrishnan et al. [22], an algorithm for dense single chains which generates new configurations by breaking and patching the chain. Grassberger et al. implemented improved versions of the pruned-enriched-Rosenbluth method (PERM) in 1997 [10] (improved versions in 1998 [1] and in 2003 [13]. PERM belongs to the Monte Carlo algorithms and is a biased chain growth algorithm that evaluates partial conformations and employs pruning and enrichment strategies to explore promising partial solutions.

An evolutionary different from the previous ones is the Ant Colony Optimization Algorithm presented in [24] by Shmygelska and Hoos for the 2D HP model, 2003, and in [25] an improved version for both 2D and 3D.

1.2. Objectives and Organization of Thesis

In this thesis, we present a hybrid genetic algorithm for the 2D HP model that is motivated by the genetic algorithm of Unger and Moult.

An introduction to topics relevant for our algorithm implementation is given in chapter 2. In particular, this chapter outlines protein folding in nature, the 2D HP Model, an widely used model for protein folding simulations our algorithm is also based on, and genetic algorithms. The following chapter gives an detailed description of our algorithm. In chapter 4, the experimental results of tests executed with our algorithm are presented. In addition, our results are compared to other algorithms dealing with the 2D HP Protein Folding Problem. Following the discussion, we present a modified version of our algorithm that employs secondary structures. The chapter further gives information about the biological background, and introduces the graphical implementation of the modified algorithm.

2. Background

2.1. Protein Folding

Experiments of Anfinsen in 1961 and 1973 implied that the amino acid sequence contains sufficient information to define the three-dimensional structure of a protein in a particular environment [9]. From this broadly accepted tenet derives the *Protein Folding Problem*: The prediction of the native structure of proteins based only on the consideration of the sequence.

2.1.1. Conformational States of Proteins

For each natural amino acid sequence, there is a unique stable native state, in which under the adequate conditions the molecule folds on its own. By heating and bringing the protein into circumstances very different from the normal physiological environment, the protein defolds and forms a disordered, biologically inactive structure. This process is usually reversible: Changing the conditions back to normal, the protein again folds into its native structure, which is the only conformational state where the protein is active (cf. [18]).

The native state

A unique primary structure has the potential of great diversity at the level of three-dimensional conformation. For example, with only three conformations defined per residue, a polypeptide chain of 166 residues would have a theoretical possibility of existing in 10^{79} different conformations [5]. However, the great majority of natural proteins exists in a unique conformation which may be determined to atomic resolution by X-ray crystallography and by Nuclear Magnetic Resonance Spectroscopy (NMR). An example of a tertiary structure obtained by X-ray defraction can be seen in Figure 2.1.

Proteins with non-homologous amino acid sequences usually have different conformations whereas homologous proteins invariably have essentially the same folded conformation. Proteins vary in the degree of flexibility which is greatest at the protein surface, where some side chains and a few loops have alternative conformations or no particularly preferred conforma-

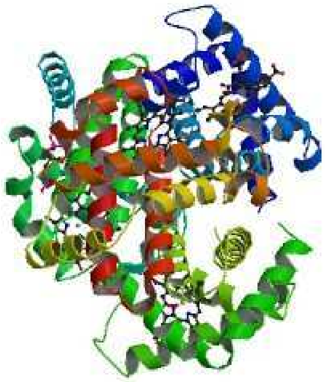


Figure 2.1.: Three-dimensional structure of the human deoxyhaemoglobin-2,3-diphosphoglycerate complex (adapted from http://www.rcsb.org/pdb/explore.do?structureId=1B86). The structure was determined by X-ray defraction. The figure shows the ribbon model of the protein, in which a ribbon is drawn along the polypeptide backbone.

tion. Although there is as well some flexibility in the interior, the basic architecture of the protein generally stays relatively close to the average structure determined by X-ray crystallography or by NMR [6].

The native folded state of a protein is assumed to correspond to the conformation which has the lowest free energy, a specific folding pattern that is thermodynamically more favourable than other conformations.

Stability, denaturation and folding

In contrast to a unique native conformation, in the denatured polypeptide chain an immense number of conformations have essentially equal energies. The denatured protein becomes less compact, and much more flexible. These conformations are very nearly random coils. In a random coil, rotation about each bond is assumed to be independent of the rotation of all other bonds distant in the covalent structure, and the rotations take place with the same freedom they would have in an analogous small molecule [5].

Proteins only possess a marginal stability, and the native state is only slightly more stable than the denatured, unfolded state. The difference corresponds to energy contribution of one or two hydrogen bonds [18]. For a protein to adapt its native structure, there are stabilizing interactions within its building blocks and between them. One of these physical forces is the hydrophobic effect.

2.1.2. Hydrophobic Force

Though there are also other types of interactions present within folded proteins, such as hydrogen bonds, van der Waals' interactions, and electrostatic interactions, the dominant driving force for determining the unique native structure of globular proteins is believed to be the hydrophobic force [16, 23]. In the folded state of a protein, hydrophobic amino acids, such as alanine, valine, leucine, isoleucine, phenylalanine, and methionine, tend to form a core shielded from the surrounding solvent by hydrophilic acids. The hydrophobic core seems to be the most critical aspect for stability of the normal folded state of a protein [6].

It is thermodynamically unfavourable for a protein to fold into a native globular conformation because this process leads to a reduction of freedom, and thus a decreased entropy. This is balanced by inducing a compact globular state where non-polar amino acids are inside the molecule and removed from contact with water. The thermodynamic consequences of the unfavourable interactions of such non-polar regions with water are defined as the hydrophobic effect: It follows a compensative increase of the entropy [26, 18].

An illustrative example of the hydrophobic effect in the molecular model of lysoszyme can be seen in Figure 2.2. The core of the protein is mainly composed of hydrophobic residues (dark) which are surrounded by mainly hydrophilic residues (light).

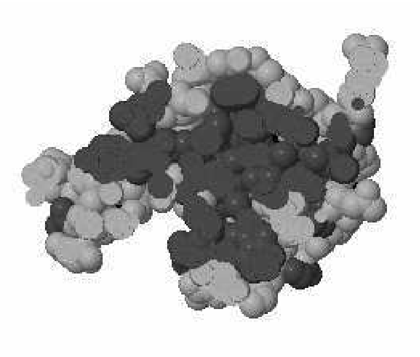


Figure 2.2.: Hydrophobic effect in the molecular model of lysoszyme: The protein consist of a core composed mainly of hydrophobic residues (dark) surrounded by mainly hydrophilic residues (light). Amino acids are drawn with space filling atoms. This figure shows the whole molecule in a cutaway section (modified from http://webhost.bridgew.edu/fgorga/proteins/hydrophobic.htm).

2.2. The two-dimensional HP Model

The assumption that the native structure of a protein corresponds to a free energy minimum forms the basis for many computational simulations which attempt to predict the tertiary structure of proteins from its amino acids sequence. Consequently, the native structure is then presumed to be a conformation which minimizes the energy function over the set of possible conformations.

One of the simplest and most predominant models used for protein folding simulations is the the hydrophobic-hydrophilic (HP) model, which was introduced by Lau and Dill [16, 17] in order to explore the full conformational and sequence spaces of proteins. Though it is both simple and low-resolution, it still exhibits the important features of real protein folding.

The HP model belongs to the so-called lattice models, which are characterized by the following simplifications:

(1) a uniform size of residues,
(2) a uniform bond length and
(3) the positions are restricted to positions in a regular lattice (cf. [3]).

2.2.1. The Model

The HP model is based on the observation that the hydrophobic force is the main force for protein folding. In the model, a protein is represented as a linear sequence being composed of amino acids of only two types: Hydrophobic amino acids represented by H and hydrophilic or polar amino acids represented by P. This sequence is folded on a two-dimensional (2D) square lattice[1]. Thus, each amino acid is occupying one lattice site, connected to its chain neighbour(s). Only self-avoiding structures are valid conformations of a sequence. There is a distinction between *connected* neighbours, which are adjacent along the chain sequence, and *topological* neighbours. The latter are neighbouring non-diagonal lattice points which are not adjacent in position along the sequence. From this it follows that each amino acid can have two topological neighbours at most, except for the first and the last amino acid in the chain, which can have three topological neighbours.

[1] There also exists a 3D HP model, in which a conformation is embedded as a self-avoiding walk in a three-dimensional lattice. This model however is not considered here.

2.2.2. Energy Function

The energy function of a conformation is simple and defined as follows: Every H-H contact between topological neighbours has a contact free energy of -1, and any other interaction of neighbours has an energy of 0. This means that the free energy of a conformation corresponds to the negative number of topological H-H neighbours. Thus, a conformations with minimal energy is a conformation that maximizes the number of H-H contacts. Figure 2.3 shows a sample conformation that has two topological H-H contacts, denoted by the dotted lines. Thus, the energy value of the structure is -2.

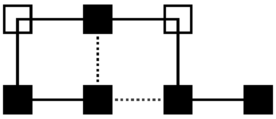


Figure 2.3.: Sample conformation for HHPHPHH. Hydrophobic amino acids (H) are represented as black squares, polar (P) as white squares. The 2 topological H-H contacts are denoted by dotted lines

Formal definition

Formally, a native state is a conformation having minimum contact energy (cf. [3])

$$E = \sum_{1 \le i+1 \le j \le n} B_{i,j} \delta(r_i, r_j)$$

$$\text{where } \delta(r_i, r_j) = \begin{cases} 1 & \text{if } \|r_i - r_j\| = 1 \text{ and } i \neq j \pm 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{and } B_{i,j} = \begin{cases} -1 & \text{if } i \text{ and } j \text{ are hydrophobic} \\ 0 & \text{otherwise.} \end{cases}$$

Compact form

Under this energy function, low energy conformations are compact with a hydrophobic core. Since H-H interactions are rewarded, the hydrophobic residues tend to be inside of a low energy structure, while polar residues are forced to the surface. The Figures 2.4 (a)-(d) illustrate this tendency in optimal conformations of a 25-mer sequence.

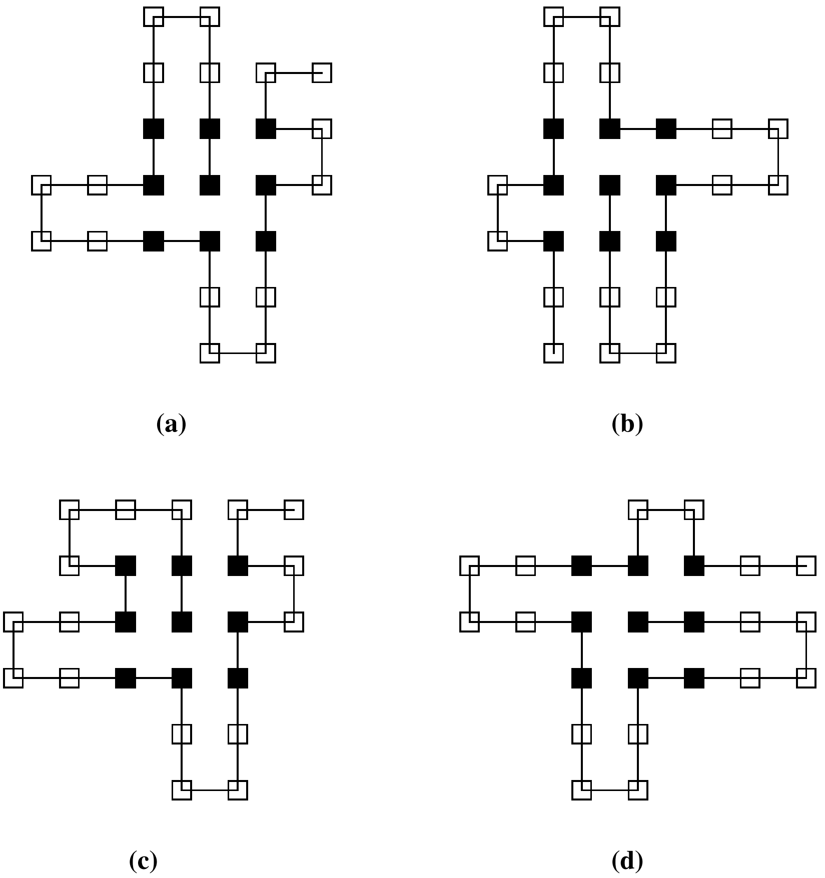


Figure 2.4.: Optimal conformations of the 25-mer sequence PPHPPHHPPPPHHPPPPHHPPPPHH. These conformations are compact with a hydrophobic core

2.2.3. Complexity

Despite the simplifications of the model, the HP Protein Folding Problem, which means finding a conformation that maximizes the number of topological H-H contacts, remains a NP complete problem. The NP-hardness has been proven for this problem and several variations of it [7, 28].

2.3. Introduction to Genetic Algorithms

Genetic algorithms were developed by John H. Holland [11, 12] in quest for a mathematical model of adaptive processes. Being derived from evolutionary biology, adaptation designates a process whereby a structure is progressively modified to make it better suited to its environment. A genetic algorithm is a search technique that is loosely based on simulated evolution (cf. [21]). It uses optimization techniques inspired by biological adaptive processes such as mutation, selection and recombination. This approach makes it possible to explore a far greater range of potential solutions to a problem than conventional programmes.

2.3.1. Definition

Genetic algorithms repeatedly generate new solutions by mutating and recombining parts of the existing solutions. The pool of candidate solutions is called the *population* and is created randomly in the first step. *Individuals* or *chromosomes* encode the potential solutions. Often, solutions are represented as binary strings of 0s and 1s, but other encodings are also possible. The population is iteratively updated by replacing the individuals by offspring of current solutions to form a new *generation*. Thereby, the size of the population is maintained.

In each generation, all individuals are evaluated according to the *fitness function*, a predefined numerical measure. The best solutions are defined as the one that optimize the fitness function. Based on their fitness, multiple solutions of the current population are then probabilistically selected to produce the next generation.

According to a certain ratio, some of the individuals are carried forward intact, others are used for creating new offspring by applying genetic operators such as mutation and crossover.

2.3.2. General Observations

Genetic algorithms often rapidly find solutions, even for difficult search spaces. Thereby, the search can move quite abruptly. This occurs if a parent solution is replaced by an offspring that may be completely different from the parent. As a consequence, the algorithm is less likely to fall into the same kind as local minima as for example gradient descent methods.

However, one problem of GAs is the so-called crowding problem: Some individual is fitted better than others, and then quickly reproduces. As a result, copies of this individual and similar individuals take over a large fraction of population, which reduces diversity. The problem can be addressed by using another selection procedure for recombination.

In general GAs often converge to local minima, rather than to global ones. There is no general solution to that problem, but it is important to maintain a diverse population e.g. by a different fitness function, by increasing the mutation rate, or by another selection techniques. Otherwise, the structures become homogeneous, and recombination does not produce new solutions (cf. [21]).

3. The Algorithm

Our hybrid genetic algorithm duplicates the algorithm implementation of Unger and Moult, 1992 [27], which seeks to find the minimal energy conformations in the 2D square model using the search strategy of genetic algorithms. As specified in the HP model, the energy function is defined by the number of topological H-H neighbours. Furthermore, we tested a second version of the algorithm which differs in the way the initial population is created. The following overview however, is applicable for both versions.

```
t=0
initialize population P(t) of m conformations

while(t < iterations || E(best) <= optimal energy) {
    t++
    best conformation = argmin { E(x) | x in P(t) }
    mutate all individuals
    n = 0

    while (n < m) {
        select 2 parents p1, p2
        produce child c by crossover of p1 and p2
        n++
    end while
    place best into next generation
end while
```

Figure 3.1.: Overview of the algorithm

The algorithm starts by creating a population of *N* initial structures. Then follows an iterative process: In each generation, each individual goes through a number of mutation steps. In our experiments, the structures were mutated 20 times each. Subsequently, the crossover operator is applied: Always two conformations are joined to generate new offspring individuals that form the next generation population. The chance of a conformation to be selected for crossover is proportional to its fitness. This step of recombining is repeated until *N-1* valid structures are created. In addition, the best energy conformation is directly replicated to the next generation without any operator applied to it. Thus, a population of N conformations is maintained. In our experiments, in most cases N was a population of 200 individuals created. If not, it is noted at respective site. Note that each mutated and recombined structure is subject to certain acceptance criteria explained below. This process is iterated for the next generations until the stopping criteria are met. This is the case when either a given number of iterations is reached or the optimal energy[1] was found. We set the maximal number of iterations to 300. The pseudo code of the algorithm is given in Figure 3.1. The following sections go into details of the algorithm.

3.1. Representing Sequences

In this genetic algorithm, the individuals or chromosomes are conformations of a protein themselves, to which the mutation and crossover operators are directly applied. The representation of a conformation always denotes a valid conformation in the 2D square lattice.

Each conformation is represented as a sequence over the alphabet $\Sigma = l, r, s$, where the letters indicate a left, right or straight fold direction. Thus, it is encoded in such a way that the fold directions are relative to the conformation itself. If the input amino acid sequence is a string of length n, then each individual in the the population is a string of length *n-1*.

The example in Figure 3.2 shows a conformation of the 20 amino acid molecule HPHPPHHPHPPHPHHPPHPH and would be denoted as *srsrrllrsrrlrllrrsr*. The first fold direction in this encoding scheme is fixed as *s* denoting straight since a change of the first symbol only means rotating the entire conformation. Accordingly, it is not necessary to be mentioned explicitly, and a given sequence of length n could be represented using *n-2* symbols as well.

3.2. Initial Population

As mentioned above, we implemented two versions of the algorithm which differ in the initialization of the population.

[1] For our experiments, we used the optimal energies from literature sources [27, 1, 20]

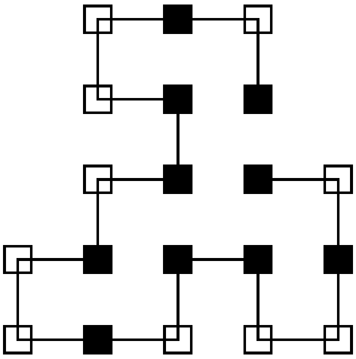


Figure 3.2.: A conformation for the 20-mer sequence HPHPPHHPHPPHPHHPPHPH. Its string representation would be *srsrrllrsrrlrllrrsr*.

3.2.1. Fully Extended Structures

The first version (ES) follows the description of the Unger-Moult algorithm as close as possible. Here, *N* fully extended structures (i.e. a straight line) are created for the initial population, which are subsequently mutated and recombined in the course of the search process [27].

3.2.2. Random Coils

In contrast, the modified second version (RC) starts with *N* strings that are created randomly from the available fold directions. The random coils correspond to the unfolded state of proteins, rather than the fully extended structures. After creation of an individual, its energy is evaluated. The second procedure requires apparently more computational effort at the beginning than the first version, especially for longer sequences. These additional expenses are due to more invalidly created conformations. However, at the same time there is more diversity among the individuals which in turn reduces the likelihood for the algorithm to drop in a local minimum.

3.3. The Operators

Every iteration, the population is updated by the mutation and the crossover operator. First all individuals are repeatedly mutated, and then the individuals are recombined to form the next generation. After each change to a conformation its energy is evaluated: Compact conformations with a low energy value are more likely to be accepted for the next generation.

3.3.1. Mutation

Mutations were performed using the Metropolis Monte Carlo Procedure. In this procedure, an amino acid of a structure S_1 with energy E_1 is randomly selected, and the successive residues of the chain are rotated around that amino acid. The rotation can either be 90° right, 90° left, or 180°. If the resulting structure S_2 is a valid conformation, the energy E_2 is evaluated. Otherwise, another rotation angle is applied. In case none of the rotation angles leads to a valid structure, another amino acid is selected as pivot for mutation and the process is repeated until a valid structure is found. Figure 3.3 shows the sequence HPHPPHHPHPPHPHHPPHPH before and after mutation.

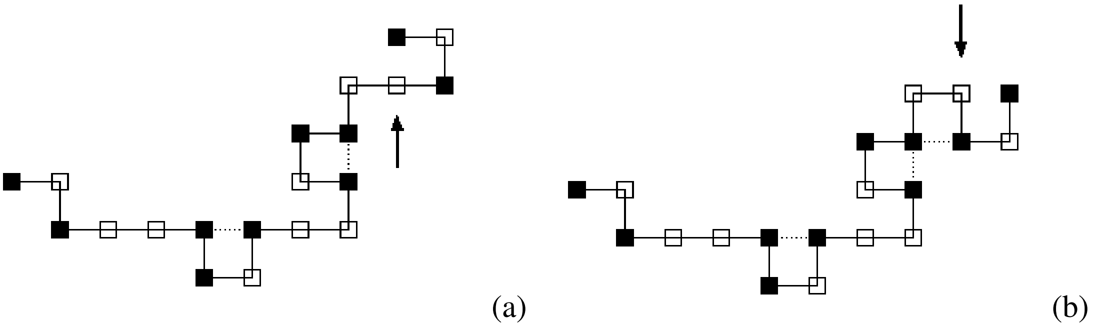


Figure 3.3.: The 20-mer sequence HPHPPHHPHPPHPHHPPHPH before (a) and after (b) mutation. Residue number 17 was randomly selected as the pivot for mutation. The successive residues of the chain were rotated 90° right around that amino acid. This rotation improves the energy from -2 in (a) to -3 in (b)

Acceptance criteria

For a valid conformation S_2 the following acceptance criteria apply. If the evaluated energy $E_2 \leq E_1$, then the conformation S_2 is accepted. Otherwise, it is non-deterministically decided whether to accept the change despite the energy increase. This second criterion is satisfied if:

$$Rnd < \exp\left[\frac{E_1 - E_2}{c_k}\right]$$

where *Rnd* is a random number between 0 and 1, and c_k is gradually decreased (cooled) during the simulation. If the change was not accepted, the former conformation S_1 is retained.
In our experiments, the cooling scheme for mutation starts with $c_k = 2$ and is cooled by $c_k = 0.97c_k$ every 5 generations.

3.3.2. Crossover

In the crossover operator, offspring is produced by the recombination of two parental individuals to form a new population. Since the lowest energy conformation is passed directly into the new generation, *N-1* new structures have to be created by crossover.

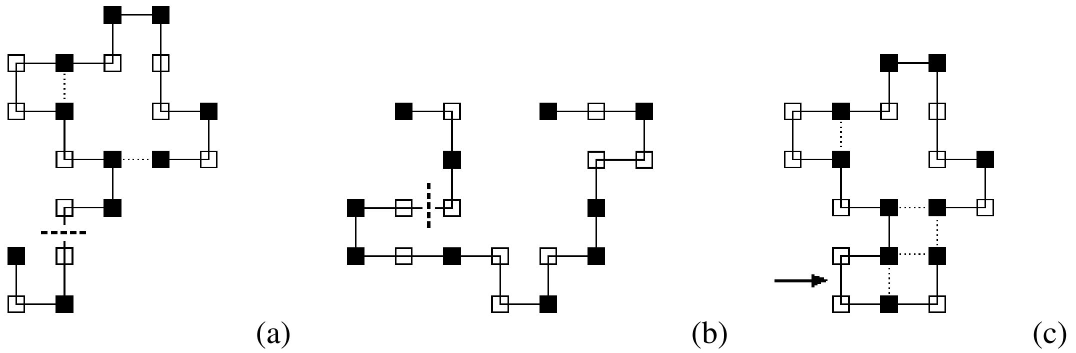


Figure 3.4.: In this example the cutpoint was randomly chosen to be after the residue number 16. The first 16 residues of the structure (a) and the last 4 residues of (b) are joined together to form the new conformation (c). In addition, the second part was rotated 90° right. The resulting structure (c) has an energy of -5 which is lower than the average energy of its parents.

The recombination of structures proceeds as follows: For a pair of structures S_{p_1} and S_{p_2} selected from the current population, a random amino acid is chosen along the sequence. From the structure S_{p_1} the part before that point and from the structure S_{p_2} the part behind that point are connected. The parts can be joined together in three ways, namely with the angles of 0°, 90° or 270°. Again, only self-avoiding structures are accepted. If none of the three angles leads to a valid structure, the two parental structures are swapped and joined together at a new random point. Figure 3.4 shows how the recombination of two structures leads to a new more compact conformation. The parents are selected for crossover using the roulette wheel technique.

Roulette wheel technique

In this technique, each individual is assigned a part of the roulette wheel. The size of each part is proportional to the fitness of the individual. The fitness $F(S_i)$ of a conformation S_i equals $-E(S_i)$. Thus, the chance of an individual being selected for crossover is proportional to its energy value, i.e.

$$p(S_i) = \frac{E_i}{\sum_{j=1}^{N} E_j}$$

The wheel is spun n times to select n individuals.
Figure 3.5 shows the implementation simulating the roulette wheel technique: In the first step, the fitnesses of all strings are summed up. Then secondly, a random number between 0 and this sum is generated. Thirdly, the first string is returned whose fitness value, added to the fitness values of the preceding strings, is greater than or equal to the random number.

```
# Step 1
for all conformations of the population
    sum += individual fitness

# Step 2
random = random number between 0 and sum

# Step 3
count = 0
conformation c
while (count < random) {

    c = next conformation
    count += fitness of c
end while
return c
```

Figure 3.5.: Implementation of the roulette wheel technique

Acceptance criteria

The two parental structures S_{p_1} and S_{p_2} have the average energy $E_{ave} = (E_{p_1} + E_{p_2}/2)$. If a valid structure S_c is found, its energy E_c is evaluated and compared to E_{ave}. If $E_c \leq E_{ave}$, the new structure is accepted. Otherwise, the increased energy will be accepted if:

$$Rnd < \exp\left[\frac{E_{ave} - E_c}{c_k}\right]$$

where *Rnd* is a random number between 0 and 1. If the resulting structure was not accepted, the crossover step is repeated with the parents being swapped.

The cooling scheme for crossover starts with $c_k = 0.3$ and is cooled by $c_k = 0.99c_k$ every 5 generations.

4. Experimental Results

4.1. Criteria of Comparison

This chapter describes the results from our algorithm on a number of HP benchmark sequences. The details of the sequences are given in Table 4.1. These sequences have already been used as benchmark by various algorithms for protein folding simulations in the 2D HP model. E_{min} describes the best-known maximum number of H-H contacts found for the corresponding sequence as reported in Unger and Moult, 1992 [27], and other, more recently published approaches [20, 2]. For some of the sequences, e.g. the 60-mer sequence, the putative ground energy was found by the pruned-enriched Rosenbluth method (PERM)[1], a chain growth approach based on the Rosenbluth-Rosenbluth (RR) method, whereby the variation introduced in [13] seems to exceed previous versions [25, 2].

Length[a]	Sequence	E_{min} [b]
20	$(HP)_2PH(HP)_2(PH)_2HP(PH)_2$	-9
24	$H_2P_2(HP_2)_6H_2$	-9
25	$P_2HP_2(H_2P_4)_3H_2$	-8
36	$P(P_2H_2)_2P_5H_5(H_2P_2)_2P_2H(HP_2)_2$	-14
48	$P_2H(P_2H_2)_2P_5H_{10}P_6(H_2P_2)_2HP_2H_5$	-23
50	$H_2(PH)_3PH_4PH(P_3H)_2P_4(HP_3)_2HPH_4(PH)_3PH_2$	-21
60	$P(PH_3)_2H_5P_3H_{10}PHP_3H_{12}P_4H_6PH_2PHP$	-36
64	$H_{12}(PH)_2((P_2H_2)_2P_2H)_3(PH)_2H_{11}$	-42
85	$H_4P_4H_{12}P_6(H_{12}P_3)_3HP_2(H_2P_2)_2HPH$	-53
100 (a)	$P(P_2H_2)_2H_2P_2H_3(PH_2)_3H_2P_8(H_6P_2)_2P_7H(PH_2)_2H_9P_2H(H_2P)_2HP(PH)_2H_2P_6H_3$	-50
100 (b)	$P_5(PH)_2HP_5H_3PH_5PH_2P_2(P_2H_2)_2(PH_5)_2H_5(PH_2)_2H_5P_{11}H_7P(PH)_2H_2P_5(PH)_2H$	-48

[a] The length denotes the number of amino acid residues of the sequence

[b] The putative ground energies are taken from literature sources [27, 1, 20]

Table 4.1.: Benchmark HP sequences

Since not for all sequences the optimal energy can be achieved, the best energy found is one criterion for a comparison between different methods. As proposed by Unger and Moult [27], the number of energy evaluations needed to find the lowest energy conformations is a further main factor to be compared. The energy is calculated once for each valid conformation, i.e. after applying the mutation and crossover operators. Thus, this number indicates how many valid conformations were scanned before the lowest energy conformation was found.[1]

4.2. Performance on Benchmark Sequences

Our algorithm was implemented in Java, and run on a computer with a 1.73GHz dual-core processor and 0.98RAM under Windows XP.

Table 4.2 gives an overview of the results we obtained on the benchmark sequences of length 20 - 64 with the two versions of our algorithm: The genetic algorithm with a population initialized as random coils (RC) and the one with fully extended structures (ES). As comparison, the results from Unger and Moult are given as well. The algorithms were run 5 times (Unger and Moult) or 10 times (RC, ES), respectively, per sequence with a population size of 200. For the most efficient run, the lowest energy value achieved and the number of energy evaluations is reported.

The results show that though the RC algorithm required more energy evaluations for some of the shorter sequences, the overall performance of this algorithm was better compared to the two other algorithms. For the sequences of length 48, 60 and 64, it found lower energy values than the Unger-Moult algorithm.

The ES algorithm reached the same energies for the sequences of length ≤ 50 as the two other algorithms. With a population size of 1000, it performed even better for the 48-mer sequence than the Unger-Moult algorithm. Though this algorithm found the optimal energies mostly faster for the shorter sequences, its performance was not as good on the two longest sequences.

4.3. Comparison of Various Methods

In addition to the RC algorithm and the Unger-Moult algorithm (GA), table 4.3 gives the results of several other algorithms for the 2D HP Protein Folding Problem, namely, the Evolutionary Monte Carlo Algorithm (EMC), 2001 [20], the Genetic Algorithm combined with

[1] Shmygelska and Hoos [25] found CPU times a more significant factor for comparison. They argue that runtime spent for backtracking and the checking of partial or infeasible conformations is not accounted for. CPU times are however dependent on the machine and sensitive to disturbances. As they are moreover not available for the most algorithms, the other two factors are kept for comparison.

Length	E_{min}	Random Coils[a]	Extended Structures[b]	Unger-Moult GA [27]
20	**-9**	**-9** (26,098)	**-9** (13,780)	**-9** (30,492)
24	**-9**	**-9** (39,627)	**-9** (22,986)	**-9** (30,491)
25	**-8**	**-8** (43,211)	**-8** (17,810)	**-8** (20,400)
36	**-14**	**-14** (61,625)	**-14** (63,086)	**-14** (301,339)
48	**-23**	**-23** (96,636)	**-23** (693,409) [c]	-22 (126,547)
50	**-21**	**-21** (101,100)	**-21** (93,757)	**-21** (592,887)
60	**-36**	-35 (124,367)	-33 (67,916)	-34 (208,781)
64	**-42**	-39 (153,667)	-35 (181,278)	-37 (187,393)

Unless otherwise noted, all three algorithms were run with a population size of 200. For the mutation stage the cooling scheme starts with $c_k = 2$ and is cooled by $c_k = 0.97$ every 5 generations. The crossover stage starts with $c_k = 0.3$ and is cooled by $c_k = 0.99$ every 5 generations. The GA algorithm from Unger and Moult [27] was run 5 times for each sequence, our two simulations were run 10 times. For the most efficient run, the lowest energy value achieved and the number of energy evaluations is reported. The energy values shown in bold face correspond to currently best known solution qualities.

[a] Population starts with random structures

[b] The initial populations consists of fully extended structures

[c] The programme was run with a population size of 1000

Table 4.2.: Results of the two versions of our algorithm in comparison with the Unger-Moult genetic algorithm

Tabu search (GTS), 2003 [14], the Protein Folding Genetic Algorithm (PFGA), 2005 [2], and the Ant Colony Optimization Algorithm (ACO), 2005 [25]. All these algorithms are based on the energy function of the standard HP model as introduced in the previous chapter.

Our algorithm reached the same best energy values as the EMC and the GTS algorithm for all tested sequences up to the length of 64. The EMC algorithm found the optimal energies faster for the shorter sequences, but more valid conformations had to be scanned until the best energy conformations for the sequences 48, 60 and 64 were found, namely 165,791, 203,729 and 564,809 [20][2]. Accordingly, all three algorithms yielded better results for the 48-mer, 60-mer and 85-mer sequences than the genetic algorithm of Unger and Moult.

Both the PFGA and the ACO algorithm found optimal energy conformations for the benchmark sequences up to the length of 85. For sequence 100 (a) they got the same result. In addition, the PFGA also obtained optimal results for the second 100-mer sequence. The ACO performed slightly worse on this sequence.

[2] In GTS, the quality of performance was only given in terms of iterations. Due to different population sizes, a direct comparison is difficult.

Length	E_{min}	RC[a]	GA[27]	EMC[20]	GTS[14]	PFGA[2]	ACO[25]
20	**-9**	**-9**	**-9**	**-9**	**-9**	**-9**	**-9**
24	**-9**	**-9**	**-9**	**-9**	**-9**	**-9**	**-9**
25	**-8**	**-8**	**-8**	**-8**	**-8**	**-8**	**-8**
36	**-14**	**-14**	**-14**	**-14**	**-14**	**-14**	**-14**
48	**-23**	**-23**	-22	**-23**	**-23**	**-23**	**-23**
50	**-21**	**-21**	**-21**	**-21**	**-21**	**-21**	**-21**
60	**-36**	-35	-34	-35	-35	**-36**	**-36**
64	**-42**	-39	-37	-39	-39	**-42**	**-42**
85	**-53**	-48				**-53**	**-53**
100 (a)	**-50**					-49	49
100 (b)	**-48**					**-48**	47

Missing entries indicate cases where the respective method has not been tested on a given instance. The energy values shown in bold face correspond to currently best known solution qualities.

[a] Since the optimal energies were not reached for the sequences of length 60, 64 and 85, we refrained from testing the two 100-mer sequences

Table 4.3.: Performance comparison of various algorithms for the 2D HP Protein Folding Problem. The results from the RC algorithm, the Unger-Moult algorithm (GA) [27], Evolutionary Monte Carlo Algorithm (EMC) [20], the Genetic Algorithm combined with Tabu search (GTS) [14], the Protein Folding Genetic Algorithm (PFGA) [2], and the Ant Colony Optimization Algorithm (ACO) [25] are compared.

5. Discussion

Our algorithm that starts with fully extended structures (ES) performs worse than the algorithm of Unger and Moult for no apparent reason. Since the ES algorithm was no direct copy of the Unger-Moult algorithm but was implemented according to the description from paper, it has probably slight differences that leaded to a different performance. As the ES algorithm found the optimal energy values either fast or not at all, it is to assume that it tends to drop in local minima.

In contrast, the RC algorithm is initialized with random conformations which leads to more diversity among individuals. Presumably this causes a better performance on longer sequences than the algorithm of Unger and Moult. Due to this change even the same best energy values are reached as for the EMC algorithm [20] and the GTS [14] algorithm. Surprisingly, as these algorithms seem to create the initial population as well randomly, though it is not explicitly mentioned in [20]. In addition, however, they applied more sophisticated operators, selection procedures and other optimization methods including an additional exchange operator (EMC) and a crossover operator that borrows the idea of tabu search (GTS).

We therefore assumed that minor optimizations such as other variations of mutation and crossover or different acceptance criteria for new structures would not significantly improve our results, and rather concentrated on secondary structures which were used in the PFGA algorithm of Bui and Sundarraj [2].

Already in [20] the usage of secondary structures for the EMC algorithm was proposed which improved their former results. They were the first to find the optimal energy for sequence 64, namely -42[1]. Secondary structures could also improve the results of the 48-mer and the 85-mer sequence. Bui and Sundarraj [2] picked up this idea and developed it further. As can be seen is the results above, PFGA is a quite efficient algorithm. It uses a secondary structure library, that the algorithm evolves, as building blocks for the conformation. Since the usage of secondary structures seems to be promising, we will further elaborate on that in the next chapter.

[1] Since PERM had as well trouble to find this energy value at first, the authors argued that this was due to the lack of a folding centre in that protein [1]. In 2003 [13] however, they succeeded when they presented a more efficient variation of PERM.

6. The Use of Secondary Structures

6.1. Preliminaries and Objectives

6.1.1. Pathways in Protein Folding

Denatured proteins can refold from their random disordered state into a well-defined unique structure, in which the biological activity is completely restored. Thereby, protein sequences fold into a unique native state within seconds. As pointed out by Levinthal [19] and Wetlauer [29], the number of possible conformations of a polypeptide chain is too large to involve an exhaustive search because proteins fold by at least tens of orders of magnitude too fast. This apparent contradiction leads to the *Levinthal paradox*: How can a protein find a globally optimal state without a globally exhaustive search? Consequently, the protein folding problem is a question of what is the physical basis of *cooperativity* by which proteins avoid exhaustive searching of conformational space [8]. So, the protein folds to its native state according to a relatively small number of *pathways*, which means that it folds by a specific sequence of molecular events from the unfolded random coil to a uniquely folded metastable state [19, 5]. To understand the mechanisms of protein folding, it is crucial to characterize the structures of folding intermediates.

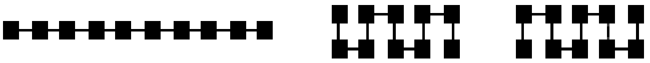
The cooperativity is mainly driven by two types of interactions: First, the secondary structure, e.g. the helix, is found by *local* processes by which each individual tetrapeptide in the sequence finds a hydrogen-bonded helical conformation, and second, the *non-local* interactions by which a compact hydrophobic core is formed [20]. However, there is controversy, whether the secondary structure of a protein forms before the growth of the hydrophobic core, as was postulated by the framework model [15] or whether or the hydrophobic residues collapse to form a compact unfolded state or a molten globule, on which the secondary structure grows, as was postulated by the hydrophobic collapse model [8].

6.1.2. Computational Implementations

Motivated by the observations of cooperativity, Liang and Wong, 2001 [20], employed secondary structures in exploring the conformation space of a protein to speed up their simulation algorithm. The idea was then further developed by Bui and Sundarraj, 2005 [2], in their Protein Folding Genetic Algorithm (PFGA).

Secondary structure are local regularly occurring structures in proteins and are mainly formed through hydrogen bonds between backbone atoms. There are three types of backbone conformations: α-helices, β-sheets and loops. α-helices and β-sheets are preferably located at the core of the protein, whereas loops are rather found in outer regions.

In the 2D HP-model, a secondary structure is a conformation of a sub-sequence consisting only of hydrophobic residues. Liang and Wong [20] chose three possible secondary structures folded by a sub-sequence that correspond to β and α structures of a real protein.

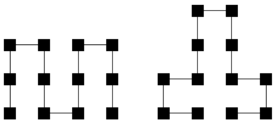


(a)

Figure 6.1.: The secondary structures of the EMC algorithm: An extended sheet and two helices with different directions.

Not to chose one secondary structure over another, a secondary structure library for the longest subsequence of hydrophobic residues is created in [2]. Due to the number of possible secondary structures even for a small subsequence of hydrophobic residues, they used a genetic algorithm to systematically evolve the secondary structures which are then used as building blocks to evolve the best conformation for the given input sequence. Additionally, they put two constraints on the conformations: The secondary structures of the hydrophobic sub-sequences are required to be symmetric to either one of the two lattice axes, and the end lattice sites should have at least one unoccupied lattice neighbour.

The secondary structures chosen for specific sub-sequences are maintained throughout the algorithm and thus are not modified by the genetic operators. In both EMC algorithms and PFGA, the use of secondary structures improved the performance significantly.



(b)

Figure 6.1.: Secondary structures of PFGA: Two conformations for the hydrophobic subsequence HHHHHHHHHHHH.

6.1.3. Objectives

Inspired by these two promising algorithms, we decided as well to employ secondary structures in our RC algorithm. However, not with the intent to improve our results but to shed light on procedure of the algorithm if some sub-sequences are forced to retain the structure assigned to them. What sub-sequences are ideally to be given a specified structure? Does it make sense to assign a structure to hydrophilic amino acids? How do conformations in general change throughout the algorithm. The modified version of the RC algorithm is meant to provide insight into both the general behaviour of the genetic algorithm, and the influence of secondary structures on that algorithm.

The implementation of secondary structures required to change certain parts of the algorithm which is explained in detail in the next section. Moreover, we assumed that a graphical implementation of the genetic algorithm is most suitable for our requirements that are rather of experimental nature and thus ought to give a means of observations. For the same reason, the length and number of sub-sequences and the favoured structure are user-driven. There is neither a restriction on what kind of amino acid sub-sequence (hydrophobic or polar) to choose nor its position within the complete sequence. Finally, the implementation should allow to observe the development of the conformations across the generations.

6.2. Changes to the Algorithm

As the changes to the algorithm are not meant to improve results, but rather provide the user a means of experiment and demonstration, sub-sequences can be selected manually and are not optimized by an algorithm.

Figure 6.2 shows the three possible formations that can be selected as sub-sequences for a certain range of arbitrarily combined polar and hydrophobic amino acids, namely a straight line, a β-sheet and a single fold. Several ranges can be chosen, whereby also the different kinds of sub-structures can be combined. The size or position of the ranges is irrelevant as long as the different ranges do not overlap. Thus, though not quite meaningful, it is theoretically possible to select the complete sequence and assigned a certain structure to it.

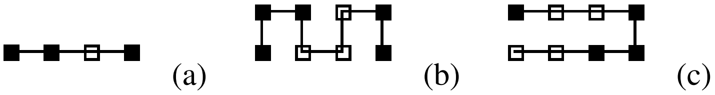


Figure 6.2.: The three formations that can be chosen as *locked* sub-sequences for an arbitrary range of amino acids: A straight line (a), a β-sheet (b) and a single fold (c)

All individuals of the population are initialized with these sub-sequences. The selected ranges are then *locked* which means that they will keep this form for the course of the algorithm. Since the individuals experience a change in their structure only by mutation and crossover, the operators are subject to the following changes.

Mutation

When the point of mutation is randomly selected and this point happens to be in a locked range, another point is randomly chosen until it falls outside the locked sub-sequences.

Crossover

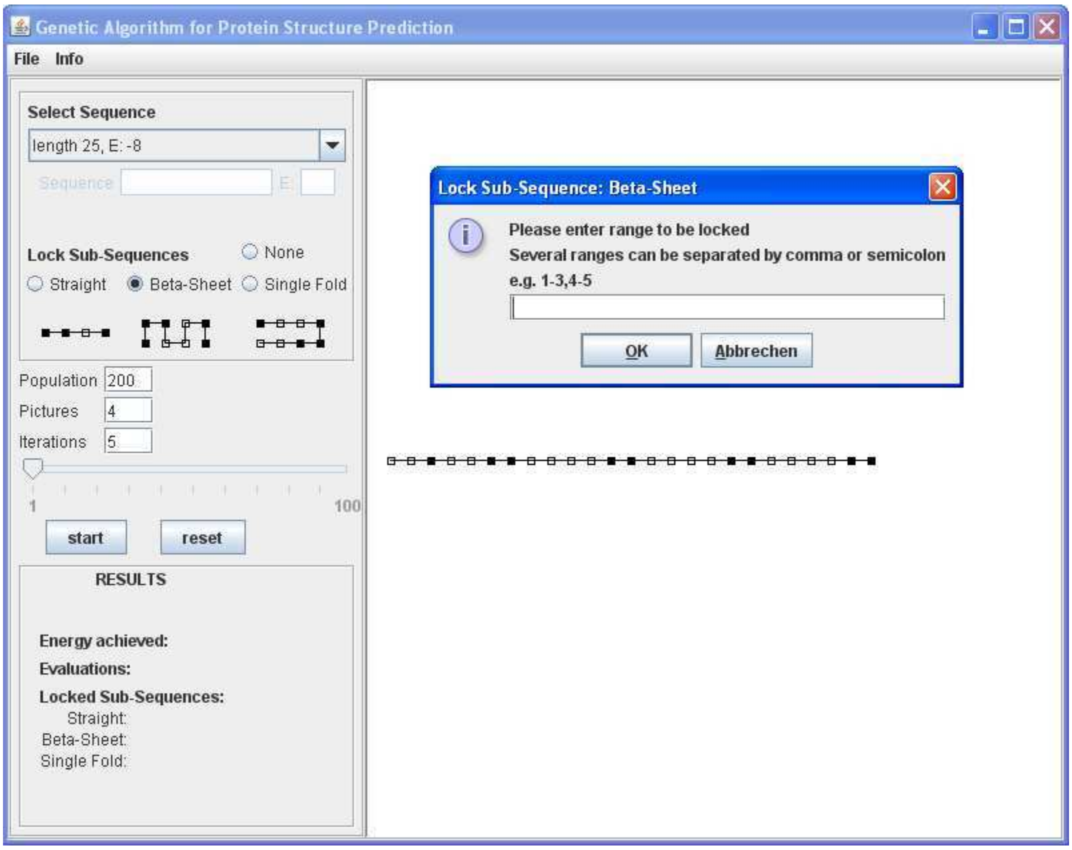
Since all individuals, including the two selected parents, are locked at the same points, two structures can still be recombined even if the cutpoint happens to be within a locked subsequence. However, in this case the second part is not turned to ensure that the newly created individual contains the respective sub-formation.

6.3. Graphical Implementation

With intent to provide an implementation that is suitable for user interactions, we decided for a graphical implementation as illustrated in 6.3. The user is provided a number of benchmark sequences with the corresponding graphical representation to select from. Moreover, an arbitrary sequence can be defined by the user. As mentioned above, the program then allows to determine sub-sequences to be locked, that means to maintain a certain structure, namely a straight line, a β-sheet and a single fold. If no sub-sequence is locked, the program is executed with the original RC algorithm.

In addition to the size of the population and the number of iterations to be performed, the user can also select how many individuals of a population should be displayed as graphical representation. Though this value can still be varied after running the algorithm.

We decided not to include modifiable cooling parameters for the genetic operators in the graphical implementation as the programme is meant to be handled as intuitively as possible. So the cooling parameters are set to the following values which are the same as used for testing: For the mutation stage the cooling scheme starts with $c_k = 2$ and is cooled by $c_k = 0.97$ every 5 generations. The crossover stage starts with $c_k = 0.3$ and is cooled by $c_k = 0.99$ every 5 generations.



Select Sequence: length 25, E: -8

Sequence
E:

Lock Sub-Sequences
○ None
○ Straight ● Beta-Sheet ○ Single Fold

Population 200

Pictures 4

Iterations 5

RESULTS

Energy achieved:

Evaluations:

Locked Sub-Sequences:
Straight:
Beta-Sheet:
Single Fold:

Lock Sub-Sequence: Beta-Sheet

Please enter range to be locked
Several ranges can be separated by comma or semicolon
e.g. 1-3,4-5

Figure 6.3.: The GUI implementation of the genetic algorithm which allows to lock sub-sequences

After running the algorithm

When the algorithm stops, either due to an optimal energy conformation found or the specified number of iterations reached, the graphical representations of the last generation conformations are displayed. The best conformation reached is always the one in the upper left corner. If any sub-sequences had been locked, the corresponding amino acid residues will be marked in blue. Figure 6.4 shows six possible structures of the 20-mer benchmark sequence after the algorithm found an optimal energy conformation (a) in the seventeenth iteration. The amino acid residues 15 to 21 had been locked as β-sheet sub-sequence which are marked in blue. In that case, all locked residues are hydrophobic.

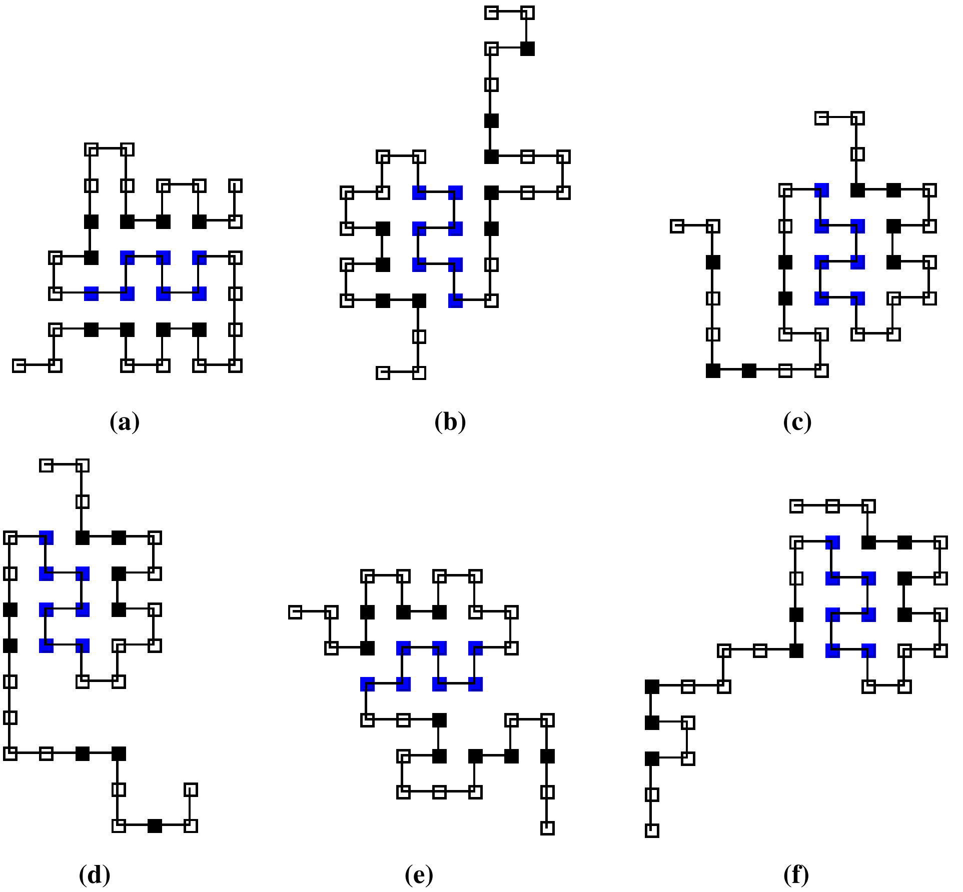
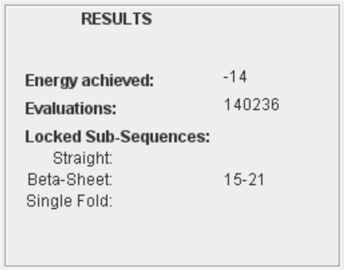


Figure 6.4.: Six possible structures of a 20 amino acids-long sequence in the 17th generation. The amino acid residues 15 to 21 had been locked as β-sheet sub-sequence. These seven amino acids, which here are all hydrophobic, are marked in blue. Structure (a) has an optimal energy of -14

Furthermore, the results of the run are reported in a field. The best energy value achieved and the number of valid conformations scanned until the best conformation was found are displayed. In addition, all locked sequences and their chosen form are summarized in the same field. Figure 6.5 gives the results for the structure of Figure 6.4 (a), which has the optimal energy -14 and required 140,236 energy evaluations.



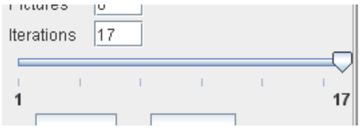
RESULTS

Energy achieved: -14
Evaluations: 140236
Locked Sub-Sequences:
Straight:
Beta-Sheet: 15-21
Single Fold:

Figure 6.5.: Results in the GUI implementation

Reviewing former generations

Another useful feature of the program is, that the individuals of all generations can be looked at. A slider allows to select the desired generation, and the respective conformations are displayed. To facilitate displaying an exact generation, the desired iteration can additionally be entered in the field *iterations* which then adjusts the slider.



Iterations 17

Figure 6.6.: Slider to switch between the generations

Passing through the generations, the development of the conformations during the course of the algorithm can be seen. The shape of the structures becomes more and more compact structure. Moreover, observations about the diversity among the individuals as well as distribution of hydrophobic and hydrophilic amino acids can be made.

7. Conclusion

In this thesis, we introduced a hybrid genetic algorithm for the protein folding problem in the 2D HP model, which outperforms the original algorithm of Unger and Moult, and is competitive with similar algorithms for the 2D HP Protein Folding Problem. Additionally, we provided the algorithm with a graphical user interface that allows to specify secondary structures for sub-sequences.

Nevertheless, we would like to optimize our genetic algorithm to improve the results for longer sequences, and achieve also optimal energy conformations for these sequences. The use of secondary structures would be one idea of achieving better energy values and reducing CPU time.

Another work of interest is the graphical version of the programme, which we would like to equip with additional features that give the user more options for executing the algorithm.

These include for example a broader range of secondary structures to choose from, as well as randomly created secondary structures or a user-defined ones. Certainly, an optimized selection of secondary structures, e.g. selecting the longest hydrophobic sub-sequence and create an structure optimized by a genetic algorithm as proposed in [2], could offer a means of comparison and demonstration.

Moreover, another feature in the programme could provide alternatives for the current genetic operators to see how these affect the conformations as such, and the diversity of the population. There are for instance other selection procedures, such as rank selection or tournament selection, and other mutation operators as the three-bead flip, crankshaft moves and rigid rotations.

Thereby, a variable ratio of crossover, mutation, and conformations, that are copied intact to the next generation, would lead to additional flexibility.

Acknowledgements

First, I would like to thank Prof. Dr. Volker Sperschneider for his extraordinary support throughout the thesis. Second, special thanks to Jun. Prof. Dr. Sigrid Knust for her second supervision. I am also very grateful to Lena Scheubert for her valuable suggestions and help.

Special thanks also to my family, especially to my brother Christoph and my mother for testing my programme and for their constructive tips improving it.

Lastly, I would like to thank my finacé Adnan Ghori for always supporting me throughout my thesis and for just being there.

Appendix

A. Best Energy Conformations

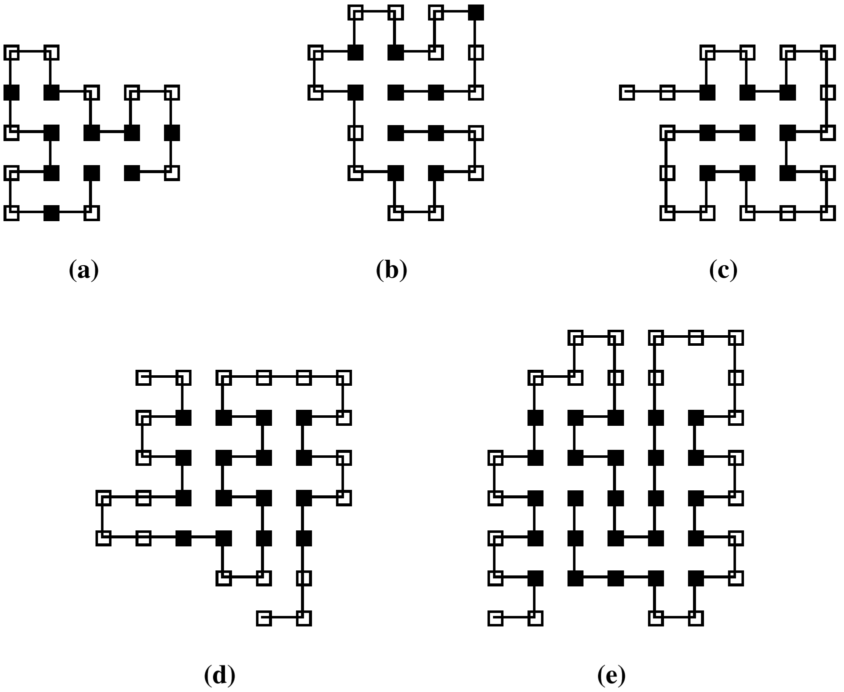
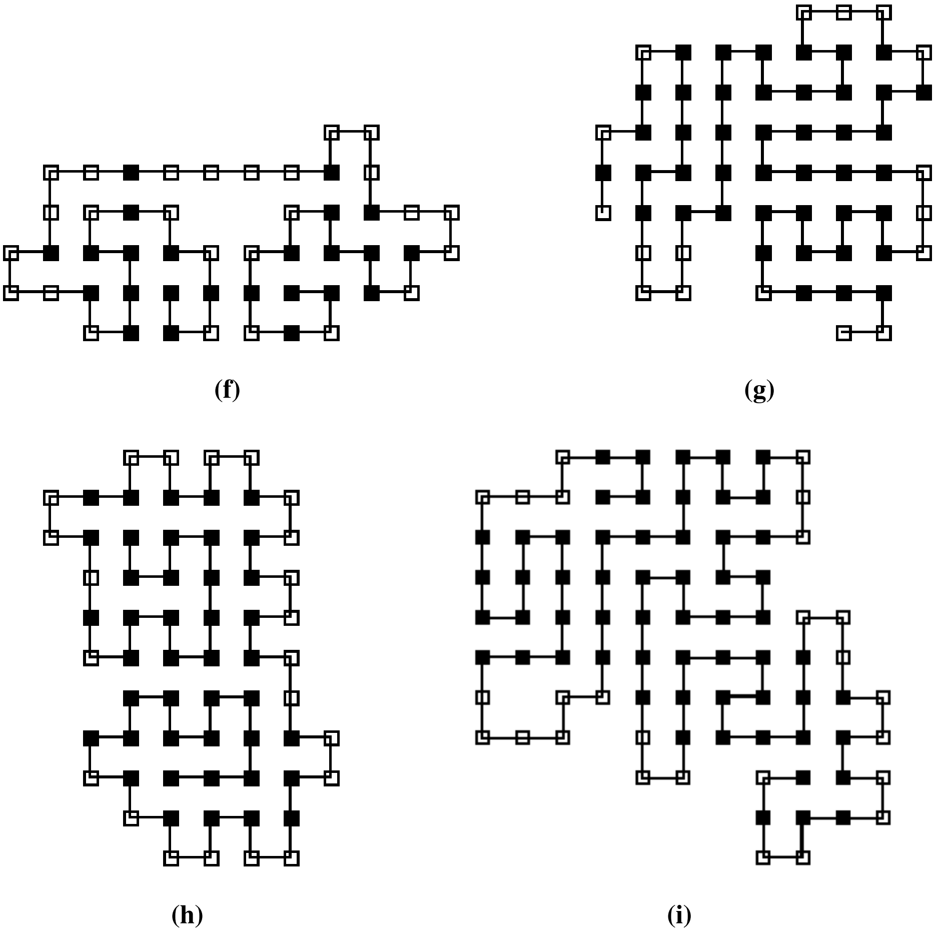


Figure 1.: Best energy conformations achieved for the benchmark sequences
(a) $(HP)_2PH(HP)_2(PH)_2HP(PH)_2$, length 20 and energy -9,
(b) $H_2P_2(HP_2)_6H_2$, length 24 and energy -9,
(c) $P_2HP_2(H_2P_4)_3H_2$, length 25 and energy -8,
(d) $P(P_2H_2)_2P_5H_5(H_2P_2)_2P_2H(HP_2)_2$, length 36 and energy -14, and
(e) $P_2H(P_2H_2)_2P_5H_{10}P_6(H_2P_2)_2HP_2H_5$, length 48 and energy -23



(f)

(g)

(h)

(i)

Figure 1.: Best energy conformations achieved for the benchmark sequences
(f) $H_2(PH)_3PH_4PH(P_3H)_2P_4(HP_3)_2HPH_4(PH)_3PH_2$, length 50 and energy -21,
(g) $P(PH_3)_2H_5P_3H_{10}PHP_3H_{12}P_4H_6PH_2PHP$, length 60 and energy -35,
(h) $H_{12}(PH)_2((P_2H_2)_2P_2H)_3(PH)_2H_{11}$, length 64 and energy -42, and
(i) $H_4P_4H_{12}P_6(H_{12}P_3)_3HP_2(H_2P_2)_2HPH$, length 85 and energy -48

Bibliography

[1] Ugo Bastolla, Helge Frauenkron, Erwin Gerstner, Peter Grassberger, and Walter Nadler. Testing a New Monte Carlo Algorithm for Protein Folding. In *Proteins: Structure, Function, and Bioinformatics*, pages 52–66. Wiley, 1998.

[2] Thang N. Bui and Gnanasekaran Sundarraj. An Efficient Genetic Algorithm for Predicting Protein Tertiary Structure in the 2D HP model. In *Genetic And Evolutionary Computation Conference*, pages 385–392. ACM, 2005.

[3] Peter Clote and Rolf Backofen. *Computational Molecular Biology: An Introduction*, pages 228–234. Wiley, 2000.

[4] Fred E. Cohen and Jeffery W. Kelly. Therapeutic approaches to proteinmisfolding diseases. *Nature*, 426:905–909, 2003.

[5] Thomas E. Creighton. Experimental Studies of Protein Folding and Unfolding. *Progress in Biophysics and Molecular Biology*, 33:231–297, 1978.

[6] Thomas E. Creighton. Protein Folding. *Biochemistry Journal*, 270:1–16, 1990.

[7] Pierluigi Crescenzi, Deborah Goldman, Christos Papadimitriou, Antonio Piccolboni, and Mihalis Yannakakis. On the Complexity of Protein Folding. *Journal of Computational Biology*, 5(3), 1998.

[8] Ken A. Dill, Klaus M. Fiebig, and Hue Sun Chan. Cooperativity in protein-folding kinetics. *Proceedings of the National Academy of Sciences USA*, 90:1942–1946, 1993.

[9] Gerald D. Fasman, editor. *Prediction of Protein Structure and the Principles of Protein Conformation*. Plenum Press, New York, 1989.

[10] Peter Grassberger. Pruned-enriched Rosenbluth method: Simulations of θ polymers of chain length up to 1 000 000. *Physical Review E*, 56(3), 1997.

[11] John H. Holland. Outline for a Logical Theory of Adaptive Systems. *Journal of the ACM*, 9(3):297–314, 1962.

[12] John H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. MIT Press, reprint edition, 1992 (originally published in 1975).

[13] Hsiao-Ping Hsu, Vishal Mehra, Walter Nadler, and Peter Grassberger. Growth algorithms for lattice heteropolymers at low temperatures. *Journal of Chemical Physics*, 118(1):444–451, 2003.

[14] Tianzi Jianga, Qinghua Cui, Guihua Shi, and Songde Ma. Protein folding simulations of the hydrophobic- hydrophilic model by combining tabu search with genetic algorithms. *Journal of Chemical Physics*, 119(8):4592–4596, 2003.

[15] Peter S. Kim and Robert L. Baldwin. Intermediates in the Folding Reactions of Small Proteins. *Annual Review of Biochemistry*, 59:631–660, 1990.

[16] Kit Fun Lau and Ken A. Dill. A Lattice Statistical Mechanics Model of the Conformational and Sequence Spaces of Proteins. *Macromolecules*, 22:3986–3997, 1989.

[17] Kit Fun Lau and Ken A. Dill. Dominant Forces in Protein Folding. *Biochemistry*, 29(31):7133–7155, 1990.

[18] Arthur M. Lesk. *Bioinformatik. Eine Einführung*. Spektrum, 2003.

[19] Cyrus Levinthal. Are there Pathways for Protein Folding? *Journal de Chimie Physique et de Physico-Chimie Biologique*, 65(1):44, 1869.

[20] Faming Liang and Wing Hung Wong. Evolutionary Monte Carlo for protein folding simulations. *Journal of Chemical Physics*, 115(7), 2001.

[21] Tom M. Mitchell. *Machine Learning*, chapter Genetic Algorithms. McGraw-Hill, 1997.

[22] R. Ramakrishnan, Bala Ramachandran, and J. F. Pekny. A dynamic Monte Carlo algorithm for exploration of dense conformational spaces in heteropolymers. *Chemical Physics*, 106(6):2418–2425, 1997.

[23] Frederic M. Richards. Areas, volumes, packing, and protein structures. *Annual Review of Biophysics and Bioengineering*, 6:151–176, 1977.

[24] Alena Shmygelska and Holger H. Hoos. An Improved Ant Colony Optimisation Algorithm for the 2D HP Protein Folding Problem. In Springer Verlag, editor, *In Proceedings of the 16th Canadian Conference on Artificial Intelligence*, pages 400–417, 2003.

[25] Alena Shmygelska and Holger H. Hoos. An ant colony optimisation algorithm for the 2D and 3D hydrophobic polar protein folding problem. *BMC Bioinformatics*, 6(30), 2005.

[26] Ruth S. Spolar, Jeung-Hoi Ha, and M. Thomas Record. Hydrophobic effect in protein folding and other noncovalent processes involving proteins. *Proceedings of the National Academy of Sciences USA*, 86:8382–8385, 1989.

[27] Ron Unger and John Moult. Genetic Algorithms for Protein Folding Simulations. *Journal of Molecular Biology*, 231:75–81, 1992.

[28] Ron Unger and John Moult. Finding the lowest free energy conformation of a protein is an NP-hard problem: Proof and implications. *Bulletin of Mathematical Biology*, 55(6):1183–1198, 1993.

[29] Donal B. Wetlaufer. Nucleation, Rapid Folding, and Globular Intrachain Regions in Proteins. *Proceedings of the National Academy of Sciences USA*, 70(3):697–701, 1973.