Proceedings of the International MultiConference of Engineers and Computer Scientists 2008 Vol I
IMECS 2008, 19-21 March, 2008, Hong Kong

A Hybrid Population based ACO Algorithm for Protein Folding

Torsten Thalheim, Daniel Merkle, Martin Middendorf, *

Abstract—**A hybrid population based Ant Colony Optimization (ACO) algorithm PFold-P-ACO for protein folding in the HP model is proposed in this paper. This is the first population based ACO algorithm in the bioinformatics. It is shown experimentally that the algorithms achieves on nearly all test sequences at least comparable results to other state of the art algorithms. Compared to the state of the art ACO algorithm PFold-P-ACO slightly better results and is faster on long sequences.**

Keywords: Population based ACO, Ant Colony Optimization, HP model, Protein Folding

1 Introduction

Proteins are one of the most important classes of biological molecules. Chemically, a protein is a chain where each element is one of only 20 different amino acids. Each amino acid consists of a central carbon atom bonded to an amino group (NH2), a carboxyl group (COOH) and a side chain or residue (R) Hence, the amino acids differ only in the residue R. One of the most important differences between the residues is their hydrophobicity, i.e., how much they are repelled from a mass of water. The properties of the residues together with the environment are responsible that the protein chain folds into a complex conformation. This conformation is called the "native" conformation of the molecule. The native conformation is thermodynamically stable, i.e., it has small Gibbs free energy, and is very important for the function of the protein.

The structure of a protein can be described on different levels: the amino acid sequence is the primary structure, the secondary structure describes characteristic structures of the backbone of the molecule within local regions (e.g., alpha-helices or beta-sheets), the tertiary structure refers to the entire 3-dimensional structure. Different types of algorithms have been developed to predict the tertiary or secondary structure of proteins. All these algorithms use a model that is an abstraction of real proteins and describes important characteristics. An important class of models are the lattice models. A lattice model consists of a lattice that describes possible positions for the amino acids and an energy function that is to be minimized and depends on the positions of the amino acids on the lattice. The most simplest lattice model is the HP model which is based on the observation that hydrophobic forces are very important factors that drive the protein folding process. Advantages of the HP model are simplicity, that it shows several aspects of real proteins, and remains the hardness features of the biological problem.

In this paper we propose a P-ACO algorithm called PFold-P-ACO for solving the protein folding in the HP model. PFold-P-ACO is the first P-ACO algorithm for the problem domain of bioinformatics.

Section 2 describes the HP model and mentions some heuristics form the literature for the protein folding problem in the HP model. An introduction to ACO and ACO approaches for the protein folding problem is given in Section 3. Population based ACO and our algorithm PFold-P-ACO are described in Section 4. The experiments and the results are presented in section 5. Conclusions are given in Section 6.

2 The HP model

The HP model is introduced by Dill [5, 11]. It is based on the fact that for folding the most important difference between amino acids is their hydrophobicity, i.e., how much they are repelled from a mass of water. The reason is that hydrophobicity is the main driving force to fold a molecule into the native conformation (at least for of small globular proteins). In the HP model all 20 different amino acids are classified into two types: hydrophobic or non-polar (H) and hydrophilic or polar (P).

A primary structure with n amino acids is viewed as a sequence $S = s_1, \ldots, s_n$ with $s_i \in \{H, P\}$ for $i = 1, \ldots, n$. A conformation is a mapping C of the amino acids s_i to the points of a cartesian lattice. Two and three dimensional cartesian lattices are used here. In the following we describe the 2-dimensional model. The definitions for the 3-dimensional model are analogous. We use the following notation: if C is a conformation then (x_i, y_i) denotes the position in the lattice to which s_i is mapped by C. All valid conformations are self-avoiding paths on the carte-

*Department of Computer Science, University of Leipzig, Postfach 100920, D-04009 Leipzig, Germany, Email: torsten.thalheim@ufz.de, {merkle,middendorf}@informatik.uni-leipzig.de, send correspondence to M. Middendorf

sian lattice. A mapping is a path when amino acids s_i, s_j that are consecutive in the molecule, i.e., $|i-j|=1$, are mapped to neighbored positions (x_i, y_i), (x_j, y_j) on the lattice, i.e., $|x_i - x_j| + |y_i - y_j| = 1$. A path is self-avoiding when all two different amino acids s_i, s_j, $i \neq j$ are mapped to different positions, i.e., $(x_i, y_i) \neq (x_j, y_j)$.

The energy function in the HP model reflects the fact that the hydrophobic amino acids have a propensity to form a hydrophobic core. Therefore, the energy function adds a value -1 for every pair of hydrophobic amino acids (H) that are adjacent on the lattice but not consecutive in the sequence. Formally, the energy $E(f)$ of a conformation C is $\sum_{1 \leq i \leq j-2 \leq n} I(i,j)$ where $I(i,j) = -1$ if $|x_i - x_j| + |y_i - y_j| = 1$ and $I(i,j) = 0$ otherwise.

The protein folding problem in the HP model — called HP-Protein Folding problem — is to find for a given protein $S = s_1 \ldots s_n$, $s_i \in \{H, P\}$ a valid conformation C on the cartesian lattice such that the energy $E(f)$ is minimum. In [3] it was shown that the HP-Protein Folding problem is NP hard, i.e., it is very unlikely that there exists a polynomial time algorithm for solving the problem. Therefore, it is interesting to find heuristics for solving the HP-Protein Folding problem.

The variety of heuristics that have been developed include (Metropolis) Monte Carlo algorithms (e.g.,[21]), chain growth algorithms (e.g.,[10]), evolutionary algorithms (e.g., [20]), memetic algorithms, immune algorithms, and ACO algorithms (e.g., [16, 17]). Due to limited space an overview on other algorithms can not be given (for recent overviews see [17, 22]).

3 Ant Colony Optimization

Ant Colony Optimization (ACO) is a metaheuristic that is inspired by the foraging behaviour of real ants ([6]). ACO has been applied successfully to solve various combinatorial optimization problems (see [7, 13]). Shmygelska and Hoos proposed several variants of an ACO algorithm for HP-Protein Folding problem (e.g., [16, 17]). The latest algorithm ACO-HPPFP-3 iterates over the following three phases: construction phase, local search phase, pheromone update phase. In the construction phase each ant constructs a candidate solution by sequentially growing a conformation of the given HP sequence, starting from a folding point that is chosen uniformly at random among all sequence positions. Since conformations are rotationally invariant, the position of the first two amino acids can be fixed without loss of generality. A candidate conformation for a HP sequence of length n corresponds to a decision sequence of length $n-2$. Each decision indicates the position of an amino acid on the 2D or 3D lattice relative to its direct predecessors in the given sequence. Possible decisions are whether the chain folds straight (S), left (L), right (R) in 2D, (and also up (U), down (D) in 3D). Each ant performs probabilistic chain-growth construction, where in every step, the structure is extended either to the left or to the right, such that the ratio of unfolded residues at each end of the protein remains (roughly) unchanged. The relative direction $d \in \{S, L, R\}$ in which the conformation is extended in construction step i is determined probabilistically based on a heuristic function $\eta_{i,d}$ and pheromone values $\tau_{i,d}$ according to the formula: $p_{i,d} = (\eta_{i,d}^{\alpha} \cdot \tau_{i,d}^{\beta}) / \sum_{e \in \{S,L,R\}} (\eta_{i,e}^{\alpha} \cdot \tau_{i,e}^{\beta})$. The pheromone values $\tau_{i,d}$ indicate the desirability of using direction d at sequence position i. Initially, all $\tau_{i,d}$ values are equal. Throughout the search process, the pheromone values are updated to bias the folding towards the use of local directions that occur in low-energy structures.

In the pheromone update phase each pheromone value $\tau_{i,d}$ is evaporated according to $\tau_{i,d} := \rho \cdot \tau_{i,d}$ where $\rho < 1$ is the pheromone persistence parameter. Subsequently, selected ants with low-energy conformations update the pheromone values according to $\tau_{i,d} := \tau_{i,d} + \Delta_{i,d,c}$ where $\Delta_{i,d,C}$ is the relative solution quality of the given ant's candidate conformation C if that conformation contains local direction d at sequence position i, and zero otherwise. For further details and information on the following parts of the see [17]: i) the heuristic method, ii) the backtracking method that is used when a direction is not possible because the chain would run into itself, iii) the local search method.

4 P-ACO and PFold-P-ACO

One of the main characteristics of an ACO algorithm is the pheromone information which stores information on good solutions that have been found by ants of former iterations. The pheromone information is what is transferred from one iteration of the algorithm to the next. An alternative to this scheme has been proposed by Guntsch and Middendorf [8] and is called Population based ACO (P-ACO). Instead of pheromone information as in ACO, in P-ACO a population of solutions is transferred from one iteration of the algorithm to the next. The ants in the new iteration use this population to construct pheromone information from it and then proceed as in standard ACO. Instead of pheromone update P-ACO uses a population update and several strategies have been proposed for a solution to enter or leave the population (see [8]). Two potential advantages of P-ACO compared to standard ACO are: i) the population update and pheromone construction needs for typical applications (e.g., for permutation problems like the Traveling Salesperson problem) time $O(n)$ where n is the problem size instead of time $O(n^2)$ that is necessary for standard ACO pheromone update, ii) the population can be used to apply operations on the solutions (e.g. crossover). A potential disadvantage of P-ACO compared to ACO is that the number of different pheromone values is small (typically, the population size) whereas for standard ACO

it is potentially infinite. Thus, a P-ACO algorithm might be faster than a standard ACO algorithm but it not clear whether it can achieve the same solution quality.

It has been shown experimentally that P-ACO works equally good as ACO even when the P-ACO algorithm uses only a small population ([8]). It was also shown that P-ACO can be used for multi-objective problems [1, 9, 4]. Since P-ACO has been tested on classical problem like TSP ([1, 2, 8, 9, 4]) or single machine scheduling problems [9] only it is interesting to apply it to other problem domains.

In the following subsections we describe our hybrid P-ACO algorithm called PFold-P-ACO. It consists of two parts: a P-ACO part and a branch-and-bound part. When the P-ACO part has not found an improvement over a certain number of iterations, the branch-and-bound part starts. The branch-and-bound part is a heuristic that does not do a complete enumeration.

4.1 ACO Part

The population that is used by PFold-P-ACO contains always the best 10 conformations that have been found. But to keep enough diversity a new found conformation has the same HH-contacts as a conformation that is already in the population is not allowed to enter the population. If a new found conformation has the same energy as a conformation in the population it replaces the one in the population with probability 0.5. Construction Phase and Local Search Phase are described in the following subsections.

Construction Phase. Similar as in ACO-HPPFP-3 from [17] each ant constructs a solution by sequentially growing a conformation of the given sequence, starting from an element that is chosen uniformly at random. Different from algorithm ACO-HPPFP-3 in PFold-P-ACO the probability that the next element to be placed is chosen in direction of the beginning or end of the sequence equals the relative length of the remaining prefix respectively suffix of the sequence. If it possible that a P subsequence can be extended or a prescribed HH-contact can be realized it is done.

The pheromone values $\tau_{i,d}$ $i = 1, \ldots, n-1$, $d = S, L, R$ are initialized with value 1. Each ant chooses randomly a conformation from the population and sets all corresponding pheromone values to 4. As heuristic values the ants use the change of the energy ΔE of the partial conformation that occurs when a decision is made, i.e., $\lambda_{i,d} = e^{\Delta E}$. An ant that has to decide how to fold at s_{i+1} makes with probability 0.05 a decision randomly with equal probability for S, L, or R (if all are possible). Otherwise, the ant considers the pheromone values and decides for direction d with probability $\tau_{i,d}^{\alpha} \cdot \lambda_{i,d}^{\beta} / (\sum_{h \in \{S,L,R\}} \tau_{i,h}^{\alpha} \cdot \lambda_{i,h}^{\beta})$ if all directions are possible where parameters α and β define the relative influence of pheromone and heuristic. If not all directions are possible only the pheromone values corresponding to the allowed directions are taken into account. If for example direction $d = L$ is forbidden by a constraint as described in the following then the probability to fold in direction d is $\tau_{i,d} \cdot \lambda_{i,d} / (\sum_{h \in \{S,R\}} \tau_{i,h} \cdot \lambda_{i,h})$.

Several constraints are used that forbid some decisions for an ant. If the ant can not make an alternative decision it makes a backtrack step and revises the former decision. This is done until the ant finds a decision that is not forbidden or until a Time-to-Live (TTL) counter that counts the number of backtrack steps stops the ant.

Constraint 1. Inspired by an idea from [12] a set of prescribed amino acid contacts is used to guide the search for good conformation. Different from [12] the set $\mathcal{H}$ that is used by the ants in the PFold-P-ACO contains only HH-contacts. Further, set $\mathcal{H}$ contains only local restrictions, i.e., for every HH-contact in $\mathcal{H}$ the distance of the two H elements in the sequence is at most 9. If during the construction phase an ant places the first element of a HH-contact in $\mathcal{H}$ it initializes a vector of counters —- one counter for every moving direction (up, down, left, right). The counters are used to check for every decision in the construction process whether the current element is placed near enough to the location of first element of the required HH-contact so that the HH-contact can still be realized with respect to distance (it is not checked if it is really possible to realize the HH-contact).

Constraint 2. P-rich subsequences make it difficult for ants to find a good conformation because their decisions are guided by forming HH-contacts. Therefore, it is required that the elements of P-rich subsequences are placed near to each other. A P-rich subsequence is defined such before and after it comes an H element and it contains at least 75% percent P elements and does not contain a singleton P element that has no P neighbor. Similar as for the required H-contacts a vector of counters is used for every P-rich subsequence. It is checked when an element of a P-rich subsequences is to be placed that its location would not be too far from the location of the first element of this subsequence. If, the location is too far the ant has to make an alternative decision. For details see [19].

Constraint 3. HP-contacts are not allowed.

Local Search Phase. A local search is used that is similar to the filter-and-fan approach from [15]. The move operator is the pull-move which is initiated by moving one node of the current conformation to one of its empty diagonal adjacent positions in the square induced by the node and one of its adjacent neighbors in the sequence. Depending on the structure of the conformation the displacement of the initiating node may require other nodes to change their positions. In a pull-move, displaced nodes

ID	E_{min}	Sequence
D100-1	-24	$P_2HP_5HPHP_{12}HP_4H_3P(P_2H_2)_2(PH)_3P_4HP_{17}HP_2HP_3H_3PHP_2HP_2(PH)_2P_2H_2P_6H$
D100-2	-42	$H_2P_4H_2(PH_5)_2(PH)_3P_{11}HPH_3P(HP_2)_2H_2P_4H_2PHP_2H_2PHPH_6P_2H4P_3(H_2P)_4P_2HP_3H_4$
D100-3	-52	$P_2H_2P_3HPH_5P_2HPH_{10}PH_2P_2H(P_2H_2)_2P_3HPH_3PH_2(P_2H_3)_2H(PH_3)_2H_2P_3HP_2H(PH_3)_2 HP_2H_2P_4H_2$

Table 1: Test HP sequences D100-x of length 100 with ID, best known energy value E_{min}, and sequence

are only allowed to occupy vacant adjacent positions in the lattice.

The η best conformations from the construction phase are selected as start conformations for local search. To each selected conformation 4 pull moves are applied. From all conformations that have been obtained the 4 best ones are selected for the next iteration of local search. After each iteration of local search it is checked whether a better conformation has been found. If so an iteration counter is set to zero, otherwise the iteration counter is increased by one. If the iteration counter equal 10 the local search procedure is stopped. Restrictions are applied during the local search: i) a conformation is selected only if it has at least 0.7 as much HH-contacts as the so far best found conformation, ii) for each conformation a tabu list that contains the last 5 pull moves is used in order to hinder that pull moves are reversed, iii) a conformation is only accepted if its diameter is at most $(4/3)\sqrt{n}$ or if it has more HH-contacts than the so far best found solution.

4.2 Branch-and-Bound Part

The branch-and-bound process starts with two traces (on starts at s_1 the other at s_n) that work independently but exchange information on the energy of new best conformations. This information is used to estimate whether a partial conformation can potentially reach a new best energy value or should be cut. A mix between breadth first search and depth first search is done. More exactly, the algorithm searches on level l of the tree (breadth first search) until it contains more than 300 nodes. The 150 best of these partial conformations are extended to level $l+1$ and so on. Only when the algorithm does not find a conformation (because all branches on corresponding subtrees are cut as described afterwards) the other partial conformations on level l are used. The following five criteria are used to heuristically decide whether the search tree is extended or cut at a leaf.

Criterion 1. The criterion uses the pheromone information as constructed from the population of the ACO part. Consider a leaf of the search tree and assume that the corresponding partial conformation C consists of $s_1, \ldots, s_{i-1}$ and the element to be placed is s_i. Let $I_{max} = \sum_{j=1}^{i} max\{S_j, L_j, R_j\}$ be the sum of the maximum pheromone values for all decisions. Then an extension of C with decision $d \in \{S, L, R\}$ for placing s_i is not considered if the sum of pheromone values corresponding to the extended partial conformation is smaller than $I_{max} \cdot \Phi_l$ where $0 < \Phi_l < 1$ is a parameter.

Criterion 2. For each H element i in the sequence a minimum energy value is computed that has to be reached by a partial conformation that consist of elements $s_1, \ldots, s_i$. This minimum value is based on the average energy value $E_{avg}(i)$ of the prefixes of length i of the conformation in the population delivered by the ACO part. The longer the conformation becomes the higher the required energy value. For $i \in [1, n/2]$ the minimum value $\lfloor E_{avg}(i) \rfloor + 1$, for i from $n/2+1$ up to the position before the last few H elements the minimum value $\lfloor E_{avg}(i) \rfloor$ is used, and for the rest of the sequence it is required that the best so far found energy value can still be obtained.

Criterion 3. It is checked whether it is possible to extend the current partial conformation so that it can become a new best found conformation. The computation considers: i) the energy value of the current partial conformation, ii) the number of free locations next to H in the partial conformation, iii) the number of H elements with even and with odd indices that have not been placed (for details see [19]). Note, that ii) and iii) can be viewed as measures for the potential that the partial conformation has for improving its energy value.

Criterion 4. For a long subsequence that consists only of P elements (pure P subsequence) there exists many possibilities how to fold it but the energy value of the partial conformation will not change during. Therefore, for all partial conformations for $s_1, \ldots, s_j$ where the last two elements of a pure P subsequence $s_i, \ldots, s_j$, $j \geq i+2$ is placed on the same location and where and which have an equal prefix of length i all those are cut which satisfy the following criterion: the weight of the prefix of length i of the conformation is less than the average weight of the prefixes of length i of the this conformations. Basically the weight is high when the energy of the partial conformation is small, the corresponding pheromone value are high, and its potential is high (see for details [19]).

Criterion 5. If the weight of a partial sequence of length k is not at least 5% higher than the weight of its prefix of length $k-5$ the node is cut.

5 Experiments and Results

The parameter values used for PFold-P-ACO are: $\alpha = 1.2$, $\beta = 1.6$. Each TTL counter has inital value $2.5 \cdot n$

Protein	PERM	F&F	HPPFP-3	PFold-P
S1-1	-9	-9	-9	-9
S1-2	-9	-9	-9	-9
S1-3	-8	-8	-8	-8
S1-4	-14	-14	-14	-14
S1-5	-23	-23	-23	-23
S1-6	-21	-21	-21	-21
S1-7	-36	-36	-36	-36
S1-8	-42	-42	-42	-42
S1-9	-53	-53	-53[b]	-53
S1-10	-50	-49	-49	-49[d]
S1-11	-48	-47	-47	-48[e]
B30-6	-13	-	-13	-13
B30-9	-18	-	-18	-18
B50-5	-22	-	-22	-22[c]
B50-7	-17	-	-17	-17[c]
D1	-19	-	-19	-19
D2	-17	-	-17	-17
D100-1	-	-	-24[a]	-24
D100-2	-	-	-42[a]	-42
D100-3	-	-	-52	-52[c]

Table 2: Energy of best conformation found with PERM [10, 17], filter-and-fan [15] (F&F), ACO-HPPFP-3 [17] (HPPFP-3), and PFold-P-ACO (PFold-P); [a] ([b],[c],[d]) energy value obtained only for 2/5 (2/5,1/2, 9/10) of the runs; [e] energy value obtained only for 3/5 of the runs, the other results obtained -47

Protein	PERM	F&F	HPPFP-3	PFold-P
S1-1	<1s	0s	<1s	0.06s
S1-2	<1s	2s	<1s	0.4s
S1-3	2s	0.5s	<1s	0.2s
S1-4	<1s	4s	4s	1.1s
S1-5	2s	10s	1m	13.3s
S1-6	3s	22s	15s	15.4s
S1-7	4s	56s	20m	4m
S1-8	78h	24s	1.5h	35m
S1-9	60s	1.3m	24h	4.5h
S1-10	-	6.8h	12h	15h
S1-11	8m	13.5m	10h	1.5h(-47)
				8.5h(-48)
B30-6	1.6s	-	70.9s	8.5h
B30-9	0.06s	-	0.06s	0.9s
B50-5	9.4s	-	13m	9.3m
B50-7	4.5m	-	2m	2.5m
D1	2s	-	4m	2.2m
D2	3.5h	-	16m	25m
D100-1	-	-	42.4h	3.5h
D100-2	-	-	38.5h	1.5h
D100-3	-	-	25.8h	14.5h

Table 3: Average computation times for PERM [10, 17], filter-and-fan [15] (F&F), ACO-HPPFP-3 [17] (HPPFP-3), and PFold-P-ACO (PFold-P); the run times for s1-11 for PFold-P-ACO are averages over the runs that produced a conformation with energy -48.

when n is the length of the sequence. The ACO part of PFold-P-ACO stops after a maximum number of iterations 100000. The population size is 10 and the number of ants per iteration is 20. Test runs have been executed on 2.8GHz Intel Xeon double processor PC with 4GB RAM. The HP test sequences are 11 standard benchmark sequences from [18] (S-1,...S-11), 4 sequences that have been used in [17] from the PDB [14] (B-30-6,B-30-9, B-50-5,B-50-7), and 2 sequences from [17] (D-1, D-2). Moreover 3 sequences that are shown in Table 1 have been created by us using a method provided in [17] (D100-x). For the sequences of length ≥ 85 10 runs have been made per test sequences and for the shorter sequences 100 runs.

In addition to the best existing ACO algorithm ACO-HPPFP-3 from [17] we compare PFold-P-ACO with another state of the art algorithm PERM [10] and with the very good algorithm filter-and-fan algorithm of Rego et al. [15] (F&F). A variant of PERM is used which folds from both sides and is called PERMt_{exp} in [17] where also the run times results for PERM and F&F can be found.

A comparison between PERM, F&F, ACO-HPPFP-3 and PFold-P-ACO can be found in tables 2 and 3. All algorithms produce very good results on the S-x sequences but only PERM found the best results for all of them. PFold-P-ACO is the second best algorithm and found the optimal results for all sequences but sequence S1-10. The other two algorithms found the optimal values for all but the two sequences S1-10 and S1-11. With respect to run time PERM is often relatively fast, but has serious problems with some sequences, e.g., symmetric sequences (see also [10]). This can be seen for sequence S1-8 where PERM needs 78h, but ACO-HPPFP-3 needs only 1.5h and the other two algorithms need less than $1h$. F&F has a similar runtime on most S-x Sequences as the ACO algorithms and is significantly faster on sequences S1-8 and S1-11. Unfortunately, so far we could not get results of F&F for the other sequences from the authors of [15].

Comparing the ACO algorithms it has to be taken into account that results on S-x, B-x, D-1 and D-2 from [17] were obtained on a one 2.4GHz processor PC, whereas for PFold-P-ACO we used a two processor 2.8GHz PC. On the other hand ACO-HPPFP-3 is written in C whereas PFold-P-ACO is written in Java. The results for sequences D-100-x have been obtained by us for PFold-P-ACO and ACO-HPPFP-3 on the same two processor PC. Talking all this into account, it seems that PFold-P-ACO is slightly faster on the S-4,...,S-11 sequences (an exception is S-10) and seems slightly slower on the B-x and the D-1 and D-2 sequence. On the long sequences D-100-x PFold-P-ACO is clearly faster. Altogether, PFold-P-ACO seems faster on long sequences whereas both algorithms are similar on small and medium length sequences.

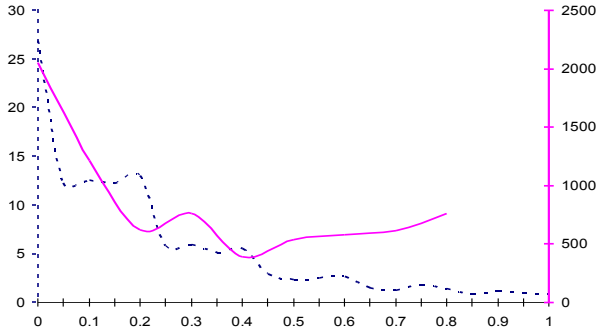

Figure 1: Run time until the optimum is found for S1-4 (left scale) and S1-7 (right scale) for different relative sizes of set $\mathcal{H}$ compared to number of estimated HH-contacts with distance at most 9 in the final conformation

Figure 1 shows the influence of relative size of $\mathcal{H}$ compared to number of estimated HH-contacts with distance ≤ 9 in the final conformation (for details see [19]). It can be seen that the size of $\mathcal{H}$ has a strong influence on the run time. The results indicate that for long sequences a medium size number of prescribed HH-contacts is advantageous whereas for small sequences a larger number of prescribed HH-contacts is better.

6 Conclusions

A hybrid population based ACO algorithm for HP-Protein folding has been proposed. It was shown experimentally that algorithm PFold-P-ACO achieves on nearly all test sequences at least comparable results to other state of the art algorithms and is slightly better and faster than other ACO algorithms.

References

[1] Angus, D., "Population-Based Ant Colony Optimisation for Multi-objective Function Optimisation," *Proceedings ACAL 2007*: pp. 232-244, 2006.

[2] Angus, D., "Crowding Population-based Ant Colony Optimisation for the Multi-objective Travelling Salesman Problem," *Proc. IEEE Symp. Comp. Intell. Multicrit. Decision Making*, pp. 333-340, 2007.

[3] Berger, B., Leight, T., "Protein folding in the hydrophobichydrophilic (HP) model is NP-complete," *J. Comp. Biol.*, V5, N1, pp. 2740, 1998.

[4] Garcia-Martinez, C., et al., "A Taxonomy and Empirical Analysis of Multiple Objective ACO for Bicriteria TSP," *EJOR*, V180, pp. 116-148, 2007.

[5] Dill, K. A., "Theory for the folding and stability of globular proteinst," *Biochemistry*, V24, pp. 15011509, 1985.

[6] Dorigo, M., Maniezzo, V., Colorni, A., "Positive feedback as a search strategy," *Tech Rep.*, 91-016, Dip Elettronica, Politecnico di Milano, Italy, 1991.

[7] Dorigo, M., Stützle, T., *Ant Colony Optimization*, The MIT Press, 2004.

[8] Guntsch, M., Middendorf, M., "A Population Based Approach for ACO," *Proc. EvoCOP-2002*, LNCS 2279, pp. 72-81, 2002.

[9] Guntsch, M., Middendorf, M., "Solving Multi-Objective Permutation Problems with Population Based ACO," *Proc. EMO 2003*, LNCS 2636, pp. 464-478, 2003.

[10] Hsu, H.-P., Mehra, V., Nadler, W., Grassberger, P., "Growth Alg. for Lattice Heteropolymers at Low Temp.," *J. Chem. Phys.* V118, pp. 444-452, 2003.

[11] Lau, K. F., Dill, K. A., "A lattice statistical mechanics model of the conformation and sequence spaces of proteins," *Macromolecules*, V22, pp. 39863997, 1989.

[12] Liebich, D., *Packungsprobleme bei Proteinen*, Shaker, 2005.

[13] Merkle, D., Middendorf, M., "Swarm Intelligence," *Search Methodologies - Introductory Tutorials* , Springer, pp. 401-435, 2006.

[14] *RCSB Protein Data Bank*, www.pdb.org

[15] Rego, C., Li, H., Glover, F., "A Filter-and-Fan Approach to the 2D Lattice Model of the Protein Folding Problem," *Manusscript under revision*, 2006.

[16] Shmygelska, A., Hoos, H. H., "An improved ant colony optimization algorithm for the 2D HP protein folding problem," *Proc. Canadian Conf. on Artificial Intelligence*, LNCS 2671, pp. 400417, 2003.

[17] Shmygelska, A., Hoos, H. H., "An ant colony optimisation algorithm for the 2D and 3D hydrophobic polar protein folding problem," *BMC Bioinformat.*, V6, p. 30, 2005.

[18] *Standard Tortilla Benchmarks*, http://www.dmi.unict.it/ mpavone/psp.html

[19] Thalheim, T., "Hybrid Ant Colony Optimization Algorithms for Solving the Protein Folding Problem," Diploma thesis, University of Leipzig, 2007.

[20] Unger, R., Moult, J, "Genetic algorithm for protein folding simulations," *J. Mol. Biol.*, V231, N1, pp. 7581, 1993.

[21] Yue, K., et al., "A Test of Lattice Protein Folding Algorithms," *PNAS*, V92, pp. 325329, 1995.

[22] Zhao, X., "Advances on protein folding simulations based on the lattice HP models with natural computing," *Applied Soft Computing*, in Press, 2007.