APPLIED MATHEMATICS REPORT: FINDING PROTEIN CONFORMATIONS WITH SIMULATED ANNEALING IN A 2D HP LATTICE MODEL

Ellian Carlos Oliveira Costa[1*]

*Corresponding author
[1] Universidade Federal Fluminense (first author)
E-mails: elliancarlos@gmail.com

ABSTRACT. Proteins are a vital part of any byological being, proteins dictacte uncountable mechanics in the human body and are a essencial part of life. Predicting the folding mechanism of a protein from its sequence of amino acids is the Protein Folding Problem (PFP). This report explores the application of Simulated Annealing (SA) to solve the Protein Folding Problem (PFP) within a 2D Hydrophobic-Polar (HP) Lattice Model.

Keywords: Protein folding problem, H-P model; Minimal energy conformation; 2D square lattice; Simulated Annealing algorithm (SA).

1 Introduction

Proteins are chains of polypeptides combined into a specific structure. Protein's structures plays a major role in its function and mechanics. The sequence of polypeptides is the primary structure of a protein and the final form of a single protein is called tertiary. A posible protein conformation that fits a fixed amino acid sequence is a fold. In nature, the process of folding a protein happens very fast and in such a way that minimizes the protein energy. The prediction of the minimal energy conformation of the protein given the sequence of amino acids is called the Protein Folding Problem (PFP).

Amino acids, which constitutes polypeptides, are either hydrophobic or hydrophilic, this single property can be used to caracterize the energy of the molecule for folding pourposes, as was Lau and Dill's H-P Lattice Model Lau (1989). This hydrophobic-hydropolar model is widely used for computationally solving the PFP. It is based on the fact that in their native state proteins minimize the internal energy in the folding process. This energy can be calculate counting the uniques H-H contact in the lattice model, so the folding process maximizes the H-H contact. In this model, for every H-H contact a energy score of -1 is given. So it is posible to rewrite the PFP as finding the protein structure in a lattice square space such that the protein native fold $s*$ is the conformation that maximizes the energy function $E(s) = -1 \sum \# \textit{ of unique H-H contacts of } s$ so: $s* = argmin\{E(s)/s \in S\}$, being S the conformational space. The protein structure conformational space is every posiblity of a Self-Avoiding Walk in a 2D Lattice. This model simplifies the PFP problem, but it is still a very complex in both 2D lattices or 3D lattices, it's unfeasable to use exact methods since this problem is a NP-complete.

Solving the PFP in the HP Model has been a great area of reasearch, many lattice models, heuristics and meta heuritics have been developed and applied to the problem. In this report we focus to solve the problem with a simulated annealing in a HP 2D Square Lattice Model.

The PFP problem is claimed to be resolved by AlphaFold in Jumper (2021), since 2021 the AlphaFold Protein Database has predicted over 200 million protein structure predictions from aminoacids sequence. Different from this work, they don't only predict the folding of the protein but its whole structure. AlphaFold is set to be the big thing for protein prediction since in a long time.

2 Code Disclaimer

The model, the proposed algorithm and every computated resource was coded in Python 3.8 with the use of `numpy` as the only external library, aside from `numpy` and the native Python libraries all code was develop by the author. The code was done trying tried to adhere to good principles and POO techniques, at least as much as Python allows for its developers. All the code was run on a laptop with the 11th Gen Intel(R) Core(TM) i5-1135G7 CPU, 8GB of RAM in a NixOS Linux. If needed the author can provide more details about the configuration. Aside from the sent code, the code was also publicly available at this repository or at the url:`https://github.com/EllianCarlos/hp-lattice-model`.

3 The HP Model Used

The used HP Model is described in a 2D Lattice Space, and the protein structure is a Self Avoiding Walk in this 2D Lattice Space where every point is Polar (P) or Non-Polar (H), in this work this property will be called affinity. Be s any solution for the protein structure in the conformational space, s_i is the $i-th$ amino acid in the protein structure, affinity $a(s_i)$ of s_i is:

$$a(s_i) = \begin{cases} H & \text{if } s_i \text{ is hydrophobic.} \\ P & \text{if } s_i \text{ is non-polar.} \end{cases}$$

The 2D Square Lattice model is such that every point in the lattice has 4 neighbors, one upwards, one downwards, one leftwards and one rightwards. A s_i that is not the last nor the first amino acid in the sequence has a s_{i-1} predecessor and a s_{i+1} sucessor, a predecessor and a sucessor of a amino acid s_i, must be a neighbor of s_i. Every amino acid in the protein, must ocuppy one node in the lattice. A single node in the lattice can have either zero or one amino acid. This defines the constraints that build up the conformation space.

The energy of a conformation can be calculated via:

$$E(s) = -1 \cdot \sum_{i=1}^{n-2} \sum_{j=i+2}^{n} x_{ij} y_{ij}$$

$$x_{ij} = \begin{cases} 1 & \text{if } s_i = H \text{ and } s_j = H \ . \\ 0 & \text{otherwise.} \end{cases}$$

$$y_{ij} = \begin{cases} 1 & \text{if } s_i \text{ and } s_j \text{ contact each other .} \\ 0 & \text{otherwise.} \end{cases}$$

For example, the following model has a energy of -2:

3.1 Instances

In this work we will cite the instances as a sequence of H and P characters representing each amino acid in the sequence and their affinity, for example: $PPHPHP$. Other used form of instance will be using chains o H and P characters alongside a expoent such that this number symbolize the number of times the last character or chain must be repeated, for example $P^2(HP)^2$ is the same as $PPHPHP$.

All available and calculated instances in this report will be alongside the delivered code under the path *data/instances*, all of them in the *.dat* extension which resolves only to a text string to the first representation in this section.

The test instances will be:

I	L	Sequence	Native Energy $E(s*)$
0	6	HP^2HPH	-2
1	20	$(\mathrm{HP})^2PH(HP)^2(PH)^2HP(PH)^2$	-9
2	24	$\mathrm{H}^2P^2(HP^2)^6H^2$	-9
3	25	$\mathrm{P}^2HP^2(H^2P^4)^3H^2$	-8
4	40	$\mathrm{P}^2H(P^2H^2)^2P^5H^{10}P^6(H^2P^2)^2HP^2H^5$	-23

Table 1: Table of instances Is with their Length, Sequence and Energy.

3.2 Solutions

All solutions will be saved locally automatically and describe in a text format with *.dat* extension and will formated with the pattern:

- Each line contains only one piece of information
- Every file there is a separator that is a long sequence of the character -
- The lines until a separator are metadata, all lines after each describe the protein conformation
- Each line contains a KEY followed by a whitespace and then a VALUE
- In each line after the separator the KEY is the affinity of the amino acid and the VALUE the direction to append the aminoacid based on its predecessor

Example of a file with the solution of the first instance $I0$:

```
NAME 57ea5767-a0cc-4c8c-b345-80b1e6f740b2_model
ID: 57ea5767-a0cc-4c8c-b345-80b1e6f740b2
Energy: -2
--------------------------------------------------
H None
P DOWN
P RIGHT
H UP
P UP
H LEFT
```

3.3 Model Construction and Representation

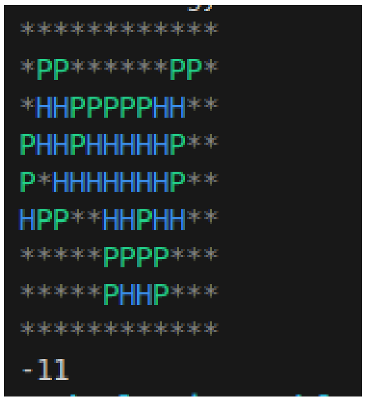
The HP Model was constructed in code with two major classes, one the `Vertice` class representing a single amino acid in the HP model, this classes controls properties of each amino acid, as the polarity (hydrophobic or non-polar), pointers to others connected amino acids, distance to any other vertice and some static methods relative to utilities of vertices, like the movement of vertices. The other main class of the model is the `Hp2dSquareModel`, which has all the utilities to control the chain of `Vertices` or amino acids, it also has the utilities to load from a file and save the model to a file and calculate energy.

4 The Algorithm

The Simulated Annealing algorithm is among the most used meta-heuristics and it's based around a local search, it draws inspiration from the annealing termodynamics phenomena. The used version of SA is based on the paper Boumedine (2020), it is called Improved Simulated Annealing, and the major differences are the neighbor function, which is constitute by a local move from a triangular lattice describe in Bockenhauer (2008) .

Algorithm 1 Simulated Annealing - SA($f(.), N(.), s, \alpha, SAmax, T_0$)

```
Require: The instance
Ensure: The best solution $s^* s$
  $s \leftarrow$ generate an initial solution in a random way;
  $s^* \leftarrow$ the best solution $s$;
  $T \leftarrow$ initial temperatue ;
  $T_{min} \leftarrow$ minimal temperature ;
  $k_{max} \leftarrow$ maximum number of iteration after the use of diversification technique ;
  $i \leftarrow 0$;
  $iterT \leftarrow 0$
  while $T \geq T_{min}$ do
      while $iterT \geq SAmax$ do
          $iterT \leftarrow iterT + 1$
          $s' \leftarrow$ some $s \in \mathbf{N(s)}$
          $\Delta \leftarrow f(s') - f(s)$
          $\delta \leftarrow e^{\Delta/T}$                                   ▷ $\delta$ is entropy
          if $\Delta < 0$ or $random\ u \in (0,1) < \delta$ then
              $s \leftarrow s'$
          end if
          if $(f(s') < f(s^*))$ then
              $s^* \leftarrow s'$
          else
              $i \leftarrow i + 1$
          end if
          if $i = k_{max}$ then
              $s \leftarrow$ some $s \in \mathbf{N(s)}$
              $i \leftarrow 0$
          end if
          $T \leftarrow \alpha T$
      end while
  end while
```



```
************
*PP******PP*
*HHPPPPPHH**
PHHPHHHHHP**
P*HHHHHHHP**
HPP**HHPHH**
*****PPPP***
*****PHHP***
************
-11
```

Figure 1: Image of model's print with a print of the energy associated with it

4.1 Algorithm Construction

The algorithm construction was not an easy task as that was no previously publicated open source code from any of the authors of the base paper Boumedine (2020). So the author had to build the HP Square Model by hand and also the so called Improved Simulated Annealing Algorithm. The SA algorithm bases itself on a neighbor function, that does a local search on a solution candidate. For every iteration on the algorithm a search $s' = N(s)$ is assigned, if $E(s') < E(s)$ a new solution s is assigned, if $E(s') >= E(s)$, s can be assigned to s' if a random uniform generator generates a number is less than the actual entropy $e^{-\Delta/T}$.

The parameters used were:

Parameter	Value
$alpha$	0.90
SA_max	100
$initial\ temperature\ T_0$	10000
K_{max}	10

Table 2: Table of parameters.

4.2 The random generation of solutions

The random generated solution is a basic Random Self Avoidance Walk (Random SAW) in the constructed model. There was tested a function `generate_biased_random_solution` that generates a SAW that is sligthly biased to turn instead of continuing in the same path. This `generate_biased_random_solution` did not generate any improvement in the model so it was not used after tests, but it was sended in the final anyway.

4.3 The Neighbor Function

The neighbor function implemented was based on Bockenhauer (2008) where a vertice of the HP Square Model is diagonally move to improve H-H contacts and so improve connections. The author's implementation was done checking every move possible for every vertice in the model and evaluating its energy. The author tested two neighbor functions one that does the previous calculations and returns the best evaluated and other which returned any posible previous movement, even if it does not improves the model evaluation. The latter one was used, because it was left to the SA algorithm to the improvement instead of the neighbor and it was way faster than evaluating every movement posible.

The paper Lin C-J. Su (2011) was cited for a local rotation in the end of the SA algorithm, instead of another neighborhood search, but the author could not implement this feature in time, so the rotation isn't implemented in this report, but with more time, this would be posible and would probably improve results.

5 Results

The implemented algorithm was compared to other algorithms cited in this report, ISA for the Improved Simulated Annealing Boumedine (2020), GA for the Genetic Algorithm Natalio Krasnogor and Pelta. (1999) and ACO for the Ant Colony Optimization Alena Shmygelska and Hoos (2002). The builded algorithm was run 100 times for each of the instances and its results are in the column of Implemented ISA.

Instance	Energy	ISA	GA	ACO	Implemented ISA
0	**-2**	No Data	No Data	No Data	**-2**
1	**-9**	**-9**	**-9**	**-9**	-7
2	**-9**	**-9**	**-9**	**-9**	-7
3	**-8**	**-8**	**-8**	**-8**	-5
4	**-14**	**-14**	-12	**-14**	-11

Table 3: Table of results to compare.

As it occurs the implemented algorithm could not find the protein structure for any instance with enough complexity.

6 Conclusions

The implemented algorithm was able to optimize in some sort the model from random solutions, but it was heavily dependent on two things, one the neighbor function, as said before the author only implemented a local move set from Bockenhauer (2008), but not the rotation from Lin C-J. Su (2011), so the results was not able to rivalize with the algorithm from Boumedine (2020), so this was a set back for the algorithm, another point was that the algorithm depends heavily on the random generated instance, as the classical SA meta-heuristic it heavily depends on the first guess to do a good local search, and based on that search, improves the solution, far from optimal solutions even with search will still be far from optimal optimized solutions, so the algorithm depends on it.

The author firmly believes the algorithm was a success since it was able to optimize a HP 2D Square Lattice model to represent folding proteins, even though it did not rivalized with the current methods it was a great learning opportunity. The author also firmly bealives that upon improving the neighborhood search method and applying the given rotation in Lin C-J. Su (2011), this algorithm can even best the current solutions.

References

Alena Shmygelska, R. A.-H. and Hoos, H. H. (2002). An ant colony optimization algorithm for the 2d hp protein folding problem. *In International Workshop on Ant Algorithms*.

Bockenhauer, H. Ullah, A. K. L. S. K. (2008). A local move set for protein folding in triangular lattice models. *Springer*.

Boumedine, N. Bouroubi, S. (2020). An improved simulated annealing algorithm for optimization of protein folding problem. *International Workshop on Human-Centric Smart Environments for Health and Well-being*.

Jumper, J. (2021). Highly accurate protein structure prediction with alphafold. *Nature*.

Lau, K. F. Dill, K. A. (1989). A lattice statistical mechanics model of the conformational and sequence spaces of proteins. *Macromolecules*.

Lin C-J. Su, S.-C. (2011). Using an efficient artificial bee colony algorithm for protein structure prediction on lattice models. *International Journal of Innovative Computing, Information and Control*, 8:2049–2064.

Natalio Krasnogor, William E. Hart, J. S. and Pelta., D. A. (1999). Protein structure prediction with evolutionary algorithms. *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation*, 2:1596–1601.