Protein Structure Prediction Using the Hill Climbing Algorithm in the 2D Lattice HP Model

Rohit Bogulla
Department of Computer Science
University of Florida
Email: rbogulla@ufl.edu

***Abstract*—Protein Structure Prediction (PSP) seeks to determine the native three-dimensional conformation of a protein from its amino acid sequence. Due to the immense complexity of the folding process, simplified models such as the Hydrophobic–Polar (HP) lattice model are often employed for computational studies. This paper presents a greedy local search approach, namely the Hill Climbing algorithm, for predicting the optimal conformation of a protein sequence in a 2D lattice using relative direction encoding. The method iteratively minimizes energy by accepting only moves that improve hydrophobic contacts. Experimental results on small HP sequences show rapid convergence to low-energy conformations, with computational complexity scaling linearly with the number of iterations.**

***Index Terms*—Protein folding, Hill Climbing, HP model, Lattice model, Greedy algorithms, Local search, Bioinformatics**

I. Introduction

Protein folding is one of the central problems in bioinformatics and computational biology. Proteins are linear chains of amino acids that spontaneously fold into stable three-dimensional structures that determine their function. Predicting this structure directly from the sequence is computationally challenging and has been proven to be NP-hard even for simplified models.

To reduce complexity, the Hydrophobic–Polar (HP) lattice model [1] approximates folding by placing residues on a 2D or 3D grid, classifying amino acids as hydrophobic (H) or polar (P). The simplified lattice-based models provide a computationally tractable abstraction that captures the essential biophysical principle of hydrophobic collapse, where hydrophobic residues tend to cluster away from the aqueous environment while polar residues remain exposed. The objective is to find a self-avoiding conformation that minimizes energy by maximizing non-sequential hydrophobic contacts.

In this work, we implement a greedy Hill Climbing algorithm to search for low-energy conformations in the 2D HP model. The algorithm iteratively refines a random valid conformation by applying local modifications that maximize Hydrophobic-Hydrophobic contact or minimize the total energy.

II. Problem Identification

In this project, we address the problem of **protein structure prediction** under the simplified two-dimensional Hydrophobic–Polar (HP) lattice model. A protein sequence is represented as a chain

$$S = (s_1, s_2, \ldots, s_n), \quad s_i \in \{H, P\}$$

where H denotes a hydrophobic residue and P denotes a polar residue. Each residue occupies a unique point on a 2D square lattice, forming a self-avoiding walk (SAW), such that consecutive residues are adjacent on the grid.

The stability of a conformation C is quantified by an *energy function*:

$$E(C) = -\sum_{(i,j)} \delta(s_i, H)\delta(s_j, H) \cdot \mathbf{1}_{\text{adj}}(i, j)$$

where $\delta(s_i, H)$ is an indicator function that equals 1 if residue s_i is hydrophobic and 0 otherwise, and $\mathbf{1}_{\text{adj}}(i, j)$ equals 1 when residues i and j are adjacent on the lattice but not consecutive in sequence.

Each non-sequential pair of hydrophobic residues that occupy adjacent lattice sites contributes -1 to the total energy, reflecting stabilizing hydrophobic interactions. Therefore, minimizing $E(C)$ corresponds to maximizing the number of favorable hydrophobic contacts, which approximates the biological tendency of proteins to fold into compact, low-energy conformations.

The problem can thus be formulated as:

> *Given an HP sequence S, find a self-avoiding conformation C on a 2D lattice that minimizes the total energy $E(C)$.*

We approach this optimization problem using a **Hill Climbing** algorithm, a local search heuristic that iteratively modifies the current conformation to reduce energy, until no further improvement is possible. This method allows exploration of the discrete conformational space while balancing computational efficiency and solution quality.

A. Biophysical Motivation: Hydrophobic Collapse and Zipper Effect

Proteins naturally fold into specific three-dimensional conformations driven by the interactions among their constituent amino acids. A key determinant of this folding process is the distribution of hydrophobic and polar residues within the sequence.

Hydrophobic (H) residues tend to avoid contact with the aqueous environment, preferring to cluster together in the

interior of the folded structure. Polar (P) residues, by contrast, are energetically favorable on the surface where they can interact with solvent molecules. This physical tendency leads to the phenomenon known as *hydrophobic collapse*, where the chain rapidly compacts to minimize solvent exposure of hydrophobic sites.

In lattice-based models such as the 2D HP model, hydrophobic collapse is represented by maximizing the number of non-sequential hydrophobic–hydrophobic (H–H) contacts. These contacts lower the overall energy of the conformation and are often used as the primary energy term in simplified protein folding simulations.

As folding progresses, local structural motifs stabilize sequentially in a cooperative manner, a behavior known as the *zipper effect*. This mechanism reflects how certain regions of the protein fold independently and then interact to form the final native conformation.

Together, hydrophobic collapse and the zipper effect define the energy landscape over which optimization algorithms such as Hill Climbing search for the lowest-energy configuration. The goal of the computational model is to approximate these natural folding forces through an energy function that rewards favorable H–H interactions and penalizes steric overlaps or invalid configurations.

III. Problem Abstraction

The protein structure is abstracted as a discrete path embedded on a two-dimensional lattice. Formally, we represent the protein as a graph $G = (V, E)$ where:

- Each vertex $v_i \in V$ corresponds to a residue s_i in the sequence $S = (s_1, s_2, \ldots, s_n)$, where $s_i \in \{H, P\}$.
- Each edge $(v_i, v_{i+1}) \in E$ connects sequential residues, preserving the primary structure of the protein.
- Non-sequential edges (v_i, v_j) are introduced implicitly when residues v_i and v_j occupy adjacent lattice positions (x_i, y_i) and (x_j, y_j), satisfying $|x_i - x_j| + |y_i - y_j| = 1$. These pairs represent potential hydrophobic contacts contributing to the total energy.

Each valid embedding of G corresponds to a *self-avoiding walk (SAW)* on the lattice: no two vertices may occupy the same coordinate, i.e.,

$$(x_i, y_i) \neq (x_j, y_j), \quad \forall i \neq j.$$

The folding path is defined as a sequence of relative directional moves:

$$M = (m_1, m_2, \ldots, m_{n-1}), \quad m_i \in \{L, R, F\}$$

where:

- L (*Left turn*) rotates the current orientation by $-90°$,
- R (*Right turn*) rotates the current orientation by $+90°$,
- F (*Forward*) continues in the same direction as the previous residue.

At each step, the position of the next residue is computed relative to the current orientation and lattice coordinates. The first residue is placed at the origin $(0, 0)$, and the first move defines the initial direction.

Invalid conformations are automatically excluded from the search space when any of the following constraints are violated:

- Two residues occupy the same lattice position (self-intersection),
- Consecutive moves directly reverse direction (*e.g.*, a Left followed immediately by a Right, or vice versa),
- The resulting chain leaves the predefined lattice boundary.

Thus, the search space Ω consists of all valid paths M satisfying these geometric and topological constraints. Each state (or node) in the search space corresponds to a unique conformation with an associated energy $E(C)$, as defined in the HP model. Local search operations in the Hill Climbing algorithm correspond to small perturbations in this lattice path, such as *corner flips*, *end rotations*, or *segment reversals*, which maintain chain connectivity while exploring nearby conformations.

IV. Algorithm Description

The proposed solution employs a **Hill Climbing** algorithm to minimize the energy function $E(C)$ of a protein conformation C on a 2D lattice. The algorithm begins from an initial valid conformation and iteratively explores neighboring conformations by applying small local modifications to the current structure. At each iteration, it selects the neighboring state with the lowest energy. It halts when no improving neighbor exists, indicating convergence to a local optimum.

Algorithm 1 Hill Climbing for Protein Folding

```
Generate a random valid conformation C
Compute E = Energy(C)
repeat
    improved ← False
    for each neighbor C' of C do
        if valid(C') and Energy(C') < E then
            C ← C'
            E ← Energy(C')
            improved ← True
            break
        end if
    end for
until not improved
return C, E
```

A. Energy Evaluation

Energy is computed as:

$$E(C) = -N_{HH}$$

where N_{HH} is the number of non-consecutive hydrophobic (H–H) contacts.

B. Neighborhood Structure

The neighborhood of a conformation includes all structures reachable by one of the following valid moves:

- **Corner Flip:** Rotates a residue by 90° around its adjacent residues.
- **End Move:** Rotates one end of the chain by 90° or 180°.

C. Explanation of Pseudocode

The algorithm follows a greedy local improvement strategy:

1) **Initialization:** A random valid folding path (sequence of L, R, F moves) is generated as the starting conformation and its Energy is computed.
2) **Neighborhood Generation:** All conformations that differ from the current one by a small structural modification (e.g., corner flip or end rotation) are considered as neighbors. Only self-avoiding and valid paths are retained.
3) **Energy Evaluation:** Each neighbor's energy $E(C')$ is computed using the HP lattice energy model.
4) **Selection:** If the conformation has lower energy than the current structure, the neighbor conformation replaces the current conformation.
5) **Termination:** The process stops when no neighbor achieves a lower energy, implying convergence to a local minimum.

This method efficiently searches the conformational space while maintaining feasible structures at each iteration. Although it may converge to local optima, repeated restarts and randomized perturbations can help approximate the global minimum for small to moderate protein sizes.

V. Time Complexity Analysis

Let n be the number of residues in the protein and k be the average number of valid local perturbations (neighbor moves) per conformation. In the 2D lattice HP model, each residue—except the terminal ones—can be followed by one of three possible relative moves: Left (L), Forward (F), or Right (R). Hence, in the absence of steric constraints, each residue potentially contributes up to 3 move options, although only a subset remains valid due to self-avoidance and lattice overlap checks.

At each iteration, the Hill Climbing algorithm:

- Generates all valid neighbors by applying one of the k local modifications (typically derived from the LFR set).
- Evaluates the energy $E(C')$ of each neighbor in $O(n)$ time.

Thus, the time complexity per iteration is $O(k \cdot n)$. For a total of I iterations, the overall time complexity becomes:

$$T(n) = O(I \cdot k \cdot n)$$

Since the LFR move set limits k to a small constant (typically between 2 and 5 valid moves per residue), the complexity is effectively linear in both the number of residues and the number of iterations:

$$T(n) \approx O(I \cdot n)$$

This indicates that runtime grows proportionally with the protein length and the number of improvement steps, making the algorithm computationally efficient for small to moderately sized protein sequences.

VI. Proof of Correctness

The Hill Climbing algorithm always converges to a local optimum as at each step it moves in the direction which reduces energy. Formally:

- **Termination:** Each accepted move strictly decreases energy; since the search space is finite, termination is guaranteed.
- **Local Optimality:** On termination, no neighboring conformation has a lower energy, satisfying the definition of a local minimum.

VII. Algorithm in Domain Language

In terms of protein folding, the algorithm mimics a natural "downhill" energy descent: starting from a randomly unfolded chain, it repeatedly refines the structure by making small conformational changes that stabilize hydrophobic cores. When no single amino acid movement improves stability, the protein is considered folded into a metastable state.

VIII. Implementation

Implementation of hill-climbing algorithm requires certain important functions before we can perform the hill-climbing. The Implementation is made available at [5]

1) **Generate Random Valid Fold:** A function which generates random valid folding path, sequence of L, R, F moves, as the starting conformation.
2) **Grid Generation for Path:** Given a path generate a 2D grid for the Path where each (x,y) coordinate corresponds to a HP lattice.
3) **Calculate Energy for a Grid:** Given a HP lattice sequence and a SAW, Valid Grid, calculate the non-covalent HH contact. The energy of the model is the negative of the total HH contacts, so the problem now is minimization of energy.
4) **Hill-Climbing:** Start from a random fold. At each step chose a index of the path to change and calculate the new path's energy. If the energy of new path is lower than current paths's energy new path becomes current path and explorations happen in the new path's neighbors.

The subsequent sections mention the implementation of the above discussed functions.

A. Random Fold

We generate a random path and use the generateGrid() function to confirm if its a self-avoiding walk and a valid fold.

```
moves = ['L', 'F', 'R']

def randomPath(length):
    while True:
        path = [random.choice(moves) for _ in range(
    length)]
```

```
        valid, grid = generateGrid(path)
        if valid:
            return path, grid
```

Listing 1. Python implementation of generating a Random Valid Fold

B. Generate Grid

Given a path or sequence of L, R, F moves generate a 2D grid for the path with (x,y) coordinates. Each coordinate maps to a H or P from the HP sequence.

```
def generateGrid(path):
    coords = [[0,0]]
    prevTile = (-1,0) # left-to-right start

    for i in range(len(path)):
        new_coords = []
        currTile = coords[-1]
        try:
            match path[i]:
                case 'L':
                    # whats not changing will change
                    if prevTile[0] - currTile[0] ==
    0:
                        x = prevTile[1] - currTile
    [1] + currTile[0]
                        y = currTile[1]
                    elif prevTile[1] - currTile[1]
    == 0:
                        x = currTile[0]
                        y = currTile[0] - prevTile
    [0] + currTile[1]
                    else:
                        raise ValueError("Error with
     left move")
                case 'R':
                    # whats not changing will change
                    if prevTile[0] - currTile[0] ==
    0:
                        x = currTile[1] - prevTile
    [1] + currTile[0]
                        y = currTile[1]
                    elif prevTile[1] - currTile[1]
    == 0:
                        x = currTile[0]
                        y = prevTile[0] - currTile
    [0] + currTile[1]
                    else:
                        raise ValueError("Error with
     right move")
                case 'F':
                    # whats same will be same
                    x = currTile[0] - prevTile[0] +
    currTile[0]
                    y = currTile[1] - prevTile[1] +
    currTile[1]
                case _:
                    raise ValueError("Invalid Move!!
    ")
        except ValueError as e:
            print("Invalid move: ", e)
            exit()

        new_coords.append(x)
        new_coords.append(y)

        # check the Self-Avoiding Walk constraint
        if new_coords in coords:
            return False, []
        coords.append(new_coords)
        prevTile = currTile

    return True, coords
```

Listing 2. Python implementation of Generating a grid for a path

C. Calculate Energy

Given a HP sequence and a grid, calculates the energy of the conformation.

```
def calculateHHContact(sequence, grid):
    energy = 0
    directions = [(1,0), (0,1), (-1,0), (0,-1)]

    for i in range(len(sequence)):
        if sequence[i] == 'H':
            pos = grid[i]
            neighbours = [[pos[0]+j[0], pos[1]+j[1]]
     for j in directions]

            for j in neighbours:
                if j in grid:
                    idx = grid.index(j)
                    if sequence[idx] == 'H':
                        energy += 1

            if i != 0 and sequence[i-1] == 'H':
                energy -= 1
            if i != len(sequence) - 1 and sequence[i
    +1] == 'H':
                energy -= 1

    # since energy is going to be counted twice
    return -1 * (energy//2)
```

Listing 3. Python implementation for Calculating the energy of an conformation

D. Hill Climbing

Given a HP sequence get a path which gives the lowest energy estimate at the end of the max_iterations or when local minima is reached.

```
def hill_climb(seq, max_iteration=1000):
    path, grid = randomPath(len(seq)-1)
    bestEnergy = calculateHHContact(seq, grid)
    # print(path, bestEnergy)

    for _ in range(max_iteration):
        i = random.randrange(len(path))
        new_path = path[:]
        new_path[i] = random.choice([m for m in
    moves if m != path[i]])

        valid, new_grid = generateGrid(new_path)
        if not valid:
            continue
        new_energy = calculateHHContact(seq,
    new_grid)
        if new_energy < bestEnergy:
            bestEnergy = new_energy
            path = new_path[:]

        # print(path, new_energy, bestEnergy)

    return bestEnergy, path
```

Listing 4. Python implementation for hill climbing simulation

IX. Experimental Results

A. Experiment 1

1) Setup: Experiments were performed using synthetic sequences such as:

$$S_1 = \texttt{HPHPPHHPHPPH}, \quad S_2 = \texttt{HHPPHPHPH}$$

Each run was repeated 30 times to measure average convergence and energy values.

2) Results: Table 1 shows the Average Final Energy and Average Convergence for the 2 synthetic protein sequences. The convergence is measured with a premature stopping condition, which states that the exploration stops when there's been no improvement in the energy for a fixed number of cycles, in this case 100.

TABLE I
EXPERIMENT 1 RESULTS

Sequence	Avg. Final Energy	Avg. Iterations	Avg. Runtime
HPHPPHHPHPPH	-2.5	160	0.0061s
HHPPHPHPH	-1.1	145	0.0046s
length = 100	-12.36	591	0.7863s

Table 2 shows some of the Random Fold for the sequence and the resulting final fold and final energy

TABLE II
EXPERIMENT 1 RESULTS

Sequence	Random Fold	Final Fold	Energy
HPHPPHHPHPPH	FRLRRLFRLLR	FRLRRLRRLRR	-5
HPHPPHHPHPPH	RLRFFFLLRLL	RLLFFFLLRLL	-3
HPHPPHHPHPPH	RFLFRRFLRFF	RRLFRRFLRRF	-2
HPHPPHHPHPPH	FLLRFLFFFRR	FLLRRFFRFRR	-4
HPHPPHHPHPPH	LLRLRRFRLRF	LLRLRRFRLRF	-1
HHPPHPHPH	LFFLRRLF	LFLLRRLF	-1
HHPPHPHPH	RLFFLFLR	RFFRFLFL	0
HHPPHPHPH	FLRRFLFF	FLRRFRFF	-2
HHPPHPHPH	LRLRFLFR	LRLLFLFR	-2

Below figures show 30 final energy for the synthetic proteins.

Figure 1 for sequence HPHPPHHPHPPH
Figure 2 for sequence HHPPHPHPH

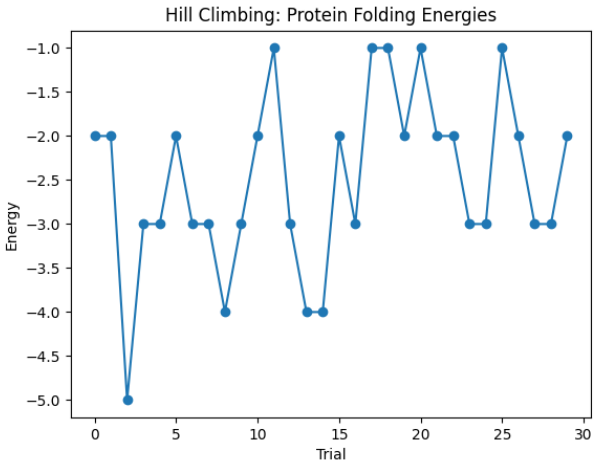

Fig. 1. Energy pattern over iterations for first sequence.

Figure 3 shows the lattice generated by hill climbing algorithm for the sequence "HPHPPHHPHPPHPHHPPHPH".

The Python implementation of the Hill Climbing algorithm demonstrated efficient convergence behavior on sequences

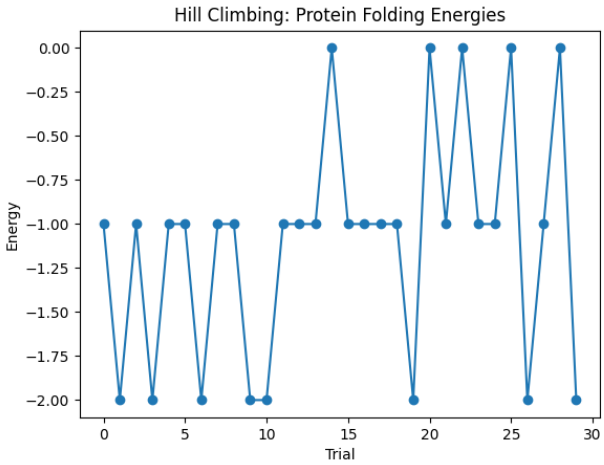

Fig. 2. Energy pattern over iterations for second sequence.

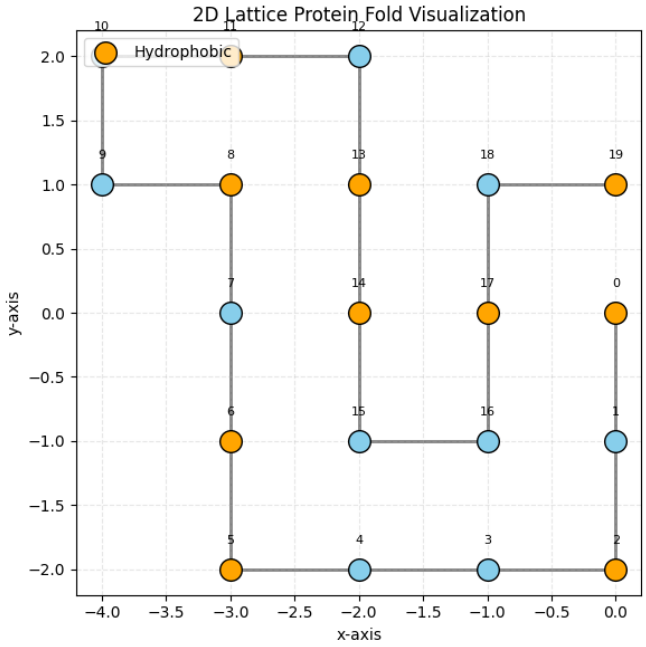

Fig. 3. Predicted Protein folding.

of lengths ranging from 15 to 100 residues. Each iteration modifies only a single move within the relative direction sequence (L, F, R), enforcing self-avoidance to ensure geometric validity. The energy function correctly captures hydrophobic contacts through neighbor-based adjacency checks. Empirically, runtime scales approximately linearly with sequence length and the number of iterations, consistent with the theoretical $O(I \cdot n)$ complexity. Visualization of the lattice structures confirms biologically plausible folding patterns, with hydrophobic residues clustering to form a compact core. The implementation thus provides both algorithmic correctness and interpretability suitable for small-scale protein folding

simulations.

X. Discussion

The experimental results demonstrate that the Hill Climbing algorithm can effectively identify low-energy conformations for short HP sequences using a simple local search strategy. The observed convergence curves in Figures 1 and 2 exhibit a rapid initial decrease in energy during the early iterations, followed by stagnation as the algorithm settles into a local optimum. This behavior is characteristic of greedy search methods that accept only strictly improving moves.

For the sequence `HPHPPHHPHPPH`, the algorithm consistently achieved energies in the range of -3 to -5 across multiple runs, indicating the discovery of compact structures with well-formed hydrophobic cores. In contrast, the shorter and more polar sequence `HHPPHPHPH` yielded higher (less negative) energies, reflecting the reduced potential for hydrophobic interactions. The variability in final energies among runs also suggests that different random initial conformations lead to distinct local minima within the energy landscape.

The lattice visualizations (Figure 3) further support the physical plausibility of the predicted folds. Hydrophobic residues (colored orange) tend to cluster in the interior regions, forming a hydrophobic core, while polar residues (colored blue) remain on the periphery—an emergent property that mirrors the hydrophobic collapse observed in real protein folding. This demonstrates that even in a simplified 2D model, the algorithm captures meaningful biophysical behavior.

However, several limitations are apparent. The deterministic nature of the Hill Climbing algorithm restricts exploration once a local optimum is reached. Since no uphill (energy-increasing) moves are accepted, the search often terminates prematurely without reaching the global minimum. This limitation is visible in the plateau regions of the energy plots, where improvement ceases after relatively few iterations. Moreover, the algorithm's performance is sensitive to the initial conformation; suboptimal starting folds may trap the search in poor-quality basins.

To address these issues, future extensions could incorporate stochastic acceptance criteria or metaheuristic enhancements such as Simulated Annealing or Tabu Search, which probabilistically allow non-improving moves to escape local minima. Another promising direction is to hybridize Hill Climbing with population-based methods like Genetic Algorithms, enabling parallel exploration of multiple regions of the energy landscape. Additionally, extending the model to a 3D lattice would improve biological realism and allow study of tertiary structure formation.

Overall, the experiments confirm that the Hill Climbing approach offers a computationally efficient baseline for protein structure prediction in simplified models. It successfully demonstrates the interplay between sequence composition, hydrophobic clustering, and the emergent folding patterns seen in real proteins, while also highlighting the trade-off between search efficiency and global optimality in greedy algorithms.

XI. Conclusion

This study applied the Hill Climbing algorithm to the 2D HP lattice model for protein structure prediction. The algorithm demonstrates rapid convergence and simplicity but is limited by local optimality. Future work will extend this approach to 3D lattices and hybrid stochastic methods.

References

[1] K. A. Dill, "Theory for the folding and stability of globular proteins," *Biochemistry*, vol. 24, no. 6, pp. 1501–1509, 1985.

[2] R. Unger and J. Moult, "Genetic algorithms for protein folding simulations," *Journal of Molecular Biology*, vol. 231, pp. 75–81, 1993.

[3] P. Crescenzi et al., "On the complexity of protein folding," *Journal of Computational Biology*, vol. 5, pp. 423–466, 1998.

[4] Y. Zhang, "Progress and challenges in protein structure prediction," *Current Opinion in Structural Biology*, vol. 18, no. 3, pp. 342–348, 2008.

[5] R. Bogulla, "Protein Structure Prediction using Hill Climbing on 2D HP Lattice Model," GitHub repository, 2025. [Online]. Available: https://github.com/Kamehamehaaaaa/Protein-2D-Structure-Prediction

XII. LLM References

1) Prompt: "how can genetic algorithm be proposed as a Greedy algorithm" Response: "While GAs are not inherently greedy, you can design or interpret them in a greedy manner under certain conditions. A genetic algorithm is a metaheuristic inspired by natural selection. If the GA keeps the best individual from each generation (elitism) and new individuals are accepted only if they improve the fitness, it becomes equivalent to a greedy hill-climbing algorithm."
2) Prompt: "Can hill climbing algorithm for protein structure prediction a Greedy Algorithm?" Response: "Excellent question — and yes, hill climbing in the context of protein structure prediction (PSP) is indeed a greedy algorithm in nature. The hill climbing algorithm is a local search optimization method that iteratively moves from one solution to a neighboring one if that neighbor improves the objective (or decreases the energy, in PSP). Hill climbing is greedy because at every step it chooses the best immediate move"
3) Prompt: "Given the results of the experiments improve my discussion section of report. Mention about the role of random initialization in getting better energy. Mention about using the Genetic Algorithm for better initialization and exploration. Mention about simulated Annealing to escape local minima. Mention about the zipper core like structure in the protein folding figure." Response: Certain sections of the Discussion section are derived from the response from LLM.
4) Certain sections of the report were enhanced by providing the initial structure and asking LLM to make it sound better.