Proceedings of the 2014 Federated Conference on
Computer Science and Information Systems pp. 413–420

DOI: 10.15439/2014F373
ACSIS, Vol. 2

Hybrid GA-ACO Algorithm for a Model Parameters Identification Problem

Stefka Fidanova
Institute of Information and
Communication Technology
Bulgarian Academy of Science,
Acad. G. Bonchev Str., bl. 25A,
1113 Sofia, Bulgaria
E-mail: stefka@parallel.bas.bg

Marcin Paprzycki
System Research Institute
Polish Academy of Sciences
Warsaw and Warsaw Management Academy
Warsaw, Poland
E-mail: marcin.paprzycki@ibspan.waw.pl

Olympia Roeva
Institute of Biophysics and
Biomedical Engineering
Bulgarian Academy of Science,
Acad. G. Bonchev Str., bl. 105,
1113 Sofia, Bulgaria
E-mail: olympia@biomed.bas.bg

Abstract—**In this paper, a hybrid scheme, to solve optimization problems, using a Genetic Algorithm (GA) and an Ant Colony Optimization (ACO) is introduced. In the hybrid GA-ACO approach, the GA is used to find a feasible solutions to the considered optimization problem. Next, the ACO exploits the information gathered by the GA. This process obtains a solution, which is at least as good as—but usually better than—the best solution devised by the GA. To demonstrate the usefulness of the presented approach, the hybrid scheme is applied to the parameter identification problem in the *E. coli* MC4110 fed-batch fermentation process model. Moreover, a comparison with both the conventional GA and the stand-alone ACO is presented. The results show that the hybrid GA-ACO takes the advantages of both the GA and the ACO, thus enhancing the overall search ability and computational efficiency of the solution method.**

Index Terms—**Genetic Algorithm; Genetic algorithms; Ant Colony Optimization; hybrid; model parameter identification; *E. coli*; fed-batch fermentation process**

I. Introduction

TO SOLVE different optimization problems we can apply various techniques and approaches, namely exact algorithms (Branch-and-Bound, Dynamic Programming, local search techniques) [1], [2], [3], heuristics [5], [6], and meta-heuristics (Genetic Algorithms, Ant Colony Optimization, Particle Swarm Optimization, Simulated Annealin, Tabu Search, etc.) [4], [7], [8]. Today, the use of meta-heuristics has received more and more attention. These methods offer good solutions, even global optima, within reasonable computing time [9]. An even more efficient behavior, and a higher flexibility when dealing with real-world and large-scale problems, can be achieved through a combination of a meta-heuristic with other optimization techniques, the so-called hybrid metaheuristic [7], [13], [12], [14], [20], [21], [19].

The main goal of all hybrid algorithms is to exploit the advantages of different optimization strategies, while avoiding their disadvantages. Choosing an adequate combination of metaheuristic techniques one can achieve a better algorithm performance in solving hard optimization problems. Developing such effective hybrid algorithm requires expertise from different areas of optimization. There are many hybridization techniques that have shown to be successful for different applications [10], [11].

In this paper, we investigate a hybrid metaheuristic method that combines the Genetic Algorithms (GA) and the Ant Colony Optimization (ACO), named GA-ACO. There already exist some applications of the ACO-GA hybrid for several optimization problems. In [15], [16] a hybrid metaheuristic ACO-GA, for the problem of sports competition scheduling is presented. In the proposed algorithm first, the GA generates activity lists, thus providing the initial population for the ACO. Next, the ACO is executed. In the next step, the GA, based on the crossover and mutation operations, generates a new population. Authors of [17] presented a hybrid algorithm in which the ACO and the GA search alternately and cooperatively in the solution space. Test examples show that the hybrid algorithm can be more efficient and robust than the traditional population based heuristic methods. In [18], the problem of medical data classification is discussed. Authors propose a hybrid GA-ACO and show the usefulness of the proposed approach on a number of benchmark real-world medical datasets. For solving NP-hard combinatorial optimization problems, in [22], a novel hybrid algorithm combining the search capabilities of the ACO and the GA is introduced. As a result a faster and better search algorithm capabilities is achieved.

Provoked by the promising results obtained from the use of hybrid GA-ACO algorithms, we propose a hybrid algorithm, i.e. collaborative combination of the GA and the ACO methods for the model parameters optimization of the *E. coli* fermentation process. The effectiveness of the GA and the ACO have already been demonstrated for model parameter optimization considering fed-batch fermentation processes (see, [24]). Moreover, parameter identification of cellular dynamics models has especially become a research field of great interest. Robust and efficient methods for parameter identification are thus of key importance.

The paper is organized as follows. The problem formulation is given in Section 2. The proposed hybrid GA-ACO technique is described in Section 3. The numerical results and a discussion are presented in Section 4. Conclusion remarks are done in Section 5.

978-83-60810-58-3/$25.00 © 2014, IEEE

II. Problem Formulation

A. E. coli Fed-batch Fermentation Model

The mathematical model of the fed-batch fermentation process of the *E. coli* is presented by the following non-linear differential equation system [28]:

$$\frac{dX}{dt} = \mu X - \frac{F_{in}}{V} X \qquad (1)$$

$$\frac{dS}{dt} = -q_S X + \frac{F_{in}}{V}(S_{in} - S) \qquad (2)$$

$$\frac{dV}{dt} = F_{in} \qquad (3)$$

where

$$\mu = \mu_{max} \frac{S}{k_S + S} \qquad (4)$$

$$q_S = \frac{1}{Y_{S/X}} \mu \qquad (5)$$

- X is the biomass concentration, [g/l];
- S is the substrate concentration, [g/l];
- F_{in} is the feeding rate, [l/h];
- V is the bioreactor volume, [l];
- S_{in} is the substrate concentration in the feeding solution, [g/l];
- μ and q_S are the specific rate functions, [1/h];
- μ_{max} is the maximum value of the specific growth rate, [1/h];
- k_S is the saturation constant, [g/l];
- $Y_{S/X}$ is the yield coefficient, [-].

For the model parameters identification, experimental data of an *E. coli* MC4110 fed-batch fermentation process can be used. The experiments providing the real-world data were performed in the Institute of Technical Chemistry, University of Hannover, Germany. The detailed description of the fermentation condition and experimental data can be found in [23], [27].

The fed-batch process starts at time t = 6.68 h, after batch phase. The initial liquid volume is 1350 ml. Before inoculation a glucose concentration of 2.5 g/l was established in the medium. Glucose concentration, in the feeding solution is 100 g/l. The temperature was controlled at 35 °C, the pH at 6.9. The stirrer speed was initially set to 900 rpm and later was increased to 1800 rpm, so that the dissolved oxygen concentration was never below 30%. The aeration rate was kept at 275 l/h and the carbon dioxide was measured in the exhaust gas. The process was stopped at time t = 11.54 h.

The bioreactor, as well as the FIA measurement system is shown in Figure 1. The feed rate profile and the dynamics of the measured substrate concentration are presented, respectively in Fgure 2 and Figure 3.

For the considered non-linear mathematical model of the *E. coli* fed-batch fermentation process (Eq. (1) - Eq. (5)) the parameters that should be identified are:

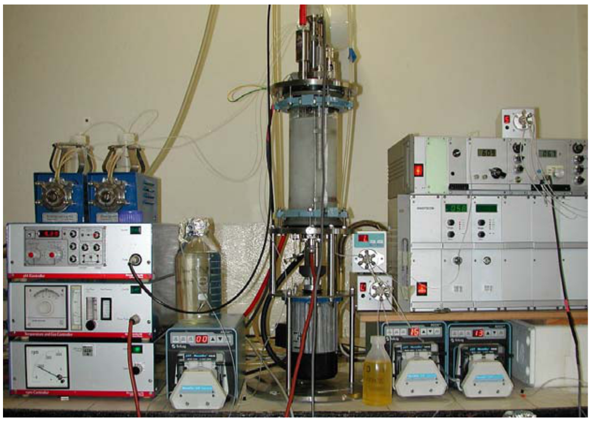

Fig. 1. *E. coli* MC4110 fed-batch fermentation process: bioreactor and FIA measurement system

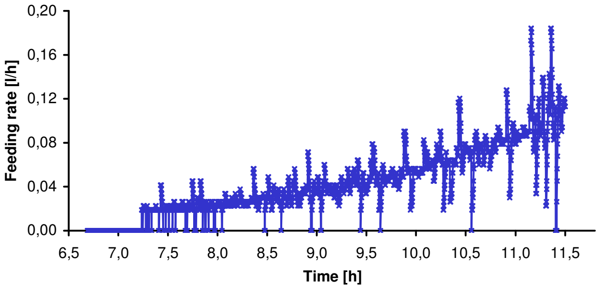

Fig. 2. *E. coli* MC4110 fed-batch fermentation process: feed rate profile

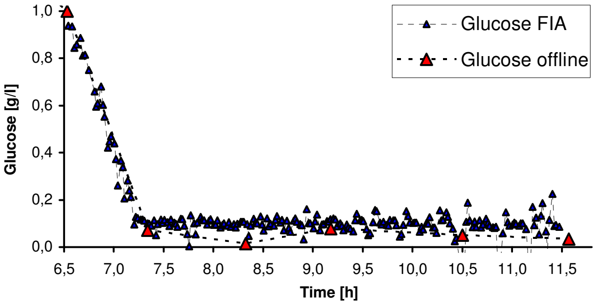

Fig. 3. *E. coli* MC4110 fed-batch fermentation process: measured substrate concentration

- maximum specific growth rate (μ_{max}),
- saturation constant (k_S),
- yield coefficient ($Y_{S/X}$).

The following upper and lower bounds of the model parameters are considered [24]:

$$0 < \mu_{max} < 0.7,$$

$$0 < k_S < 1,$$

$$0 < 1/Y_{S/X} < 30.$$

In the model identification procedures, measurements of main process variables (biomass and glucose concentration) are used. For the on-line glucose determination the FIA system has been employed. For the biomass, an off-line analysis was performed [27].

B. Optimization Criterion

The objective of the optimization consists of adjusting the parameters (μ_{max}, k_S and $Y_{S/X}$) of the non-linear mathematical model function (Eq. (1) - Eq. (5)) to best fit a data set. The objective function is presented as a minimization of a distance measure J between the experimental and the model predicted values of the main state variables (biomass X and substrate S):

$$\begin{aligned} J = \sum_{i=1}^{m} \left(X_{\exp}(i) - X_{\text{ mod }}(i)\right)^2 + \\ + \sum_{i=1}^{m} \left(S_{\exp}(i) - S_{\text{ mod }}(i)\right)^2 \to \min \end{aligned} \tag{6}$$

where m is the number of experimental data; $X_{\exp}$ and $S_{\exp}$ is the known experimental data for biomass and substrate; $X_{\text{ mod}}$ and $S_{\text{ mod}}$ are the model predictions for the biomass and the substrate with a given set of parameters (μ_{max}, k_S and $Y_{S/X}$).

III. Methodology

A. Genetic Algorithm

Genetic Algorithm is a metaheuristic technique based on an analogy with the genetic structure and behaviour of chromosomes within a population of individuals using the following foundations [33]:

- chromosomes in a population compete for resources and mates;
- those chromosomes most successful in each "competition" will produce more off-spring than those chromosomes that perform poorly;
- genes from "good" chromosomes propagate throughout the population so that the two good parents will sometimes produce offspring that are better than either parent;
- thus, each successive generation will become more suited to their environment (will move closer to an optimal solution).

The structure of the GA, shown by the pseudocode, is presented in Figure 4).

```
**Genetic Algorithm**
$i = 0$
Initial population $Pop(0)$
Evaluate $Pop(0)$
**while** (not done) **do** (test for termination criterion)
    $i = i + 1$
    Select $Pop(i)$ from $Pop(i-1)$
    Recombine $Pop(i)$
    Mutate $Pop(i)$
    Evaluate $Pop(i)$
**end while**
Final solution
```

Fig. 4. Pseudocode for GA

The GA, mainly operates on binary strings and using a recombination operator with mutation. It is based on a population of chromosomes, $Pop(t) = x_1^t, ..., x_n^t$ for generation t. Each chromosome introduces a potential solution to the problem and is implemented as some data structure S. Each solution is evaluated according its "fitness." Fitness of a chromosome is assigned proportionally to the value of the objective function of the chromosomes. Then, a new population (generation $t+1$) is formed by selecting better chromosomes (the selection step).

A roulette wheel, developed by Holland [30] is the most often used selection method. The probability, P_i, for each chromosome to be selected is defined by:

$$P[\text{Individual } i \text{ is chosen}] = \frac{F_i}{\sum\limits_{j=1}^{PopSize} F_j}, \tag{7}$$

where F_i equals the fitness of the chromosome i and $PopSize$ is the population size.

Selected members of the new population have been subjected to transformations by means of "genetic" operators to form a new solution. There are unary transformations m_i (mutation type), which create new chromosomes by a small change in a single chromosome ($m_i : S \to S$), and higher order transformations c_j (crossover type), which create new chromosomes by combining parts from several chromosomes ($c_j : S \times \ldots \times S \to S$). The combined effect of selection, crossover and mutation gives so-called reproductive scheme growth equation (the schema theorem) [29]:

$$\begin{gathered} \xi\left(S, t+1\right) \geq \\ \xi\left(S, t\right) \cdot eval\left(S, t\right) / \bar{F}\left(t\right) \left[1 - p_c \cdot \frac{\delta\left(S\right)}{m-1} - o\left(S\right) \cdot p_m\right]. \end{gathered}$$

A good schemata receives an exponentially increasing number of reproductive trials in successive generations.

B. Ant Colony Optimization

The ACO is a stochastic optimization method that mimics the social behavior of real ants colonies, which try to find shortest rout to feeding sources and back. Real ants lay down quantities of pheromone (chemical substance) marking the path that they follow. An isolated ant moves essentially at random but an ant encountering a previously laid pheromone will detect it and decide to follow it with high probability and reinforce it with a further quantity of pheromone. The repetition of the above mechanism represents the auto-catalytic behavior of a real ant colony, where the more ants follow a given trail, the more attractive that trail becomes. Hence, the overall idea of the optimization approach comes from observing such behavior, in which ants are collectively able to find the shortest path to the food.

The ACO is implemented by instantiating a team of software agents, which simulate the ants behavior, walking around the graph representing the problem to solve. The requirements of the ACO algorithm are as follows [25], [26]:

- The problem needs to be represented appropriately, to allow the ants to incrementally update the solutions through the use of a probabilistic transition rules, based on the amount of pheromone on the trail and other problem specific knowledge.
- Existence of a problem-dependent heuristic function that measures the quality of components that can be added to the current partial solution.
- Explication of a set of rules for pheromone updates, which specify how to modify the pheromone value in specific situations.
- A probabilistic transition rule, based on the value of the heuristic function and the pheromone value, that is used to iteratively construct a solution needs to be provided.

The structure of the ACO algorithm, represented as a pseudocode, is depicted in Figure 5. The transition probability $p_{i,j}$, to choose the node j, when the current node is i, is based on the heuristic information $\eta_{i,j}$ and the pheromone trail level $\tau_{i,j}$ of the move, where $i, j = 1, \ldots, n$.

$$p_{i,j} = \frac{\tau_{i,j}^{a} \eta_{i,j}^{b}}{\sum\limits_{k \in \mathit{Unused}} \tau_{i,k}^{a} \eta_{i,k}^{b}}, \tag{8}$$

where $Unused$ is the set of unused nodes of the graph.

The higher the value of the pheromone and the heuristic information, the more profitable it is to select this move and to continue the search. In the beginning, the initial pheromone level (across the graph) is set to a small positive constant value τ_0; later, the ants update this value after completing the solution construction stage. Different ACO algorithms adopt different criteria to update the pheromone level.

The pheromone trail update rule is given by:

$$\tau_{i,j} \leftarrow \rho \tau_{i,j} + \Delta \tau_{i,j}, \tag{9}$$

where ρ models pheromone evaporation (a process that takes place in the nature) and $\Delta \tau_{i,j}$ is a new added pheromone, which is proportional to the quality of the solution. Thus better solutions will receive more pheromone than others and will be more desirable in a next iteration.

```
Ant Colony Optimization
Initialize number of ants;
Initialize the ACO parameters;
while not end-condition do
    for $k = 0$ to number of ants
        ant $k$ choses start node;
        while solution is not constructed do
            ant $k$ selects higher probability node;
        end while
    end for
    Update-pheromone-trails;
end while
```

Fig. 5. Pseudocode for ACO

IV. Hybrid GA-ACO Algorithm

We propose to combine two metaheuristics, namely the GA [29], [30] and the ACO [31]. The GA is a population-based method where initial population is randomly generated. Thus the randomly generated initial solutions are further genetically evaluated. As seen above, the ACO algorithm is a population-based as well. The difference, as compared with the GA, is that the ACO does not need initial population. The ACO is a constructive method, in which the ants look for good solutions guided by the parameter called the pheromone. At the beginning the initial pheromone is the same for the all arcs of the graph representing the problem. After every iteration, the pheromone levels are updated (in all arcs; in arcs traveled by the ant the pheromone level is increasing, while in abandoned arcs it it evaporating). As the result, the elements representing better solutions receive more pheromone than others and become more desirable in a next iteration. In our hybrid algorithm the solutions constructed (proposed) by the GA are treated as solutions achieved by the ACO in some previous iteration, and we use them to specify the initial pheromone level in the solution graph. After that we search for the solution using the ACO algorithm. The structure of the proposed hybrid GA-ACO algorithm is shown by the pseudocode in Figure 6.

V. Numerical Results and Discussion

The theoretical background of the GA and the ACO is presented in details[24]. For the considered here model problem of parameter identification, we used real-value coded GA instead binary encoding. Therefore the basic operators in the applied GA are as follows:

- encoding – real-value,
- fitness function – linear ranking,
- selection function – roulette wheel selection,
- crossover function – extended intermediate recombination,
- mutation function – real-value mutation,

```
GA-ACO hybrid algorithm
$i = 0$
Initial population $Pop(0)$
Evaluate $Pop(0)$
while not end-condition do
        $i = i + 1$
        Select $Pop(i)$ from $Pop(i - 1)$
        Recombine $Pop(i)$
        Mutate $Pop(i)$
        Evaluate $Pop(i)$
end while
Best GA solution for ACO
Initialize number of ants;
Initialize the ACO parameters;
Initialize the pheromone
while not end-condition do
        for $k = 0$ to number of ants
            ant $k$ choses start node;
            while solution is not constructed do
                    ant $k$ selects higher probability node;
            end while
        end for
        Update-pheromone-trails;
end while
Final solution
```

Fig. 6. Pseudocode for Hybrid GA-ACO

- reinsertion – fitness-based.

In the applied ACO algorithm, the problem is represented by graph and the artificial ants try to construct the shortest path (under specified conditions). In our case the graph of the problem is represented by three partity graph. There are not arcs inside a level and there are arcs between (three) levels. Every level corresponds to one of the model parameters we identify (μ_{max}, k_S and $Y_{S/X}$). Every level consists of 1000 vertexes, which corresponds to 1000 uniformly distributed points in the domain (interval) of every one of the considered model parameters. The pheromone is positioned on the arcs. The ants create a solution starting from random node from the first level. They chose nodes from other levels applying the probabilistic rule. In this application the probabilistic rule uses only the pheromone value. We can think that the heuristic information is constant. Thus the ants will prefer the nodes with maximal quantity of the pheromone.

To set the optimal settings of the GA and the ACO algorithms parameters, we performed several runs of the algorithms with varying parameters, according to the considered here optimization problem. The resulting optimal settings of the GA and the ACO parameters are summarized in Table I and in Table II.

The computer, used to run all identification procedures, was an Intel Core i5-2329 3.0 GHz, with 8 GB Memory, Windows 7 (64bit) operating system and Matlab 7.5 environment.

TABLE I
PARAMETERS OF GA

Parameter	**Value**
ggap	0.97
xovr	0.7
mutr	0.05
maxgen	200
individuals	100
nvar	3
inserted rate	100 %

TABLE II
PARAMETERS OF ACO ALGORITHM

Parameter	**Value**
number of ants	20
initial pheromone	0.5
evaporation	0.1
generations	200

We performed 30 independent runs of the hybrid GA-ACO. The hybrid algorithm started with population of 20 chromosomes. We used 40 generations to find the initial solution. Next, we took the achieved best GA solution to specify the ACO initial pheromones. Next, the ACO was used to obtain the best model parameters vector using 20 ants for 100 generations (see, Table III).

For comparison of performance of the hybrid algorithm we used the pure GA and the pure ACO. They were run (30 times) with (optimized) parameters shown in Table I and in Table II.

The main numerical results, obtained when solving the parameter identification problem, are summarized in Table IV. In this table we show the best, worst and average values of the objective function achieved by the pure ACO, the pure GA and the hybrid GA-ACO algorithms after 30 run of every one of them, as well as their running times. The obtained average values of the model parameters (μ_{max}, k_S and $Y_{S/X}$) are summarized in Table V.

As it can be seen, from Table IV, the hybrid GA-ACO achieves values of the objective function that are similar to these obtained by the pure GA and the pure ACO algorithms. In the same time, the running time of the proposed hybrid algorithm is about two times shorter. The pure ACO algorithm starts with an equal initial pheromone distribution for all problem elements. In the case of the hybrid GA-ACO we use the best solution found by the GA to specify the initial distribution of the pheromone (used by the ACO). Thus our ACO algorithm uses the GA "experience" and starts from a "better" pheromone distribution. This strategy helps the ants

TABLE III
PARAMETERS OF GA-ACO ALGORITHM

Parameter	**Value**
ggap	0.97
xovr	0.7
mutr	0.05
GA maxgen	40
individuals	20
nvar	3
inserted rate	100 %
number of ants	20
initial pheromone	0.5
evaporation	0.1
ACO generations	100

TABLE IV
RESULTS FROM MODEL PARAMETERS IDENTIFICATION PROCEDURES

Value	**Algorithm**	**Algorithm performance**	
		T, [s]	J
best	GA	67.5172	4.4396
	ACO	67.3456	4.9190
	GA-ACO	38.7812	4.3803
worst	GA	66.5968	4.6920
	ACO	66.6280	6.6774
	GA-ACO	41.4495	4.6949
average	GA	67.1370	4.5341
	ACO	69.5379	5.5903
	GA-ACO	39.4620	4.5706

to find “good solutions” using less computational resources (e.g. like computer time and memory). As a matter of fact, our hybrid algorithm uses more than three times less memory than the pure ACO and the pure GA algorithms.

In Table VI we compare results achieved in current work with results obtained in our earlier work [32]. There, we had run the ACO algorithm for several iterations and used it to generate an initial populations for the GA algorithm. Thus the GA started from a population that was closer to the good (optimal) solution than a randomly generated population. We observe that the ACO-GA and the GA-ACO algorithms achieve very similar results, and in a similar running time. We run the ANOVA test to measure the relative difference between the two algorithms. The two hybrid algorithms achieves statistically equivalent results, but the GA-ACO algorithm uses 30% less memory. Thus we can conclude that hybrid GA-ACO algorithm performs better than the ACO-GA hybrid algorithm.

TABLE V
PARAMETERS’ ESTIMATIONS OF THE *E. coli* FED-BATCH FERMENTATION PROCESS MODEL

Value	**Algorithm**	**Model parameters**		
		μ_{max}	k_S	$1/Y_{S/X}$
average	GA	0.4857	0.0115	2.0215
	ACO	0.5154	0.0151	2.0220
	GA-ACO	0.4946	0.0123	2.0204

TABLE VI
RESULTS FROM MODEL PARAMETERS IDENTIFICATION PROCEDURES: ACO-GA

Value	**ACO-GA performance**	
	T, [s]	J
best	35.5212	4.4903
worst	41.4495	4.6865
average	36.1313	4.5765

In Figure 7, the comparison of the dynamics of measured and modeled biomass concentration is shown. With a solid line we show the modeled biomass during the fermentation process, while with stars we show the measured biomass concentration. We put only several stars because the two line are almost overlapped. In Figure 8 the comparison between the time profiles of measured and modeled substrate concentration, during the fermentation process, is shown. On both figures we observe how close are the modeled and the measured data. Thus we illustrate the quality of our hybrid GA-ACO algorithm.

VI. CONCLUSION

In this paper we propose a hybrid GA-ACO algorithm for parameter identification of the *E. coli* fed-batch fermentation process. In the proposed approach, first, we start the GA for several generations with a small population. Next, we use the best solution found by the GA, to instantiate the initial pheromone distribution for the ACO algorithm. We observe that our hybrid GA-ACO algorithm achieves results similar to the pure GA and the pure ACO algorithms, but it is using less computational resources (time and memory). The used time is two times smaller while the used memory is three times smaller. With this algorithm we understand how important is the pheromone distribution for good performance of the ACO algorithm. We compare our hybrid GA-ACO approach, with a hybrid ACO-GA algorithm. Both hybrid algorithms

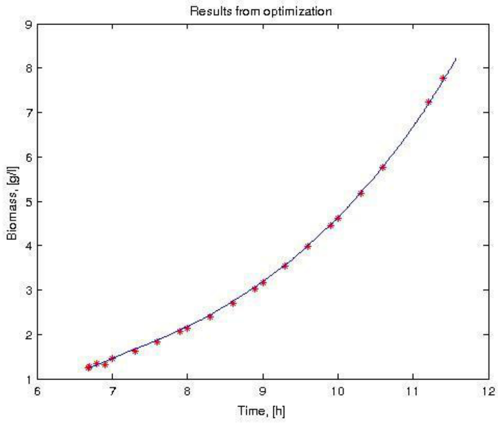

Fig. 7. *E. coli* fed-batch fermentation process: comparison between measured and modeled biomass concentration

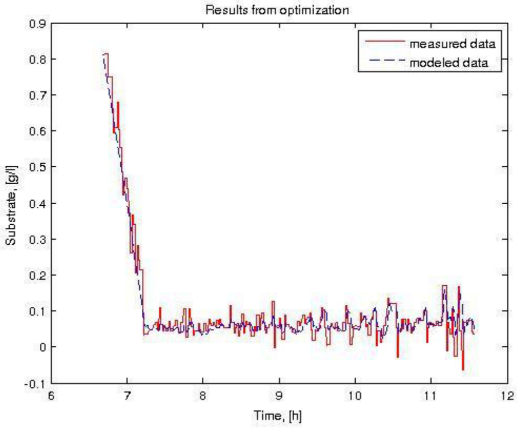

Fig. 8. *E. coli* fed-batch fermentation process: comparison between measured and modeled substrate concentration

achieve statistically similar results for a similar running time, but GA-ACO algorithm uses about 30% less memory, which is important when one is to solve large problems.

Acknowledgment

Work presented here is a part of the Poland-Bulgarian collaborative Grant ”Parallel and distributed computing practices” and the project AComIn “Advanced Computing for Innovation,” Grant 316087, funded by FP7 Capacity Programme (Research Potential of Convergence Regions).

References

[1] G. J. Woeginger, “Exact Algorithms for NP-Hard Problems: A Survey”, Lecture Notes in Computer Science, Volume 2570, 2003, pp. 185–207.

[2] M. Battarra , A. A. Pessoa, A. Subramanian and E. Uchoa, “Exact Algorithms for the Traveling Salesman Problem with Draft Limits”, European Journal of Operational Research, Volume 235, Issue 1, 2014, pp. 115–128.

[3] I. Dumitrescu and T. Sttzle, “Combinations of Local Search and Exact Algorithms”, G.R. Raidl (Ed.) et al., Applications of Evolutionary Computation, Lecture Notes in Computer Science, Vol. 2611, 2003, pp. 211-223.

[4] F. Glover and G. Kochenberger (Eds.), “Handbook of Metaheuristics”, International Series in Operations Research and Management Science, Kluwer Academic Publishers, Vol. 57, 2003.

[5] N. Harvey, ”Use of Heuristics: Insights from Forecasting Research”, Thinking & Reasoning, Vol. 13 Issue 1, 2007, pp. 5–24.

[6] H. Smith, ”Use of the Anchoring and Adjustment Heuristic by Children”, Current Psychology: A Journal For Diverse Perspectives On Diverse Psychological Issues, Vol. 18 Issue 3,1999, pp. 294–300.

[7] C. Blum and A. Roli, “Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison”, ACM Computing Surveys, Vol. 35(3), 2003, pp. 268-308.

[8] I. Boussaid, J. Lepagnot and P. Siarry, “A Survey on Optimization Metaheuristics”, Information Sciences, Vol. 237, 2013, pp. 82-117.

[9] J. Toutouh, “Metaheuristics for Optimal Transfer of P2P Information in VANETs”, MSc Thesis, University of Luxembourg, 2010.

[10] P. Tangpattanakul, N. Jozefiwiez and P. Lopez, ”Biased Random Key Genetic Algorithm with Hybride Decoding for Multi-objective Optimization”, In Proc. of FedCSIS conference, Poland, 2013, pp. 393 – 400.

[11] E. Deniz Ulker and A. Haydar, ”A Hybrid Algorithm Based on Differential Evolution, Particle Swarm Optimization and Harmony Search Algorithms”, n Proc. of FedCSIS conference, Poland, 2013, pp.417 – 420.

[12] E. G. Talbi and El-ghazali (Ed.), “Hybrid Metaheuristics”, Studies in Computational Intelligence, Vol. 434, 2013, XXVI, 458 p. 109 illus.

[13] E. G. Talbi, “A Taxonomy of Hybrid Metaheuristics”, Journal of Heuristics, 8, 2002, pp. 541-564.

[14] A. Georgieva and I. Jordanov, “Hybrid Metaheuristics for Global Optimization using Low-discrepancy Sequences of Points”, Computers and Operation Research, Vol. 37(3), 2010, pp. 456–469.

[15] H. Guangdong, P. Ling and Q. Wang, “A Hybrid Metaheuristic ACO-GA with an Application in Sports Competition Scheduling”, Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, Vol. 3, 2007, pp. 611–616.

[16] H. Guangdong and Q. Wang, “A Hybrid ACO-GA on Sports Competition Schedulingby Ant Colony Optimization - Methods and Applications”, Edited by Avi Ostfeld, 2011, pp. 89–100.

[17] A. Csebfalv, “A Hybrid Meta-heuristic Method for Continuous Engineering Optimization”, Civil Engineering, Vol. 53/2, 2009, pp. 93-100.

[18] S. AlMuhaideb and M. El B. Menai, “A New Hybrid Metaheuristic for Medical Data Classification”, Int. J. of Metaheuristics, Vol. 3(1), 2014, pp. 59–80.

[19] Yi H., Q. Duan and T. Warren Liao, “Three Improved Hybrid Metaheuristic Algorithms for Engineering Design Optimization”, Applied Soft Computing, Vol. 13(5), 2013, pp. 2433-2444.

[20] M. Lukasiewycz, M. Gla, F. Reimann, and J. Teich, “Opt4J - A Modular Framework for Meta-heuristic Optimization”, In Proc. of the Genetic and Evolutionary Computing Conference (GECCO 2011), Dublin, Ireland, 2011, pp. 1723-1730.

[21] S. Masrom, S. Z. Z. Abidin, P. N. Hashimah, and A. S. Abd. Rahman, “Towards Rapid Development of User Defined Metaheuristics Hybridisation”, International Journal of Software Engineering and Its Applicatons, Vol. 5, 2011.

[22] A. Acan, “A GA + ACO Hybrid for Faster and Better Search Capability”, In: Ant Algorithms: Proc. of the Third International Workshop, ANTS 2002, Lecture Notes in Computer Science, 2002.

[23] O. Roeva, T. Pencheva, B. Hitzmann, St. Tzonkov, “A Genetic Algorithms Based Approach for Identification of Escherichia coli Fed-batch Fermentation”, Int. J. Bioautomation, Vol. 1, 2004, pp. 30–41.

[24] O. Roeva and S. Fidanova, “Chapter 13. A Comparison of Genetic Algorithms and Ant Colony Optimization for Modeling of E. coli Cultivation Process”, In book “Real-World Application of Genetic Algorithms”, O. Roeva (Ed.), InTech, 2012, pp. 261–282.

[25] E. Bonabeau, M. Dorigo and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*, New York,Oxford University Press, 1999.

[26] M. Dorigo and T. Stutzle, *Ant Colony Optimization*, MIT Press, 2004.

[27] M. Arndt and B. Hitzmann, “Feed Forward/feedback Control of Glucose Concentration during Cultivation of *Escherichia coli*”, 8th IFAC Int. Conf. on Comp. Appl. in Biotechn, Canada, 2001, pp. 425–429.

[28] O. Roeva, “Improvement of Genetic Algorithm Performance for Identification of Cultivation Process Models”, Advanced Topics on Evolutionary Computing, Book Series: Artificial Intelligence Series-WSEAS, 2008, pp. 34–39.

[29] D. E. Goldberg, “Genetic Algorithms in Search, Optimization and Machine Learning”, Addison Wesley Longman, London, 2006.

[30] J. H. Holland, “Adaptation in Natural and Artificial Systems”, 2nd Edn. Cambridge, MIT Press, 1992.

[31] M. Dorigo and T. Stutzle, “Ant Colony Optimization”, MIT Press, 2004.

[32] O. Roeva, S. Fidanova, V. Atanassova, ”Hybrid ACO-GA for Parameter Identification of an E. coli Cultivation Process Model”, Large-Scale Scientific Computing, Lecture Notes in Computer Science 8353, Springer, Germany, ISSN 0302-9743, 2014, 288 – 295.

[33] http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol1/hmw/article1.html (last accessed April 14, 2014)