Efficient Computation of Fitness Function by Pruning in Hydrophobic-Hydrophilic Model

Md. Tamjidul Hoque, Madhu Chetty, and Laurence S. Dooley

Gippsland School of Information Technology
Monash University, Churchill VIC 3842, Australia
{Tamjidul.Hoque,Madhu.Chetty,Laurence.Dooley}
@infotech.monash.edu.au

Abstract. The use of Genetic Algorithms in a 2D Hydrophobic-Hydrophilic (HP) model in protein folding prediction application requires frequent fitness function computations. While the fitness computation is linear, the overhead incurred is significant with respect to the protein folding prediction problem. Any reduction in the computational cost will therefore assist in more efficiently searching the enormous solution space for protein folding prediction. This paper proposes a novel pruning strategy that exploits the inherent properties of the HP model and guarantee reduction of the computational complexity during an ordered traversal of the amino acid chain sequences for fitness computation, truncating the sequence by at least one residue.

1 Introduction

Proteins are made up of an alphabet set of 20 different amino acids [1]. Variations in protein conformation depend upon the different combination of amino acids in the sequence and their properties [2]. In addition to these variations, a number of chemical bonds, variations in side-chain, and a number of dihedral angles with a number of degrees of freedom within the amino acid chain make the search space for the optimum folding intractable [3]. This provides motivation to design an effective search algorithm.

Initially, the focus was upon computer-based protein folding prediction algorithms [4], with Molecular Dynamics (MD) and Monte Carlo (MC) technique being heavily employed, though these conformational search methods proved to be too slow. Subsequently, improvements in the speed and efficiency of the search methods became the primary concern [4], with Unger *et al.* [5] designing a Genetic Algorithm (GA) implementation that was much faster than the traditional MC technique. Other strategies including, Hydrophobic Zipper (HZ) [6], Contact Interaction methods (CI) [7], Constraint Programming [8], have developed statistical approaches successfully but only for sequences having limited length around 60 residues or less.

One basic, yet highly effective [8] representation of lattice models for protein folding investigation is the 2D Hydrophobic-Hydrophilic (HP) model proposed by Dill [9], which uses two letter alphabets, namely H and P. Based on dominating hydrophobic force this model has been designed which is well accepted, and used for evaluating search strategies. H indicates the hydrophobic amino acid, while P represents the polar or hydrophilic amino acids. The energy function for the HP-model is calculated as follows. If two residues are Topological Neighbours (TN) - indicated by

J.L. Oliveira et al. (Eds.): ISBMDA 2005, LNBI 3745, pp. 346–354, 2005.

the dotted lines in Figure 1, and they are both H then an ε energy contribution is made where ε is having a value -1. The sum of ε in a conformation becomes the *fitness function* (F) of that particular conformation.

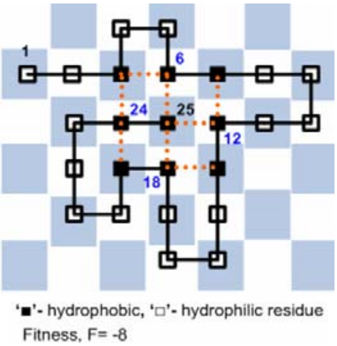


Fig. 1. HP coordinate model, presenting a sequence of amino acids connected by solid line

Searching for the optimum conformation using an HP model is an NP-complete [10] problem, which has motivated researchers to explore alternative solutions such as, the application of GAs [2, 11-14]. The search space however, is enormous and convergence takes a significant time even for short sequences [5]. Pruning strategies to reduce the search space [15] have therefore recently been developed. Caching techniques [16-17] within GA has also proved to be able to reduce the computational load, further. The paper shows pruning residues or truncated traversal while computing fitness. The pruning affords the potential to reduce the computational overhead; as such a traversal is a repetitive process during the search algorithm, irrespective of the dimension (2D or 3D) of structure prediction. Hoque *et al.* [11-12] have previously proposed an improved fitness computation, and this paper extends the work to show that it is not essential to traverse all the hydrophobic residues in computing the fitness computation, which will reduce the computational load further.

The remainder of the paper is organized as follows. Section 2 explains the nomenclature used in the paper, while section 3 describes the fitness computation in the HP model. Section 4 defines *lemma* for the identification of pruning residues and bounds while section 5 explores the searching strategies for pruning and presents the new pruning algorithm. Section 6 examines the impact of pruning and finally, section 7 provides key conclusions.

2 Nomenclature

An amino acid chain traversal for fitness function computation can either be from a higher-numbered residue to a lower numbered residue or vice versa. Throughout this paper, the amino acid chain traversal direction is indicated by *L2H* and *H2L* to respectively represent travel from a lower to a higher-numbered residue and vice versa. In a *L2H* traversal, after pruning, the highest numbered remaining H residue is represented by LCR_{L2H}, while LCR_{H2L} indicates the last computable residue in a H2L traversal after pruning. $Prune_{L2H}$, $Prune_{H2L}$, *MaxPrune* are respectively the number of pruned residues in a L2H traversal, the number of pruned residues in H2L traversal and the maximum of ($Prune_{L2H}$, $Prune_{H2L}$).

For clarity, the amino acid sequence is represented as a binary string, $S = [s_1, s_2, s_3, \cdots, s_m]$ where m is the total number of residues. s_i can either have a value of '1' indicating a hydrophobic residue, or '0' representing a hydrophilic residue. Let E be an ordered number set holding the index i of s_i where s_i is '1'; thus $E = [e_1, e_2, e_3, \cdots, e_n]$, where n is the number of total hydrophobic residues in S, and $n \leq m$.

3 Fitness Computation in the HP Model

In a 2D HP model, possible protein folding conformations are represented by the amino acid chain on a square lattice model forming a self-avoiding walk as shown in Figure 1. For a particular sequence, a number of valid conformations are possible, with the corresponding Fitness function F defined [2], as the negative of the sum of all the TN pairs possible in a particular conformation. Hence, the conformation with the highest number of TN pairs has the lowest energy.

In fitness computation, two possible directions of traversal (L2H and H2L) are considered. The amino acid sequence is numbered for ordered traversal, so for example, a L2H traversal starts from the first hydrophobic residue (Number 3 in the sequence in Figure 1) and searches for the TN amongst its four possible neighbours (in 2D and six in the 3D representation). A residue is identified if and only if, there is a TN from a lower numbered hydrophobic residue to a higher numbered residue. In the example in Figure 1, for a L2H traversal, a TN is encountered from residue number 3 to 6 (3, 6) but not (6, 3), while (12, 25) is a TN, while (25, 12) is not. Thus for a L2H traversal, 8 TNs; (3, 6), (3, 24), (6, 25), (7, 12), (12, 25), (13, 18), (18, 25) and (19, 24) are obtained so F = -8 for the conformation in Figure 1.

This fitness computation is performed after every crossover and mutation operations when a GA is applied to the HP model, which makes it an extremely time-consuming process. Any improvement in the fitness computational cost will therefore reduce the overall computational load significantly.

4 Identification of Pruning Residues and Bounds

During any ordered L2H or H2L traversal, it is clear that the final hydrophobic residue will not encounter a TN, so the last hydrophobic residue 25 for instance in Figure 1 will encounter no TN. Similar reasoning applies to a H2L traversal, so the hydrophobic residue that is traversing last can *always* be omitted from the residue list, which guarantees at least one fewer hydrophobic residues to be traversed.

4.1 Pruning Residues

The objective in this paper is to identify the number of hydrophobic residues that can be pruned in any arbitrary sequence during traversal from one end to the other. The following *lemmas* form the basis for this pruning strategy when searching for either LCR_{L2H} or, LCR_{H2L} in n_H, where n_H is the total number of hydrophobic residues in a sequence.

Lemma 1: To have a TN, the minimum sequence distance between two hydrophobic residues must be greater than 2.

Proof. Let $x \in E$ and $y \in E$. If |x-y|=1 then residues are sequentially connected so no TN is possible (see in Figure 2(a)). If |x-y|=2, then *x* and *y* can at best be diagonally positioned so again no TN is possible (see Figure 2 (b)). However, if |x-y|=3 then placement of two residues at two non-diagonal lattice points of a unit square is feasible as shown in Figure 2(c). □

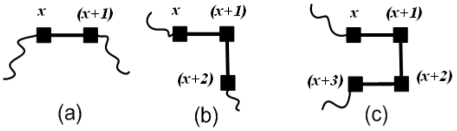


Fig. 2. Instances attempting a TN. The residues in (a) and (b) cannot have TN, but x and (x+3) in (c) can

Lemma 2: The distance between any two TN candidates must be odd.

Proof. Consider any two TN candidates, one hydrophobic residue must be odd while the other will be even in the sequence number. Since in a lattice presentation, even sequenced hydrophobic residues can only be surrounded by odd sequenced [18] residues and visa versa, the distance between two candidates of TN is therefore always odd. □

To illustrate *lemma 2*, consider Figure 1. Residue 25 (odd indexed) is surrounded by residues 6, 12, 18 and 24 (all of which are even indexed). Since, shaded square (odd indexed) is always surrounded by white squares (even indexed) and visa versa. It shows that, opposite indexed residues are separated by odd distance.

Lemma 3: The minimum pruning for any sequence is always 1.

Proof. Based on traversal direction *lemma 1* is further extended. During *L2H* traversal, for *x* the TNs (*y*) can be encountered provided $x < y$. Hence, if *x* is the last hydrophobic residue then there will be no *y*, such that $x < y$. Therefore, visiting the last hydrophobic residue is not required and can be pruned. A similar conclusion applies to H2L traversal. □

Lemma 4: The maximum pruning for traversal is equal to the total number of hydrophobic residue for a particular sequence.

Proof. If there exists a sequence such that all the hydrophobic residues are an even distant apart with respect to each other, then according to *lemma 2*, there will be no TN. Therefore, for such sequences there is no need to traverse any hydrophobic residue. □

4.2 Defining the Pruning Bounds

For defining the lower bound of pruning, assume a sequence of *m* residues. For a number, *r is such that* $r \leq m$*, and we have the* r^{th} residue as the last hydrophobic residue in a sequential traversal. Also, if the r^{th} and *(r-3)*$^{\text{th}}$ residues are hydrophobic but *(r-1)*$^{\text{th}}$ and *(r-2)*$^{\text{th}}$ are hydrophilic, then from *lemma 1*, TN pair of r^{th} or last residue with the shortest distance will be the *(r-3)*$^{\text{th}}$ residue. Using *lemma 3*, we can thus prune the r^{th} single residue so the minimum pruning for any sequence is always 1.

To develop an upper-bound for this new pruning strategy, consider a sequence in which all the hydrophobic residues are either only even or odd indexed. From *lemma 4,* there will be no TN at all for these residues, so every hydrophobic residue is pruned.

Hence, the pruning bound is [1, *n*), where *n* is the number of hydrophobic residues. For short sequences, pruned traversal for fitness computations will always be significant, while for relatively larger sequences (having length around 100 residue or more) with trivial patterns, such as mostly odd indexed or even indexed residues at a location preferably at the start or at the end of the sequence as discussed for upper-bound, will be extremely significant for pruned traversal.

5 Pruning Algorithms

Lemmas 1 and 2 (section 4) allow us to define the following two functions, which are used in the pruning algorithm-1 (given below) to detect LCR_{L2H} or LCR_{H2L}.

$$f_{LCR_{L2H}}(x) = \begin{cases} i & \text{where } 1 \le i < x \le n \text{, and } d \in (e_x - e_i) \text{ such that } d \text{ is minimum} \\ 0 & \text{provided } d > 2 \text{ and } d \text{ is odd. If no such } i \text{ exists then return } 0. \end{cases}$$

$$f_{LCR_{H2L}}(x) = \begin{cases} i & \text{where } 1 \le x < i \le n \text{, and } d \in (e_i - e_x) \text{ such that } d \text{ is minimum} \\ (n+1) & \text{provided } d > 2 \text{ and } d \text{ is odd. If no such } i \text{ exists then return } (n+1). \end{cases}$$

Algorithm 1. To find maximal truncated traversal sequence

```
Input: Sequence S and  Traversal Direction TD.
Output: Number of pruned residue.
Step 1: If  TD = L2H then
Step 2:         Compute $e_a$ and $e_b$, respectively maximum odd, maximum even in E.
Step 3:         $LCR_{L2H}$ = maximum of $\{f_{LCR_{L2H}}(a), f_{LCR_{L2H}}(b)\}$
Setp 3:         Return: $Prune_{L2H} = (n - LCR_{L2H})$
        else
Step 4:         Compute $e_u$ and $e_v$, respectively minimum odd, minimum even in E.
Step 5:         $LCR_{H2L}$ = minimum of $\{f_{LCR_{H2L}}(u), f_{LCR_{H2L}}(v)\}$
                Return: $Prune_{H2L} = (LCR_{H2L} - 1)$
        endif
```

The Algorithm-1 assumes that both the sequence and the traversal direction are given as input and it will return number of pruned residue. Depending on odd and even hydrophobic residue groups in a sequence, the function $f_{LCR_{L2H}}(x)$ is invoked twice (Step 3), with *x=a* and *x=b,* indicating the index of the maximum odd and maximum even numbered hydrophobic residue (Step 2) respectively. With *x=a*, the function will return an even indexed candidate of LCR$_{L2H}$ and with $x=b$, the function will return an odd indexed candidate of LCR$_{L2H}$. The maximum return of the two is the LCR$_{L2H.}$ Similarly we, invoked function $f_{LCR_{H2L}}(x)$ (Step 5) to detect the LCR$_{H2L}$, where x is being assigned the indices of *first odd* and *first even* hydrophobic residues (Step 4) and in this case the minimum of the two return is taken. These two functions help choosing traversal direction to have maximum pruning from either direction.

For example, in Figure 1, S = [0010011000011000011000011] and m = 25, that is, E = [3, 6, 7, 12, 13, 18, 19, 24, 25] and n = 9. The value of e_a, e_b, e_u and e_v are respectively 25, 24, 3 and 6. According to Algorithm 1, (LCR_{L2H} = 7) as maximum of $\{(f_{LCR_{L2H}}(9)=6),(f_{LCR_{L2H}}(8)=7)\}$. Similarly, ($LCR_{H2L}$ = 4) as minimum of $\{(f_{LCR_{H2L}}(1)=4),(f_{LCR_{H2L}}(2)=5)\}$. So, Prune$_{H2L}$ = 3, Prune$_{L2H}$ = 2, hence MaxPrune = 3. Therefore, the traversal direction is H2L will provide the *MaxPrune* and it saves fitness computation traversal by 33.33%.

6 Simulation Results

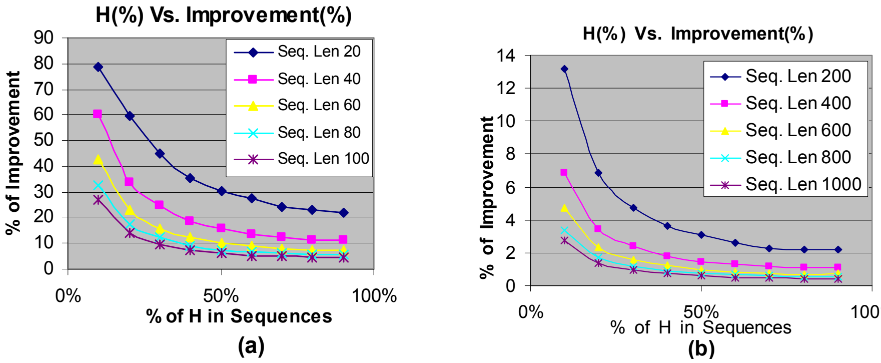
For each sequence length, the occurrence frequency of H (the total number in a sequence) is considered as a percentage of sequence length. So for a sequence length of 1000 and having 20 H residues means H% = 20. To identify the impact of H%, it is varied from between 10% to 90% in steps of 10. Since these residues are randomly distributed, for each value of H%, the average improvement is computed from 1000 simulation runs with the measure of pruning defined as $I=\frac{k}{n}\times 100\%$, where n is the total number of hydrophobic residues in a sequence and k the number of pruned hydrophobic residues from that sequence. To establish the significance and impact of pruning on computational throughput, simulations were undertaken using both randomly generated sequences for analyzing robustness and popular benchmark sequences [19] used by the broader research community for testing the practical impact.

a) Randomly generated sequences:

The studies are performed to investigate the impact upon both the sequence length as well as the occurrence of H in the sequence. For this purpose, the length is varied from 20 till 1000 with a step increase of 20. It can be seen from Figures 3 and 4(a) that the significance of pruning depends on the occurrence frequencies and pattern of H and P in the sequence. It is observed that the lower the frequency of H in a sequence, the higher the pruning improvement I. Also, it is observed that the pruning performance is higher for shorter sequences which are less than around 100 residues. Figure 4(b) shows the maximum, average and minimum percentage ‘improvement’, which also reveals that for a sequence with a relatively lower number of hydrophobic residues, the line showing maximum improvement (%) has higher value. For any sequence, the minimum number of pruned residue is at least 1, so the significance of the minimum pruning decreases uniformly with increasing numbers of hydrophobic residues in a sequence.

b) Benchmark sequences [19]:

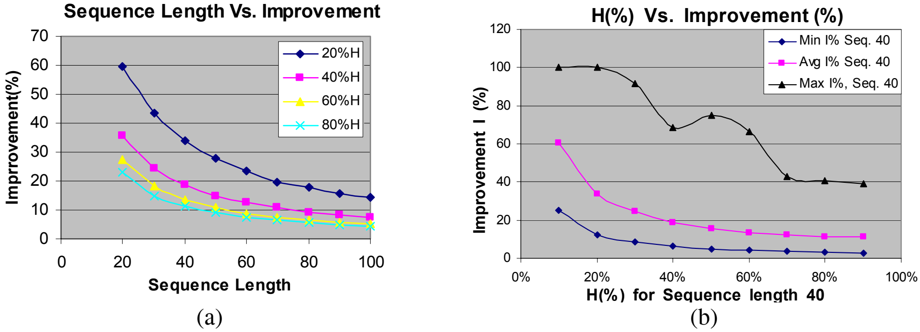
The pruning algorithm was next applied to a selection of the 2D benchmark sequences given in Table 1. The Table also shows the corresponding improvement in results with respect to all hydrophobic residue traversal thus establishing the practical significance of the pruning technique presented here in this paper.



(a)

(b)

Fig. 3. Percentage of H and corresponding pruning improvement (%). (a) Sequence length 20 to 100, step 20. (b) Sequence length 200 to 1000, step 200



(a)

(b)

Fig. 4. (a) Sequence length and corresponding pruning improvement (%). (b) Improvement I(%) comparing maximum, average and minimum

7 Conclusions

This paper has presented a novel pruning strategy for Hydrophobic-Hydrophilic (HP) model to reduce the computation overhead during an ordered traversal of amino acid chain sequences. The new approach guarantees a minimum pruning for any sequence, thereby ensuring a speed up in the search process for protein folding prediction using GA. A series of lemma have been postulated in the development of the theoretical basis of this new strategy and simulation results for both randomly-generated and benchmark sequences confirm the improvement achieved. While the focus of the pruning algorithm has been on a 2D HP model, the strategy can be extended in a straightforward manner to a 3D HP model. Also, our algorithm can be easily extended to be embedded within current caching approaches [16-17], thereby providing a further reduction in the computational load.

Table 1. Pruning results for 2D benchmark sequences. Pruning improvement is shown respect to all hydrophobic residues traversal

Sequence Length	Total # of H	Max-Prune	Improvement (%)	Traversal Direction	Sequence
20	10	3	**30.00**	H2L	HHHPPHPHPHPPHPHPHPPH
20	10	2	**20.00**	L2H / H2L	HPHPPHHPHPPHPHHPPHPH
24	10	2	**20.00**	L2H / H2L	HHPPHPPHPPHPPHPPHPPHPPHH
25	9	2	**22.22**	L2H	PPHPPHHPPPPHHPPPPHHPPPPHH
36	16	2	**12.50**	H2L	PPPHHPPHHPPPPPHHHHHHHPPHHPPPPHHPPHPP
45	27	3	**11.11**	H2L	PHHHPHHHPPPHPHHPHHPPHPHHHHPHPPHHHHHPHPHHPPHHP
48	25	3	**12.00**	L2H	PPHPPHHPPHHPPPPPHHHHHHHHHHPPPPPPHHPPHHPPHPPHHHHH
50	24	2	**8.33**	L2H / H2L	HHPHPHPHPHHHHPHPPPHPPPHPPPPHPPPHPPPHPHHHHPHPHPHPHH
57	30	2	**6.67**	L2H	HPHHHPHHHPPHHPHPHHPHHHPHPHPHHPPHHHPPHPHPPPPHPPHPPHHPPHPPH
60	43	3	**6.98**	L2H	PPHHHPHHHHHHHHPPPHHHHHHHHHHPHPPPHHHHHHHHHHHHPPPPHHHHHHPHHPHP
64	42	3	**7.14**	L2H / H2L	HHHHHHHHHHHHPHPHPPHHPPHHPPHPPHHPPHHPPHPPHHPPHHPPHPHPHHHHHHHHHHHH
102	37	2	**5.41**	H2L	PHHPPPPPHHPPHHPHPPHPPPPPPPPHPPPHHPHHPPPPPPHPPHPHPPHPPPPPHHHPPPPHHPHHPPPPPHHPPPPHHHHPHPPPPPPPPPHHHHHHPPHPP
123	47	5	**10.65**	L2H	PPHHHPHPPPPHPPPPPHHPPPPHHPPHHPPPPHPPPPHPPHPPHHPPPHHPHPHHHPPPPHHHPPPPPPHHPPHPPHPHPPHPPPPPPPPHPPHHHPPPPHPPPHHHHHHPPPPHHPHPHPHPH
136	50	2	**4.00**	L2H / H2L	HPPPPPHPPPPHPHHPHHPPPPHPHHHPPPPHPHPHHHHPPPPPPPPPPHPPHPPPHPHHPPPHHPPHPPHPHPHPPPPPPPPPHPPPHHHHHHHPPPHHPPHHHPPPHHPHHHHHHPPPPPPPPPPHPPPPHPHPPPP

References

1. Allen, et al.: Blue Gene: A vision for protein science using a petaflop supercomputer. IBM System Journal (2001), Vol 40, No 2
2. Gary, B.F. and David, W.C. (eds.): Evolutionary Computation in Bioinformatics. Elsevier Science (2003) USA
3. Lathrop, R.H.: Protein Structure Prediction. http://helix-web.stanford.edu/psb98/lathrop.pdf (1998)
4. Beutler, T.C. and Dill, K.A.: A fast conformational search strategy for finding low energy structures of model proteins. Protein Science (1996)
5. Unger, R. and Moult J.: Genetic Algorithm for Protein Folding Simulations, J. Mol. Biol. (1993), 231, 75-81
6. Dill, K.A., Fiebig, K.M., Chan H.S.: Cooperativity in Protein-Folding Kinetics. Proceedings of the National Academy of Sciences USA, Biophysics (1993) Vol 90, pp.1942-1946
7. Toma, L. and Toma, S.: Contact interactions method: A new algorithm for protein folding simulations. Protein Science (1996) 5: 147-153
8. Backofen, R.: Using Constraint Programming for Lattice Protein Folding. http://helix-web.stanford.edu/psb98/backofen.pdf (1998)

9. Dill, K.A.: Theory for the Folding and Stability of Globular Proteins. Biochemistry (1985) 24: 1501
10. Berger, B. and Leighton, T.: Protein Folding in the Hydrophobic-Hydrophilic (HP) model is NP-Complete. ACM, Proceedings of the second annual international conference on Computational molecular biology (1998)
11. Hoque M.T., Chetty, M. and Dooley L.S.: An Efficient Algorithm for Computing the Fitness Function of a Hydrophobic-Hydrophilic Model. 4th International Conference on Hybrid Intelligent Systems (HIS 2004), pp. 285-290, ISBN 0-7695-2291-2.
12. Hoque M.T., Chetty, M. and Dooley L.S.: Partially Computed Fitness Function Based Genetic Algorithm for Hydrophobic-Hydrophilic Model. 4th International Conference on Hybrid Intelligent Systems (HIS 2004), pp. 291-296, ISBN 0-7695-2291-2.
13. König, R. and Dandekar, T.: Refined Genetic Algorithm Simulation to Model Proteins. Journal of Molecular Modeling (1999)
14. Takahashi, O., Kita, H. and Kobayashi, S.: Protein Folding by A Hierarchical Genetic Algorithm. 4th Int. Symp. On Artificial Life and Robotics (AROB) (1999)
15. Voelz, V.: Zipping as a fast conformational search strategy for protein folding. http://laplace.compbio.ucsf.edu/~voelzv/orals/orals_proposal.pdf (2004)
16. Santos, E.E. and Santos, E.J.: Effective and Efficient Caching in Genetic Algorithms. International Journal on Artificial Intelligence Tools, © Worlds Scientific Publishing Company (2000)
17. Santos, E.E. and Santos, E.J.: Reducing the Computational Load of Energy Evaluations for Protein Folding", 4th IEEE Symp. on BIBE'04.
18. Newman, A.: A new algorithm for protein folding in the HP model. Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete Algorithms (2002)
19. Hart, W. and Istrail, S.: HP Benchmarks,
http://www.cs.sandia.gov/tech_reports/compbio/tortilla-hp-benchmarks.html