Memorial
University of Newfoundland

A Genetic Algorithm for Protein Structure Prediction Using the 2D-HP-Lattice Model

Project Report

Evolutionary Computation COMP 6776

Dr. Ting Hu

Birgit Kuehbacher and Aida Ghayour-Khiavi

April 20, 2018

Contents

Abstract

The prediction of a protein's tertiary structure only with knowledge of its primary structure is of high importance in the field of molecular biology. However, finding such a global minimum can be extremely laborious. In fact, the protein structure prediction (PSP) problem is considered NP-hard. [1] This course project proposes a genetic algorithm (GA) for predicting the folded state of the protein. First, a brief overview is given over protein structure. Then, the problems suitability for evolutionary computing methods is discussed. The GA design is laid out and a novel combination of semi-systematic crossover and λ-mutation is to explore and exploit the search space. In the last section, the results of our approach are compared to previous PSP algorithms.

1 Introduction

Proteins perform many essential functions in living organism [2]. Being able to predict their structure and therefore their function can yield great benefits for understanding how human bodies work. Furthermore, knowing a protein's folded state can be very useful for gene therapy and drug designing as most of the drug-able targets in the human body are proteins. In addition, protein structure prediction also helps scientists design proteins for specific tasks. [3] In the field of renewable energy, researches hope recreate a cells energy metabolism and understanding what structures are involved is essential. [4] However, experimental assessment of protein structures is difficult and expensive [5]. Even though more than 50 years of research have been dedicated to PSP, researchers are still not able to reliably predict a predict a proteins 3-dimensional structure with knowing only its amino acid sequence. Therefore, a model-based approach is needed. Evolutionary methods are widely employed in the field of molecular biology. Their stochastic and population based approach especially lends itself for PSP as it is an NP-hard problem. Section 2 gives a quick overview over protein folding, PSP's suitability for evolutionary computing, and the HP-model, and Section 3 describes the design of the GA proposed in this project. Section 4 describes the results of our GA and Section 5 compares them to other genetic algorithms for solving PSP.

2 Protein Folding

A protein's function is determined by its structure. Therefore, in order to be able to accurately predict a protein's function, we must first know its 3-dimensional structure. Protein structure is divided into four hierarchical layers [6]. Primary structure describes the amino acid sequence of the polypeptide chain. An amino acid that is part of a polypeptide chain is referred to as an amino acid residue. Secondary structure refers to a local conformation of the polypeptide chain. The most common secondary structure types are α-helices and β-sheets. Tertiary protein structure describes the overall 3-dimensional arrangement of the polypeptide chain. If a protein consists of more than one polypeptide chain, the arrangement of the multiple chains defines the quaternary structure of the protein. The process of a polypeptide chain arranging itself in its folded state in a 3-dimensional space is called protein folding. There are many different interactions between amino acids that influence protein folding [5]. There can be non-covalent interactions such as van der Waals or electrostatic (ionic) interactions, hydrogen bonding, and hydrophobic interactions, as well as covalent bonds, e.g. disulfide bonds between cysteine residues. Furthermore, steric hinderances between the molecules, in particular the amino acid side chain, also contribute to the 3-dimensional arrangement of the polypeptide chain. Even little changes in the tertiary structure may cause a protein to lose its functions. In 1973, Anfinsen experimentally determined that tertiary protein structure is decided by its primary structure alone, i.e. its amino acid sequence [7].

2.1 The HP-Model

We do know that hydrophobic interactions are the main driving force behind protein folding. Water is a polar solvent and cells provide an aqueous environment for proteins [6]. Any hydrophobic amino acid residues are repulsed by the polar water molecules, while hydrophilic (Greek, “water-loving") amino acid residues form interactions with water molecules. Hence, hydrophobic amino acid residues form a hydrophobic core in the tertiary structures and hydrophilic residues tend to arrange themselves on the outside of the structure to maximize their interactions with water. The hydrophobic core is further stabilized by hydrophobic interactions between the residues.

Since protein folding is too complex to actually model in an algorithm, there are different abstraction models that are used to do PSP [8]. For this project, we are using the 2D-HP-lattice model introduced by [9]. While there are actually 20 different amino acids, the HP-model only distinction that is made between amino acids is whether they are polar (hydrophilic) or hydrophobic (non-polar). Furthermore, the amino acid residues have discrete positions in a 2-dimensional lattice. Even though the 2D-HP-lattice model very much abstracts from real-life protein folding, it is capable of presenting useful characteristics of real amino acid sequence.

2.2 GA Suitablity

The folded state of a protein, also called its native state, is the conformation that contains the minimum amount of internal energy. Proteins actively seek this state. However, it is unfeasible that they attain it by trying all possible conformations. The search space of possible conformations is enormous: An amino acid take on at least six conformations and most proteins consist of at least 60 amino acids. Thus, a protein can take on at least $6^{60} \approx 4.8 \times 10^{46}$ conformations. The Levinthal Paradox further illustrates this problem: If a protein with 4.8×10^{46} possible conformations were to sequentially sample all conformations and were able to sample a million of them in one second, it would still take about $1.5_{\times}10^{33}$ years for it to find the conformation with the lowest energy.

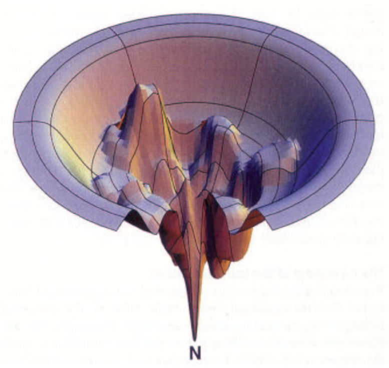


Figure 1: The funnel-shaped energy landscape of protein conformations. The unfolded amino acid sequence with high internal energy is located at the top of the funnel. Intermediate states are biased towards the bottom of the funnel which describes a protein's native state, i.e. the folded conformation with the lowest internal energy. **Source:** [10]

Current research proposes a funnel shaped energy landscape for a protein's conformations with multiple local optima and minima (see Figure 1) [10]. The global minimum at the bottom of the funnel represents the native state of the protein. Thus, each intermediate conformation is biased towards the native conformation. Using the energy landscape as adaptive landscape, the usage of evolutionary computing for solving PSP is relatively intuitive. The individuals as units of selection are different conformations of a protein, and are located at the points in the landscape that correspond to their energy potential. It is the goal, to evolve the population of conformations in such a way that it migrates towards the global minimum of the funnel shaped landscape, i.e. the native state of the protein. Therefore, the fitness function is defined as the energy content of a conformation and we have an optimization problem where we are trying to find the individual with the lowest energy.

3 Methodology and GA Design

We are using the HP-model for representing protein conformations. White beads represent hydrophobic amino acid residues and black ones represent hydrophilic amino acid residues. The sequence is mapped onto a 2-dimensional square lattice. Each point in the chain can turn 90 degrees left or right, or continue straight ahead [5]. One square in the lattice can only contain one bead. An example of a sequence in the 2D-HP-model is shown in Figure 2.

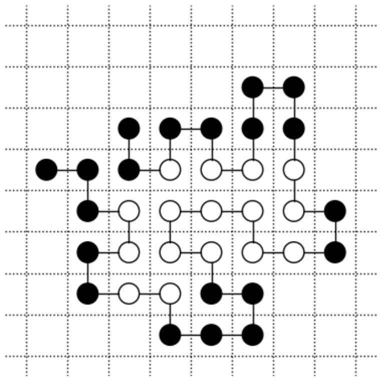


Figure 2: An example of a folded amino acid sequence in the 2D-HP-model. White beads represent hydrophobic amino acid residues, black ones represent polar amino acid residues.

3.1 Representation

An individual consists of an array of numbers which are from the set $\{0, 1, 2\}$. The elements of the array represent the direction of edges between beads: *1 = straight*, *0 = left*, *2 = right*. Orientation at each bead is relative to its position to the previous bead and we start facing east at $(0, 0)$ The position of the second bead is also fixed at $(1, 0)$. Therefore, the first element of the array is always 1 and can be omitted. Hence, an n-amino acid sequence can be described by an array of length $n-2$ with elements from $\{0, 1, 2\}$.

3.2 Initialization

The initial population is generated by doing a self-avoiding walk on the lattice. For each step, a direction is randomly chosen from $\{0, 1, 2\}$. Then, the individual is mapped onto the lattice and it is checked whether there are any clashes, i.e. whether any square is occupied by more than one bead. If there is a clash, we backtrack and choose a different direction until a valid next step is found.

3.3 Fitness

The fitness of an individual is the internal energy of the conformation. The energy ϵ of interactions between amino acids is defined as $\epsilon_{HH} = -1$ for interactions between two hydrophobic amino acids, $\epsilon_{HP} = 0$ for one hydrophobic and one polar amino acid, and $\epsilon_{PP} = 0$ for two polar amino acids [1]. Hence, only hydrophobic interactions contribute to the internal energy. Furthermore, there can only be interactions between adjacent amino acids that are not sequence neighbours. The overall fitness is therefore defined as

$$E = \sum_{i<j} \epsilon_{ij} \Delta_{ij}, \text{where}$$

$$\Delta_{ij} = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are topological neighbours,} \\ & \text{but are not sequence neighbours} \\ 0 & \text{otherwise.} \end{cases}$$

The aim of the the GA is to minimize E.

3.4 Parent and Survivor Selection

Tournament selection is used to select individuals for reproduction. k individuals are chosen randomly from the populations and tournament competitors. The competitors are ranked according to their fitness and the two best ones are chosen to undergo crossover. The resulting two children replace the two worst competitors of the tournament in the population.

In order to preserve the best individual, elitism is used, i.e. the best individual of one generations is always preserved to the next generation.

3.5 Semi-Systematic-Crossover

The semi-systematic crossover is based on the crossover operator introduced in [11]. One-point crossover is applied m times (depending on a crossover rate) to the same pair of parents. Generated children are checked for clashes and should a clash occur, the crossover is redone. If the number of clashes during one crossover exceeds a certain threshold, the parents are simply chosen as offspring. m crossovers result in $2 \times m$ number of offspring which are ranked according to their fitness. The best two children are chosen to replace the two worst tournament competitors in the population.

3.6 λ-Mutation

λ-Mutation was introduced in [12]. In each iteration, λ individuals of the population are chosen at random to undergo mutation. There is no mutation rate, i.e. chosen individuals are going be mutated. The resulting mutant is check for clashes. If there is a clash, the mutant is discarded and the individual undergoes mutation again, until either a valid individual has been created or a clash threshold has been reached. If no valid mutant was created, the individual is put back into the population unmodified.

There are four mutation methods that are performed with different probabilities [8].

3.6.1 In-plane rotation

A mutation point is chosen at random and substring following the mutation point is rotated by ±90 or ±180 degrees around the xy-plane. Depending on the value of the mutation point, in-plane rotation is implemented by replacing 0 with an element from $\{1, 2\}$, 1 with an element from $\{0, 2\}$, or 2 with an element from $\{0, 1\}$ The mutant will only differ from the original individual at one element. Thus, this mutation can keeps most of the adjacency.

3.6.2 Out-of-plane rotation

A mutation point is chosen at random and the following substring is rotated by 180 degrees either around the xz- or yz-plane, depending on the orientation of the mutation point to the x- and y-axis. Out-of-plane rotation is implemented by changing all 0's (left) to 2's (right) and vice versa.

3.6.3 Crank shaft rotation

All crank shaft structures in the individual are recorded and one is chosen at random for mutation. A crank shaft structure can be either $0, 2, 2, 0$ or $2, 0, 0, 2$. The mutation rotates the structure by 180 degrees on either the xz- or yz-plane by changing $0, 2, 2, 0$ to $2, 0, 0, 2$ or vice versa.

3.6.4 Kink motion

Kink motion involves reversing a bend by moving the bead that is part of the kink, from its current location to the diagonal opposite coordinate. The connections to its neighbours remain unchanged. For the implementation, eight types of subsequences must be detected and mutated to the corresponding form. All possible cases are illustrated in Figure 3.

3.7 Termination

The GA terminates either after a specified number of iterations or if the absolute minimum energy has been found.

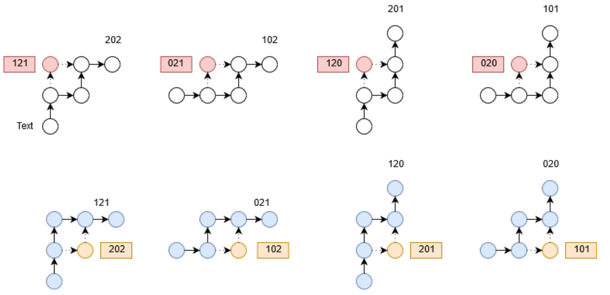


Figure 3: All eight possible mutations when doing kink motion.

A schematic summary of the execution of the GA is described in Figure 4.

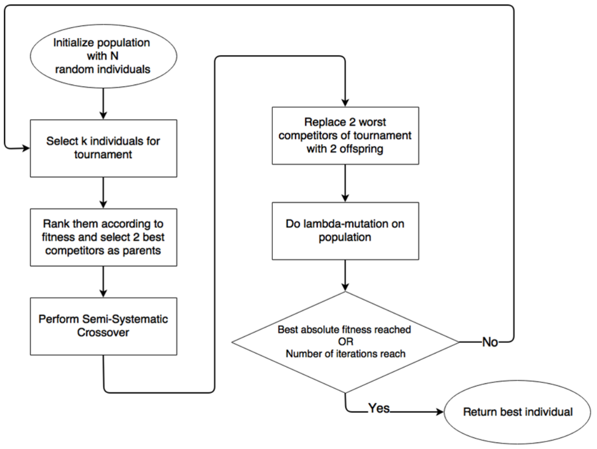

Initialize population with N random individuals
Select k individuals for tournament
Rank them according to fitness and select 2 best competitors as parents
Perform Semi-Systematic Crossover
Replace 2 worst competitors of tournament with 2 offspring
Do lambda-mutation on population
Best absolute fitness reached OR Number of iterations reach
No
Yes
Return best individual

Figure 4: A flowchart describing the execution of our GA.

4 Results

Table 1 shows eight sequences that have been used as benchmark sequences for various PSP algorithms over the last 25 years. Their absolute minima are based on the 2D-HP-lattice model and the fitness function described in Section 3.3. The energy minima of the sequences of lengths 20, 24, and 25 have been determined by exhaustive search, while those of sequence of length ≥ 36 have not been proven conclusively. For example, the minima of the sequences of lengths 50, 60, and 64 have only been discovered by [1] in 2010. We use these sequences to compare our algorithm to others. The measure of an algorithm's performance is quite easily defined as whether or not it can find the absolute energy minimum. If an algorithm is able to determine the absolute minimum, it can be further investigated how often the energy minimum is found and how many iterations it takes for the algorithm to find it.

Length	Energy Minimum	Sequence
20	-9	$HPHPPHHPHPPHPHHP$ $PHPH$
24	-9	$HHPPHPPHPPHPPHPP$ $HPPHPPHH$
25	-8	$PPHPPHH(P)_4HH(P)_4HH$ $(P)_4HH$
36	-14	$PPPHHPPHH(P)_4(H)_4HH$ $PPHH(P)_4HHPPHPP$
48	-23	$PPHPPHHPPHH(P)_5(H)_{10}$ $(P)_6HHPPHHPPHPP(H)_4$
50	-36	$HHPHPHPHP(H)_4PH(P)_3$ $H(P)_3H(P)_4H(P)_3H(P)_3HP$ $(H)_4PHPHPHPHH$
60	-42	$PP(H)_3P(H)_8(P)_3(H)_{10}PH$ $(P)_3(H)_{12}(P)_4(H)_6PHHPHP$
64	-52	$(H)_{12}PHPHPPHHPPHHPP$ $HPPHHPPHHPPHPPHHP$ $PHHPPHPHP(H)_{12}$

Table 1: Eight often used benchmark sequences and their corresponding minimal energy values. The energy minima for the sequences of lengths 20, 24, and 25 have been determined by exhaustive search. Those for sequences of length ≥ 36 are approximate values and have not been proven.

4.1 Rotation Crossover and Pioneer Search

For our project, we tried to combine a variant of the *systematic crossover* from [11] with *λ-mutations* from [12]. While the latter is not an evolutionary algorithm for PSP, the idea of a λ-mutation operator seemed promising as it introduces more diversity into the population. One problem with systematic crossover is, that it is very greedy and tends to get trapped in local optima. In [11], the authors tried to counteract this behaviour by introducing a strategy called *pioneer search*: Every 10 generations, the children of this generation are compared to each individual of the population. If a child is a duplicate of an individual, it is discarded and new offspring is created until it differs from every individual in the population. Our design hopes to circumvent this by *a)* using a semi-systematic crossover which does m crossovers with m randomly chosen crossover points, and *b)* using λ-mutations in each generation. Since the semi-systematic crossover is not testing all possible crossover points and not choosing from all possible children, it is less greedy than the systematic crossover.

However, when testing the algorithm with the HP-20 sequence, which has an energy minimum of -9, the minimal fitness found by the algorithm was -8 and changes of the parameter m for the semi-systematic crossover and

λ did not lead to better results. In order to improve our algorithm, we implemented an additional rotation crossover. When a clash occurs during crossover, the rotation crossover operator does not immediately dismiss the offspring but tries to piece the two fragments together with different orientations. Rotation crossover can be viewed as the normal crossover operator but with additional systematic in-plane mutations if a clash should occur. We also implemented a version of the pioneer search from [11] in order to promote exploration of the search space. Figure 5a shows the mean minimum fitness for combinations of normal crossover, normal crossover with pioneer search, rotation crossover, and rotation crossover with pioneer search and varying λ-values. All four combinations found -8 as minimum but with different frequencies. Figure 5b shows the number of runs out of 30 that found -8 as best fitness for different λ-values. Since the results did not show any significant improvements with rotation crossover or pioneer search and both increase the algorithm's computational costs, we decided to go forward with the original semi-systematic crossover and λ-mutations.

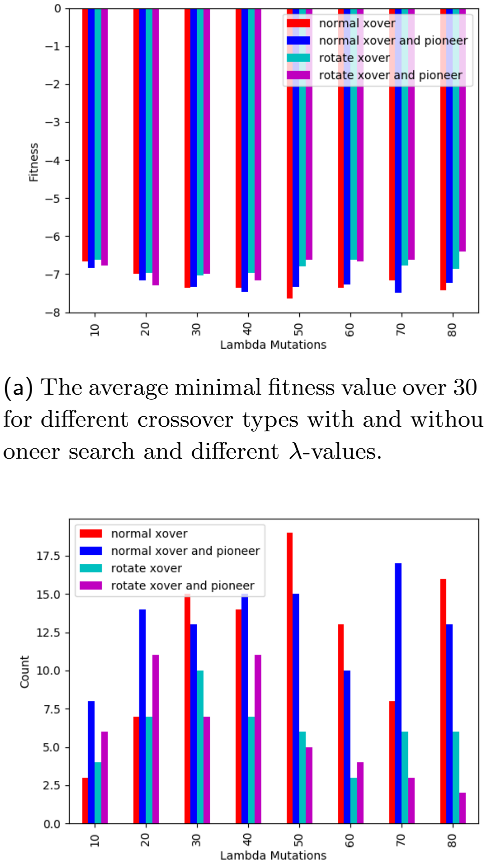
4.2 Population Size and λ-values

The algorithm was run for different combinations of population size and λ-values for the HP-20 sequence. Figure 6a shows the mean minimum fitness over 30 runs. Again, the best fitness found is -8 and no combination found the absolute minimal energy of -9. Figure 6a suggests that λ should not be bigger than about $\approx$ 40% of the population size. Too many mutations disrupt the continuous evolution of the population.

Figure 8b depicts how many runs out of 30 found the minimum energy of -8. λ-values between 20 and 70 seem to be most beneficial.

4.3 Final Runs

Using the knowledge from the previous two sections, we ran the algorithm 30 times for each sequence with a population size of 500 and three different λ-values, 30, 40, and 50. The results can be seen in Figures 8a and 8b. All λ-value combinations found the same minima for the first four sequences (-8, -7, -5, and -13, respectively). Curiously, for sequence three, the energy value of -5 was found fairly quickly (after about 150-200 iterations) and did not improve over the next 400-600 iterations. Sequence three stands out as having the biggest ratio of polar to hydrophobic amino acids. There are three stretches with four contiguous polar amino acid residues, while the longest contiguous stretch of hydrophobic residues is only of length 2. Figure 7 shows the best individual for a run with 1000 iterations and $\lambda = 50$.



(a) The average minimal fitness value over 30 runs for different crossover types with and without pioneer search and different λ-values.

(b) The number of times the best minimal fitness value (-8) is detected over 30 runs for different crossover types with and without pioneer search and different λ-values.

Figure 5: Comparison of GA performance when using normal crossover, normal crossover and pioneer search, rotation crossover, or rotation crossover and pioneer search. Other parameters: sequence = HP-20, energy minimum = -9, population size = 100, tournament size = 10, iterations = 1000, clash limit = 10, $m-$crossover = 5, crossover rate = 0.8, mutation probabilities: in-plane = 0.4, out-of-plane = 0.2, crank shaft = 0.1, kink motion = 0.3.

Since there are few hydrophobic residues, there are only few configurations with maximal contact between hydrophobic amino acid residues. Unfortunately, our algorithm seems to not be able to find those places in the energy landscape.

For the sequences HP-48, HP-50, HP-60, and HP-64, runs with $\lambda = 40$ consistently found minima lower than those found by runs with $\lambda = 30$ and $\lambda = 50$. Table 2 summarizes the best results found by our algorithm for each sequence.

Length	Energy Minimum	Our GA	Unger and Moult[14]
HP-20	-9	-8	-9
HP-24	-9	-7	-9
HP-25	-8	-5	-8
HP-36	-14	-13	-14
HP-48	-23	-19	-22
HP-50	-36	-18	-21
HP-60	-42	-32	-32
HP-64	-52	-31	-37

Length	Energy Minimum	König and Dandekar[11]	Cox et al.[8]	Huang et al.[1]
HP-20	-9	-9	-9	-9
HP-24	-9		-9	-9
HP-25	-8		-8	-8
HP-36	-14	-14	-14	-14
HP-48	-23	-23	-23	-23
HP-50	-36		-21	-36
HP-60	-42	-37		-42
HP-64	-52			-52

Table 2: Comparison of our GA's performance with other evolutionary algorithms for PSP. Parameters of our GA: population size = 500, tournament size = 10, iterations = 1000, λ = 40, clash limit = 10, $m-$crossover = 5, crossover rate = 0.8, mutation probabilities: in-plane = 0.4, out-of-plane = 0.2, crank shaft = 0.1, kink motion = 0.3.

5 Comparison and Discussion

Table 2 compares our results to those by Unger and Moult (1993), König and Dandekar (1999), Cox et al. (2004), and Huang et al. (2010) [1], [8], [11], [14]. While it was to be expected that our algorithm would not be able to find the minima of the longer benchmark sequences, it is somewhat disappointing that it is also not able to determine the energy minima of the sequences HP-20, HP-24, HP-25, and HP-36. It is our opinion that the algorithm does not do enough exploration of the search space. However, pioneer search should have counteracted this flaw, as it forces the algorithm to create children which are not yet contained in the population. Rotation crossover could also introduce new individuals by means of its additional in-plane rotations. However, both showed no improvements in our algorithm. This is especially surprising considering that König and Dandekar are using systematic crossover and pioneer search in their implementation and their GA outperforms ours. Notably, the algorithm by Huang et al. outperforms all others. In fact, the assumed minimal energies of HP-50, HP-60, and HP-64 had not been known before Huang et al.'s publication in 2010. The big difference of their approach is that they are using knowledge of secondary protein structures in order to predict the tertiary structure. We also considered scanning the amino acid sequences for possible secondary structure elements but decided against it. Since the HP-model only differentiates between hydrophobic and polar amino acids, Huang et al. make their prediction of secondary structures based solely on the existence of contiguous stretches of hydrophobic amino acids in the primary sequence. Furthermore, it has been noted that secondary structure elements will develop naturally during the folding process. Therefore, Huang et al. shorten computational time by predetermining secondary structure elements but also limit the possibilities for tertiary structure conformations by defining parts of it in advance.

Since our algorithm did not find a minimal energy value, it always ran for 1000 iterations until the termination condition is satisfied. Algorithms in Table 2 which found a minimum energy value, did not run through all iterations. Therefore, their computational cost is less than for our algorithm. However, by using multiple processors for crossover, mutation, and fitness evaluation, execution times of our algorithm are pleasantly low. For HP-20, population size= 500, λ = 40, and 1000 iterations, the average execution time for one run is $\approx$ 39.5 seconds. For HP-64 with the same parameters, the average execution time is $\approx$ 3.6 minutes.

While our algorithm was not able to predict the minimal energy conformations of the shorter benchmark sequences, there are still adjustments that can be made. For this project we focused mainly on λ values, crossover type, and pioneer search. We also made changes to crossover rate and the probability distribution between in-plane rotation, out-of-plane rotation, crankshaft rotation, and kink motion. Concerning the probability distribution between mutation types, we discovered that the algorithm shows slightly better performance when in-plane rotation, which preservers adjacency information the most, has a higher probability than the others. We did not find parameter combinations that led to significant improvements in the algorithm's result. However, our search was not exhaustive and so there are still combinations to be tried out.

6 Conclusion

We implemented a genetic algorithm for protein structure prediction. Protein folding was abstracted using the 2D-HP-lattice model. We tried a novel approach by combining $m-$systematic crossover with $\lambda-$mutations. We determined that a λ value between 30 and 70 is most beneficial for the results of the algorithm. By sampling combinations of normal or rotation crossover and with or without pioneer search, we concluded that usage of normal crossover without pioneer search is superior considering the tradeoff between performance and computational time. We tested our genetic algorithm on eight benchmark sequences. While the algorithm gave good approximations for shorter sequences, it did not find the absolute minimum energy. There are still parameter combinations left which can be tested in order to improve performance. For future work, we think a different energy definition could enhance structure prediction abilities. Currently, only interactions between adjacent hydrophobic amino acid residues which are not sequence neighbours are considered. Additionally, there are interactions between polar amino acid residues and water molecules which should be considered. Designing a GA with such an energy function was infeasible for this course project, as there are no benchmark sequences with known energy minima using such a function definition. Furthermore, diagonal interactions in the lattice can be considered in addition to vertical and horizontal interactions.

References

[1] C. Huang, X. Yang, and Z. He, “Protein folding simulations of 2d hp model by the genetic algorithm based on optimal secondary structures”, *Computational Biology and Chemistry*, vol. 34, no. 3, pp. 137–142, 2010.

[2] T. W. de Lima, P. H. R. Gabriel, A. C. B. Delbem, R. A. Faccioli, and I. N. da Silva, “Evolutionary algorithm to ab initio protein structure prediction with hydrophobic interactions”, in *2007 IEEE Congress on Evolutionary Computation.* 2007, pp. 612–619.

[3] unkown author, *Pdb - renewable energy*, Accessed April 20, 2018, Research Collaboratory for Structural Bioinformatics (RCSB. [Online]. Available: `http://pdb101.rcsb.org/browse/renewable-energy`.

[4] ——, *Pdb - nanotechnology*, Accessed April 20, 2018, Research Collaboratory for Structural Bioinformatics (RCSB. [Online]. Available: `http://pdb101.rcsb.org/browse/nanotechnology`.

[5] G. W. Greenwood and J.-M. Shin, “Chapter 6 - on the evolutionary search for solutions to the protein folding problem”, in *Evolutionary Computation in Bioinformatics*, ser. The Morgan Kaufmann Series in Artificial Intelligence, G. B. Fogel and D. W. Corne, Eds., 1st ed., San Francisco: Morgan Kaufmann, 2003, pp. 115–136.

[6] D. L. Nelson and M. M. Cox, *Lehninger - Principles of Biochemistry*, 5th ed. New York, NY, USA: WH. Freeman and Company, 2008.

[7] C. B. Anfinsen, “Principles that govern the folding of protein chains”, *Science*, vol. 181, no. 4096, pp. 223–230, 1973.

[8] R. P. T. Graham A. Cox Thomas V. Mortimer Jones and R. L. Johnston, “Development and optimisation of a novel genetic algorithm for studying model protein folding”, *Theoretical Chemistry Accounts, Springer*, pp. 163–178, 2004.

[9] K. F. Lau and K. A. Dill, “A lattice statistical mechanics model of the conformational and sequence spaces of proteins”, *Macromolecules*, vol. 22, no. 10, pp. 3986–3997, 1989.

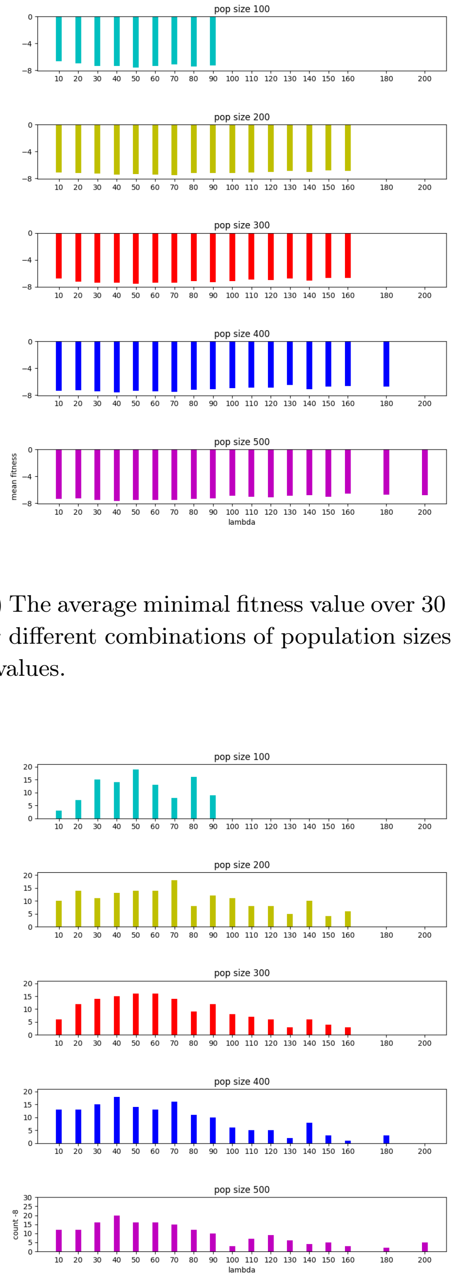
[10] K. A. Dill and H. S. Chan, “From levinthal to pathways to funnels”, *Nature Structural Biology*, vol. 4, no. 4, pp. 10–19, 1997, Nature Publishing Group.

[11] T. D. Rainer König, “Improving genetic algorithms for protein folding simulations by systematic crossover”, *BioSystems,Elsevier*, no. 50, pp. 17–25, 1999.

[12] L. F. Sotto, V. V. de Melo, and M. P. Basgalupp, “An improved λ-linear genetic programming evaluated in solving the santa fe ant trail problem”, in *SAC*, 2016, pp. 103–108.

[13] G. A. Cox and R. L. Johnston, “Analyzing energy landscapes for folding model proteins”, *The Journal of Chemical Physics*, vol. 124, no. 20, p. 204 714, 2006.

[14] R. Unger and J. Moult, “Genetic algorithms for protein folding simulations”, *Journal of molecular biology*, vol. 231, no. 1, pp. 75–81, 1993.



(a) The average minimal fitness value over 30 runs for different combinations of population sizes and λ-values.

(b) The number of times the best minimal fitness value (-8) is detected over 30 runs for different combinations of population sizes and λ-values.

Figure 6: Comparison of GA performance for different combinations of population sizes and $\lambda-$values. Other parameters: sequence = HP-20, energy minimum = -9, tournament size = 10, iterations = 1000, clash limit = 10, $m-$crossover = 5, crossover rate = 0.8, mutation probabilities: in-plane = 0.4, out-of-plane = 0.2, crank shaft = 0.1, kink motion = 0.3.

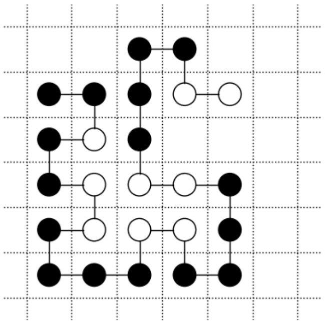
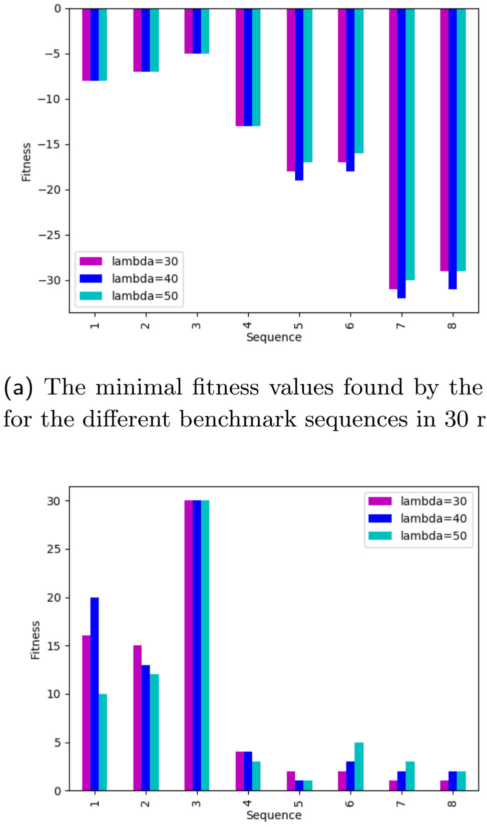


Figure 7: Best individual for sequence HP-25, $\lambda = 50$, population size = 500, iterations = 600 with energy = -5.



(a) The minimal fitness values found by the GA for the different benchmark sequences in 30 runs.

(b) The number of times the minimal fitness value was found in 30 runs.

Figure 8: Comparison of GA performance for the individual benchmark sequences. Parameters: population size = 500, tournament size = 10, iterations = 1000 for $\lambda = 40$ and iterations = 600 for $\lambda = 30$ and $\lambda = 50$, clash limit = 10, $m-$crossover = 5, crossover rate = 0.8, mutation probabilities: in-plane = 0.4, out-of-plane = 0.2, crank shaft = 0.1, kink motion = 0.3.