

# Parallel Ant Colony Optimization for 3D Protein Structure Prediction using the HP Lattice Model

D. Chu, M. Till, A. Zomaya  
School of Information Technologies  
Madsen Building, F09  
The University of Sydney  
Sydney 2006, NSW Australia

## Abstract

*The Protein Folding Problem studies the way in which a protein - a chain of amino acids - will 'fold' into its natural state. Predicting the way in which various proteins fold can be fundamental in developing treatments of diseases such as Alzheimers and Systic Fibrosis. Classical solutions to calculating the final conformation of a protein structure are resource-intensive. The Hydrophobic-Hydrophilic (HP) method is one way of simplifying the problem. We introduce a novel method of solving the HP protein folding problem in both two and three dimensions using Ant Colony Optimizations and a distributed programming paradigm. Tests across a small number of processors indicate that the multiple colony distributed ACO (MACO) approach is scalable and outperforms single colony implementations.*

## 1 Introduction

Proteins are complex molecules that consist of a chain of amino acids. They are essential to the structure and function of all living cells and perform functions such as acting as a catalyst for biomedical reactions, as structures of cells and receptors for hormones. By understanding how proteins achieve their native three dimensional structure, we can also help develop treatments for diseases such as Alzheimers and Cystic Fibrosis (diseases that cause proteins to change their native conformation).

There have been many developments in the field of protein folding. There is still much work to be done in simulating proteins with many chains. Computations of this kind still remain infeasible with current processing technology. The Hydrophobic-Hydrophilic (HP) lattice based protein model simplifies the problem whilst maintaining behavioural relevance. Until recently the 3D HP model has been largely ignored in favour of the 2D HP model due to the

simplicity of the search space and easier visualization.

Simplified HP models have been proven to be NP-Complete [1] thus are perfect for evaluating and improving heuristic based algorithms that will assist future development of expanded protein folding problems.

Much of the previous work in this field focuses on developing and implementing solutions to 2d protein folding without considering 3D implementations. We propose an algorithm that can solve the 3D HP protein folding problem by extending a solution to the 2D HP problem. We use an algorithm that suits loosely coupled distributed programming paradigms.

## 2 Background

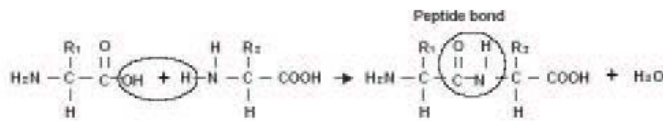
### 2.1 The Protein Folding Problem

The Protein Folding Problem (or the Protein Structure Prediction Problem) is defined as the prediction of the 3D native structure of a given protein from its amino acid chain (primary structure). The native structure or state is the 3D structure that proteins vibrate around when equilibrium is formed almost immediately after protein creation. Experiments by Anfinsen et al in 1961 showed that a protein in its natural environment will fold into its native state regardless of the starting confirmation.

The native structure of proteins is classically determined by techniques such as Nuclear-magnetic resonance imaging (NMRI) and xray / electron crystallography which are expensive in terms of computation , time and equipment.

### 2.2 Protein Structures

Proteins are polymer chains of amino acids. Each amino acid consists of a central carbon atom and four connecting bonds, namely a hydrogen atom, a carboxylic acid group, an amino group and a side chain. The carboxylic acid (COOH)



**Figure 1. Two amino acids forming a peptide chain**

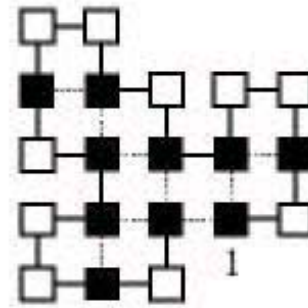
and the amino group ( $\text{NH}_2$ ) can dissociate or accept a hydrogen ion respectively, thus allowing the amino acid to act either as acid or base. The side chain also known as the R-Group acts as the distinguishing feature of the amino acid such that all twenty amino acids found in proteins have unique side chains.

During protein synthesis, amino acids are joined end-to-end by the formation of peptide bonds when the carboxyl group of one amino acid condenses with the amino group of the next to eliminate water. This process is repeated as the chain elongates. The amino group of the first amino acid and the carboxylic group of the last amino acid in the polypeptide chain remains unchanged and is named the amino terminus and carboxyl terminus respectively. In biology, protein structures are usually described in four levels in order to reduce complexity. The amino acid sequence of a protein chain is called the primary structure. Different regions of the sequence form local secondary structures such as alpha helices and beta strands. The tertiary structure is formed by packing such structure into one or several compact globular units called domains. Tertiary structure can also be regarded as the full 3D folded structure of the polypeptide chain. The final quaternary structure describes the interconnections and organization of multiple peptide chains and therefore is only used when there is more than one polypeptide chain. It is widely believed that secondary, tertiary and quaternary structures of the proteins native state can be determined by the primary structure.

### 2.3 Hydrophobic-Hydrophilic Lattice Model

The Hydrophobic-Hydrophilic (HP) Lattice Model [4,7] is a NP-Complete [1] free energy model that is motivated by a number of well known facts about the pivotal role of hydrophobic and polar amino-acids for protein structure -

1. Hydrophobic interaction is the driving force for protein folding and the hydrophobicity of amino acids is the main force for development of a native conformation of small globular proteins.
2. Native structures of many proteins are compact and have well-packed cores that are highly enriched in the



**Figure 2. A sample protein conformation in the 2D HP Model. The black squares represent Hydrophobic (H) residues. Dashed lines indicate the non-adjacent H residue contacts. The 1 indicates a terminating residue.**

hydrophobic residues as well as minimal solvent exposed non-polar surface areas. [7,11]

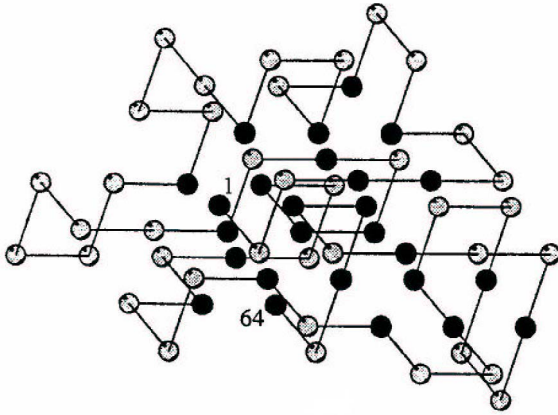
In the HP model the amino acid sequence is abstracted to a sequence of hydrophobic (H) and hydrophilic (P) residues. The protein conformations of this sequence are then restricted to self-avoiding paths on a lattice; for our purpose the 2D HP model considered here consists of a 2D square lattice and the 3D HP model consists of a 3D cube lattice.

Based on the biological motivation given above, the energy of a conformation is defined as a number of topological contacts between hydrophobic amino-acids that are not neighbors in the given sequence. Specifically a conformation  $c$  with exactly  $n$  such contacts would have an energy value of  $E(c) = n(-1)$ .

The HP Protein Folding Problem can be formally defined as follows - Given an amino-acid sequence  $s = s_1 s_2 s_3 \dots s_n$ , find an energy minimizing conformation of  $s$ , i.e., find  $c^* \in C(s)$  such that  $E^* = E(c^*) = \min\{E(c) \mid c \in C\}$ , where  $C(s)$  is set of all valid conformations for  $s$  [12].

### 2.4 Existing Algorithms for the HP Lattice Model

Over the years several well known heuristic based optimization algorithms have been applied to solve protein folding in the HP lattice model. This includes Evolutionary algorithms (EAs) and Monte Carlo (MC) algorithms. Ant Colony Optimization has been used in 2D HP lattice models [12]. Tabu searching (Hill climbing optimizations) has been combined with GAs. Three dimensional lattice models have also been used with success. None of these solutions have focused on distributed programming environments. With the assembly of computing systems with



**Figure 3. A sample protein conformation in the 3D HP Model. The black squares represent Hydrophobic (H) residues. Dashed lines indicate the non-adjacent H residue contacts. The 1 indicates a terminating residue.**

1. Initialize pheromone trails
2. While optimal solution not found do
  - (a) Construct candidate solutions
  - (b) Perform local search
  - (c) Update pheromone trails
3. Return best found solution

**Figure 4. Single process Ant colony algorithm**

hundreds and even thousands of processors it is advantageous to harness this distributed processing power.

### 3 Ant Colony Optimization

Ant Colony optimization (ACO) is a class of natural algorithms inspired by the foraging behavior of ant colonies. First proposed [8] as an approach to difficult optimization problems such as TSP [5], ACO has since been applied to many other problems [2][3][6][9]. The algorithm mimics the behavior of a colony of ants trying to find the shortest path between nest and food source by breaking down the problem into a shortest path problem and creating a colony of artificial ants to search for the shortest path (which is the optimal solution).

It is well known that biological ants form and maintain paths primarily through creation of a pheromone

trail. Whilst traveling, ants deposit a certain amount of pheromone trail and probabilistically chooses the direction richest with pheromone. Because shorter paths between the nest and food would be reached faster, the pheromones of the shorter path would also be stronger, resulting in other ants being more likely to choose the shorter path thus slowly eliminating the longer path.

The behavior of the artificial ants in ACO is similar to that of biological ants. In each iteration of the algorithm each artificial ant acts as an agent traversing the graph in order to construct a higher quality solution. At the end of each iteration the quality of all candidate solutions are evaluated based on a heuristic function and the top solutions are then employed to update the pheromone matrix.

In general ACO uses the following three ideas from natural ant behavior -

1. The preference for solutions or parts of solution with a higher pheromone level
2. The higher rate of growth of the amount of pheromone on better solutions
3. Trail mediated communication amongst ants

#### 3.1 Pheromone Matrix

The pheromone matrix is how ACO keeps track of the pheromone values of given routes. All values within the matrix are initially set to zero, and the matrix is updated with pheromone values of the better paths with each iteration of the algorithm. The amount of pheromone added to the matrix is usually dependent on the quality of the ant, therefore proportional to the objective function value of a given solution. It is also possible to multiply a constant  $\rho$  (where  $0 < \rho < 1$ ) to the matrix every iteration to simulate evaporation of the pheromone trail when a path is not used.

#### 3.2 Local search

Many applications of ACO include a local search element as a means of by-passing local minima and preventing the algorithm converging too quickly. Usually algorithms utilized for local search are mutations of the acquired solutions or problem specific optimizations such as 2-opt for TSP.

#### 3.3 Population based ACO

Rather than retaining a pheromone matrix at the end of the iteration, a population of solutions is kept. At the start of each iteration the population of solutions from previous iterations are used to construct the pheromone matrix which is then used to create the population at the next iteration. This

allows the combination of ACO and other population based algorithms such as genetic and evolutionary algorithms.

### 3.4 Multi ACO (MACO)

Multi colony algorithms (MACOS) utilize multiple colonies of artificial ants. This approach utilizes separate pheromone matrices for each colony and allow limited cooperation between different colonies. MACOS are especially suited for distributed computing due to the limited amount of information exchange and synchronization required. Methods of information exchange include -

1. Exchange of the global best solution every  $n$  iterations, where the best solution is broadcast to all colonies and becomes the best local solution for each colony
2. Circular exchange of best solutions every  $n$  iterations. A virtual neighborhood is established such that colonies form a directed ring structure. Every  $n$  iterations every colony sends its best local solution to the successor colony in the ring. The best local solution is updated accordingly.
3. Circular exchange of  $m$  best solutions every  $n$  iterations. The colonies form a virtual directed ring. Every  $n$  iterations every colony compares its  $m$  best ants with the  $m$  best ants of its successor in the ring. The best  $m$  ants are allowed to update the pheromone matrix.
4. Circular exchange of the best solution plus  $m$  best local solutions every  $n$  iterations.

## 4 Distributed Algorithms

Various distributed programming paradigms are suitable for both single and multiple ant colony implementations for the HP protein folding problem.

### 4.1 Centralized periodic update

This model suits the controller/worker paradigm whereby the worker processors are given a set of paths to evaluate. After evaluating these paths, the workers return them to the master who is responsible for co-ordinating the experiment.

### 4.2 Round Robin - single colony

A federated system with no single controller - every processor works on its own local solutions and shares the best solution to a single neighbor in a ring topology.

### 4.3 Round Robin - Multiple colonies

Every processor has its own pheromone matrix and separate colony of ants. At the end of each iteration a processor will share its best solution with one neighbor in the ring.

### 4.4 Round Robin - Multiple colonies multiple updates

Separate colonies of each processor and multiple updates of solutions per iteration.

## 5 ACO Implementation

The code was written in the C programming language using the Message Passing Interface (MPI) toolkit to achieve interprocess communication.

### 5.1 Construction phase

In the construction phase, each ant randomly selects a starting point within the given protein sequence. The sequence is then folded in both directions one amino acid at a time. The probability of extending the solution in each direction is equal to the number of unfolded amino acids in the respective direction divided by the total number of unfolded residues. This method encourages both directions of the chain completely folding within a few construction steps of one another.

In each construction step the relative direction is determined probabilistically using heuristic value  $\eta_{i,d}$  and the pheromone values  $\tau_{i,d}$  where  $d$  is the relative direction of folding at position  $i$  of the protein sequence.

When the conformation is extended in the reverse direction (from  $i$  to  $i - 1$ ) the pheromone values  $\tau_{i,d}$  and heuristic  $\eta_{i,d}$  are used. In our implementation we define  $\tau_{i,L} = \tau_{i,R}$ ,  $\tau_{i,R} = \tau_{i,L}$ ,  $\tau_{i,S} = \tau_{i,S}$ ,  $\tau_{i,U} = \tau_{i,U}$ ,  $\tau_{i,D} = \tau_{i,D}$  and analogous equalities for the respective  $\eta$  values. This reflects the symmetry of travelling in opposite directions.

Once  $\eta_{i,d}$  is calculated for the current amino acid the relative direction  $d$  of  $s_{i+1}$  with respect to  $s_i$  and  $s_{i-1}$  is determined according to

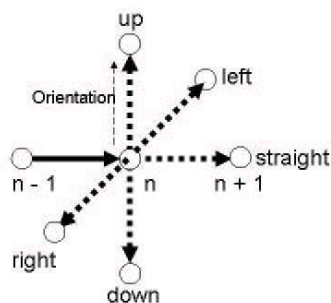
$$p_{i,d} = \frac{[\tau_{i,d}]^\alpha [\eta_{i,d}]^\beta}{\sum_{e \in \{L,R,S,U,D\}} [\tau_{i,e}]^\alpha [\eta_{i,e}]^\beta}$$

### 5.2 Heuristics

The heuristic value  $\eta_{i,d}$  guides the construction process towards high quality solutions. In our implementation  $\eta_{i,d}$  is defined as  $\eta_{i,d} = h_{i+1,d} + 1$  where  $h$  is the number of new

1. for each ant do
  - (a) Randomly select starting positions
  - (b) while (candidate solution not complete) do
    - i. Select next amino acid
    - ii. Calculate possible directions and heuristics
    - iii. if (no possible direction found) backtrack; else select next direction
2. Return list of conformations

**Figure 5. ACO construction phase**



**Figure 6. Relative direction system for the candidate solution**

$H - H$  contacts achieved by placing  $s_{i+1}$  in the direction  $d$  relative to  $s_i$  and  $s_{i-1}$  when folding forward. Since only  $H - H$  bonds contribute to the heuristics,  $h_{i+1,d} = 0$  if the current amino acid is a  $P$  molecule and therefore  $1 < i < n - 1$ ,  $1 \leq \eta_{i,d} \leq 5$  and  $1 \leq \eta_{n,d} \leq 6$ .

### 5.3 Coding

As in other work [12] candidate conformations are represented through relative directions straight, left, right, up and down ( $S, L, R, U$  and  $D$ ) for the 3D lattice. Each direction in the candidate conformations indicates the position of the next amino acid relative to the direction projected from the previous to the current amino acid. The number of directions required for a conformation of size  $n$  is  $n - 2$ . An orientation value is also required to determine the upward direction at a given amino acid. During the construction phase this indicator for the upward direction is stored.

### 5.4 Local search

Local search is implemented similar to previous work [12] where we initially select a uniformly random position

within a candidate solution and randomly change the direction of that particular amino acid.

### 5.5 Pheromone updates

Selected ants update the pheromone values by the following -

$$\tau_{i,d} = (1 - \rho)\tau_{i,d} + E(c)/E^*$$

$\rho$  is the pheromone persistence that determines how much pheromone evaporates each iteration of the algorithm,  $E(c)/E^*$  is the relative solution quality of the given candidate conformation where  $E^*$  is the known minimal energy for the given protein. If this value is unknown an approximation is calculated by counting the number of  $H$  residues in the sequence. This technique prevents stagnation of solutions and ensures that lesser quality candidate solutions contribute proportionally lower amounts of pheromone.

## 6 Implementations

We constructed four implementations of our 3D ACO - a reference implementation and three distributed implementations. The distributed models both use master / slave paradigms. The solution was assembled to report the number of cpu ticks that the program's master process took to find an improved solution as well as the score associated with that conformation.

### 6.1 Single process single colony

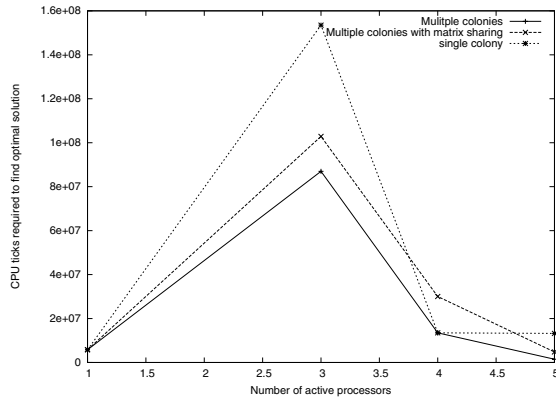
The reference implementation which uses a single processor, single colony and single pheromone matrix. Every distributed implementation would function in this fashion if it was to be run on a single processor.

### 6.2 Distributed Single Colony

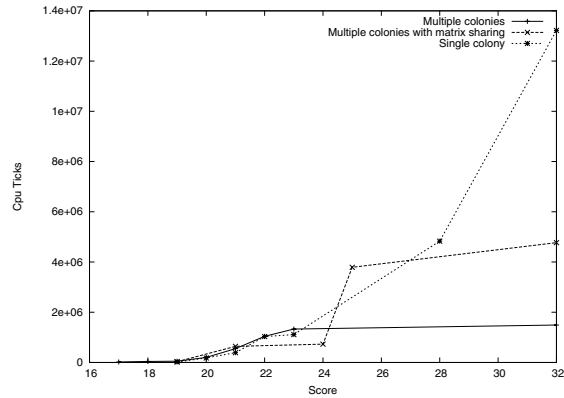
At end of construction and local search phases, all client systems transfer selected conformations to update the centralized pheromone matrix and receive a copy of the updated pheromone matrix.

### 6.3 Distributed Multi Colony with Circular exchange of migrants

All pheromone matrices are stored within the master process, every iterations at end of construction and local search phases the client transmits selected conformations for pheromone updates and receives a updated pheromone matrix. For every  $n$  iterations for each colony, their neighbouring colony is also updated.



**Figure 7. Optimal solution cpu ticks vs number of active processors for each implementation**



**Figure 8. Optimum solution score vs cpu ticks for 5 processors for each implementation**

#### 6.4 Distributed Mult Colony with pheromone matrix sharing

Every  $n$  iterations counted on the server, each of the pheromones matrices are updated by the following formula:

$$\tau_{C,i,d} = \tau_{C,i,d} + \frac{(\sum_{n=1}^{Ctotal} \tau_{n,i,d} - \tau_{C,i,d})\gamma}{Ctotal - 1}$$

### 7 Results

The program was compiled and run on an IBM Blade center comprising of 9 nodes. Each node comprised of 2 2.4 Ghz Intel processors with 1 Gbyte of shared ram. the blade center contained an extremely fast dedicated interconnect. We used lam-mp 1.2 for our implementation of MPI.

The first implementation was run on a single processor. The distributed implementations were run on more than 2 processors due to the nature of their master/slave components. Tests were run on a protein sequence obtained from the HP Protein folding benchmark site [13]. The execution time regards how much wall clock time the executions took until either no more optimal solutions were found or the optimal solution was equal to the best known score for that protein sequence.

Due to the master / slave implementation, we did not test two processors - the distributed implementation would function the same as the single processor version.

### 8 Discussions and Future Work

The single processor implementations would not find the optimal solution in all cases. This was to be expected considering the small search space that the single processor

algorithm covers. In running single processor implementations we terminated executing the test once no further improvements in the solutions were found. Both Multiple colony implementations outperformed the single colony implementation across 5 processors by a large margin. The tests run across five processors for multiple colonies resulted in execution times of less than a second - we did not test on more processors in these cases.

We have developed a framework for testing a variable number of colonies for ACO for the HP protein folding problem. We have shown that good 2D solutions for this problem can be extended to the 3D case. Our solution works well across a number of processors. Multiple colony solutions offer a definite improvement over the single colony implementation. We hope to harness other properties of ACOs by extending our solution to work across loosely coupled distributed systems such as grids.

### References

- [1] Bonnie Berger and Tom Leighton. Protein folding in the hydrophobic - hydrophilic (HP) model is np-complete. *Journal of Computational Biology*, 5(1):27-40,1998.
- [2] R.F. Hartl Bullnheimer D and C Strauss. *Ant Colony Optimization in Vehicle Routing*. PhD thesis, University of Vienna, 1999.
- [3] Costa D. and A. Hertz. Ants can Colour Graphs. *Journal of Operational Research Society*, 48:295-305, 1997.
- [4] K.A.Dill. *Biochemistry*. 1985

- [5] Dorigo M.L.M Gambardella. Ant Colonies for the Travelling Salesman Problem. *BioSystems*, 43:73-81, 1997.
- [6] E.Taillard Gambardella L.M and M.Dorigo. Ant colonies for the quadratic assignment problem. *Journal of the Operational Research Society*, 50:167-176, 1999
- [7] K.F.Lau and K.A.Dill. A lattice statistical mechanics model of the conformation and sequence space for proteins. *Macromolecules*, 22, 1989.
- [8] Dorigo M. Optimization, Learning and Natural Algorithms. PhD thesis, Politecnico di Milano, 1992.
- [9] Gambardella L.M and M. Dorigo. Has-sop: An hybrid ant system for the sequential ordering problem. Technical Report IDSIA 97-11, Lugano, Switzerland , 1997
- [10] H. Schmeck M.Middendorf, F. Reischle. Information exchange in multi colony ant algorithms. *Parallel and Distributed Computing: Proceedings in the Fifteenth IPDPS Workshops 2000*, 2000
- [11] F.M Richards. Areas, volumes, packing and protein structures. *Annu. Rev. Biophys. Bioeng.*:6:151-176,1977.
- [12] A Shmygelska and H.H. Hoos. An improved ant colony optimization algorithm for the 2d HP protein folding problem. *Canadian Conference on AI 2003*, pages 400-417, 2003.
- [13] W. Hart, S. Istrail. HP Benchmarks [http://www.cs.sandia.gov/tech\\_reports/compbio/tortilla-hp-benchmarks.html](http://www.cs.sandia.gov/tech_reports/compbio/tortilla-hp-benchmarks.html)  
last accessed October 2004