Modeling of 2D Protein Folding using Genetic Algorithms and Distributed Computing

Andriy Sadovnychyy

DMAS, DCNI, UAM Cuajimalpa,
Artificios 40, col. Hidalgo, Delegación Álvaro Obregón,
México D. F., C.P. 01120, Mexico
asadovnychyy@gmail.com

Abstract. In this work, it is presented an application of parallel programming for studying of a scalable problem in the area of bioinformatics (Protein Folding) using a genetic algorithm. The properties of proteins depend on its configuration in space. Calculation of its configuration is a hard problem so we used some approximations models like 2D square lattice. For finding an optimal configuration of the protein in the space (configuration with minimal energy) genetic algorithm is used. It makes it possible to find an optimal configuration several times faster than a full calculation of interaction between atoms. Disadvantage of genetic algorithms is computation load. Therefore the parallel genetic algorithms are applied in this work. Parallel genetic algorithms operations (like mutation, crossover and fitness) are realized in parallel mode. For this the multi-agent system are useful. They make it possible to realize many independent functions in parallel mode.

Keywords: Protein folding, PH model of protein, genetic algorithm, parallel computing.

1 Introduction

There are many problems of bioinformatics, such as folding, docking and molecular design can be classified as "difficult problem of optimization". This family of problems is those for which not guaranteed to find the best solution in a reasonable time. Therefore the term NP-hard problem is used in the context of the complexity of algorithms. Recently, several studies have shown that hybrid algorithms, metaheuristics and parallel patterns have improved the search optimization methods, resulting in quite acceptable solutions, and even robust, which solved many increasingly large and complex problems with tolerable computing time [1, 2]. Examples of such problems are the problems which we deal with. This work proposes the development of a conceptual and computational infrastructure that allows integration through metaheuristics and hybrid algorithms, key models, heuristics and algorithms for the study of biologically interesting problems. In

this work we postulate that many of these problems can be effectively modeled through a conceptual and computational infrastructure based on multi-agent systems (MAS) that supports the work of algorithms based on metaheuristics.

Using techniques for developing parallel models allows solving large and complex problems in tolerable computing times. But to use all their advantages, parallel genetic algorithms must submit all data structures into a specific model. We have to develop computational algorithms using parallelization techniques. All tools can be heterogeneous and distributed. It means that the simulation system can use the advantages of each of its elements assigning the task to more suitable block. Using metaheuristic implemented through the interaction of agents in the computing infrastructure allows interrelate the properties of all molecules with the activity or function to be optimized, generating a functional mathematical approach to describe such relationship. The simulations of protein folding have the benefits of using heterogeneous systems. Applying this kind of processing power changes the whole dynamic of the simulation and could significantly reduce the time required to conduct research. In this work we develop the methods, algorithms and computational model structures that allow, through metaheuristics and hybrid algorithms, undertaking the study of biologically interesting problems mentioned above.

2 Methods

The modeling of physics and chemistry processes is very complex. This problem can be studied through alternative strategies, such as methods, techniques and models of intelligence artificial (IA), science and computer engineering, and sciences information, among other disciplines. It is noteworthy that such strategies have left aside those traditional approaches in physics and chemical commonly biological science researchers have used as study models [3]. Depending on empirical knowledge engaged in theoretical model, this relationship will be more successful to reproduce and predict the properties of a complex system of this nature. In this sense, we will mention some of the problems of biological interest that we intend to address in this work and specify why this complexity and strategies have been employed to get a solution.

The folding or folding of a protein is the physical process by which these biomolecules are rearranged in three-dimensional structure. The prediction of protein folding has been commonly addressed through combination of a model for the random generation of conformations proteins in a 2D or 3D space (hydrophobic-hydrophilic model), and an approximation algorithm for finding the closest conformation native state protein model. Both algorithms were used a criterion of minimizing some function of free energy. Genetic algorithm is one of the most commonly used techniques for approximation [4]. When this technique is used, the prediction of three-dimensional structure of a protein can be formulated as a search through its conformation space to find a global minimum of the energy state.

The multi-agent systems paradigm is very effective in designing a methodology for cover the entire spectrum of construction of a simulation, from design to development, implementation and control (dynamic) runtime [5, 6]. Critical points of biological systems - concerned with structures, activities and interactions - can be captured directly by abstractions that are "Kept alive" design at run time, supported by appropriate infrastructure. The simulation and modeling can then be framed as an experiment in online or in simulation system, where researchers can observe and interact dynamically with the system and its environment through changes in their structures, or directly modifying the global biological process.

Agent-based systems and multi-agent systems (MAS) are considered strategies to manage the correct level of abstraction modeling and building complex systems. In fact, biological systems exhibit all the characteristics of complex systems. However, as noted in [1], a biological system is much more than a complex system, it is a system consistent, as it implies levels of functionality, localization, individualization and therefore specialization, dimensionality (micro and macro levels) and other scenarios for these levels of operation.

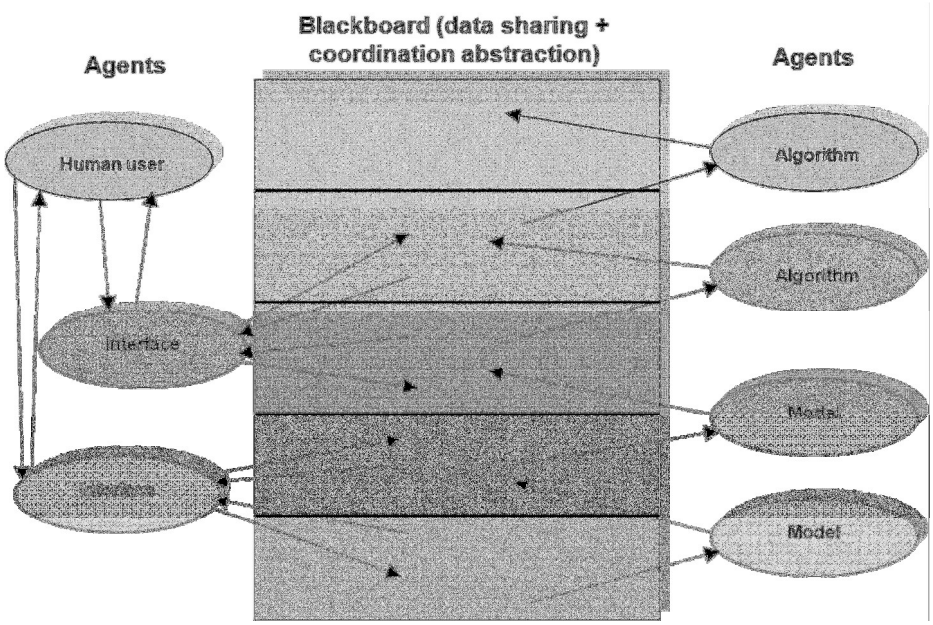

Fig. 1. Principal structure of simulation system.
Interaction among agents is realized indirectly through a blackboard, which represents the communication coordination abstraction.

In this work we will use an infrastructure for modeling MAS biological systems strongly based on location, distribution and interaction (communication) of the components of the system and therefore provide for better solutions compared to generating strategies traditionally used. We will also make an analysis from a viewpoint theory to explain the reasons why the infrastructure is better and which of its properties can be used in other evolutionary systems.

3 Multi-agent System

The principal structure of simulation system can be seen in Figure 1. There its main components, the agents and the blackboard, are presented. As can be seen in this figure, the architecture provides three types of agents: model agents, algorithm agents and interface agents [8]. According to this Multi-Agent System architecture, the interaction among agents is realized by indirect communication, through a communication-coordination abstraction, represented in this figure by the blackboard.

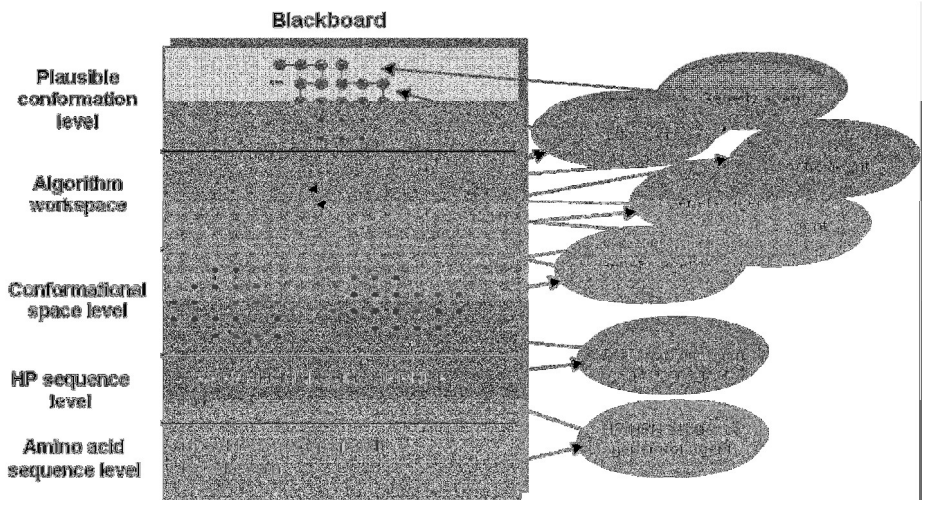

Fig. 2. Semantics assigned to the different blackboard levels and types of agents.

Figure 2 shows the semantics assigned to the different blackboard levels and types of agents when protein folding is modeled. Evolution, the bioinformatics framework, allows inducing folding of molecular-chain representations with the aim to explore both the behavior and efficiency of EA by following the most relevant variables of chain folding. Regarding the software development, our work will progress by using the bioinformatics framework as a scaffold or mainframe devoted to the development and testing of

gradually more complex and robust EA and their associated parameters, with more realistic representations of the problem and with improved fitness functions. This framework would also allow adding functionalities for the user to tract results and algorithm efficiency, to interactively bias the conformational search with biological sense, or to perform graphical and numerical analysis.

4 Experimental Results

The goal of experiments is to compare the performance of our simulation system with similar simulations and others previously published. Also to see if our simulation system can shed some light in the study of how proteins tend to fold and if there is some phenomena that could be identified and explained.

The HP-48 problem. This problem is defined by the sequence

$HP\text{-}48 = P_2H(P_2H_2)_2P_5H_{10}P_6(H_2P_2)_2HP_2H_5 =$
$PPHPPHHPPHHPPPPPHHHHHHHHHH$

In this problem the optimal sequences have a square shape in the H beads with several options for the P beads. This problem was reported in [9] as very difficult.

According to the results in previous sections, this problem is likely to be difficult simply because of this square shape. Any algorithm should have problems here.

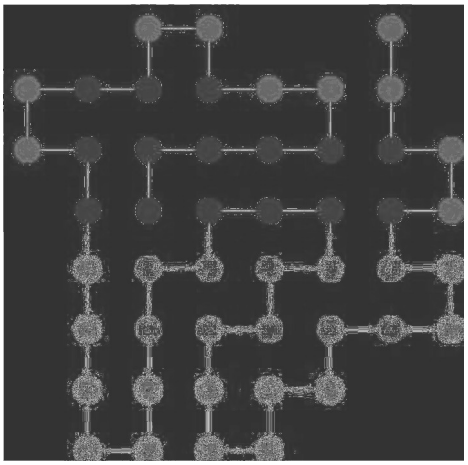

Fig. 3. A near optimal configuration for HP-48.

Figure 3 shows one nearly optimal configuration found. One can see that the shape of the H beads is more or less close to a rounded square. This is the kind of shapes that are more easily found by our algorithm and we think that this is also the one that is more likely to be present in nature. The 2D square model induces a bias that makes it difficult

to find square shaped optima. So we think that it is not good to try and solve this kind of problems. It should suffice with the use of more rounded shapes to be able to study the traits present in real life protein folding.

The genetic algorithm for find the optimal configuration was used. Each of gen presents the direction of connection in the protein structure. The gen shows three directions: straight, left and right. On basis of this information we construct the model protein. We start construct the model from the first amino acid and first direction to straight, then we continue construct in concordance with the chromosome.

Figure 4 shows the principal structure of genetic algorithm. On phases "Initial population", "Crossover and mutation", "Selection" and "Termination" the algorithm works with whole population. On these stages the processing of each chromosome is depended of others.

On phase "Fitness" the algorithm calculates the level (fitness function) of each chromosome. On this stage the processing of chromosome is independent. And we calc the fitness function for each chromosome simultaneously, en parallel.

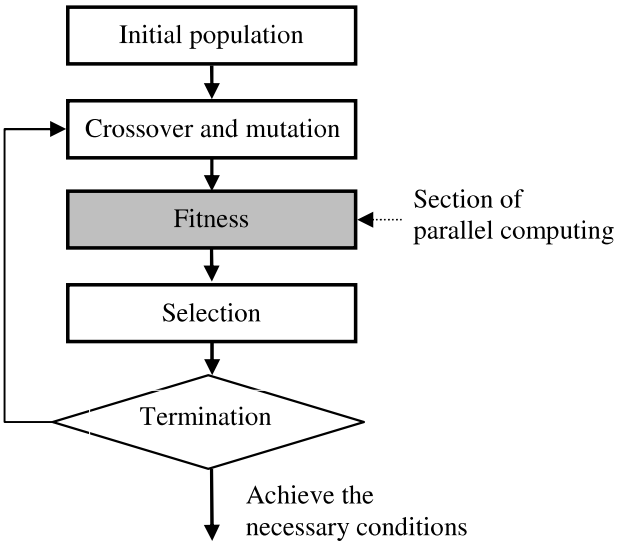

Fig. 4. Principal structure of genetic algorithm.

The main algorithm waits for termination of all parallel calculations, receives all results (values of each fitness functions) and makes "selection".

During the phase "selection", the algorithm assigns the rank for each genome and selects pairs for "Crossover" stage.

Figure 5 shows cluster's structure that we are used for realize computing. The cluster has Server, which controls and distributes tasks for nodes. Each node has 2 CPUs with 8

cores (16 cores for node) and work like SMP machine. We consider that each core like a separate CPU.

We use Open MPI [10, 11] library and Open MP API [11, 12] for computing parallelization.

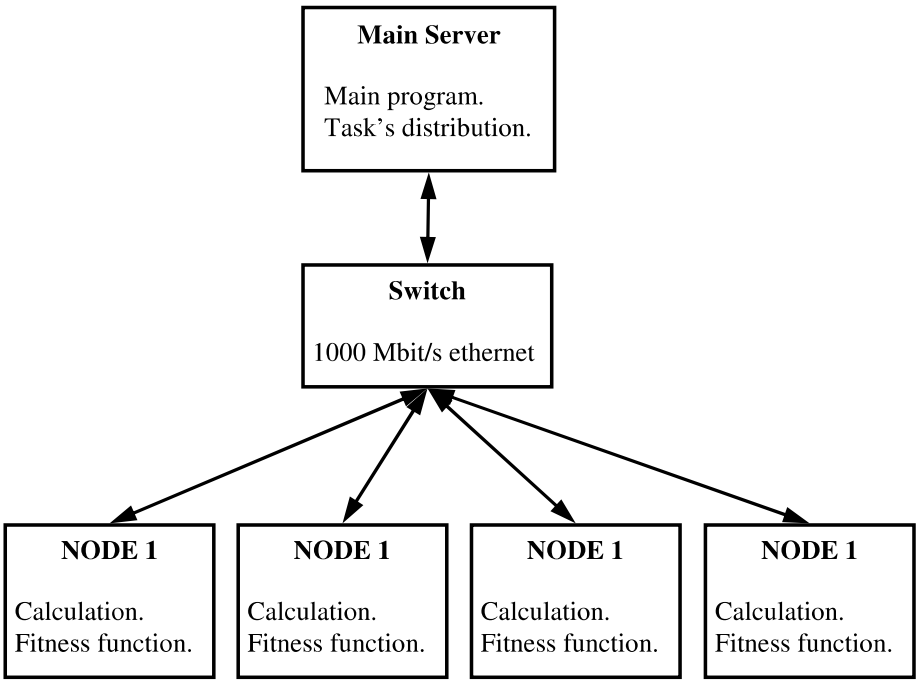

Fig. 5. The cluster's structure

The Open MPI library provides tools for communication between the nodes. It uses sockets for sends and receives data. Open MPI automatically analyses the topology of distributed system (cluster) and then chooses the best (fastest) way for send data, messages.

The open MPI model uses distribute memory, so all processes have own local memory and have not access to memory of others. They can communicates by the send/receive function (send/receive massages).

The main program (in the server) has whole generation of population. Divides one for parts and send them for each node. The nodes from 1 to P-1 receive N/P genomes, where: N – number of genome in the population and P – number of node. And node P receives the rest $N - \frac{N}{P} * (P - 1)$ of genome.

Then in the node we use Open MP API. Open MP realize shared memory paradigm. So each thread has access to global memory.

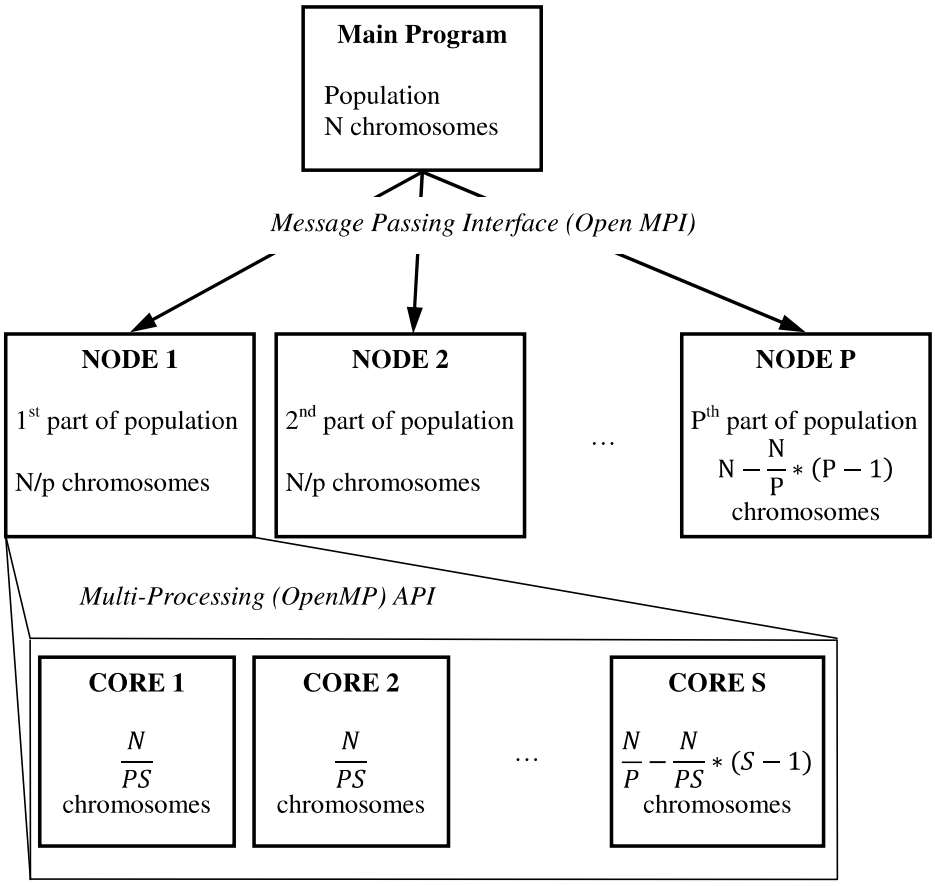

Fig. 6. Task distribution in the cluster.

MP starts one thread for each CPU (core) y divides task of node between S threads.

The threads from 1 to S-1 calculate $\frac{N}{PS}$ part and the last thread S calculate $\frac{N}{P} - \frac{N}{PS} * (S-1)$ part of task of node.

Table 1. Time of computing. Only MP API

CPUs	1	2	3	4	5	6	7	8
Seconds	94.2	66.9	57.7	53.2	50.4	48.6	47.3	46.3
CPUs	9	10	11	12	13	14	15	16
Seconds	45.5	45.0	44.4	44.0	43.7	43.4	43.1	42.9

The table 1 and figure 7 show the time of computing using only one node. In this case we use only Open MP API.

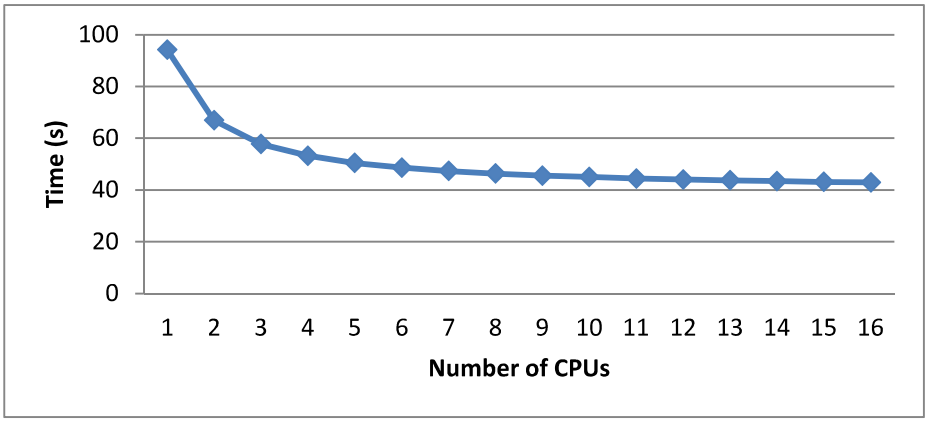

Fig. 7. Time of computing. Only MP API.

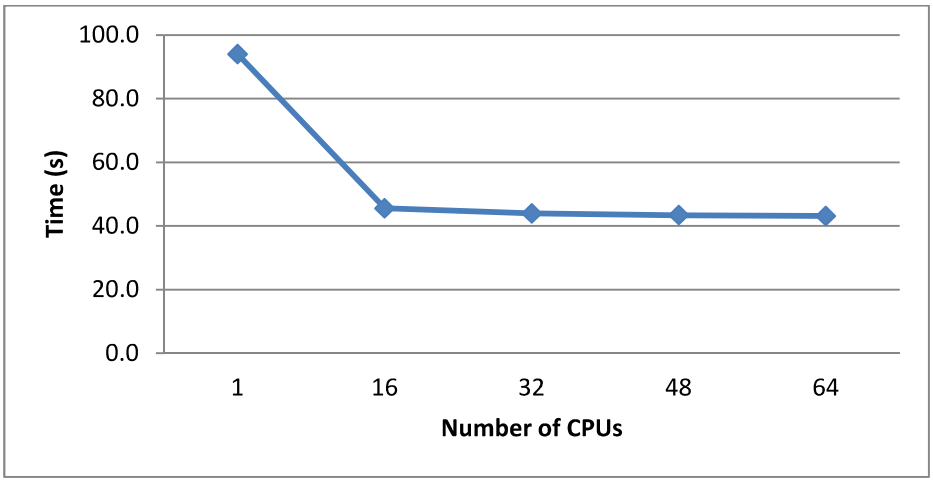

Fig. 8. Time of computing. MPI library with MP API.

Table 2. Time of computing. MPI library with MP API.

Nodes	1	1	2	3	4
CPUs	1	16	32	48	64
Seconds	95.3	45.5	43.9	43.4	43.1

The table 2 and figure 8 show the time of computing using Open MPI for distribute the tasks between the nodes, and Open MP API for calculate the part of task in parallel mode. In this case we use mixed mode Open MPI library+ Open MP API.

5 Conclusions

The methods of genetic algorithms allow finding optimal configurations of the proteins in space. That makes it possible to model proteins properties. Using parallel genetic algorithms, it is possible to analyze the whole of population in multiprocessors system. All of these methods together decrease, by several times, time of modeling.

In the experiment we used different techniques for parallelize computing. The experiment shows that than we incise number of CPU (processing units) the time of computing is decreased. But there is limit of number of processing unit in our experiment. According to Amdahl's law $\frac{1}{(1-P)+\frac{P}{S}}$ where S is a number of processing unit and P is a parallel part of the algorithm, this limit depends on the parallel part of algorithm.

The directions for future research are to change the algorithm for increasing the parallel part and to design methods and models for simulation of protein folding with parallel genetic algorithms in distributed system like clusters or grid.

References

1 Kitano, H.: Computational System Biology. Nature, Vol. 420, pp. 206-210 (2002)
2 Oliveto, P.S., Lehre P.K., Neumann, F.: Theoretical Analysis of Rank Based Mutation – Combining Exploration and Exploitation. Proceedings of the Eleventh Congress on Evolutionary Computation (2009)
3 C. Clementi: Coarse-grained models of protein folding: toy models or predictive tools?, Current Opinion in Structural Biology. 18, 10-15 (2008)
4 Cervantes, J., Stephens, C.R.: Rank Based Variation Operators for Genetic Algorithms. Proceedings of the 2008 international Genetic and Evolutionary Computation Conference (GECCO08) pp. 905-912 (2008)
5 Zambonelli, F. and A. Omicini: Challenges and research directions in agent-oriented software engineering. Autonomous Agents and Multi-Agent Systems, 9(3):253–283 (2004)
6 Parunak, V.D., R. Savit, and R. L. Riolo: Agent-based modelling vs. equation based modelling: A case study and users' guide. In J. S. Sichman, N. Gilbert, and Conte, R., editors, Multi-Agent Systems and Agent-Based Simulation, pages 10–26, Springer-Verlag (1998)
7 Omicini, A., A. Ricci, M. Viroli, C. Castelfranchi, and L. Tummolini. 2004. Coordination artifacts: Environment-based coordination for intelligent agents. In N. R. Jennings, C. Sierra, L. Sonenberg, and M. Tambe, editors, 3rd international Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004), volume 1, pages 286–293, New York, USA, 19–23 ACM (2004)

8 Cervantes, J., Sánchez, M., González, P.P.: Emerging traits in the application of an evolutionary algorithm to a scalable bioinformatics problem, Evolutionary Computation (CEC), 2010 IEEE Congress. Barcelona, pp 1–8 (2010)
9 Cox, G.A., Mortimer-Jones, T.V., Taylor, R.P. Johnston R.L.: Development and Optimization of a novel Genetic Algorithm for Studying Model Protein Folding. Theor Chem Acc (2004) 112: 163–178 DOI 10.1007/s00214-004-0601-4 (2004)
10 Open MPI v1.4.3 documentation, http://www.open-mpi.org/doc/v1.4/
11 Michael Jay Quinn: Parallel programming in C with MPI and OpenMP, McGraw-Hill Higher Education, 529 p. (2004)
12 Barbara Chapman, Gabriele Jost and Ruud van der Pas: Using OpenMP. Portable Shared Memory Parallel Programming. Massachusetts Institute of Technology, 353 p. (2008)
13 Andriy Sadovnychyy, Pedro Pablo Gonzales Pérez and Jorge Cervantes: Design parallel genetic algorithms for multi-agent systems. GESTS International Transactions on Computer Science and Engineering. Volume 64, Number 1, pp 67-72 (2011)