2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications

Collaborative Parallel Local Search for Simplified Protein Structure Prediction

Mahmood A Rashid*†§, M.A.Hakim Newton*§, Md Tamjidul Hoque‡§ and Abdul Sattar*†§
*Institute for Integrated and Intelligent Systems (IIIS), Griffith University, Australia
†Queensland Research Laboratory, National ICT of Australia
‡Computer Science, University of New Orleans, USA
§Email: {m.rashid,hakim.newton,a.sattar}@griffith.edu.au, thoque@uno.edu

***Abstract*—Protein structure prediction is a challenging optimisation problem to the computer scientists. A large number of existing single-point search algorithms attempt to solve the problem by exploring possible structures and finding the one with the minimum free energy. However, these algorithms perform poorly on large sized proteins due to an astronomically wide search space. In this paper, we present a multi-point local search framework that uses parallel processing techniques to expedite exploration by starting from different points. In our approach, a set of random initial solutions are generated and distributed to different threads. We allow each thread to run for a pre-defined period of time. The improved solutions are stored thread-wise. When the threads finish, the solutions are merged together and the duplicates are removed. A selected distinct set of solutions are then split to different threads again. We tested our approach on large sized proteins for simplified models. The experimental results show that our new parallel framework significantly improves over the results obtained by the state-of-the-art single-point search approaches for the hydrophobic-polar energy model and the three dimensional face-centred-cubic lattice.**

***Keywords*—Parallel Computing; Multi-threading; Local Search; Protein Structure Prediction; HP Model; 3D FCC Lattice**

I. Introduction

Proteins are essentially sequences of amino acids. They adopt specific folded three-dimensional structures to perform specific tasks. The function of a given protein is determined by its *native* structure, which has the lowest possible free energy level. Nevertheless, misfolded proteins cause many critical diseases such as Alzheimer's disease, Parkinson's disease, and Cancer [1], [2]. Protein structures are important in drug design and biotechnology.

Protein structure prediction (PSP) is computationally a very hard problem [3]. Given a protein's amino acid sequence, the problem is to find a three dimensional structure of the protein such that the total interaction energy amongst the amino acids in the sequence is minimised. The protein folding process that leads to such structures involves very complex molecular dynamics [4] and unknown energy factors. To deal with the complexity in a hierarchical fashion, researchers have used discretised lattice-based structures and simplified energy models [5]–[7] for PSP. However, the complexity of the simplified problem still remains challenging.

There are a large number of existing search algorithms that attempt to solve the PSP problem by exploring feasible structures called *conformations*. For population based approaches, a genetic algorithm (GA^+ [8]) reportedly produces the state-of-the-art results. However, for local search approaches, spiral search (SS-Tabu) [9], which is a tabu-based local search, produces the best results.

In general, the success of single-point search or population based search algorithms crucially depends on the balance of diversification and intensification of the exploration. However, these algorithms often get stuck in local minima. As a result, they perform poorly on large sized ($length > 100$ amino acids) proteins. Any further progress to these algorithms require addressing the above issues appropriately.

In a collaborative human team, each member may work individually on his/her own way to solve a problem. They may meet together occasionally to discuss the possible ways they could find and may then refocus only on the more viable options in the next iteration. We envisage this approach to be useful in finding a suitable solution when there are enormously many alternatives that are very close to each other. We therefore try this in the context of conformational search for protein structure prediction.

In this paper, we present a multi-threaded search technique that runs SS-Tabu in each thread. The search starts with a set of random initial solutions by distributing these solutions to different threads. We allow each thread to run for a pre-defined period of time. The interim improved solutions are stored thread-wise and merged together when all threads have finished. After removing the duplicates form the merged-solutions, a selected distinct set of solutions is then considered for next iteration. In our approach, multi-point start first helps find some promising results. For the next set of solutions to be distributed, the most promising solutions from the merged-list are selected. Therefore, multi-point parallelism reduces the search space by exploring the vicinities of the promising solutions recursively. In our parallel local search, we use the hydrophobic-polar (HP) energy model and the three dimensional (3D) face-centred-cubic (FCC) lattice space. The experimental results show that our new approach significantly improves over the results obtained by the state-of-the-art single-point search approaches for the same model.

The rest of the paper is organized as follows: Section II describes the protein structure structure problem and simplified models for PSP; Section III presents the related work; Section IV presents the SS-Tabu framework used in our parallel search approach; Section V describes our parallel framework in detail; Section VI discusses and analyses the experimental results; and finally, Section VII presents the conclusions and outlines the future work.

978-0-7695-5022-0/13 $26.00 © 2013 IEEE
DOI 10.1109/TrustCom.2013.118

II. Background

Homology modeling [10], *protein threading* [11], [12] and *ab initio* [13], [14] are three computational approaches used in protein structure prediction. Prediction quality of *homology modeling* and *protein threading* depend on the sequential similarity of previously known protein structures. However, our work is based on the *ab initio* approach that only depends on the amino acid sequence of the target protein. Levinthal's paradox [15] and Anfensen's hypothesis [16] are the basis of *ab initio* method for PSP. The idea was originated in 1970 when it was demonstrated that all information needed to fold a protein resides in its amino acid sequence. In our simplified protein structure prediction model, we use 3D FCC lattice for conformation mapping, HP energy model for conformation evaluation, and a hydrophobic-core centric local search algorithm (SS-Tabu) for conformation search. The simplified models, multi-threading, local search, and are described below.

A. Simplified Model

In our approach, we use 3D FCC lattice points for conformation mapping and the HP energy model for conformation evaluation. The 3D FCC lattice and the HP energy model are briefly described below.

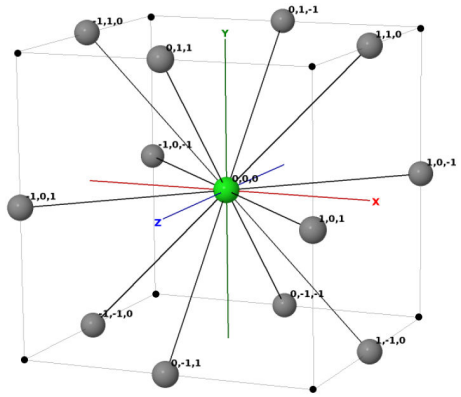


Figure 1: A unit 3D FCC lattice with 12 basis vectors on the Cartesian coordinate space.

1) 3D FCC Lattice: The FCC lattice has the highest packing density compared to the other existing lattices [17]. In FCC, each lattice point (the origin in Figure 1) has 12 neighbours with 12 *basis vectors* $(1,1,0)$, $(-1,-1,0)$, $(-1,1,0)$, $(1,-1,0)$, $(0,1,1)$, $(0,1,-1)$, $(1,0,1)$, $(1,0,-1)$, $(0,-1,1)$, $(-1,0,1)$, $(0,-1,-1)$, and $(-1,0,-1)$. The hexagonal closed pack (HCP) lattice, also known as cuboctahedron, was used in [18]. In HCP, each lattice point has 12 neighours that correspond to 12 basis vertices with real-numbered coordinates. The real numbers cause the loss of structural precision for PSP. In simplified PSP, conformations are mapped on the lattice by a sequence of basis vectors, or by the *relative vectors* that are relative to the previous basis vectors in the sequence.

2) HP Energy Model: The 20 amino acid monomers are the building block of protein polymers. These amino acids are broadly divided into two categories based on their hydrophobicity: (a) hydrophobic amino acids (*Gly*, *Ala*, *Pro*, *Val*, *Leu*, *Ile*, *Met*, *Phe*, *Tyr*, *Trp*) denoted by H; and (b) hydrophilic or polar amino acids (*Ser*, *Thr*, *Cys*, *Asn*, *Gln*, *Lys*, *His*, *Arg*, *Asp*, *Glu*) denoted by P. In the HP model [19], when two non-consecutive hydrophobic amino acids become topologically neighbours, they contribute a certain amount of negative energy, which for simplicity is shown as -1 in Figure 2. The total energy (E) of a conformation based on the HP model becomes the sum of the contributions of all pairs of non-consecutive hydrophobic amino acids as shown in Equation 1.

	H	**P**
H	-1	0
P	0	0

Figure 2: HP energy model [19]

$$E = \sum_{i<j-1} c_{ij}.e_{ij} \tag{1}$$

Here, $c_{ij}=1$ if amino acids i and j are non-consecutive neighbours on the lattice, otherwise 0; and $e_{ij} = -1$ if ith and jth amino acids are hydrophobic, otherwise 0.

B. Multi-threading

Multi-threading is the ability of an operating system to execute different parts of a program, called threads, simultaneously. The programmer must carefully design the program in such a way that all the threads can run at the same time in parallel without interfering with each other. Creating a new thread is similar but not the same as to forking a new process. Compared to a forked process, a new thread shares much more information with other threads.

C. Local Search

Starting from an initial solution, local search algorithms move from one solution to another to find a better solution. Local search algorithms are well known for efficiently producing high quality solutions, which are difficult for systematic search approaches. However, they are incomplete [20], and suffer from revisitation and stagnation. Restarting the whole or parts of a solution remains the typical approach to deal with such situations.

D. Tabu Meta-heuristic

Tabu meta-heuristic [21], [22] enhances the performance of local search algorithms. It maintains a short-term memory storage to remember the local changes of a solution. Then any further local changes for those stored positions are forbidden for a certain number of subsequent iterations (known as tabu tenure).

III. Related Work

Different types of metaheuristic have been used in solving the simplified PSP problem. These include Monte Carlo Simulation [23], Simulated Annealing [24], Genetic Algorithms (GA) [25], [26], Tabu Search with GA [27], Tabu Search with Hill Climbing [28], Ant Colony Optimisation [29], Immune Algorithms [30], Tabu-based Stochastic Local Search [31], [32], and Constraint Programming [33]. Cebrian *et al.* [31] used tabu-based local search, and Shatabda *et al.* [32] used memory-based local search with tabu heuristic and achieved the state-of-the-art results. However, Dotu *et al.* [33] used constraint programming and found promising results but only for smaller sized ($length < 100$ amino acids) proteins. Besides local search, Unger and Moult [25] applied population based genetic algorithms to PSP and found their method to be more promising than the Monte Carlo based methods [23]. They used absolute encodings on the square and cubic lattices for HP energy model. Later, Patton [34] used relative encodings to represent conformations and a penalty method to enforce the self-avoiding walk constraint. GAs have been used by Hoque *et al.* [18] for cubic, and 3D HCP lattices. They used DFS-generated pathways [35] in GA crossover for protein structure prediction. They also introduced a twin-removal operator [36] to remove duplicates from the population to prevent the search from stalling. Ullah *et al.* in [37] and [38] combined local search with constraint programming. They used a 20×20 energy model [39] on FCC lattice and found promising results. In another hybrid approach [40], tabu meta-heuristic was combined with genetic algorithms in two-dimensional HP model to observe crossover and mutation rates over time.

However, for the simplified model (HP energy model and 3D FCC lattice) that is used in this paper, a new genetic algorithm GA$^+$ [8] and a tabu-based local search algorithm Spiral Search [9] currently produce the state-of-the-art results.

IV. SS-Tabu: Spiral Search

SS-Tabu is composed of H and P move selections, random-walk [41], and relay-restart [9]. The search progresses with the application of a series of diagonal-moves in a spiral fashion.

A. Applying Diagonal-move

In a tabu-guided local search (see Algorithm 1), we use the diagonal move operator (shown in Figure 3) to build H-core. A diagonal move displaces ith amino acid from its position to another position on the lattice without changing the position of its succeeding $(i+1)$th and preceding $(i-1)$th amino acids in the sequence. The move is just a corner-flip to an unoccupied lattice point.

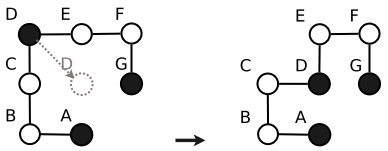


Figure 3: Diagonal move operator. For easy understanding, the figures are presented in 2D space.

B. Forming H-core

Protein structures have hydrophobic cores (H-core) that hide the hydrophobic amino acids from water and expose the polar amino acids to the surface to be in contact with the surrounding water molecules [42]. H-core formation is an important objective for HP based protein structure prediction models. In our work, we repeatedly use the diagonal-move to aid forming the H-core. We maintain a tabu list to control the amino acids from getting involved in the diagonal moves. SS-Tabu performs a series of diagonal-moves on a given conformation to build the H-core around the hydrophobic core centre (HCC) as shown in Figure 4. The Cartesian distance between the HCC and the current position or a new position is denoted by d_1 and d_2 respectively. The diagonal-move squeezes the conformation and quickly forms the H-core in a spiral fashion.

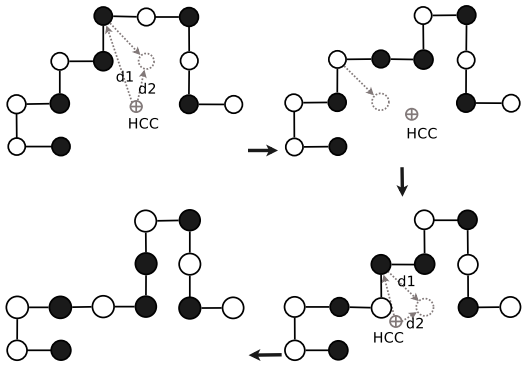


Figure 4: Spiral search comprising a series of diagonal moves with tabu meta-heuristics. For simplification and easy understanding, the figures are presented in 2D space.

C. Selecting Moves

In H-move selection, the HCC is calculated by finding arithmetic means of x, y, and z coordinates of all hydrophobic amino acids using Equation 2. The selection is guided by the Cartesian distance d_i (as shown in Equation 3) between HCC and the hydrophobic amino acids in the sequence. For the ith hydrophobic amino acid, the common topological neighbours of the $(i-1)$th and $(i+1)$th amino acids are computed. The topological neighbours (TN) of a lattice point are the points at unit lattice-distance apart from it. From the common neighbours, the unoccupied points are identified. The Cartesian distance of all unoccupied common neighbours are calculated from the HCC using Equation 3. Then the point with the shortest distance is picked. This point is listed in the possible H-move list for ith hydrophobic amino acid if its current distance from HCC is greater than that of the selected point. When all hydrophobic amino acids are traversed and the feasible shortest distances are listed in H-move list, the amino acid having the shortest distance in H-move list is chosen to apply the diagonal move on it. A tabu list is maintained for each hydrophobic amino acid to control the selection priority amongst them. For each successful move, the tabu list is updated for the respective amino acid. The process stops when no H-move is found. In this situation, the control is transferred to select and apply P-moves.

$$x_{\mathrm{hcc}} = \frac{1}{n_{\mathrm{h}}} \sum_{i=1}^{n_{\mathrm{h}}} x_i,\ y_{\mathrm{hcc}} = \frac{1}{n_{\mathrm{h}}} \sum_{i=1}^{n_{\mathrm{h}}} y_i,\ z_{\mathrm{hcc}} = \frac{1}{n_{\mathrm{h}}} \sum_{i=1}^{n_{\mathrm{h}}} z_i \qquad (2)$$

Algorithm 1: *SpiralSearch* **(mxIter, mxRetry, mxRW, c)**

```
1  //H and P are hydrophobic & polar amino acids.
2  initTabuList()
3  for (i = 1 to maxIter) do
4      mv ⟵ selectMoveForH()
5      if (mv != null) then
6          applyMove(mv)
7          updateTabuList(i)
8      else
9          mv ⟵ selectMoveForP()
10         if (mv != null) then
11             applyMove(mv)
12         end
13     end
14     evalute(AA)  //AA-amino acid array
15     if (!improved) then
16         retry++
17     else
18         improvedList ⟵ addTopOfList()
19         retry = 0
20         rw = 0
21     end
22     if retry ≥ maxRetry then
23         randomWalk(maxPull)
24         resetTabuList()
25         rw++;
26     end
27     if rw ≥ maxRW then
28         relayRestart(improvedList)
29         resetTabuList()
30     end
31 end
```

where n_{h} is the number of H amino acids in the protein.

$$d_i = \sqrt{(x_i - x_{\mathrm{hcc}})^2 + (y_i - y_{\mathrm{hcc}})^2 + (z_i - z_{\mathrm{hcc}})^2} \tag{3}$$

However, in P-move selection, the same kind of diagonal moves are applied as H-move. For each ith polar amino acid, all free lattice points that are common neighbours of lattice points occupied by $(i-1)$th and $(i+1)$th amino acids are listed. From the list, a point is selected randomly to complete a diagonal move for the respective polar amino acid. No hydrophobic-core-center is calculated, no Cartesian distance is measured, and no tabu list is maintained for P-move. After one try for each polar amino acid the control is returned to select and apply H-moves.

D. Handling Stagnation

For hard optimisation problems such as protein structure prediction, local search algorithms often face stagnation. Thus, handling such situation intelligently is important to proceed further. To deal with stagnation, in SS-Tabu, a random-walk [41] and a relay-restart techniques are used on an on-demand basis.

Random-walk: Premature H-cores are observed at local minima. To escape local minima, a random-walk [41] algorithm is applied. This algorithm uses pull moves [43] to break the premature H-cores and to create diversity.

Relay-restart: When the search stagnation situation arises, a new relay-restart technique is applied instead of a fresh restart or restarting from the current best solution [31], [32]. We use relay-restart when random-walk fails to escape from the local minima. The relay restart starts from an improving solution. We maintain an improving solution list that contains all the improving solutions after the initialisation.

Algorithm 2: *evaluate***(AA)**

```
1  for (i = 1 to seqLength − 1) do
2      for (k = i + 2 to seqLength − 1) do
3          if AAType[i] = AAType[k] =H then
4              nodeI ⟵ AA[i]
5              nodeJ ⟵ AA[k]
6              sqrD ⟵ getSqrDist(nodeI, nodeJ)
7              if sqrD = 2 then
8                  fitness ⟵ fitness − 1
9              end
10         end
11     end
12 end
13 return fitness
```

E. Further Implementation Details

Like other search algorithms, SS-Tabu requires initialisation. It also needs evaluation of the solution in each iteration. It starts with a randomly generated or parameterised initial solution and enhance it in a spiral fashion. Further, it needs to maintain a tabu meta-heuristic to guide the local search.

1) Tabu Tenure: Intuitively we use different tabu-tenure values based on the number of hydrophobic amino acids (hCount) in the sequence. We calculate tabu-tenure using the formula at Equation 4:

$$\mathbf{tenure} = \left(10 + \frac{\mathsf{hCount}}{10}\right) \tag{4}$$

2) Evaluation: After each iteration, the conformation is evaluated by counting the H-H contacts (topological neighbours) where the two amino acids are non-consecutive. The *pseudocode* in Algorithm 2 presents the algorithm of calculating the free energy of a given conformation. Note that the energy value is negation of the H-H contact count.

V. Our Approach

In our approach, we designed a multi-threaded local search framework (SS-Parallel) that run the spiral search (SS-Tabu) [9] in all threads. The details of the framework are described below.

A. Multi-threaded Framework

Figure 5 shows the architecture of our parallel search algorithm. In this framework, the search starts with a set of randomly generated initial solutions (Line 2 in Algorithm 3).

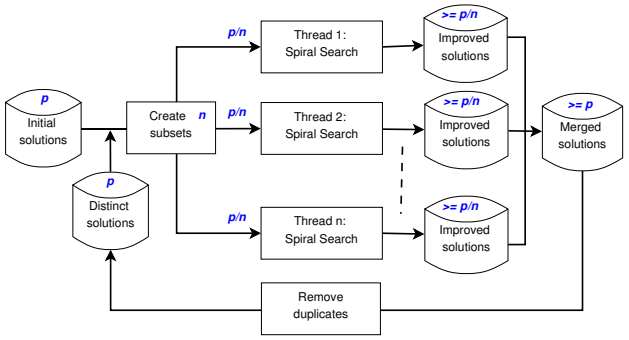


Figure 5: Parallel Spiral Search Framework.

The solutions are then divided in subsets (Line 4 in Algorithm 3) and are distributed to different threads. We allow each thread to run for a pre-defined period of time. The improved solutions are stored thread-wise and are merged together (Line 10 in Algorithm 3) when all threads finish. After removing the duplicates (Line 11 in Algorithm 3) form the merged-solutions, a selected distinct set of solutions are taken (Line 12 in Algorithm 3) for the next iteration. The iterative process continues until the terminating criteria (Line 3 in Algorithm 3) is satisfied.

Algorithm 3: *SSParallel*(**thrCount,cputime,size,repeat**)

```
1  //thr- Thread
2  currSet ← initialise()
3  for (i = 1 to repeat) do
4      subSet ← genSubSet(currSet)
5      for (i = 1 to thCount) do
6          thr[i] = createSSThread(subSet[i], time)
7          thr[i].start()
8      end
9      if (noAliveThread) then
10         mrgLst = mergeImprovedLists()
11         distinctLst = removeDuplicate(mrgLst)
12         currSet ← genCurrSet(distinctLst)
13     end
14 end
```

B. Initialisation

Our algorithm starts with a feasible set of conformations known as population. We generate each initial conformation following a randomly generated self-avoiding walk (SAW) on FCC lattice points. The *pseudocode* of the algorithm is presented in Algorithm 4. It places the first amino acid at $(0, 0, 0)$. It then randomly selects a basis vector to place the successive amino acid at a neighbouring free lattice point. The mapping proceeds until a self avoiding walk is found for the whole protein sequence.

VI. Experiments

This section includes the experimental setup, benchmarks, experimental results, and analysis and discussion on the results.

A. Experiment Setup

We implemented our framework in Java (J2EE). We ran our experiments on the NICTA[1] cluster. The cluster consists of a number of identical Dell PowerEdge R415 computers, each equipped with 2×AMD 6-Core Opteron 4184 processors, 2.8GHz clock speed, 3M L2/6M L3 Cache, 64 GB memory and running Rocks OS (a Linux variant for cluster). For each protein, we ran each algorithm 50 times.

[1]NICTA website: www.nicta.com.au

Algorithm 4: *initialise*()

```
1  //AA-amino acid array of the protein
2  //SAW- Self-avoiding-walk
3  basisVec[12] ⟵ getTwelveBasisVectors()
4  AA[0] ⟵ AminoAcid(0, 0, 0)
5  while (!SAW) do
6      for (i = 1 to seqLength − 1) do
7          k ⟵ getRandom(12)
8          basis ⟵ basisVec[k]
9          node ⟵ AA[i − 1] + basis
10         if isFree(node) then
11             AA[i] ⟵ AminoAcid(node)
12         else
13             SAW ⟵ false
14             break
15         end
16     end
17 end
18 return AA[ ]
```

B. Benchmark

In our experiment, the protein instances (Table I), *S*, *F180*, and *R* are taken from Peter Clote laboratory website[2]. These instances have been used in [8], [9], [31]–[33] for evaluating different algorithms. Moreover, we use other six larger sequences that are taken from the CASP[3] competition. The corresponding CASP target IDs for proteins *3mse*, *3mr7*, *3mqz*, *3no6*, *3no3*, and *3on7* are *T0521*, *T0520*, *T0525*, *T0516*, *T0570*, and *T0563*. These CASP targets are also used in [32]. To fit in the HP model, the CASP targets are converted to HP sequences based on the hydrophobic properties of the constituent amino acids. The lower bounds of the free energy values (in Column *LBFE* of Table I) are obtained from [31], [32]; however, there are some unknown values (presented as n/a) of lower bounds of free energy for large sequences.

C. Data Table (Table I)

In Table I, we present three different sets of result obtained from *i)* our parallel local search framework that runs on four parallel threads (30 minutes for Protein S and 1 hour 15 minutes for others), *ii)* a local search (SS-Tabu) that runs on a single thread (2 hours for Protein S and 5 hours for others), and *iii)* a genetic algorithm (GA^+) that runs on a single thread (2 hours for Protein S and 5 hours for others). In the table, the *Size* column presents the number of amino acids in the sequences, and the *LBFE* column shows the known lower bounds of free energy for the corresponding protein sequences in Column *ID*. However, a lower bound of free energy for protein *3on7* is not known. The best and average free energy values for three different algorithms are presented in the table

[2]Peter Clote Lab: bioinformatics.bc.edu/clotelab/FCCproteinStructure
[3]CASP website: predictioncenter.org/casp9/targetlist.cgi

Protein Info			Our approach		The current state-of-the-art approaches					
			SS-Parallel		**SS-Tabu** [9]			**GA$^+$** [8]		
			No. of threads = **4**		No. of thread = 1			No. of thread = 1		
Seq	**Size**	**LBFE**	**Best**	**Avg**(E_t)	**Best**	**Avg**(E_r)	**RI**	**Best**	**Avg**(E_r)	**RI**
			User time/run = **0.50** hrs		User time/run = 2.00 hrs			User time/run = 2.00 hrs		
			CPU time/run = 2.00 hrs		CPU time/run = 2.00 hrs			CPU time/run = 2.00 hrs		
S1	135	-357	**-355**	**-350**	-355	-347	34%	-355	-348	27%
S2	151	-360	**-356**	**-351**	-354	-347	30%	-356	-349	18%
S3	162	-367	**-360**	**-354**	-359	-350	29%	-361	-349	28%
S4	164	-370	**-364**	**-358**	-358	-350	**37%**	-364	-352	**31%**
			User time/run = **1.25** hrs		User time/run = 5.00 hrs			User time/run = 5.00 hrs		
			CPU time/run = 5.00 hrs		CPU time/run = 5.00 hrs			CPU time/run = 5.00 hrs		
F180_1	180	-378	**-359**	**-344**	-357	-340	12%	-351	-341	8%
F180_2	180	-381	**-364**	**-352**	-359	-345	19%	-362	-346	16%
F180_3	180	-378	**-368**	**-356**	-362	-353	13%	-361	-350	20%
R1	200	-384	**-366**	**-353**	-359	-345	21%	-355	-346	20%
R2	200	-383	**-368**	**-355**	-358	-346	24%	-360	-346	24%
R3	200	-385	**-369**	**-353**	-365	-345	20%	-363	-344	22%
3mse	179	-323	**-296**	**-285**	-289	-280	11%	-290	-279	14%
3mr7	189	-355	**-332**	**-319**	-328	-313	15%	-328	-316	8%
3mqz	215	-474	**-430**	**-414**	-420	-402	17%	-427	-410	7%
3no6	229	-455	**-429**	**-407**	-411	-391	25%	-420	-400	14%
3no3	258	-494	**-422**	**-404**	-412	-393	**11%**	-421	-402	**3%**
3on7	279	n/a	**-516**	**-500**	-512	-485	n/a	-515	-485	n/a

Table I: Three different sets of excremental data are presented here: *i)* our parallel local search framework (SS-Parallel), *ii)* the tabu guided spiral search (SS-Tabu), and *iii)* the genetic algorithms (GA$^+$). The *RI* Columns present the relative improvements of parallel local search over the single-thread local search and the genetic algorithm.

under the specific column headers (SS-Parallel, SS-Tabu, and GA$^+$). The bold-faced values indicate better performance in comparison to the other algorithms for corresponding proteins. The experimental results show that our SS-Parallel wins over SS-Tabu, and GA$^+$ on all 16 proteins of various length with significant margins on average search results.

D. Relative Improvement

The difficulty of improving energy level is increased as the improved energy level approaches to the lower bound of free energy. For example, if the lower bound of free energy of a protein is -100, the efforts to improve energy level from -80 to -85 is much less than that to improve energy level from -95 to -100 though the change in energy is the same (-5). Relative Improvement (RI) explains how close our predicted results to the lower bound of free energy with respect to the energy obtained from the state-of-the-art approaches.

$$\mathsf{RI} = \frac{E_t - E_r}{E_l - E_r} * 100\% \tag{5}$$

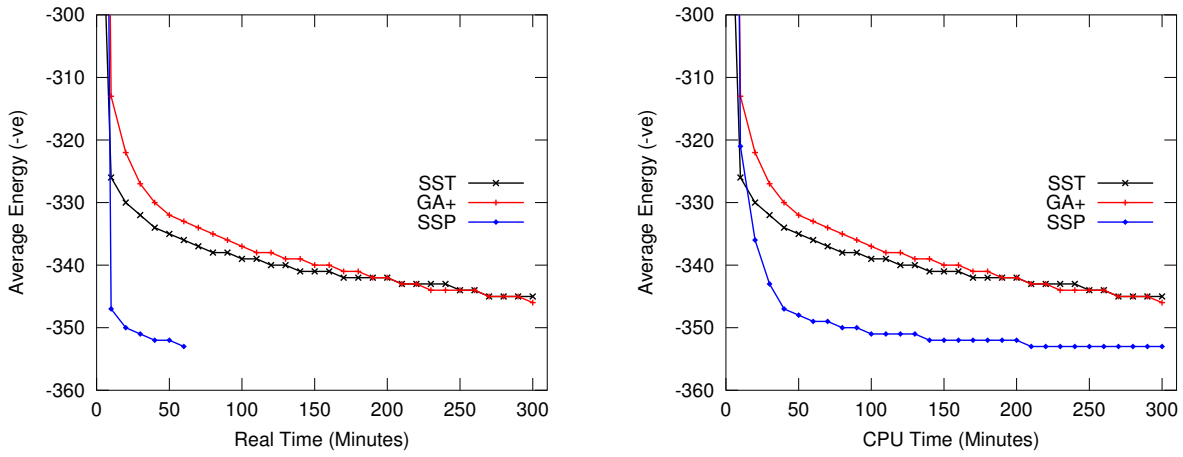
In Table I, we also present a comparison of improvements (%) on average conformation quality (in terms of free energy levels). We compare SS-Parallel (target) with SS-Tabu and GA$^+$ (references). For each protein, the *RI* of the target (t) w.r.t. the reference (r) is calculated using the formula in Equation 5, where E_t and E_r denote the average energy values achieved by the target and the reference respectively, and E_l is the lower bound of free energy for the protein in the HP model. We present the relative improvements only for the proteins having known lower bounds of free energy values. We test our new approach on 16 different proteins of various length. The bold-faced values are the minimum and the maximum improvements for the same column.

1) Improvement w.r.t. SS-Tabu: The experimental results in Table I, at column *RI* under SS-Tabu shows that our SS-Parallel is able to improve the search quality in terms of minimising the free energy level over all the 16 proteins considered for the test. The relative improvements with respect to SS-Tabu range from 11% to 37%.

2) Improvement w.r.t. GA$^+$: The experimental results in Table I, at column *RI* (relative improvement) under GA$^+$ shows that our SS-Parallel is able to improve the search quality in terms of minimising the free energy level over all 16 proteins considered for the test. The relative improvements with respect to GA$^+$ range from 3% to 31%.

E. Search Progress

We compare the search progresses of different approaches; SS-Tabu, GA$^+$, and SS-Parallel over real time. Figure 6-a shows the average energy values obtained with times by the algorithms for protein R1. The graph shows that the progress of SS-Parallel stops at 75 minutes (1.25 hours). As we mentioned earlier, we run parallel threads (four threads) in our SS-Parallel for 1.25 hours to keep total CPU time equals to five ($1.25 \times 4 = 5$) hours. From the graph, it is clear that multi-point local search with four parallel threads dramatically outperforms the local search and genetic algorithms within $\frac{1}{4}^{th}$ of the execution time.



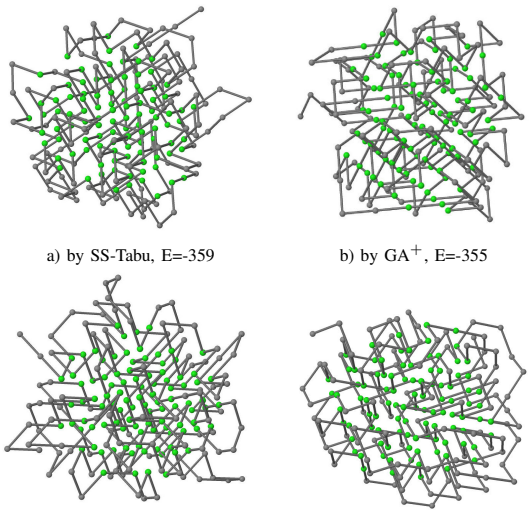
(a) (b)

Figure 6: Search progress for Protein R1 with (a) real time and (b) CPU time of 4 threads (4 x real time). SST, GA$^+$, and SSP represents tabu-based spiral search [9], genetic algorithms [8], and multi-point parallel spiral search respectively.

However, in Figure 6-b, we compare the search progresses of SS-Tabu, GA$^+$, and SS-Parallel over CPU time. The CPU time of SS-Parallel is calculated by summing up the individual times of all threads (time per thread $\times$ 4) in different instances.

F. On-lattice Simplified Structure

In Figure 7, we show the best structures obtained by SS-Parallel, SS-Tabu and GA$^+$ for protein $R1$. Each algorithm runs over a period of 5 hours equivalent of CPU time to achieve these results. However, the structure in Figure 7-d is collected from the literature [31], [32]. To view structures, we use *Jmol*[4]: an open-source Java viewer for chemical structures in 3D. The figures present only α-carbon positions of protein *R1* without side-chains.



a) by SS-Tabu, E=-359 b) by GA$^+$, E=-355

c) by SS-Parallel, E=-366 d) LBFE, E=-385

Figure 7: 3D structures of protein R1 obtained by different approaches.

[4]Jmol website: www.jmol.org

VII. Conclusion

In this paper, we present a multi-point parallel local search framework (SS-Parallel) that runs tabu-based hydrophobic-core directed local search in parallel threads. In our low-resolution *ab initio* method, we use hydrophobic-polar energy model and face-centred-cubic lattice for protein structure prediction. Collaboration and negotiation play vital roles for dealing with real world challenges. In our research, we try to adopt this analogy by considering each threads as a collaborator. We allow each thread to run for a pre-defined period of time. The threads are met in an assembly-point when finish their execution and donate or accept better solutions to proceed with. The SS-Parallel starts with a set of random initial solutions by distributing a sub set of solutions to different threads. The interim improved solutions are stored thread-wise and merged together when the threads finish. After removing the duplicates form the merged-solutions, a selected distinct set of solutions is considered for the next iteration. In our approach, multi-point start helps find some promising solutions. For the next working set of solutions from the merged-list, the most promising solutions are selected. Therefore, multi-point parallelism reduces the search space by exploring around the promising solutions in every iteration. The experimental results show that our new approach significantly improves over the results obtained by the state-of-the-art single-point search approaches. In future, we intend to apply our SS-Parallel in high resolution protein structure prediction.

Acknowledgment

We would like to express our great appreciation to the people managing the *Cluster Computing Services* at National ICT Australia (NICTA) and Griffith University. They helped a lot in preparing this article on time by taking care of our submitted jobs in clusters.

REFERENCES

[1] Adam Smith, "Protein misfolding," *Nature reviews drug discovery*, vol. 426, no. 6968, pp. 78–102, December 2003.

[2] C. M. Dobso, "Protein folding and misfolding," *Nature*, vol. 426, no. 6968, pp. 884–890, 2003.

[3] The Science Editorial, "So much more to know," *The Science*, vol. 309, no. 5731, pp. 78–102, July 2005.

[4] R. Bonneau and D. Baker, "*Ab initio* protein structure prediction: progress and prospects," *Annual Review of Biophysics and Biomolecular Structure*, vol. 30, no. 1, pp. 173–89, 2001.

[5] C. Rohl, C. Strauss, K. Misura, and D. Baker, "Protein structure prediction using ROSETTA," *Methods in enzymology*, vol. 383, pp. 66–93, 2004.

[6] J. Lee, S. Wu, and Y. Zhang, "Ab initio protein structure prediction," *From protein structure to function with bioinformatics*, pp. 3–25, 2009.

[7] Y. Xia, E. S. Huang, M. Levitt, and R. Samudrala, "*Ab initio* construction of protein tertiary structures using a hierarchical approach," *Journal of Mol. Biology*, 2008.

[8] M. A. Rashid, M. Hoque, M. A. H. Newton, D. Pham, and A. Sattar, "A new genetic algorithm for simplified protein structure prediction," in *AI 2012: Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science, 2012.

[9] M. A. Rashid, M. A. H. Newton, M. T. Hoque, S. Shatabda, D. Pham, and A. Sattar, "Spiral search: a hydrophobic-core directed local search for simplified PSP on 3D FCC lattice," *BMC Bioinformatics*, vol. 14, no. Suppl 2, p. S16, 2013.

[10] Y. Zhang and J. Skolnick, "The protein structure prediction problem could be solved using the current PDB library," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, no. 4, p. 1029, 2005.

[11] J. U. Bowie, R. Luthy, and D. Eisenberg, "A method to identify protein sequences that fold into a known three-dimensional structure," *Science*, vol. 253, no. 5016, p. 164, 1991.

[12] A. Torda, "Protein threading," *The proteomics protocols handbook*, pp. 921–938, 2005.

[13] K. T. Simons, R. Bonneau, I. Ruczinski, and D. Baker, "Ab initio protein structure prediction of CASP III targets using ROSETTA," *PROTEINS: Structure, Function, and Bioinformatics*, vol. Suppliment 3, pp. 171–176, 1999.

[14] D. Baker and A. Sali, "Protein structure prediction and structural genomics," *Science*, vol. 294, no. 5540, pp. 93–96, 2001.

[15] C. Levinthal, "Are there pathways for protein folding?" *Journal of Medical Physics*, vol. 65, no. 1, pp. 44–45, 1968.

[16] C. B. Anfinsen, "The principles that govern the folding of protein chains," *Science*, vol. 181, no. 4096, pp. 223–230, 1973.

[17] T. Hales, "A proof of the Kepler conjecture," *The Annals of Mathematics*, vol. 162, no. 3, pp. 1065–1185, 2005.

[18] M. T. Hoque, M. Chetty, and A. Sattar, "Protein folding prediction in 3D FCC HP lattice model using genetic algorithm," vol. 2007. IEEE Congress on Evolutionary Computation, 2007, pp. 4138–4145.

[19] K. F. Lau and K. A. Dill, "A lattice statistical mechanics model of the conformational and sequence spaces of proteins," *Macromolecules*, vol. 22, no. 10, pp. 3986–3997, 1989.

[20] B. Berger and T. Leightont, "Protein folding in the hydrophobic-hydrophilic (HP) model is NP-complete," *Journal of Computational Biology*, vol. 5, no. 1, pp. 27–40, 1998.

[21] F. Glover and M. Laguna, *Tabu search*. Kluwer Academic Pub, 1998, vol. 1.

[22] F. Glover, "Tabu search - part I ," *ORSA Journal on Computing*, vol. 1, no. 3, pp. 190–206, 1989.

[23] C. Thachuk, A. Shmygelska, and H. H. Hoos, "A replica exchange Monte Carlo algorithm for protein folding in the HP model," *BMC bioinformatics*, vol. 8, no. 1, p. 342, 2007.

[24] A. Tantar, N. Melab, and E.-G. Talbi, "A grid-based genetic algorithm combined with an adaptive simulated annealing for protein structure prediction," *Soft Computing-A Fusion of Foundations, Methodologies and Applications*, 2008.

[25] R. Unger and J. Moult, "A genetic algorithm for 3D protein folding simulations," Morgan Kaufmann Publishers. The 5th International Conference on Genetic Algorithms, 1993, p. 581.

[26] M. T. Hoque, "Genetic Algorithm for *Ab initio* protein structure prediction based on low lesolution Models," Ph.D. dissertation, Gippsland School of Information Technology, Monash University, Australia, Sep. 2007.

[27] H.-J. Böckenhauer, A. Z. M. D. Ullah, L. Kapsokalivas, and K. Steinhöfel, "A Local move set for protein folding in triangular lattice models," in *WABI*, ser. Lecture Notes in Computer Science, vol. 5251. Springer, 2008, pp. 369–381.

[28] G. W. Klau, N. Lesh, J. Marks, and M. Mitzenmacher, "Human-guided tabu search," in *The Eighteenth National Conference on Artificial Intelligence (AAAI-02)*, 2002.

[29] C. Blum, "Ant colony optimization: Introduction and recent trends," *Physics of Life reviews*, vol. 2, no. 4, pp. 353–373, 2005.

[30] V. Cutello, G. Nicosia, M. Pavone, and J. Timmis, "An immune algorithm for protein structure prediction on lattice models," *IEEE Trans on Evolutionary Computing*, vol. 11-1, pp. 101–117, 2007.

[31] M. Cebrián, I. Dotú, P. Van Hentenryck, and P. Clote, "Protein structure prediction on the face centered cubic lattice by local search," in *Proceedings of the 23rd national conference on Artificial intelligence - Volume 1*, 2008, pp. 241–246.

[32] S. Shatabda, M. A. H. Newton, D. N. Pham, and A. Sattar, "Memory-based local search for simplified protein structure prediction," in *ACM Conference on Bioinformatics, Computational Biology and Biomedicine*. ACM, 2012, pp. 345–352.

[33] I. Dotu, M. Cebrián, P. Van Hentenryck, and P. Clote, "On lattice protein structure prediction revisited," *IEEE Transactions on Comp. Bio. and Bioinformatics*, 2011.

[34] A. L. Patton, W. F. Punch III, and E. D. Goodman, "A standard GA approach to native protein conformation prediction." Int. Conf. on Genetic Algorithms, 1995.

[35] M. T. Hoque, M. Chetty, A. Lewis, A. Sattar, and V. M. Avery, "DFS-generated pathways in GA crossover for protein structure prediction," *Neurocomputing*, vol. 73, no. 13-15, pp. 2308–2316, 2010.

[36] M. T. Hoque, M. Chetty, A. Lewis, and A. Sattar, "Twin removal in genetic algorithms for protein structure prediction using low-resolution model," *Transactions on Computational Biology and Bioinformatics*, vol. 8, no. 1, pp. 234–245, 2011.

[37] A. D. Ullah, L. Kapsokalivas, M. Mann, and K. Steinhöfel, "Protein folding simulation by two-stage optimization," in *Computational Intelligence and Intelligent Systems*, ser. Communications in Computer and Information Science.

[38] A. D. Ullah and K. Steinhöfel, "A hybrid approach to protein folding problem integrating constraint programming with local search," *BMC bioinformatics*, vol. 11, no. Suppl 1, p. S39, 2010.

[39] M. Berrera, H. Molinari, and F. Fogolari, "Amino acid empirical contact energy definitions for fold recognition in the space of contact maps," *BMC bioinformatics*, vol. 4, no. 1, p. 8, 2003.

[40] T. Jiang, Q. Cui, G. Shi, and S. Ma, "Protein folding simulations of the hydrophobic–hydrophilic model by combining tabu search with genetic algorithms," *The Journal of chemical physics*, vol. 119, p. 4592, 2003.

[41] M. A. Rashid, S. Shatabda, M. A. H. Newton, M. T. Hoque, D. N. Pham, and A. Sattar, "Random-walk: a stagnation recovery technique for simplified protein structure prediction," in *ACM Conference on Bioinformatics, Computational Biology and Biomedicine*, 2012, pp. 620–622.

[42] K. Yue and K. A. Dill, "Sequence-structure relationships in proteins and copolymers," *Physical Review E*, vol. 48, no. 3, p. 2267, 1993.

[43] N. Lesh, M. Mitzenmacher, and S. Whitesides, "A complete and effective move set for simplified protein folding," in *Research in comp. mol. biology (RECOMB)*, 2003.