Parallel Artificial Bee Colony Algorithm Approaches for Protein Structure Prediction Using the 3DHP-SC Model

César Manuel Vargas Benítez and Heitor Silvério Lopes

Abstract. This paper reports the use of the Artificial Bee Colony algorithm (ABC) for protein structure prediction using the three-dimensional hydrophobic-polar model with side-chains (3DHP-SC). Two parallel approaches for the ABC were implemented: a master-slave and a hybrid-hierarchical. Experiments were done for tuning the parameters of the ABC, as well as to adjust the load balance in a cluster-based computing environment. The performance of the parallel models was compared with a sequential version for 4 benchmark instances. Results showed that the parallel models achieved a good level of efficiency and, thanks to the co-evolution effect, the hybrid-hierarchical approach improves the quality of solutions found.

Keywords: Swarm Intelligence, Parallel Computing, Artificial Bee Colony, Protein Folding, 3DHP-Side Chain.

1 Introduction

The specific biological function of a protein depends on its three-dimensional shape, which, in turn, is a function of its primary structure (linear sequence of amino acids). Failure to fold into the intended three-dimensional shape usually leads to proteins with different properties that can become inactive or even harmful to the organism. Therefore, better understanding the protein folding process can result in important medical advancements and development of new drugs. Only a small amount of such proteins have its three-dimensional structure known. This fact is due to the cost and difficulty in unveiling the structure of proteins, from the biochemical point of view. This is an important motivation for developing computational methods for predicting the structure of these proteins. Computer science has an important role here,

César Manuel Vargas Benítez · Heitor Silvério Lopes
Federal University of Technology Paraná (UTFPR), Curitiba, PR, Brazil
e-mail: cesarvargasb@gmail.com, hslopes@utfpr.edu.br

M. Essaaidi et al. (Eds.): Intelligent Distributed Computing IV, SCI 315, pp. 255–264.
springerlink.com

proposing models for studying the Protein Structure Prediction (PSP) problem. The simplest computational model for the PSP problem is known as Hydrophobic-Polar (HP) model, both in two (2D-HP) and three (3D-HP) dimensions [4]. Although simple, the computational approach for searching a solution for the PSP using the HP models was proved to be *NP*-complete. This fact has motivated the development of many evolutionary computation approaches for dealing with the problem [8].

Currently, there are a variety of algorithms inspired by the bee foraging behavior found in literature, such as: Bee System [12], Honey Bee Algorithm [10], and the Artificial Bee Colony Algorithm (ABC) [6], among others. The ABC is a population-based optimization algorithm based on the foraging behavior of honeybee swarms. Currently, the ABC is one of the most widely used bee algorithm for problem solving. Some successful applications found in the literature using the ABC include, for instance, multi-objective problems [11], and template matching in digital images [3].

To the best of our knowledge, the current work is the first to apply parallel ABC approaches to the PSP using the 3DHP-SC model. This work compares the performance of the standard ABC sequential algorithm with two parallel models, namely: Master-Slave ABC (MS–ABC) and Hybrid Hierarchical ABC (HH–ABC). Experiments were done using four protein sequences. Therefore, one can evaluate the benefits of the parallelized approaches for the ABC algorithm in comparison with the regular sequential version.

2 Artificial Bee Colony Algorithm

Social insects, such as ants and bees, spend most of their life foraging for food. Honey bee colonies have a decentralized system to collect the food and can adjust the searching pattern precisely so that to enhance the collection of nectar.

The ABC algorithm [6] works with a swarm of n solutions x (food sources) of dimension d that are modified by artificial bees. The bees aim at discovering places of food sources v (locations in the search space) with high amount of nectar (good fitness). In the ABC algorithm, a colony of bees is divided into three groups: employed bees (forager bees) that exploit the neighborhood of their food sources selecting a random solution to be perturbed; onlooker bees (observer bees) that are placed on the food sources by using a probability based selection process; and scouts bees that randomly fly in the search space without guidance. As the nectar amount of a food source increases, the probability value P_i with which the food source is preferred by onlookers also increases. If the nectar amount of a new source is higher than that of the previous one in their memory, they update the new position and forget the previous one. If a solution is not improved by a predetermined number of trials controlled by the parameter *limit*, then the food source is abandoned by the corresponding employed bee and it becomes a scout bee. Each search cycle consists of moving the employed and onlooker bees onto the food sources and calculating their nectar amounts; and determining the scout bees and directing them onto possible food sources. The ABC pseudo-code is shown in Algorithm 1. The ABC algorithm

attempts to balance exploration and exploitation by combining local search methods, carried out by employed and onlooker bees, with global search methods, managed by scout bees.

Two remarkable features of such swarm-based systems are self-organization and decentralized control that leads to an emergent behavior. Emergent behavior is a property that emerges through interactions among system components (bees) and it is not possible to be achieved by any of the components of the system by itself. Interactions between bees occurs through a waggle dance performed inside the hive. It is used to recruit other bees to exploit a food source. The intensity of the waggle dance performed by a bee is proportional to the quality of the food source. Hence, best food sources lead to a more intense waggle dance, and this can reinforce the exploitation of the best food locations.

3 The 3D-HP Side-Chain Model (3D-HP-SC)

The Hydrophobic-Polar (HP) model proposed by Dill [4] divides the 20 standard amino acids into two classes, according to their affinity to water: Hydrophilic (or Polar) and Hydrophobic. When a protein is folded in its native conformation, most hydrophobic amino acids tend to group themselves in the inner part of the protein, in such a way to get protected from the solvent by the polar amino acids that are positioned preferably outwards. Therefore, a hydrophobic core is usually formed,

Algorithm 1 Pseudo-code of Artificial Bee Colony Algorithm (ABC).

Parameters: n, $limit$
Objective function $f(x)$, $x = [x_1, x_2, ..., x_d]^T$
Initialize the food positions randomly x_i $i = 1, 2, ..., n$
Evaluate fitness $f(x_i)$ ***of the Bees***
while stop condition not met **do**
  Employed phase:
    Produce new solutions with k, j and ϕ at random
      $v_{ij} = x_{ij} + \phi_{ij} \cdot (x_{ij} - x_{kj})$ $k \in \{1, 2, ..., n\}, j \in \{1, 2, ..., d\}, \phi \in [0,\ 1]$
    Evaluate solutions
    Apply greedy selection process for the employed bees
  Onlooker phase:
    Calculate probability values for the solutions x_i
      $P_i = \frac{f_i}{\sum_{j=i}^{n} f_j}$
    Produce new solutions from x_i selected using P_i
    Evaluate solutions
    Apply greedy selection for the onlookers
  Scout phase:
    Find abandoned solution: If limit exceeds, replace it with a new random solution
  Memorize the best solution achieved so far
end while
Postprocess results and visualization

especially in globular proteins. In this model, the folding or conformation of a protein is represented in a lattice, usually square (for the 2D-HP) or cubic (for the 3D-HP). Both 2D-HP and 3D-HP models have been frequently explored in the recent literature [8].

Since the expressiveness of the HP models is very poor, from the biological point of view, a further improvement is to include a bead to represent the side-chain (SC) of the amino acids [7]. Therefore, a protein is modeled by a common backbone and a side-chain, either Hydrophobic (H) or Polar (P).

To compute the energy of a given conformation using this model, [7] proposed an equation that takes into account the spatial position of the side-chain and the backbone, as shown in Equation 1.

$$H = \varepsilon_{bb} \sum_{i=1,j>i+1}^{N} \delta_{r_{ij}^{bb}} + \varepsilon_{bs} \sum_{i=1,j\neq i}^{N} \delta_{r_{ij}^{bs}} + \varepsilon_{ss} \sum_{i=1,j>i}^{N} \delta_{r_{ij}^{ss}} \quad (1)$$

In this equation, ε_{bb}, ε_{bs} and ε_{ss} are the weights of the energy for each type of interaction: backbone/backbone (BB-BB), backbone/side-chain (BB-SC) and side-chain/side-chain (SC-SC) In a chain of n residues, the distance (in the three-dimensional space) between the i^{th} and j^{th} residue interacting with each other is represented by r_{ij}^{**}. For the sake of simplification, in this work we used unity distance between residues ($r_{ij}^{**} = 1$). Therefore, δ is an operator that returns 1 when the distance between the i^{th} and j^{th} side-chain is the unity, or 0 otherwise. During the folding process, interactions between amino acids take place and the free energy of the conformation tends to decrease, conversely, the conformation tends to converges to its native state, in accordance with the Anfinsen's thermodynamic hypothesis [1]. In this work we consider the symmetric of H to turn the problem to a maximization.

It is known that the number of hydrophobic contacts (HnC) is inversely proportional to the free-energy of a given conformation. Therefore, any algorithmic procedure for the folding that maximizes the HnC will, conversely, take the molecule to the smallest possible free-energy state.

4 Solution Encoding and Fitness Function

The encoding of the candidate solutions (Bees) should be carefully implemented because it can have a strong influence not only in the size of the search space, but also in the hardness of the problem. To model the PSP, the solution represents the spatial position of the amino acids chain in a lattice, using internal coordinates [8]. In this coordinate system, given conformation is represented by a set of movements of one amino acid relative to its predecessor in the chain. Therefore, for a protein with n amino acids, a folding encoded in an artificial bee will have $n-1$ elements. As mentioned before, in the 3DHP-Side Chain model, the amino acids of the protein are represented by a backbone (BB) and a side-chain (SC), either hydrophobic (H) or polar (P). In the three-dimensional space there are five possible relative movements for the backbone (**L**eft, **F**ront, **R**ight, **D**own, **U**p), and other five for the side-chain,

relative to the backbone (**l**eft, **f**ront, **r**ight, **d**own, **u**p). The combination of possible movements for backbone and side-chain gives a set of 25 possibilities. Instead of the traditional binary alphabet, a set of 25 numbers and letters was used to encode the chromosome. More details about the encoding was reported in a previous work [2].

To evaluate a possible solution encoded in a bee, the chromosome of relative coordinates must be previously decoded into a set Cartesian coordinates representing backbones and respective side-chains. The fitness function used in this work was first proposed by [9], and later adapted to the 3DHP-SC by [2]. This function has three terms (as shown in Equation 2). The first one is relative to the free-energy of the conformation H (see Equation 1), the number of collisions NC (number of points in the three-dimensional lattice that is occupied by more than one element) and the penalty weight (*PenaltyValue*) The following terms represent the gyration radius of the hydrophobic and hydrophilic side-chains, respectively. Radius of gyration is a measure of compactness of a set of points (in this case, the side-chains of the amino acids in the lattice). This is done in such a way to favor conformations in which hydrophobic side-chains are compacted within the core, and polar side-chains are pushed outwards of the conformation. Equations 3, 4, and 5 show how it is computed.

$$fitness = [H - (NC \cdot PenaltyValue)] \cdot RadiusG_H \cdot RadiusG_P \tag{2}$$

$$RG_{aa} = \sqrt{\frac{\sum_{i=1}^{N_{aa}}[(x_i - \overline{X})^2 + (y_i - \overline{Y})^2 + (z_i - \overline{Z})^2]}{N_{aa}}} \tag{3}$$

$$RadiusG_H = maxRG_H - RG_H \tag{4}$$

$$RadiusG_P = \begin{cases} 1 & \text{if } (RG_P - RG_H \geq 0) \\ \frac{1}{1-(RG_P - RG_H)} & \text{else} \end{cases} \tag{5}$$

5 Parallel Implementations of the ABC Algorithm

The first parallel approach was the Master-Slave (MS–ABC). It is a global single-population system where a master process distributes the processing load into several slave processes, each one running in a different processor of a cluster-based processing environment. The master is responsible for initializing the food positions randomly, generating new solutions in the algorithm phases (Employed, Onlooker and Scout – see Algorithm 1), applying a greedy selection procedure, and distributing bees (solutions) to slaves. Slaves, in turn, are responsible for computing the fitness function of received bees. The computation of the fitness function is very intensive, thus justifying the need to distribute computing. In Algorithm 1 the bold and italic instructions are where the parallelization takes place. Figure 1(a) illustrates the MS–ABC model. As shown, the MS–ABC model has exactly the same functionalities of the sequential version of the ABC.

The second parallel approach, the Hybrid Hierarchical (HH–ABC) was implemented has two levels. In the upper level, there are multiple-population coarse-grained islands (such as a multi-hive model). In the lower level, there are global single-population master-slaves (such as the MS–ABC). These two levels can be seen as an hierarchy of hives. This combination aims at taking advantage of the benefits of both models in a single approach. At the lower level there is a master process that distributes the computational effort into several slaves. In the upper level each population (master and corresponding slaves) is seen as an island that works independently. A migration policy defines the sporadic migrations that occur between islands. It has four parameters: Migration gap (number of generations between successive migrations), Migration rate (number of bees that will migrate at each migration event), Selection/Substitution criteria for migrants, and topology of connectivity between islands. Figure 1(b) illustrates the HH–ABC model with four hives at the upper level and n slaves per hive at the lower level.

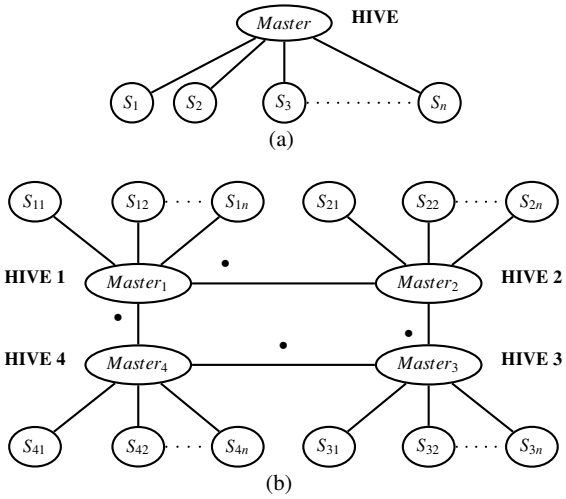

Fig. 1 MS–ABC model (a) and HH–ABC model (b)

6 Computational Experiments and Results

The experiments reported in this work were run in a cluster of networked computers with 124 processing cores, running Linux. The software was developed in ANSI-C programming language, using the Message Passing Interface (MPI) MPICH2 package for the communication between processes [1].

Similarly to other evolutionary computation algorithms, the ABC also has parameters to adjust and there is no specific procedure for doing this for a given problem.

[1] Available at: http://www.mcs.anl.gov/research/projects/mpich2/

In this work, the following parameters of the ABC algorithm were adjusted using a benchmark sequence: Colony size (*Colonysize*), Maximum Cycle Number (*MCN*), and the percentage of Employed (n_e) and Onlookers Bees (n_o). Three values for *Colonysize* and *MCN* were tested, respectively {250, 500, 1000} and {6000, 3000, 1500}, thus keeping constant the number of function evaluations (done by employed and onlooker bees). For each combination, three pairs of values for n_e and n_o were tested, as follows: {75%, 25%}, {50%, 50%}, and {25%, 75%}, resulting in 9 sets of parameters. For each set, a total of 100 independent runs was done, and the average of the best fitness found in the runs was used as criterion of quality. The best performing set of parameters was: *Colonysize*=250, *MCN*=6000, n_e=50% and n_o=50%. These values were fixed for the remaining experiments.

In a cluster-based parallel processing environment, the time needed to transmit control messages and data between processes through the network is significantly high, when compared with the processors' speed. Hence, it is necessary to establish a suitable balance between the processing load of each processor and the amount of communication between processes. After several preliminary experiments, we adjusted *Colonysize* and the number of slaves (for each master-slave) in such a way to establish the best trade-off between communication and processing.

6.1 Migration and Coevolution in the HH–ABC Model

At the upper level, the HH–ABC model uses a topology with 4 islands connected by a unidirectional ring. This number of islands was set due two facts: availability of hardware (allocating one processing core per process) and the expectance of observing the coevolution effect as soon as possible. From the literature of parallel evolutionary algorithms, it is known that the topology is an important factor in the performance of the algorithm because it determines how fast a good solution disseminates to other islands (hives). A small number of islands in a ring topology accelerate the coevolution process, thus requiring a reduced number of cycles/generations.

The Migration gap was set to 10% of Maximum Cycle Number (*MCN*), that is, a migration event occurs at every 600 cycles (recall that the selected *MCN* is 6000 cycles). The migration rate was set to 2 bees, such that a copy of the best bee and a random one are emigrated to the next island of the topology and replace randomly chosen bees of the receiving hive. Two versions were made changing the amount of bees per hive (*Colonysize*). One uses 250 bees per island (HH–ABC_1) and another uses 63 bees per island (HH–ABC_2). The HH–ABC_1 and HH–ABC_2 totalizes 1,000 and 252 bees, respectively. The reason for these two approaches is that the sequential ABC model uses 250 bees and, if we divide this population into four hives, we have 62.5 bees per hive (since only integer numbers make sense, we set to 63 bees per hive). Indices 1 and 2 indicate only different *Colonysize*.

At the lower level, the HH–ABC uses a MS–ABC configuration with one master and 25 slaves. The reason to set this configuration is the same as for the number of islands.

The effect of coevolution between hives is reflected after a migration cycle. When a good quality bee immigrates to a hive, it not only improves the local best solution, but also, through recombination, induces a further improvement of quality in the hive.

The three parallel approaches (MS–ABC, HH–ABC_1 and HH–ABC_2) were run 100 times, keeping fixed all other parameters. Results are not shown here due to space restrictions. However, we observed that the HH–ABC (HH–ABC_1 and HH–ABC_2) approach leads to better results when compared with the MS–ABC (recall that the MS–ABC and sequential versions have exactly the same functionalities). These results strongly suggests that the coevolution effect of the migration between hives is advantageous for the parallel ABC algorithm, and it is used henceforth.

6.2 Benchmark Results and Discussion

In our experiments, four synthetic 27 amino acids-long sequences were used as benchmark, as shown in the columns of the first row of the Tables 1 and 2. These sequences have been used by other researchers for the 3DHP model [13, 5], and to the best of our knowledge, these sequences were used for the first time by [2] for the 3DHP-SC model. Since the ABC algorithm is an stochastic algorithm, experiments were run 100 times with different random seeds. The performance of each approach takes into account the best found solution an the processing time. The MS–ABC model was run with 50 slaves and 250 bees. The HH–ABC was run with 25 slaves per hive (4) and, therefore, each slave computes the fitness function for 10 bees in the HH–ABC_1, and 2 or 3 bees in the HH–ABC_2. Results are shown in Tables 1 and 2.

Comparing the MS–ABC and the ABC (sequential), it is noticed that the quality of solutions was almost the same with statistically insignificant differences. How-

Table 1 Statistical results for benchmark sequences S_1 and S_2

Sequence	S_1 $(PH)^3H^2P^2(HP)^2P^{10}H^2P$		S_2 $PH^2P^{10}H^2P^2H^2P^2HP^2HPH$	
Model	*fitness*	Tp(s)	*fitness*	Tp(s)
ABC (Sequential)	523.27±26.32	58052.15	529.48±23.44	57221.32
MS–ABC	524.15±25.04	1554.38	530.51±22.76	1546.22
HH-ABC$_1$	581.83±35.25	1849.98	673.95±27.42	1848.15
HH-ABC$_2$	517.21±27.04	886.23	588.74±19.86	902.56

Table 2 Statistical results for benchmark sequences S_3 and S_4

Sequence	S_3 $H^4P^5HP^5H^3P^8H$		S_4 $H^3P^2H^4P^3(HP)^2PH^2P^2HP^3H^2$	
Model	*fitness*	Tp(s)	*fitness*	Tp(s)
ABC (Sequential)	621.11±35.74	57358.08	938.36±30.06	57147.24
MS–ABC	620.05±36.86	1547.44	940.91±29.31	1406.21
HH-ABC$_1$	772.36±20.22	1758.01	1002.59±22.21	1788.19
HH-ABC$_2$	711.37±14.48	871.44	989.55±18.59	926.82

ever, the MS–ABC model achieved a significantly speedup ($\sim$40 for sequence S4) and efficiency ($\sim$0.8 for sequence S4). In fact, speedup and efficiency are a direct consequence of the load balancing between master and slaves. Ideally, efficiency should be close to the unity (but not above it). However, in practice, this is not always possible, since processors are not used 100% of time for processing, but also for communication, memory allocation and other tasks of the underlying operating system. This is due to the communication overload caused in the master processor.

The difference between HH–ABC_1 and HH–ABC_2 is the size of their hives. Results indicate that the higher the hive, the better the quality of results. However the computational cost increases linearly with the size of the hive.

7 Conclusions

This paper reports the first work using parallel approaches of the ABC algorithm for the PSP problem using the 3DHP-SC model. PSP is still an open problem in Bioinformatics. Therefore, an important contribution of this work are the results regarding this issue. The use of parallel computing is justified in this work due to the complexity of the fitness function.

Experiments showed that the migration between hives leads to better results due to the effect of coevolution. The *Colonysize* (number of bees) per hive has a significant influence in the quality of solutions. Results suggested that larger colony leads to better results than the smaller ones. Further work will focus on other topologies (i.e. hypercubes, toroidal mesh and fully connected), as well as a deep study of the migration policy between hives. Since the experiments were performed using a cluster computing environment, the communication overhead takes significant influence upon the computational time. Consequently, an appropriate load balance procedure have to be done. To overcome such drawback, future work will consider the use of reconfigurable computing and General-Purpose Graphics Processing Units (GPGPU) to accelerate processing.

We believe that this work is an useful contribution to this area of research, since that the directions pointed out can be useful for other Swarm Intelligence approaches. Future work will also investigate parallel versions of other Swarm Intelligence, such as Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), and Firefly Algorithm (FA), so as to compare with the parallel ABC presented in this study. Overall, results were good enough and promising to support the continuity of the work.

References

1. Anfinsen, C.B.: Principles that govern the folding of protein chains. Science 181 (1973)
2. Benítez, C.M.V., Lopes, H.S.: A parallel genetic algorithm for protein folding prediction using the 3DHP side-chain model. In: Proc. of IEEE Congress on on Evolutionary Computation, pp. 1297–1304 (2009)

3. Chidambaram, C., Lopes, H.S.: A new approach for template matching in digital images using an artificial bee colony algorithm. In: World Congress on Nature and Biologically Inspired Computing, pp. 146–151 (2009)
4. Dill, K.A., Bromberg, S., Yue, K., et al.: Principles of protein folding - a perspective from simple exact models. Protein Science 4(4), 561–602 (1995)
5. Guo, Y.-Z., Feng, E.-M.: The simulation of the three-dimensional lattice hydrophobic-polar protein folding. Journal of Chemical Physics 125, 234–703 (2006)
6. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical report, Department Computer Engineering Department, Erciyes University (2005)
7. Li, M.S., Klimov, D.K., Thirumalai, D.: Folding in lattice models with side chains. Computer Physics Communications 147(1), 625–628 (2002)
8. Lopes, H.S.: Evolutionary algorithms for the protein folding problem: A review and current trends. In: Computational Intelligence in Biomedicine and Bioinformatics, vol. I, pp. 297–315. Springer, Heidelberg (2008)
9. Lopes, H.S., Scapin, M.P.: An enhanced genetic algorithm for protein structure prediction using the 2D hydrophobic-polar model. In: Talbi, E.-G., Liardet, P., Collet, P., Lutton, E., Schoenauer, M. (eds.) EA 2005. LNCS, vol. 3871, pp. 238–246. Springer, Heidelberg (2006)
10. Nakrani, S., Tovey, C.: On honey bees and dynamic server allocation in internet hosting centers. Adaptive Behavior 12(3-4), 223–240 (2004)
11. Pawar, P.J., Rao, R.V., Shankar, R.: Multi-objective optimization of electro-chemical machining process parameters using artificial bee colony (ABC) algorithm. In: Advances in Mechanical Engineering (AME 2008) (December 2008)
12. Sato, T., Hagiwara, M.: Bee system: Finding solution by a concentrated search. In: Proc. of the IEEE Int. Conf. on Systems, Man, and Cybernetics, vol. 4(C), pp. 3954–3959 (1997)
13. Unger, R., Moult, J.: A genetic algorithm for 3D protein folding simulations. In: Proc. 5^{th} Annual Int. Conf. on Genetic Algorithms, pp. 581–588 (1993)