THE USV ANNALS OF ECONOMICS AND PUBLIC ADMINISTRATION | VOLUME 14, ISSUE 2(20), 2014

PARALLEL HYBRID METHODS USED IN OPTIMIZATION PROBLEMS SOLVING

Lecturer PhD **Ionut BALAN**
„Ştefan cel Mare" University of Suceava, Romania
Faculty of Economic Sciences and Public Administration
ionutb@seap.usv.ro

Abstract:
This paper presents different models of hybrid algorithms that can be run on parallel architectures being used in optimization problems solving. In these models we used several techniques: genetic algorithms, ant colony and tabu search. Optimization problems can achieve a high degree of complexity, which is the main reason for the necessity of using of these methods in such incursions. With their cooperation, we tried to obtain satisfactory results in much better running time than the sequential versions. These models have been run using various parallel configurations on a cluster cores, which belong to „Stefan cel Mare" University. The results obtained for these models were compared with each other and with the results obtained for models described in other personal papers. The paper highlights the advantages of the parallel hybrid cooperation in solving of complex optimization problems. This paper is structured in four chapters: Introduction, Cooperative heterogeneous model, Cooperative hybrid models and Conclusions.

Key words: parallel, optimization, hybrid, tabu search, genetic algorithm,ant colony

JEL classification: C61, C63

INTRODUCTION

The optimization theory is the mathematical study of a given problem, within we seek a maximum or minimum value for an objective-based function in a given area. This involves the process of searching a possible solution to the problem studied properties and aspects of the algorithm aimed to be solved. The importance of studying the theory of optimization has grown with time. This was due, in particular, to many areas in which it operates, including applied mathematics, computer science, engineering, economics, etc.

Among several methods used in optimization, the population-based methods provide the greatest degree of competition, thus a high number of evaluations of the objective-based function can be achieved in parallel in order to obtain desired solutions after going through a number of generations (LEWIS,2004). The emergence of these methods has been imposed by computational limitations that were impossible to solve optimization problems in the real world using conventional methods.

The population-based methods involve a large number of evaluations of the objective-based function, providing solutions in every generation. A particular set of these methods is the one related to evolutional computation, which aims to use natural processes insight in solving certain computational problems. The evolutional algorithms are divided into three major classes (LEWIS,2004): genetic algorithms, evolutional strategies and evolutional programming.

The genetic algorithms are commonly used in solving engineering problems, but not only (they can solve economic problems, etc). They can be easily parallelized, with a high probability of convergence.

Within the genetic algorithms (ZAHARIE,2004), population is represented by states from the problem space, states which representing potential solutions of the problem. Normally, population elements are binary coded. The main operator is the crossover operator, the mutation having a secondary role. Genetic algorithms were introduced by Holland (1960), initially as models for evolution of natural systems, or their adaptation to the environment. It was later proved to be effective models of computation, especially in solving combinatorial optimization problems.

With regard to the parallel implementations of these methods, there has been a lot of research in the field, in order to obtain satisfactory results. However, we cannot consider this area a closed chapter.

Using evolutionary algorithms for solving optimization problems derives from using of population on these methods, concept useful in obtaining sets of solutions to the problem to be optimized.

We can obtain a hybrid algorithm by combining of the deterministic techniques, heuristics, or a these two techniques between them. In this way we can solve optimization problems achieving superior results compared with other techniques. Usability of such algorithms has increased in recent years, primarily, because it has proven to be an accessible technique that gave superior results for a wide range of optimization problems. (TANTAR, 2007 and LIEFOOGHE, 2010)

This paper is divided into the following parts: Introduction, Cooperative heterogeneous model, Cooperative hybrid model, Conclusions and References. In this paper there are treated three techniques very popular in recent years: genetic algorithms, ant colony and tabu search. After Introduction, where we present the necessity of this methods usage for optimization problems, in next chapters, there are taken results for this methods, runed in different configurations, on different architectures. Results obtained had the same input data (DD_SDST50_ta058) (SYSTEMAS, 2011). The input data are related to flow-shop scheduling problems, where are made the scheduling of 50 processes on 20 resources. All the results were obtained by running the algorithms in the configurations described, on nodes being part of the IBM RoadRunner architecture from „Stefan cel Mare" University. This architecture use IBM PowerXCell 8i procesors having a real computing power of 6.45 TFlops in double precision and a total storage capacity of 6936 TB

ALGORITHM PARALLELIZATION

The model used for the parallelization of the evolutionary algorithm is the island model (LIEFOOGHE, 2010), where the population is divided into several subpopulations which are distributed to several processors. Each processor is responsible for the evolution of each subpopulation, performing all the steps of the metaheuristic. After a predetermined number of steps (in a synchronous communication), or when it is reached a certain threshold (asynchronous communication) it is activated the migration process. As a result of this process, the received individuals are integrated into local populations, helping the populations from the next generations to improve the solution (BALAN, 2012).

The methods described in this paper were run, several times, respecting multiple configuration. Thus, there were run, several times, sequentially and respecting the island model for two and four nodes. Number of runs for each studied variant was 5, among the results that have some degree of significance were the maximum and minimum values obtained, the mean and the median. Each of them, taken separately, have a degree of importance for the studied problem. The number of runs higher than 1 is recommended due to the random behaviour met in those two methods, which lead to different results, obtained after each new run, so the mean and/or the median of the obtained results can be considered references in the caracterization of the solutions. Being solved the same problem, but using different techniques (genetic algorithms and ant colony), to establish certain parameters, we took into acount the ability of these methods to work with populations, that are actually points in the search space, populations that can evolve over a number of generations. In this case, the number of generations for each trial was predetermined, being equal to 200. For those two methods, which are working with populations, the number of individuals involved in evolution from each population is 25, this thing, besides the one related to the number of generations, making it possible to explore spaces of solutions identical in size. This was set to start from a common point for the methods that solve the considered problem (BALAN, 2012).

The number of mutations and cross-overs for the genetic algorithm depends on the number of individuals from the population. In all variants studied these two numbers are identical, for 25 individuals we have three cross-over (number calculated as integer value of (1+nrindividuals/10))

and the number of mutations will be 7 (integer value for (1+nrindividuals/4)). Regarding to the individuals selection method (individuals on which genetic operators will be apllied), for sequential and island variants on two and four nodes, are used the best 50% of the chromosomes, depending by the fitness function. In the integration of the new chromosomes in the population, for the cross-over operator, if the resulted chromosome is superior to the weakest solution from the population, the second one will be replaced, if not, nothing will happen. Similar behaviour occurs for the mutation operator. For the parallel versions we are dealing with specific issues, so to enable the information exchange, we considered that it is preferable the migration of only one chromosome, represented by the best solution from the population in terms of objective value, which is currently transferred by each node at defined intervals, for five times during the application execution. Weakest individuals from the current population are replaced by these new arrivals.

As stated in (BALAN, 2012), for the ant colony, certain characteristics must be specified. Thus, an important issue is to select the routes for this type of algorithm. The first location of each route of ant colony is generated randomly. For the next points of the route, from the pheromone matrix are selected top five values and their corresponding points. The next step is the calculation of the pheromone intensity from each of these five points to the points that are already in the route. Finally, it is selected the point which has the maximum level of intesity. They also use calculations to find the point which should be in the route, before the existing points (LI, 2011). Regarding the update process for the pheromone matrix, any of those generated individuals leads to a increasing of the intensity for that route, while the influence of the elite is higher (a five times multiplication for the route). Also, at every generation there is a pheromone evaporation (in our case it is halved). With regard to the communications between nodes, for ant colony, the elite migrate, the number of tranfers being identical to that found in the genetic algorithm described above.

Using this techniques, and a deterministic method, in the next chapters I present five models, with some differences between them and reasons for their results.

COOPERATIVE HETEROGENEOUS MODEL

A first suggested model is presented in figure no. 1. In this model we can see that the second algorithm (GA), being faster, has lower runtime than the first algorithm (ANT). This is due to the number of iterations equal met in the case of both algorithms.

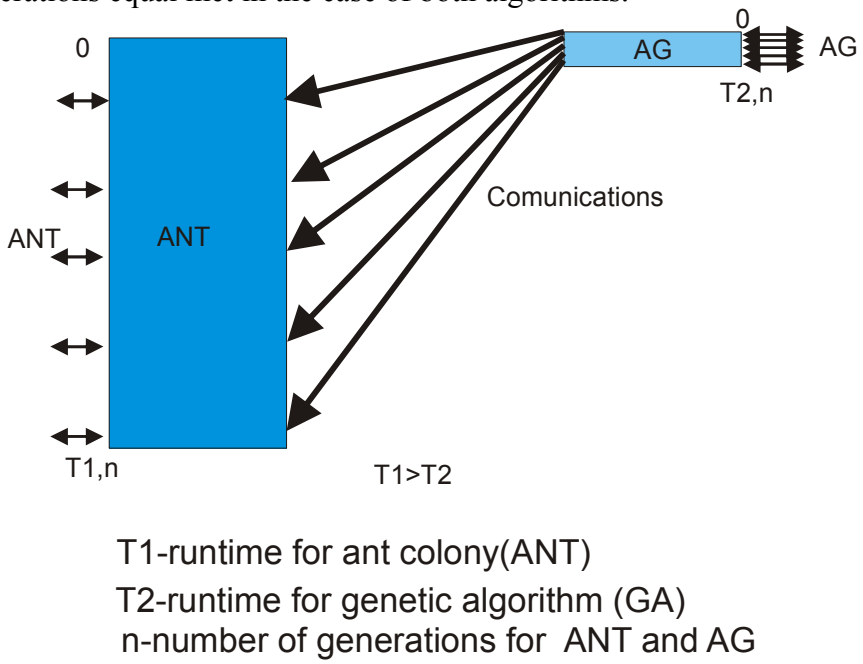


Figure 1. Heterogeneous parallel model with a fixed number of iterations without Idle state on nodes

Returning to communications, we can see some differences in comparison with the models described in (BALAN,2012) and (BALAN,2013):

- The processes corresponding to slow algorithm communicate with each other at predetermined intervals (in this case after a number of iterations);
- The processes corresponding to fast algorithm communicates as well at predetermined intervals. The number of iterations is taken as communication reference, and this algorithm is faster than the previous one, we can conclude that the timing of communication in this class of algorithms do not coincide with moments of communication found in the first case;
- The communication between nodes running different algorithms is one-way communication, the information is transmitted only in one direction: from fast algorithm to slow algorithm. Say that in this case, even if the information is sent by the fast process at a time that would coincide with the initialization of the slow process due to the use of non-blocking messages, they can reach their destination at certain time intervals, without damaging undesirably the running of a participant process to obtain the final results.

Basically, in the above mentioned model the genetic algorithms cooperate between them, without being influenced by the ant colony. However, the ant colony even if it has a cooperative connection with some other processes that solve the same algorithm, it can receive influences from genetic algorithms, within well-established time intervals. In contrast to the above quoted models, in this case the total flow of information transfer is low and can be calculated with the expression no.1.

$$N_{com} = size_{ANT} \times (size_{ANT} - 1) + size_{AG} \times (size_{AG} - 1) + size_{ANT} \times size_{AG} \quad [1]$$

The entire time (cost) of data processing on all nodes will be:

$$C_{Total} = size_{ANT} \times T_1 + size_{AG} \times T_2 \quad [2]$$

The results obtained using this configuration are lower than the models presented in (BALAN,2012) and (BALAN,2013). The causes for these results are:

- In the model from figure no. 1. we only have communication from AG to ANT, and these items passed from AG to ANT are obtained only from a strong collaboration between AG's, unlike the cases found in the quoted sources, where we have ANT influences;
- In the figure no. 1. we may note that the number of generations is sharply lower than the asyncron model of (BALAN,2013), and thus resulting a much weaker trend.

Another suggested model is presented in figure no. 2. Between this model and the one presented above there are some similarities, among which we might include the running time for each algorithm on the corresponding node and the direction of communication. The difference between the two models is given by moments of the data sending from the AG to the ANT. These take place without delay, so that the slow algorithm will have influences from AG at the beginning of the evolution. This behavior will lead to achieving poorer results than previous cases presented.

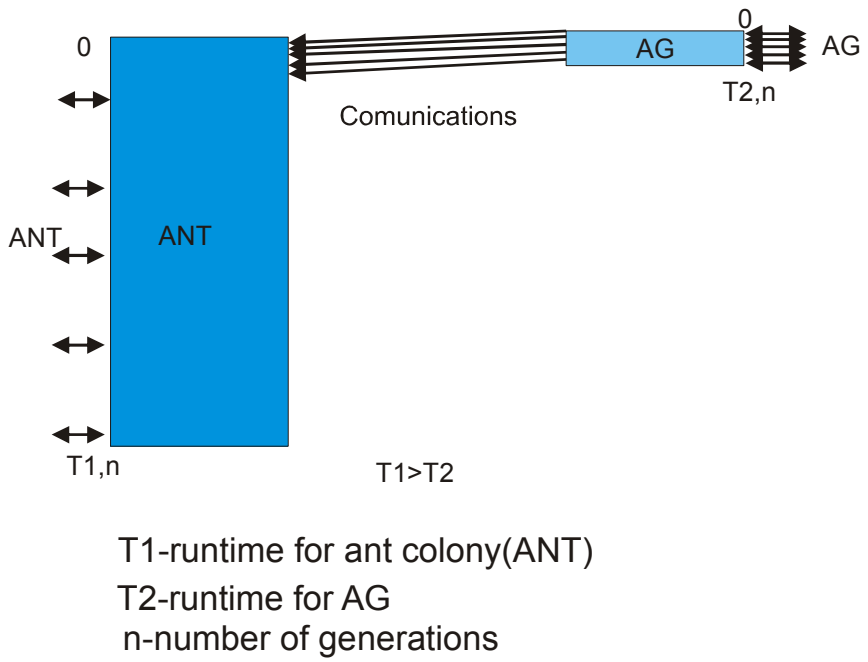


Figure 2. Heterogeneous model with blocking communication from fast algorithm

COOPERATIVE HYBRID MODEL

One of the most common features in the above described models is the interruption of the fast process while other processes run all kind of operations. This could be considered a problem in some cases. Using models like those from figures 3. and 4. I'm trying to remove this possible problem.

In the case of the hybrid model based on cooperation and diffusion, with a non-blocking communication from the AG, we are dealing with the same type of communication as that encountered in the model from figure no. 1. This distinction is given by the process that dealt with solving the algorithm with less effort. Thus, this process will inform the slow process that it is approaching its last calculations; therefore the slow process will try dividing its task into two main activities. Due to its random nature it is impossible to share the task of a process between two processes. In this case, after running the algorithm AG, the ANT process shall send the pheromone matrix in each generation towards the uneven node. It is observed that data diffusion takes place between nodes, two by two, so we recommend using an even number of processes. After data transfer, both the master and the slave process will try to work with an equal number of items. Both processes will follow the specific steps of the algorithm, so that, towards the end of the generation, to make possible the transfer of pheromone matrix in reverse, from the AG to the ANT (from slave to master). All these operations will be performed until the exhaustion of the generations.

During its evolution, even if the fast process finished long before the transfer to the slow operations, the data is set so as to arrive at certain pre-set generation. The results in this case are comparable to those of the model from Figure no. 1.

Regarding the entire number of streams of transferred data, this model is superior to all other models, because in this case, in addition to transfers representing the best item of a certain generation, obtained by using an algorithm; it has also transfers of pheromone matrix. The number of transfers of the matrix in a single direction may be even equal to the number of generations within the slow algorithm.

The total cost of data processing on all nodes can be expressed by expression 4. (Ideally, the slow process tasks are divided equally between the master and slave process):

$$C_{Total} = size_{ANT} \times (T_1 - T_2)/2 + size_{ANT} \times T_2 + size_{AG} \times T_2 + size_{AG} \times (T_1 - T_2)/2 \quad [3]$$

$$C_{Total} = (T_1 / 2 + T_2 / 2) \times (size_{ANT} + size_{AG}) \quad [4]$$

In the case of architecture with only two processes we shall have:

$$C_{Total} = (T_1 + T_2) \quad [5]$$

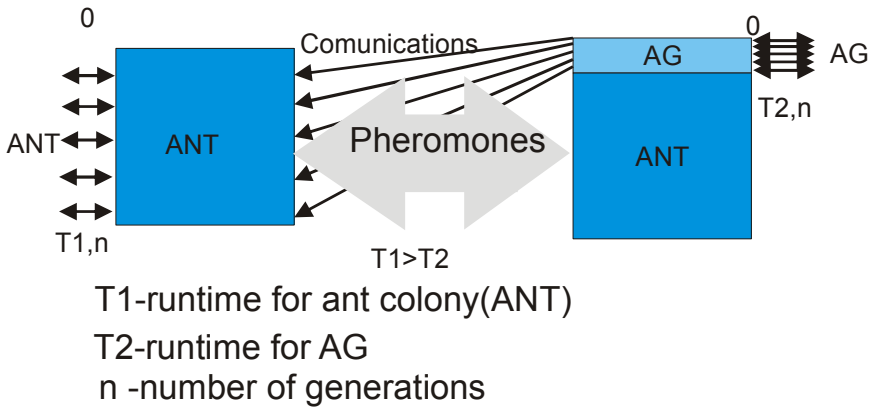


Figure 3. Cooperative diffuse hybrid model, with non-blocking communication from AG

A similar model to that of figure no. 3. is presented in figure no. 4. The starting point for this model was the heterogeneous model with blocking communication from fast algorithm shown in figure no. 2. In this case the process is similar to the behavior of the hybrid model based on cooperative and diffusion with non-blocking communication from fast algorithm. In comparison to this, we have blocking communications, which are made at the beginning of the slow process, trying to avoid blocking the standby state.
The results in this case are comparable to those of the model from figure no. 2, and in terms of the total number of streams of data transfer, it is equal to that of the hybrid model shown in figure no. 3.

The entire cost of data processing on all nodes can be expressed, in this case, in the expression no. 4.

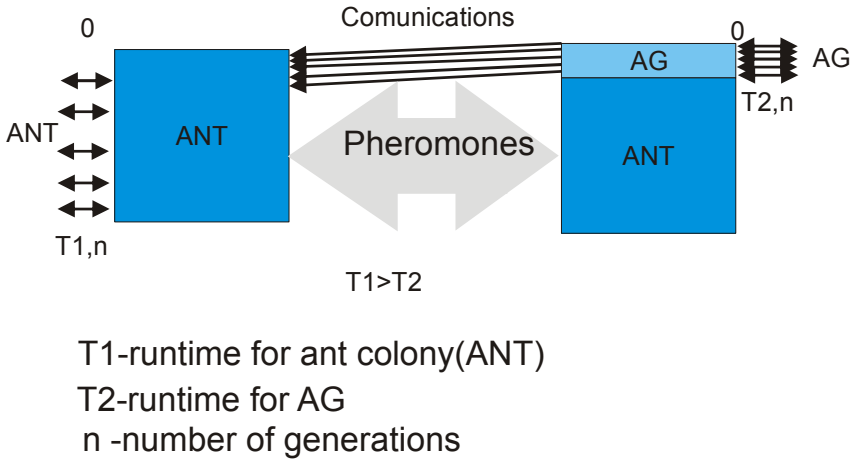


Figure 4. Cooperative diffuse hybrid model, with blocking communication from AG

Figure no. 5. illustrates the total costs for the last four models discussed. Gain is observed for the hybrid method based on cooperation and diffusion. And in terms of execution time we can observe that the last two methods are executed in a short time.

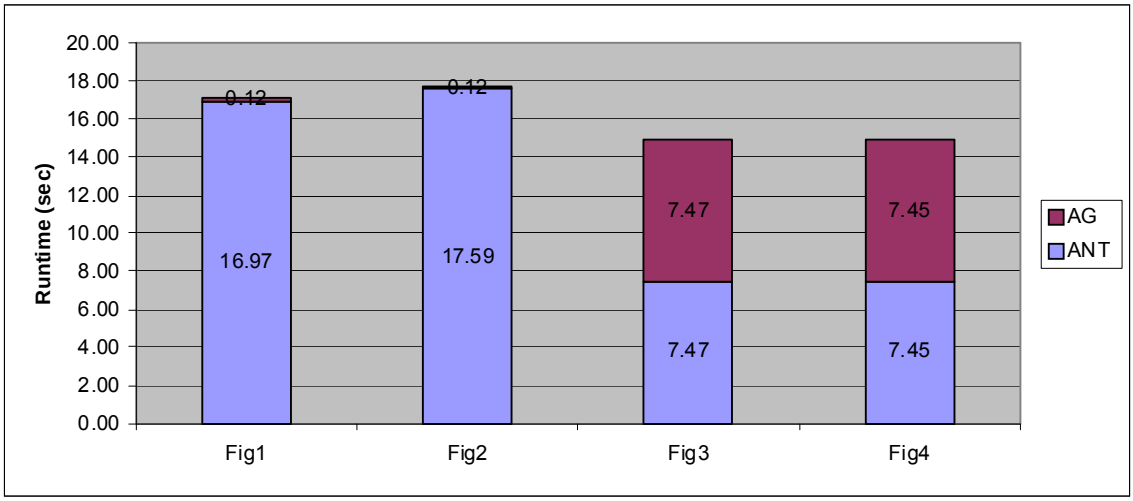


Figure 5. Costs of execution of different cooperative models

The heterogeneous model with blocking communications from fast algorithm has the same cost and the same number of streams transmitted as in the case of the model from figure no. 1. The only gains made by these two models are included in Runtime, and this can be a starting step in approaching other models.

Another method that leads to satisfactory results, even quite good results in reducing the standby time corresponding state on the node running the fast algorithm, would be the hybridization of this algorithm by introducing local search techniques. Such type of a technique is represented by Tabu Search (TS) (PARARACH,2011). The model obtained in this way has been illustrated in Figure no. 6.

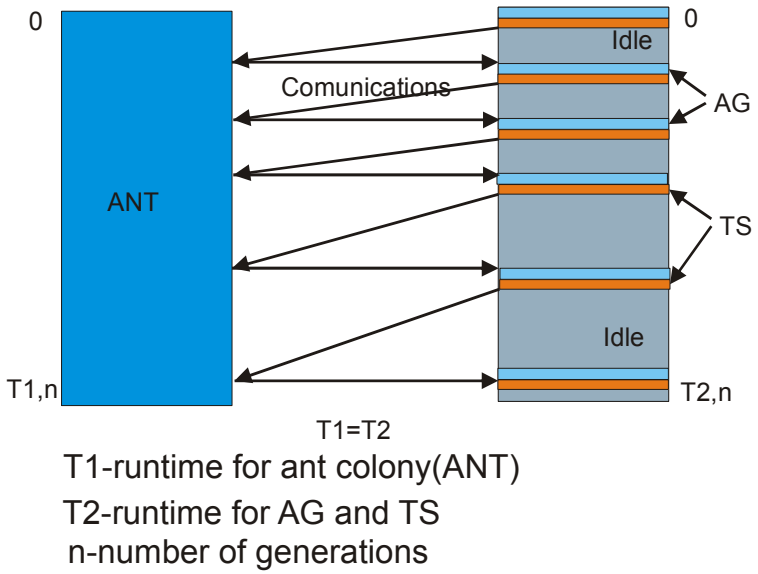
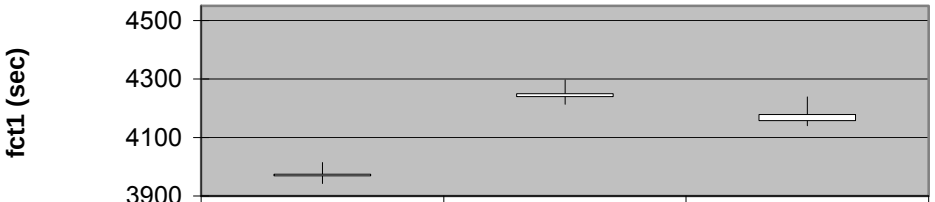


Figure 6. Hybrid cooperative model ANT-AG-TS for flow-shop scheduling problem

The model presented here is characterized by the presence of blocking transmissions of data between processes, which determines a reduction of time wasted by the processes in standby, especially because the newly introduced technique does not cover the whole time.

The introduction of local search techniques can improve results for a specific problem. In our case, this improvement took place, and in figure no. 7., I present these results, making a comparison with those obtained in synchronous and asynchronous version of (BALAN,2013). These comparisons were made for the following reasons:

- in the case of the synchronous variant - we are dealing with similar architectural model and the distinction constitutes the presence of TS before transfers of data;
- in the case of the asynchronous variant - fast processes are dealing with many more operations than synchronous.



	ANT-AG-TS	ANT-AG synchronous	ANT-AG asynchronous
Median	3968	4239	4157
Maximum	4015	4297	4239
Minimum	3943	4214	4140
Average	3974	4250	4179

Figure 7. Results obtained for flow-shop scheduling problem in hybrid version ANT-AG-Tabu Search

CONCLUSIONS

Two different techniques can cooperate with each other in order to determine an optimal solution within a given problem. To strengthen this claim we tried to use in this paper two different algorithms (ANT and AG). In addition, a simple method of cooperation could be obtained by using an island model, a model that enables collaboration between all processes that are executed. A first problem that may be encountered in such cooperation process is the standby times, specific to fast execution processes. This leads to a cost (total time) big enough for all the processes that contribute to the final result. A solution that leads to a significant reduction of standby times consists of a continuous execution of the fast process, stopping during data transfers and slow process interruption. In this case it comes to delivering better results superior to simple island models, to the entire exclusion of the standby times specific to fast processes, but instead, the cost of processing remains quite high. In order to reduce costs, there were suggested two parallel models to communications only from fast algorithm to slow algorithm. In these cases, because the faster algorithm should not expect information from slow algorithm, the algorithm can terminate the corresponding first predetermined time (a fixed number of generations). Thus, apart from reducing costs it can be excluded the standby time, but the results obtained are not as good as in the previous cases.

In order to reduce the total cost related to a problem by keeping the size of the search space specific to each algorithm, it may direct the tasks associated to slow algorithm towards a much faster algorithm. In this way they will get better results in a short time and the costs can be considered satisfactory.

Similarly, as in the case of algorithms using message cooperation blockers, it may use the techniques for reducing standby times specific to fast processes. The suggested solution in this case includes a local search technique (Tabu Search), the results are superior to other methods, but the final cost is higher than those of other models.

Thus, by using heterogeneous cooperative parallel models, it can be achieved satisfactory results in a fairly small period of time, for any problem whose solution requires large computing power.

REFERENCES

[1] Balan, I., – *A Parallel Hybrid Cooperative Model for Optimization Problems Solving*, International Journal of Academic Research, Baku, Azerbaijan, iulie 2012, ISSN 2075-4124, p. 57-62
[2] Balan, I., ş.a., - *Parallel cooperative models for optimization problems*, Journal of Applied Computer Science & Mathematics, nr.14, 2013
[3] Lewis, A., - *Parallel Optimisation Algorithms for Continuous Non-Linear Numerical Simulations*, School of Computing and Information Technology, Grifftih University, Nathan Campus, Brisbane, Australia, thesis, 2004
[4] Li, B., ş.a., - *Communication latency tolerant parallel algorithm for particle swarm optimization*, Parallel Computing, Vol. 37, nr.1, p.1-10, 2011
[5] Liefooghe, A., ş.a., - *A software framework based on a conceptual unified model for evolutionary multiobjective optimization: ParadisEO-MOEO,* European Journal of Operational Research, Elsevier, 2010
[6] Pararach, S., - *A tabu search approach for makespan minimization in a permutation flow shop sheduling problems*, Proceedings of the 2011 International Conference on Industrial Engineering and Operations Management, Kuala Lumpur, Malaezia, p. 300-305, 2011
[7] Sistemas de Optimizacion Aplicada, *http://soa.iti.es/instances-results-and-other-materials-for-multi-objective-pfsp-with-sdst*, September 2011
[8] Tantar, A.A., ş.a., - *A parallel hybrid genetic algorithm for protein structure prediction on the computational grid,* Future Generation Computers Systems, vol.23, nr.3, p. 398-409, 2007
[9] Zaharie, D., - *Calcul neural şi calcul evolutiv*, Universitatea de Vest Timişoara, 2004