13

Application of Genetic Algorithms and Ant Colony Optimization for Modelling of *E. coli* Cultivation Process

Olympia Roeva[1] and Stefka Fidanova[2]
[1]*Institute of Biophysics and Biomedical Engineering, Bulgarian Academy of Sciences*
[2]*Institute of Information and Communication Technologies, Bulgarian Academy of Sciences*
Bulgaria

1. Introduction

Classical biotechnology is the science of production of human-useful processes and products under controlled conditions, applying biological agents – microorganisms, plant or animal cells, their exo- and endo- products, e.g. enzymes, etc. (Viesturs et al., 2004). The conventional agriculture or chemistry cannot perform these processes as efficiently or at all. In fact, conventional biotechnology has been the largest industrial activity on earth for a very long time. Modern biotechnology goes much further with respect to control of the biological processes.

Particularly microorganisms have received a lot of attention as a biotechnological instrument and are used in so-called cultivation processes. Numerous useful bacteria, yeasts and fungi are widely found in nature, but the optimum conditions for growth and product formation in their natural environment is seldom discovered. In artificial (in vitro) conditions, the biotechnologist can intervene in the microbial cell environment (in a fermenter or bioreactor), as well as in their genetic material, in order to achieve a better control of cultivation processes. Because of their extremely high synthetic versatility, ease of using renewable raw materials, great speed of microbial reactions, quick growth and relatively easy to modify genetic material, many microorganisms are extremely efficient and in many cases indispensable workhorses in the various sectors of industrial biotechnology.

Cultivation of recombinant micro-organisms e.g. *Escherichia coli*, in many cases is the only economical way to produce pharmaceutical biochemicals such as interleukins, insulin, interferons, enzymes and growth factors. Simple bacteria like *E. coli* are manipulated to produce these chemicals so that they are easily harvested in vast quantities for use in medicine. *E. coli* is still the most important host organism for recombinant protein production. Scientists may know more about *E. coli* than they do about any other species on earth. Research on E. coli accelerated even more after 1997, when scientists published its entire genome. They were able to survey all 4,288 of its genes, discovering how groups of them worked together to break down food, make new copies of DNA and do other tasks. But despite decades of research there is a lot more we need to know about *E. coli*. To find out more, *E. coli* experts have been joining forces. In 2002, they formed the *International E-coli Alliance* to organize projects that many laboratories could do together. As knowledge of *E. coli* grows, scientists

are starting to build models of the microbe that capture some of its behavior. It is important to be able to predict how fast the microbe will grow on various sources of food, as well as how its growth changes if individual genes are knocked out. Here is the place of mathematical modelling. Some of recent researches and developed models of *E. coli* are presented in (Covert et al., 2008; Jiang et al., 2010; Karelina et al., 2011; Opalka et al., 2011; Petersen et al., 2011; Skandamis & Nychas, 2000).

Modelling of biotechnological processes is a common tool in process technology. Development of adequate models is an important step for process optimization and high-quality control. In an ideal world, process modelling would be a trivial task. Models would be constructed in a simple manner just to reproduce the true process behaviour. In the real world it is obvious that the model is always a simplification of the reality. This is especially true when trying to model natural systems containing living organisms. For many industrial relevant processes however detailed models are not available due to insufficient understanding of the underlying phenomena. The mathematical models, which naturally could be incomplete and inaccurate to a certain degree, can still be very useful and effective tools in describing those effects which are of great importance for control, optimization, or for understanding of the process. At present the models can be applied in practice since computers allow numerical solution of process models of such complexity that could hardly be imagined a couple of decades ago. Thus numerical solution of the models is the fundament for the development of economic and powerful methods in the fields of bioprocess design, plant design, scale-up, optimization and bioprocess control (Schuegerl & Bellgardt, 2000).

The mathematical modelling of biotechnological processes is an extremely wide field that covers all important kinds of processes with many different microorganisms or cells of plants and animals. The mathematical model is a tool that allows to be investigated the static and dynamic behaviour of the process without doing (or at least reducing) the number of practical experiments. In practice, an experimental approach often has serious limitations that make it necessary to work with mathematical models instead.

Modelling approaches are central in system biology and provide new ways towards the analysis and understanding of cells and organisms. A common approach to model cellular dynamics is the sets of nonlinear differential equations. Real parameter optimization of cellular dynamics models has especially become a research field of great interest. Such problems have widespread application.

The principle of mathematical optimization consists in choice of optimization criteria, choice of control parameters and choice of exhaustive method. Parameter identification of a nonlinear dynamic model is more difficult than the linear one, as no general analytic results exist. The difficulties that may arise are such as convergence to local solutions if standard local methods are used, over-determined models, badly scaled model function, etc. Due to the nonlinearity and constrained nature of the considered systems, these problems are very often multimodal. Thus, traditional gradient-based methods may fail to identify the global solution. In this case only direct optimization strategies can be applied, because they exclusively use information about values of the goal function. These optimization methods provide more guarantees of converging to the global optimal solution. Although a lot of different global optimization methods exist, the efficacy of an optimization method is always problem-specific. A major deficiency in computational approaches to design and optimization of bioprocess systems is the lack of applicable methods.

There are many possible variants such as numerical methods (Lagarias et al., 1998; Press et al., 1986). But while searching for new, more adequate modeling metaphors and concepts, methods which draw their initial inspiration from nature have received the early attention. During the last decade metaheuristic techniques have been applied in a variety of areas. Heuristics can obtain suboptimal solution in ordinary situations and optimal solution in particular. Since the considered problem has been known to be NP-complete, using heuristic techniques can solve this problem more efficiently. Three most well-known heuristics are the iterative improvement algorithms, the probabilistic optimization algorithms, and the constructive heuristics. Evolutionary algorithms like Genetic Algorithms (GA) (Goldberg, 2006; Holland, 1992; Michalewicz, 1994) and Evolution Strategies, Ant Colony Optimization (ACO) (Dorigo & Di Caro, 1999; Dorigo & Stutzle, 2004; Fidanova, 2002; Fidanova et al., 2010), Particle Swarm Optimization (Umarani & Selvi, 2010), Tabu Search (Yusof & Stapa, 2010), Simulated Annealing (Kirkpatrick et al., 1983), estimation of distribution algorithms, scatter search, path relinking, the greedy randomized adaptive search procedure, multi-start and iterated local search, guided local search, and variable neighborhood search are - among others - often listed as examples of classical metaheuristics (Bonabeau et al., 1999; Syam & Al-Harkan, 2010; Tahouni et al., 2010), and they have individual historical backgrounds and follow different paradigms and philosophies (Brownlee, 2011). In this work the GA and ACO are chosen as the most common direct methods used for global optimization.

The GA is a model of machine learning deriving its behaviour from a metaphor of the processes of evolution in nature. This is done by the creation within a machine of a population of individuals represented by chromosomes. A chromosome could be an array of real numbers, a binary string, a list of components in a database, all depending on the specific problem. Each individual represents a possible solution, and a set of individuals form a population. In a population, the fittest are selected for mating. The individuals in the population go through a process of evolution which is, according to Darwin, made up of the principles of mutation and selection; however, the modern biological evolution theory distinguishes also crossover and isolation mechanisms improving the adaptiveness of the living organisms to their environment. The principal advantages of GA are domain independence, non-linearity and robustness. The only requirement for GA is the ability to calculate the measure of performance which may be highly complicated and non-linear. The above two characteristics of GA assume that GA is inherently robust. A GA has a number of advantages. It can work with highly non-linear functions and can cope with a great diversity of problems from different fields. It can quickly scan a vast solution set. Bad proposals do not effect the end solution negatively as they are simply discarded. The inductive nature of the GA means that it doesn't have to know any rules of the problem - it works by its own internal rules. This is very useful for complex or loosely defined problems. However, the conventional GA has a very poor local performance because of the random search used. To achieve a good solution, great computational cost is inevitable. The same qualities that make the GA so robust also can make it more computationally intensive and slower than other methods.

On the other hand ACO is a rapidly growing field of a population-based metaheuristic that can be used to find approximate solutions to difficult optimization problems. ACO is applicable for a broad range of optimization problems, can be used in dynamic applications (adapts to changes such as new distances, etc) and in some complex biological problems (Fidanova & Lirkov, 2009; Fidanova, 2010; Shmygelska & Hoos, 2005). ACO can compete with other global optimization techniques like genetic algorithms and simulated annealing. ACO algorithms have been inspired by the real ants behavior. In nature, ants usually wander

randomly, and upon finding food return to their nest while laying down pheromone trails. If other ants find such a path, they are likely to not keep traveling at random, but to follow the trail instead, returning and reinforcing it if they eventually find food. However, as time passes, the pheromone starts to evaporate. The more time it takes for an ant to travel down the path and back again, the more time the pheromone has to evaporate and the path becomes less noticeable. A shorter path, in comparison will be visited by more ants and thus the pheromone density remains high for a longer time. ACO is implemented as a team of intelligent agents which simulate the ants behavior, walking around the graph representing the problem to solve using mechanisms of cooperation and adaptation.

In this chapter GA and ACO are applied for parameter identification of a system of nonlinear differential equations modeling the fed-batch cultivation process of the bacteria *Escherichia coli*. A system of ordinary differential equations is proposed to model *E. coli* biomass growth and substrate (glucose) utilization. Parameter optimization is performed using real experimental data set from an *E. coli* MC4110 fed-batch cultivation process. The cultivation is performed in *Institute of Technical Chemistry, University of Hannover, Germany* during the collaboration work with the *Institute of Biophysics and Biomedical Engineering, BAS, Bulgaria*, granted by *DFG*.

The experimental data set includes records for substrate feeding rate, concentration of biomass and substrate (glucose) and cultivation time. In considered here nonlinear mathematical model the parameters that should be estimated are maximum specific growth rate (μ_{max}), saturation constant (k_S) and yield coefficient ($Y_{S/X}$).

The parameter estimation is performed based upon the use of Hausdorff metric (Rote, 1991), in place of the most commonly used metric – Least Squares regression. Hausdorff metrics are used in geometric settings for measuring the distance between sets of points. They have been used extensively in areas such as computer vision, pattern recognition and computational chemistry (Chen & Lovell, 2010; Nutanong et al., 2010; Sugiyama et al., 2010; Yedjour et al., 2011). A modified Hausdorff Distance is proposed to evaluate the mismatch between experimental and model predicted data.

The results from both metaheuristics GA and ACO are compared using the modified Hausdorff Distance. The algorithms accuracy (value of the objective function) and the resulting average, best and worst model parameter estimations are compared for the model identification of the *E. coli* MC4110 fed-batch cultivation process.

The chapter is organized as follows: In Section 2 the problem definition is formulated. As a case study an fed-batch cultivation of bacteria *E. coli* is presented. Further optimization criteria is defined. In Section 3 the theoretical background of the GA is presented. In Section 4 the theoretical background of the ACO is presented. The numerical results and a discussion are presented in Section 5. The GA and ACO adjustments for considered parameter identification problem application are discussed too. Conclusion remarks are done in Section 6.

2. Problem definition

Cultivation process are known to be very complex and modeling may be a rather time consuming. However, it is neither necessary nor desirable to construct comprehensive mechanistic process models that can describe the system in all possible situations with a high accuracy. In order to optimize a real biotechnical production process, the model must be

regarded as a step to reach more easily the final aim. The model must describe those aspects of the process that significantly affect the process performance.

The costs of developing mathematical models for bioprocesses improvement are often too high and the benefits too low. The main reason for this is related to the intrinsic complexity and non-linearity of biological systems. In general, mathematical descriptions of growth kinetics assume hard simplifications. These models are often not accurate enough at describing the underlying mechanisms. Another critical issue is related to the nature of bioprocess models. Often the parameters involved are not identifiable. Additionally, from the practical point of view, such identification would require data from specific experiments which are themselves difficult to design and realize. The estimation of model parameters with high parameter accuracy is essential for successful model development.

The important part of model building is the choice of a certain optimization procedure for parameter estimation, so with a given set of experimental data to calibrate the model in order to reproduce the experimental results in the best possible way.

Real parameter optimization of simulation models has especially become a research field of great interests in recent years. Nevertheless, this task still represents a very difficult problem. This mathematical problem, so-called inverse problem, is a big challenge for the traditional optimization methods. In this case only direct optimization strategies can be applied, because they exclusively use information about values of the goal function. Additional information about the goal function like gradients, etc., which may be used to accelerate the optimization process, is not available. Since an evolution of a goal for one string is provided by one simulation run, proceeding of an optimization algorithm may require a lot of computational time. Thus or therefore, various metaheuristics are used as an alternative to surmount the parameter estimation difficulties.

2.1 *E. coli* fed-batch cultivation process

To maximize the volumetric productiveness of bacterial cultures it is important to grow *E. coli* to high cell concentration. The use of fed-batch cultivation in the fermentation industry takes advantage of the fact that residual substrate concentration may by maintained at a very low level in such a system.

The general state space dynamical model described by Bastin and Dochain (Bastin & Dochain, 1991) is accepted as representing the dynamics of an n components and m reactions bioprocess:

$$\frac{dx}{dt} = K\varphi(x,t) - Dx + F - Q \tag{1}$$

where x is a vector representing the state components; K is the yield coefficient matrix; φ is the growth rates vector; the vectors F and Q are the feed rates and the gaseous outflow rates. The scalar D is the dilution rate, which will be the manipulated variable, defined as follows:

$$D = \frac{F_{in}}{V} \tag{2}$$

where F_{in} is the influent flow rate and V is the bioreactor's volume.

Application of the general state space dynamical model (Bastin & Dochain, 1991) to the *E. coli* cultivation fed-batch process leads to the following nonlinear differential equation system

(Roeva, 2008b):

$$\frac{dX}{dt} = \mu_{max}\frac{S}{k_S + S}X - \frac{F_{in}}{V}X \tag{3}$$

$$\frac{dS}{dt} = -\frac{1}{Y_{S/X}}\mu_{max}\frac{S}{k_S + S}X + \frac{F_{in}}{V}(S_{in} - S) \tag{4}$$

$$\frac{dV}{dt} = F_{in} \tag{5}$$

where:

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

The growth rate of bacteria *E. coli* is described according to the classical Monod equation:

$$\mu = \mu_{max}\frac{S}{k_S + S} \tag{6}$$

The mathematical formulation of the nonlinear dynamic model (Eqs. (3) - (5)) of *E. coli* fed-batch cultivation process is described according to the mass balance and the model is based on the following a priori assumptions:

- the bioreactor is completely mixed;
- the main products are biomass, water and, under some conditions, acetate;
- the substrate glucose mainly is consumed oxidatively and its consumption can be described by Monod kinetics;
- variation in the growth rate and substrate consumption do not significantly change the elemental composition of biomass, thus balanced growth conditions are only assumed;
- parameters, e.g. temperature, pH, pO_2 are controlled at their individual constant set points.

For the parameter estimation problem real experimental data of the *E. coli* MC4110 fed-batch cultivation process are used. Off-line measurements of biomass and on-line measurements of the glucose concentration are used in the identification procedure. The cultivation condition and the experimental data have been presented in (Roeva et al., 2004). Here a brief description is presented.

The fed-batch cultivation of *E. coli* MC4110 is performed in a 2l bioreactor (Bioengineering, Switzerland), using a mineral medium (Arndt & Hitzmann, 2001), in *Institute of Technical Chemistry, University of Hannover*. Before inoculation a glucose concentration of 2.5 g/l is established in the medium. Glucose in feeding solution is 100 g/l. Initial liquid volume is 1350 ml, pH is controlled at 6.8 and temperature is kept constant at 35°C . The aeration rate is kept at 275 l/h air, stirrer speed at start 900 rpm, after 11h the stirrer speed is increased in steps of 100 rpm and at end is 1500 rpm. Oxygen is controlled around 35%.

Off-line analysis
For off-line glucose measurements as well as biomass and acetate concentration determination samples of about 10 ml are taken roughly every hour. Off-line measurements are performed by using the Yellow Springs Analyser (Yellow Springs Instruments, USA).

On-line analysis
For on-line glucose determination a flow injection analysis (FIA) system has been employed using two pumps (ACCU FM40, SciLog, USA) for a continuous sample and carrier flow rate. To reduce the measurement noise the continuous-discrete extended Kalman filter are used (Arndt & Hitzmann, 2001).

Glucose measurement and control system
For on-line glucose determination a FIA system has been employed using two pumps (ACCU FM40, SciLog, USA) for a continuous sample and carrier flow rate at 0.5 ml/min and 1.7 ml/min respectively. 24 ml of cell containing culture broth were injected into the carrier stream and mixed with an enzyme solution of 350 000 U/l of glucose oxidase (Fluka, Germany) of a volume of 36 ml. After passing a reaction coil of 50 cm length the oxygen uptake were measured using an oxygen electrode (ANASYSCON, Germany). To determine only the oxygen consumed by cells no enzyme solution were injected. Calculating the difference of both dissolved oxygen peak heights, the glucose concentration can be determined. The time between sample taking and the measurement of the dissolved oxygen was Δt = 45 s.

For the automation of the FIA system as well as glucose concentration determination the software CAFCA (ANASYSCON, Germany) were applied. To reduce the measurement noise the continuous-discrete extended Kalman filter were used. This program was running on a separate PC and got the measurement results via a serial connection. A PI controller was applied to adjust the glucose concentration to the desired set point of 0.1 g/l (Arndt & Hitzmann, 2001).

The initial process conditions are (Arndt & Hitzmann, 2001):
t_0 = 6.68 h, $X(t_0)$ = 1.25 g/l, $S(t_0)$ = 0.8 g/l, S_{in} = 100 g/l.

The bioreactor, as well as FIA measurement system and the computers used for data measurement from the FIA system and for the process control are presented in Figure 1.

2.2 Optimization criterion

In practical view, modelling studies are performed to identify simple and easy-to-use models that are suitable to support the engineering tasks of process optimization and, especially, of control. The most appropriate model must satisfy the following conditions:

(i) the model structure should be able to represent the measured data in a proper manner;
(ii) the model structure should be as simple as possible compatible with the first requirement.

On account of that the cultivation process dynamic is described using simple Monod-type model, the most common kinetics applied for modelling of cultivation processes (Bastin & Dochain, 1991).

The optimization criterion is a certain factor, whose value defines the quality of an estimated set of parameters. The parameter estimation is performed based on Hausdorff metric. To evaluate the mishmash between experimental and model predicted data a modified Hausdorff Distance is proposed.

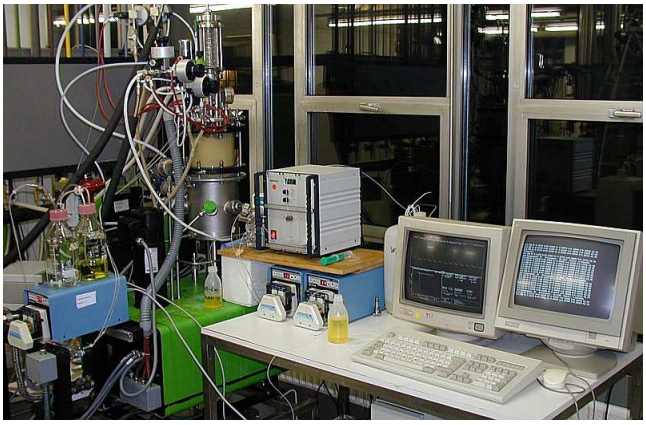


Fig. 1. Experimental equipment

When talking about distances, we usually mean the shortest: for instance, if a point X is said to be at distance D of a polygon P, we generally assume that D is the distance from X to the nearest point of P. The same logic applies for polygons: if two polygons A and B are at some distance from each other, we commonly understand that distance as the shortest one between any point of A and any point of B. That definition of distance between polygons can become quite unsatisfactory for some applications. However, we would naturally expect that a small distance between these polygons means that no point of one polygon is far from the other polygon. It's quite obvious that the shortest distance concept carries very low informative content.

In mathematics, the Hausdorff distance, or Hausdorff metric, also called Pompeiu-Hausdorff distance, (Rote, 1991) measures how far two subsets of a metric space are from each other. It turns the set of non-empty compact subsets of a metric space into a metric space in its own right. It is named after Felix Hausdorff. Informally, two sets are close in the Hausdorff distance if every point of either set is close to some point of the other set. The Hausdorff distance is the longest distance you can be forced to travel by an adversary who chooses a point in one of the two sets, from where you then must travel to the other set. In other words, it is the farthest point of a set that you can be to the closest point of a different set. More formally, Hausdorff distance from set A to set B is a maxmin function defined as:

$$h(A,B) = \max_{a \in A} \left\{ \min_{b \in B} \{ d(a,b) \} \right\}, \tag{7}$$

where a and b are points of sets A and B respectively, and $d(a,b)$ is any metric between these points. For simplicity, we will take $d(a,b)$ as the Euclidean distance between a and b. If sets A and B are made of lines or polygons instead of single points, then $h(A,B)$ applies to all defining points of these lines or polygons, and not only to their vertices. Hausdorff distance gives an interesting measure of their mutual proximity, by indicating the maximal distance between any point of one set to the other set. Better than the shortest distance, which applied only to one point of each set, irrespective of all other points of the sets.

In this work the Hausdorff metric is used for first time for solving of model parameter optimization problem regarding cultivation process models.

3. Genetic Algorithm

GA originated from the studies of cellular automata, conducted by John Holland and his colleagues at the University of Michigan. Holland's book (Holland, 1992), published in 1975, is generally acknowledged as the beginning of the research of genetic algorithms. The GA is a model of machine learning which derives its behavior from a metaphor of the processes of evolution in nature (Goldberg, 2006). This is done by the creation within a machine of a population of individuals represented by chromosomes. A chromosome could be an array of real numbers, a binary string, a list of components in a database, all depending on the specific problem. The GA are highly relevant for industrial applications, because they are capable of handling problems with non-linear constraints, multiple objectives, and dynamic components – properties that frequently appear in the real-world problems (Goldberg, 2006; Kumar et al., 1992). Since their introduction and subsequent popularization (Holland, 1992), the GA have been frequently used as an alternative optimization tool to the conventional methods (Goldberg, 2006; Parker, 1992) and have been successfully applied in a variety of areas, and still find increasing acceptance (Akpinar & Bayhan, 2011; Al-Duwaish, 2000; Benjamin et al., 1999; da Silva et al., 2010; Paplinski, 2010; Roeva & Slavov, 2011; Roeva, 2008a).

Basics of Genetic Algorithm

GA was developed to model adaptation processes mainly operating on binary strings and using a recombination operator with mutation as a background operator. The GA maintains a population of individuals, $P(t) = x_1^t, ..., x_n^t$ for generation t. Each individual represents a potential solution to the problem and is implemented as some data structure S. Each solution is evaluated to give some measure of its "fitness". Fitness of an individual is assigned proportionally to the value of the objective function of the individuals. Then, a new population (generation $t+1$) is formed by selecting more fit individuals (selected step). Some members of the new population undergo transformations by means of "genetic" operators to form new solution. There are unary transformations m_i (mutation type), which create new individuals by a small change in a single individual ($m_i : S \rightarrow S$), and higher order transformations c_j (crossover type), which create new individuals by combining parts from several individuals ($c_j : S \times \ldots \times S \rightarrow S$). After some number of generations the algorithm converges - it is expected that the best individual represents a near-optimum (reasonable) solution. The combined effect of selection, crossover and mutation gives so-called reproductive scheme growth equation (Goldberg, 2006):

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

Differences that separate genetic algorithms from the more conventional optimization techniques could be defined as follows (Goldberg, 2006):

1. Direct manipulation of a coding – GA works with a coding of the parameter set, not the parameter themselves;
2. GA searches in a population of points, not a single point;
3. GA uses payoff (objective function) information, not derivatives or other auxiliary knowledge;
4. GA uses probabilistic transition rules (stochastic operators), not deterministic rules.

Compared with traditional optimization methods, GA simultaneously evaluates many points in the parameter space. This makes convergence towards the global solution more probable. A

genetic algorithm does not assume that the space is differentiable or continuous and can also iterate many times on each data received. A GA requires only information concerning the quality of the solution produced by each parameter set (objective function value information). This characteristic differs from optimization methods that require derivative information or, worse yet, complete knowledge of the problem structure and parameters. Since GA do not demand such problem-specific information, they are more flexible than most search methods. Also GA do not require linearity in the parameters which is needed in iterative searching optimization techniques. Genetic algorithms can solve hard problems, are noise tolerant, easy to interface to existing simulation models, and easy to hybridize. Therefore, this property makes genetic algorithms suitable and more workable in use for a parameter estimation of considered here cultivation process models. Moreover, the GA effectiveness and robustness have been already demonstrated for identification of fed-batch cultivation processes (Carrillo-Uretaet al., 2001; Ranganath et al., 1999; Roeva, 2006; 2007).

The structure of the GA is shown by the pseudocode below (Figure 2).

```
begin
    i = 0
    Initial population P(0)
    Evaluate P(0)
    while (not done) do (test for termination criterion)
    begin
        i = i + 1
        Select P(i) from P(i − 1)
        Recombine P(i)
        Mutate P(i)
        Evaluate P(i)
    end
end
```

Fig. 2. Pseudocode for GA

The population at time t is represented by the time-dependent variable P, with the initial population of random estimates being $P(0)$. Here, each decision variable in the parameter set is encoded as a binary string (with precision of binary representation). The initial population is generated using a random number generator that uniformly distributes numbers in the desired range. The objective function (see Eq. (16)) is used to provide a measure of how individuals have performed in the problem domain.

4. Ant colony optimization

ACO is a stochastic optimization method that mimics the social behaviour of real ants colonies, which manage to establish the shortest route to feeding sources and back. Real ants foraging for food lay down quantities of pheromone (chemical cues) 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 thereby 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 the ants follow a trail, the more attractive that trail becomes. The original idea comes from observing the exploitation of food resources among ants, in which ants' individually limited cognitive

abilities have collectively been able to find the shortest path between a food source and the nest.

Basics of Ant Algorithm

ACO is implemented as a team of intelligent agents which simulate the ants behavior, walking around the graph representing the problem to solve using mechanisms of cooperation and adaptation. The requirements of ACO algorithm are as follows (Bonabeau et al., 1999; Dorigo & Stutzle, 2004):

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

The structure of the ACO algorithm is shown by the pseudocode below (Figure 3).

```
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 chooses start node;
            while solution is not constructed do
                    ant k selects higher probability node;
            end while
        end for
        Update-pheromone-trails;
end while
```

Fig. 3. Pseudocode for ACO

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 $i, j = 1, \ldots, n$.

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

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 resume the search. In the beginning, the initial pheromone level is set to a small positive constant value τ_0; later, the ants update this value after completing the construction stage. 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 evaporation in the nature and $\Delta\tau_{i,j}$ is 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.

5. Numerical results and discussion

For parameter identification of model parameters (μ_{max}, k_S, $Y_{S/X}$) of *E. coli* fed-batch cultivation process model, GA and ACO algorithms are applied.

5.1 Application of GA for parameter optimization of *E. coli* cultivation process model

On this subsection we will describe in more details about the application of GA for parameter optimization of *E. coli* cultivation process model.

Solution Representation
The strings of artificial genetic systems are analogous to chromosomes in biological systems. The total genetic package (genotype) in artificial genetic systems is called a structure. In natural systems, the organism formed by interaction of the genotype with its environment is called the phenotype. In artificial genetic systems, the structures decode to form a particular parameter set, solution alternative, or point (in the solution space). Thus a chromosome representation is needed to describe each individual in the population of interest. The representation scheme determines how the problem is structured in the GA and also determines the genetic operators that are used. Each individual or chromosome is made up of a sequence of genes from a certain alphabet. Here applied alphabet consists of binary digits 0 and 1. Binary representation is the most common one, mainly because of its relative simplicity. A binary 20 bit representation is considered here. It has been shown that more natural representations are more efficient and produce better solutions (Chipperfield & Fleming, 1995; Goldberg, 2006; Michalewicz, 1994). The representation of the individual or chromosome for function optimization involves genes with values within the variables upper and lower bounds.

Three model parameters are represent in the chromosome - maximum specific growth rate (μ_{max}), saturation constant (k_S) and yield coefficient ($Y_{S/X}$). The following upper and lower bounds are considered (Cockshott & Bogle, 1999; Levisauskas et al., 2003):

$$0 < \mu_{max} < 0.7,$$
$$0 < k_S < 1,$$
$$0 < Y_{S/X} < 30.$$

Selection Function
The next question is how to select parents for crossover. The selection of individuals to produce successive generations plays an extremely important role in a GA. A probabilistic selection is performed based upon the individual's fitness such that the better individuals have an increased chance of being selected. An individual in the population can be selected more than once with all individuals in the population having a chance of being selected to reproduce into the next generation. There are several schemes for the selection process - roulette wheel selection and its extensions, scaling techniques, tournament, elitist models,

and ranking methods (Chipperfield & Fleming, 1995; Goldberg, 2006; MathWorks, 1999; Michalewicz, 1994). The selection method used here is the roulette wheel selection.

A common selection approach assigns a probability of selection, P_j, to each individual, j based on its fitness value. A series of N random numbers is generated and compared against the cumulative probability, $C_i = \sum_{j=1}^{i} P_j$ of the population. The appropriate individual, i, is selected and copied into the new population if $C_{i-1} < U(0,1) \leq C_i$. Various methods exist to assign probabilities to individuals: roulette wheel, linear ranking and geometric ranking. Roulette wheel, developed by Holland (Holland, 1992) is the first selection method. The probability, P_i, for each individual is defined by:

$$P[\text{ Individual } i \text{ is chosen}] = \frac{F_i}{\sum_{j=1}^{PopSize} F_j}, \tag{10}$$

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

The fitness function, is normally used to transform the objective function value into a measure of relative fitness. A commonly used transformation is that of proportional fitness assignment.

Genetic Operators

The genetic operators provide the basic search mechanism of the GA. The operators are used to create new solutions based on existing solutions in the population. There are two basic types of operators: crossover and mutation. The crossover takes two individuals and produces two new individuals. The crossover can be quite complicated and depends (as well as the technique of mutation) mainly on the chromosome representation used. The mutation alters one individual to produce a single new solution. By itself, mutation is a random walk through the string space. When used sparingly with reproduction and crossover, it is an insurance policy against premature loss of important notions.

Let $\overline{X}$ and $\overline{Y}$ be two m-dimensional row vectors denoting individuals (parents) from the population. For $\overline{X}$ and $\overline{Y}$ binary, the following operators are defined: binary mutation and simple crossover.

Binary mutation flips each bit in every individual in the population with probability p_m according to Eq. (11) (Houck et al., 1996):

$$x_i = \begin{cases} 1 - x_i, & \text{if } U(0,1) < p_m \\ x_i, & \text{otherwise} \end{cases}. \tag{11}$$

Simple crossover generates a random number r from a uniform distribution from 1 to m and creates two new individuals $\overline{X'}$ and $\overline{Y'}$ according to Eqs. (12) and (13) (Houck et al., 1996).

$$x_i' = \begin{cases} x_i, & \text{if } i < r \\ y_i, & \text{otherwise} \end{cases}. \tag{12}$$

$$y_i' = \begin{cases} y_i, & \text{if } i < r \\ x_i, & \text{otherwise} \end{cases}. \tag{13}$$

In proposed genetic algorithm fitness-based reinsertion (selection of offspring) is used (Pohlheim, 2003).

Initialization, Termination, and Evaluation Functions
The GA must be provided an initial population as indicated in step 3 of Figure 2. The most common method is to randomly generate solutions for the entire population. However, since GA can iteratively improve existing solutions (i.e., solutions from other heuristics and/or current practices), the beginning population can be seeded with potentially good solutions, with the remainder of the population being randomly generated solutions (Houck et al., 1996).

The GA moves from generation to generation selecting and reproducing parents until a termination criterion is met. The most frequently used stopping criterion is a specified maximum number of generations.

Evaluation functions of many forms can be used in a GA, subject to the minimal requirement that the function can map the population into a partially ordered set. As stated, the evaluation function is independent of the GA (i.e., stochastic decision rules) (Houck et al., 1996).

Genetic Parameters
Some adjustments of the genetic parameters, according to the regarded problem, have to be done to improve the optimization capability and the decision speed. Primary choice of the genetic operators and parameters depends on the problem, as well as on the chosen encoding. The inappropriate choice of operators and parameters in the evolutionary process makes the GA susceptible to premature convergence. Based on performed pre-test procedures and other results in (Roeva, 2008a;b), the GA parameters are set as follows.

There are two basic parameters of genetic algorithms - crossover probability and mutation probability. Crossover probability (xovr) should be high generally, about 65-95%, here – 75%. Mutation probability (mutr) is randomly applied with low probability – 0.01 (Obitko, 2005; Pohlheim, 2003). The rate of individuals to be selected (generation gap – ggap) should be defined as well. In proposed genetic algorithm generation gap is 0.97 (Obitko, 2005; Pohlheim, 2003).

Particularly important parameters of GA are the population size (nind) and number of generations (maxgen). If there are too low number of chromosomes, GA has a few possibilities to perform crossover and only a small part of search space is explored. On the other hand, if there are too many chromosomes, GA slows down. To solve the considered optimization problem the population size is chosen to be 100 after several algorithm performance pre-tests. In the same manner the number of generations is set at 200.

For the considered here parameter optimization, the type of the basic operators in GA is summarized in Table 1. The values of genetic algorithm parameters are listed in Table 2.

5.2 Application of ACO for parameter optimization of *E. coli* cultivation process model

On this subsection we will describe in more details about the application of ACO for parameter optimization of *E. coli* cultivation process model. First we represent the problem by graph. We need to find optimal values of three parameters which are interrelated. Therefore we represent the problem with three-partitive graph. The graph consists of three levels. Every level represents a search area of one of the parameter we optimise. Every area is discretized thus, to consists of 1000 points (nodes), which are uniformly distributed in the search interval of every of the parameters. The first level of the graph represents the parameter μ_{max}. The

Operator	Type
encoding	binary
fitness function	linear ranking
selection function	roulette wheel selection
crossover function	simple crossover
mutation function	binary mutation
reinsertion	fitness-based

Table 1. Operators of GA

Parameter	Value
ggap	0.97
xovr	0.75
mutr	0.01
nind	100
maxgen	200

Table 2. Parameters of GA

second level represents the parameter k_S. The third level represents the parameter $Y_{S/X}$. There are arcs between nodes from consecutive levels of the graph and there are no arcs between nodes from the same level. The pheromone is deposited on the arcs, which shows how good is this parameter combination.

Our ACO approach is very close to real ant behaviour. When starting to create a solution, the ants choose a node from the first level in a random way. Than for nodes from second and third level they apply probabilistic rule. The transition probability consists only of the pheromone. The heuristic information is not used. Thus the transition probability is as follows:

$$p_{i,j} = \frac{\tau_{i,j}}{\sum_{k \in Unused} \tau_{i,k}}, \tag{14}$$

The ants prefer the node with maximal probability, which is the node with maximal quantity of the pheromone on the arc, starting from the current node. If there are more than one candidate for next node, the ant choses randomly between the candidates. The process is iterative. At the end of every iteration we update the pheromone on the arcs. The quality of the solutions is represented by the value of the objective function. In our case the objective function is the mean distance between simulated data and experimental data which are the concentration of the biomass and the concentration of the substrate. We try to minimize it, therefore the new added pheromone by ant i in our case is:

$$\Delta\tau = (1 - \rho)/J(i) \tag{15}$$

Where $J(i)$ is the value of the objective function according the solution constructed by ant i. Thus the arcs corresponding to solutions with less value of the objective function will receive more pheromone and will be more desirable in the next iteration.

The values of the parameters of ACO algorithms are very important, because they manage the search process. Therefore we need to find appropriate parameter settings. They are the number of ants, in ACO we can use a small number of ants between 10 and 20 without having to increase the number of iterations to achieve good solutions; initial pheromone, normally it

has a small value; evaporation rate, which shows the importance of the last found solutions according to the previous ones. Parameters of the ACO were tuned based on several pre-tests according to the considered here optimization problem. After tuning procedures the main algorithm parameters are set to the optimal settings. The parameter setting for ACO is shown in Table 3.

Parameter	Value
number of ants	20
initial pheromone	0.5
evaporation	0.1

Table 3. Parameters of ACO algorithm

5.3 Objective function

To form the objective function we apply modified Hausdorff distance, which is conformable to our problem. We have two sets of points, simulated and measure data, which formed two lines. We calculated the Euclidean distance $d(t)$ between points from two lines corresponding to the same time moment t. After that we calculate the Euclidean distance from point of one of the lines in time t to the points from other line in the time interval $(t - d(t), t + d(t))$ and we take the minimal of this distances. This is the distance between two lines in time moment t. Thus we decrease the number of calculations comparing with traditional Hausdorff distance because it is obvious that the distance to the points out of the interval $(t - d(t), t + d(t))$ will be large. At the end we sum all this distances between the points and the lines. Thereby we eliminate eventual larger distance in some time moment because of not precise measurement.

When the Least Squares regression is applied as metric, the distance between two lines can be very big and in the same time it is seen that they are geometrically close to each other. This can happen especially in the steep parts of the lines. Applying Hausdorff metrics avoids this, because it measures the geometrical similarity.

Thus, the objective function is presented as a minimization of a modified Hausdorff distance measure J between experimental and model predicted values of state variables, represented by the vector $\mathbf{y}$:

$$J = \sum_{i=1}^{m} h\left(\mathbf{y}_{\text{exp}}(i), \mathbf{y}_{\text{ mod }}(i)\right)^2 \rightarrow \min \tag{16}$$

where m is the number of state variables; $\mathbf{y}_{\text{exp}}$ – known experimental data; $\mathbf{y}_{\text{ mod}}$ – model predictions with a given set of the parameters.

5.4 Numerical calculation

Computer specification to run all identification procedures are Intel Core 2 2.8 GHz, 3.5 GB Memory, Linux operating system and Matlab 7.5 environment. Matlab is a technical computing environment for high computation. Matlab integrates numerical analysis, matrix computation and graphics in an easy-to-use environment. User-defined Matlab functions are simple text files of interpreted instructions. Therefore, Matlab functions are completely portable from one hardware architecture to another without even a recompilation step.

Because of the stochastic characteristics of the applied algorithms a series of 30 runs for each algorithm is performed. For comparison of the GA and ACO the best, the worst and the

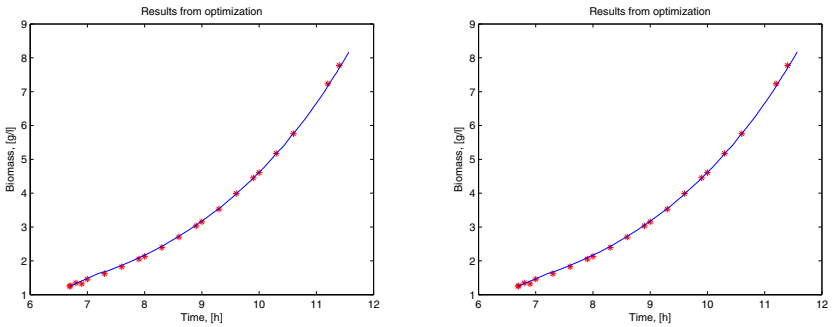


Fig. 4. Time profiles of the biomass, respectively GA and ACO

average results of the 30 runs, for the J value and execution time are watched. For realistic comparison the execution time is fixed to be 1h.

The obtained results are presented in Tables 4 and 5. Regarding the Tables 4 and 5 we observe

Parameters	Average GA	Best GA	Worst GA
μ_{max}	0.5266	0.5537	0.5253
k_S	0.0163	0.0187	0.0164
$Y_{S/X}$	2.0295	2.0318	2.0536
J	2.0699	1.7657	2.3326

Table 4. Results from parameter identification using GA

Parameters	Average ACO	Best ACO	Worst ACO
μ_{max}	0.5444	0.5283	0.5313
k_S	0.0223	0.0174	0.0209
$Y_{S/X}$	2.0256	2.0300	2.0100
J	1.8744	1.6425	2.5322

Table 5. Results from parameter identification using ACO

that the average value of the objective function achieved by ACO algorithm is better than this achieved by GA algorithm. The best value of the objective function achieved by the ACO algorithm is better than this achieved by GA algorithm, but the worst result achieved by ACO algorithm is worst than this achieved by the GA. Thus the interval where the value of the objective function varies is larger when we apply ACO algorithm than GA algorithm. But regarding the average value we can say that the most achieved values of the objective function are close to the best found value. Therefore we can conclude that the ACO algorithm performs better for this problem than GA algorithm.

The objective function is a sum of the modified Hausdorff distance between the modeled and measured data of the biomass and substrate. On Figure 4 with line are represented the values of the modelled biomass and with stars are represented the values of the measured biomass. In most cases, graphical comparisons clearly show the existence or absence of systematic deviations between model predictions and measurements. It is evident that a quantitative measure of the differences between calculated and measured values is an important criterion for the adequacy of a model. We observe that with both algorithms there is coincidence

between modelled and measured data. Hence the difference between the values of the objective function achieved by different algorithms comes from the value of the substrate, achieved by them.

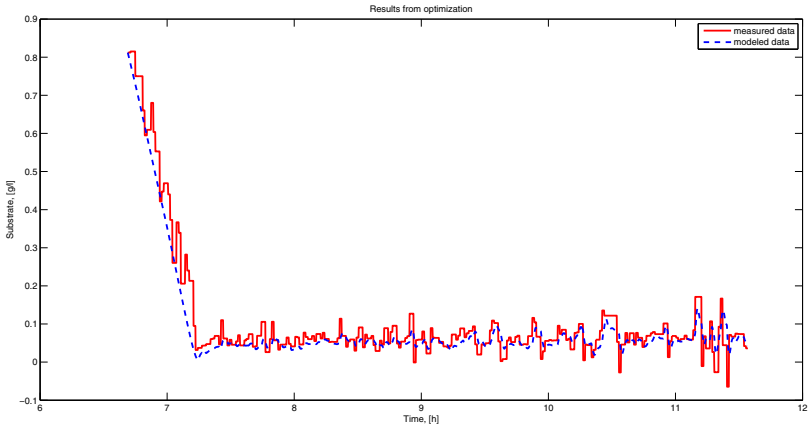


Fig. 5. Time profiles of the substrate: experimental data and models predicted data - best GA result

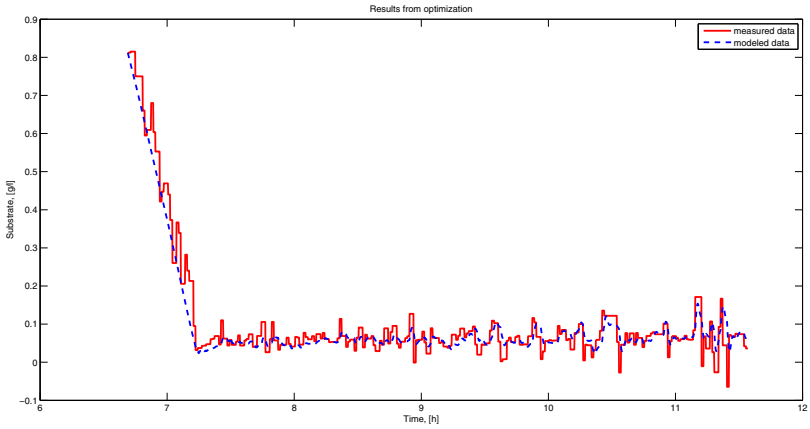


Fig. 6. Time profiles of the substrate: experimental data and models predicted data - best ACO result

On Figures 5 and 6 the modelled substrate is represented by dash line, by solid line is represented the measured substrate. We observe that the modelled data by the ACO algorithm are closer to the measured data than this by the GA algorithm.

On Figure 7 is represented the improvement of the value of the objective function during the execution time. With dash line is represented the improvement of the objective function by GA. With dash-dot line is represented the improvement of the objective function by ACO

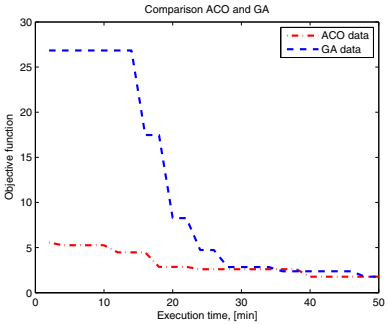


Fig. 7. Improving the objective function during the time

algorithm. The ACO algorithm achieves much better solution at the beginning, because it is constructive method. During the time the achieved values of the objective function by both algorithms become close to each other.

6. Conclusion

In this chapter GA and ACO are applied for parameter identification of a system of nonlinear differential equations modeling the fed-batch cultivation process of the bacteria *E. coli*. A system of ordinary differential equations is proposed to model *E. coli* biomass growth and substrate (glucose) utilization. Parameter optimization is performed using real experimental data set from an *E. coli* MC4110 fed-batch cultivation process. In considered nonlinear mathematical model the parameters that should be estimated are maximum specific growth rate (μ_{max}), saturation constant (k_S) and yield coefficient ($Y_{S/X}$). The parameter estimation is performed based upon the use of a modified Hausdorff metric, in place of most common used metric – Least Squares regression. Parameters of the two algorithms (GA and ACO) were tuned based on several pre-tests according considered here optimization problem. Based on the obtained result it is shown that the best value of the objective function J is achieved by the ACO algorithm. Comparison of the worst obtained results from the two metaheuristics is shown that the GA achieved better estimations than ACO. Analysing of average results it could be concluded that the ACO algorithm performs better for the problem of parameter optimization of an *E. coli* fed-batch cultivation process model.

7. Acknowledgements

This work has been partially supported by the Bulgarian National Scientific Fund under the grants High quality control of biotechnological processes with application of modified conventional and metaheuristics methods DMU 02/4 and TK-Effective Monte Carlo Methods for large-scale scientific problems DTK 02/44.

8. References

Akpinar, S. & Bayhan, G. M. (2011). A Hybrid Genetic Aalgorithm for Mixed Model Assembly Line Balancing Problem with Parallel Workstations and Zoning Constraints. *Engineering Applications of Artificial Intelligence*, Vol. 24, No. 3, pp. 449–457.

Al-Duwaish, H. N. (2000). A Genetic Approach to the Identification of Linear Dynamical Systems with Static Nonlinearities. *International Journal of Systems Science*, Vol. 31, No. 3, pp. 307–313.

Arndt, M. & Hitzmann, B. (2001). Feed Forward/feedback Control of Glucose Concentration during Cultivation of *Escherichia coli*. *8th IFAC Int. Conf. on Comp. Appl. in Biotechn*, Canada, pp. 425–429.

Bastin, G. & Dochain, D. (1991). *On-line Estimation and Adaptive Control of Bioreactors*. Els. Sc. Publ.

Benjamin, K. K.; Ammanuel, A. N.; David, A. & Benjamin, Y. K. (2008). Genetic Algorithm using for a Batch Fermentation Process Identification. *J of Applied Sciences*, Vol. 8, No. 12, pp. 2272–2278.

Bonabeau, E.; Dorigo, M. & Theraulaz, G. (1999). *Swarm Intelligence: From Natural to Artificial Systems*, New York,Oxford University Press.

Brownlee, J. (2011). *Clever Algorithms. Nature-Inspired Programming Recipes*. LuLu, p. 436, 978-1-4467-8506-5.

Carrillo-Ureta, G. E.; Roberts, P. D. & Becerra, V. M. (2001). Genetic Algorithms for Optimal Control of Beer Fermentation. In: *Proc. of the 2001 IEEE International Symposium on Intelligent Control*, Mexico City, Mexico, pp. 391–396.

Chen, Sh. & Lovell, B. C. (2010). Feature space Hausdorff distance for face recognition. In: *Proc. of 20th International Conference on Pattern Recognition (ICPR)*, Istanbul, Turkey, pp. 1465–1468.

Chipperfield, A. J. & Fleming, P. J. (1995). The Matlab Genetic Algorithm Toolbox. *IEE Colloquium Applied Control Techniques Using MATLAB*, pp. 10/1–10/4.

Cockshott, A. R. & Bogle, I. D. L. (1999). Modelling the effects of glucose feeding on a recombinant *E. coli* fermentation. *Bioprocess Engineering*, Vol. 20, pp. 83–90.

Covert, M. W.; Xiao, N.; Chen, T. J. & Karr J. R. (2008). Integrating Metabolic, Transcriptional Regulatory, and Signal Transduction Models in *Escherichia coli*. *Bioinformatics*, Vol. 24, No. 18, pp. 2044–2050.

da Silva, M. F. J.; Perez, J. M. S.; Pulido, J. A. G. & Rodriguez, M. A. V. (2010). AlineaGA - A Genetic Algorithm with Local Search Optimization for Multiple Sequence Alignment. *Appl Intell*, Vol. 32, pp. 164–172.

Dorigo, M. & Di Caro, G. (1999). The Ant Colony Optimization Meta-heuristic. *In: Corne, D, Dorigo, M., Glover, F. (eds).: New Idea in Optimization*, McGrow-Hill, pp. 11–32.

Dorigo, M. & Stutzle, T. (2004). *Ant Colony Optimization*, MIT Press.

Fidanova, S. (2002). Ant Colony Optimization: Additional reinforcement and convergence. *Tech. report IRIDIA-2002-30*, Free university of Bruxelles, Belgium, 12.

Fidanova, S. & Lirkov, I. (2009). 3D Protein Structure Prediction. *J. Analele Universitatii de Vest Timisoara, Seria Matematica-Informatica*, Vol XLVII(2),ISSN 1224-970X, pp. 33–46.

Fidanova, S. (2010). An Improvement of the Grid-based Hydrophobic-hydrophilic Model. *Int. J. Bioautomation*, ISSN 1312-451X, Vol. 14, No. 2, pp. 147–156.

Fidanova, S.; Alba E. & Molina, G. (2010). Hybrid ACO Algorithm for the GPS Surveying Problem, *Large Scale Scientfic Computing*, Springer, Berlin, Vol. 5910, pp. 318–325.

Goldberg, D. E. (2006). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley Longman, London.

Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems*. 2nd Edn. Cambridge, MIT Press.

Houck, Ch. R.; Joines, J. A. & Kay, M. G. (1996). A Genetic Algorithm for Function Optimization: A Matlab Implementation, *Genetic Algorithm Toolbox Toutorial*, available at: http://read.pudn.com/downloads152/ebook/662702/gaotv5.pdf

Jiang, L.; Ouyang, Q. & Tu, Y. (2010). Quantitative Modeling of *Escherichia coli* Chemotactic Motion in Environments Varying in Space and Time. *PLoS Comput Biol*, Vol. 6, No. 4, e1000735. doi:10.1371/journal.pcbi.1000735.

Karelina, T. A.; Ma, H.; Goryanin, I. & Demin, O. V. (2011). EI of the Phosphotransferase System of *Escherichia coli*: Mathematical Modeling Approach to Analysis of Its Kinetic Properties. *Journal of Biophysics*, Vol. 2011, Article ID 579402, doi:10.1155/2011/579402.

Kirkpatrick, S.; Gelatt, C. D. & Vecchi, M. P. (1983). Optimization by Simulated Annealing, *Science, New Series*, Vol. 220, No. 4598, pp. 671–680.

Kumar, S. M.; Giriraj, R.; Jain, N.; Anantharaman, V.; Dharmalingam, K. M. M. & Sheriffa, B. (2008). Genetic algorithm based PID controller tuning for a model bioreactor. *Indian Chemical Engineer*, Vol. 50, No. 3, pp. 214–226.

Lagarias, J. C.; Reeds, J. A.; Wright, M. H. & Wright, P. E. (1998). Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions, *SIAM Journal of Optimization*, Vol. 9, No. 1, pp. 112–147.

Levisauskas, D.; Galvanauskas, V.; Henrich, S.; Wilhelm, K.; Volk, N. & Lubbert, A. (2003). Model-based Optimization of Viral Capsid Protein Production in Fed-batch Culture of recombinant *Escherichia coli*. *Bioprocess and Biosystems Engineering*, Vol. 25, pp. 255–262.

MathWorks Inc. (1999). *Genetic Algorithms Toolbox, User's Guide*.

Michalewicz, Z. (1994). *Genetic Algorithms + Data Structures = Evolution Programs*. Second, Exended Edition, Springer-Verlag, Berlin, Heidelberg.

Nutanong, S.; Jacox, E. H. & Samet, H. (2011) An Incremental Hausdorff Distance Calculation Algorithm. In: *Proc. of the VLDB Endowment*, Vol. 4, No. 8, pp. 506–517.

Obitko, M. (2005). *Genetic Algorithms*, available at http://cs.felk.cvut.cz/∼ xobitko/ga

Opalka, N.; Brown, J.; Lane, W. J.; Twist, K.-A. F.; Landick, R.; Asturias, F. J. & Darst, S. A. (2010). Complete Structural Model of *Escherichia coli* RNA Polymerase from a Hybrid Approach. *PLoS Biol*, Vol. 8, No. 9, e1000483. doi:10.1371/journal.pbio.1000483.

Paplinski, J. P. (2010). The Genetic Algorithm with Simplex Crossover for Identification of Time Delays. *Intelligent Information Systems*, pp. 337–346.

Parker, B. S. (1992). *Demonstration of using Genetic Algorithm Learning*. Information Systems Teaching Laboratory.

Petersen, C. M.; Rifai, H. S.; Villarreal, G. C. & Stein, R. (2011). Modeling *Escherichia coli* and Its Sources in an Urban Bayou with Hydrologic Simulation Program – FORTRAN, *Journal of Environmental Engineering*. Vol. 137, No. 6, pp. 487–503.

Pohlheim, H. (2003). Genetic and Evolutionary Algorithms: Principles, Methods and Algorithms. *Genetic and Evolutionary Toolbox*, http://www.geattb.com/docu/algindex.html.

Press, W. H.; Flannery, B. P.; Teukolsky, S. A. & Vetterling, W. T. (1986). *Numerical Recipes - The Art of Scientific Computing*, Cambridge University Press.

Ranganath, M.; Renganathan, S. & Gokulnath, C. (1999). Identification of Bioprocesses using Genetic Algorithm. *Bioprocess Engineering*, Vol. 21, pp. 123–127.

Roeva, O. & Ts. Slavov (2011). Fed-batch Cultivation Control based on Genetic Algorithm PID Controller Tuning, *Lecture Notes on Computer Science*, Springer-Verlag Berlin Heidelberg, Vol. 6046, pp. 289–296.

Roeva, O. (2006). A Modified Genetic Algorithm for a Parameter Identification of Fermentation Processes. *Biotechnology and Biotechnological Equipment*, Vol. 20, No. 1, pp. 202–209.

Roeva, O. (2007). Multipopulation genetic algorithm: A tool for parameter optimization of cultivation processes models. *Lecture Notes on Computer Science*, Springer-Verlag Berlin Heidelberg, Vol. 4310, pp. 255–262.

Roeva, O. (2008a). Improvement of Genetic Algorithm Performance for Identification of Cultivation Process Models. *Advances Topics on Evolutionary Computing, Book Series: Artificial Intelligence Series-WSEAS*, pp. 34–39.

Roeva, O. (2008b). Parameter Estimation of a Monod-type Model based on Genetic Algorithms and Sensitivity Analysis. *Lecture Notes on Computer Science*, Springer-Verlag Berlin Heidelberg, Vol. 4818, pp. 601–608.

Roeva, O.; Pencheva, T.; Hitzmann, B. & Tzonkov, St. (2004). A Genetic Algorithms Based Approach for Identification of *Escherichia coli* Fed-batch Fermentation. *Int. J. Bioautomation*, Vol. 1, pp. 30–41.

Rote, G. (1991). Computing the minimum Hausdorff distance between two point sets on a line under translation. *Information Processing Letters*, Vol. 38, pp. 123–127.

Schuegerl, K. & Bellgardt, K.-H. (2000). *Bioreaction Engineering: Modeling and Control*. Berlin Heidelberg, Springer-Verlag.

Shmygelska, A.& Hoos, H. H. (2005). An ant colony optimization algorithm for the 2D and 3D hydrophobic polar protein folding problem. *BMC Bioinformatics*, Vol. 6, No. 30, doi:10.1186/1471-2105-6-30.

Skandamis, P. N. & Nychas, G. E. (2000). Development and Evaluation of a Model Predicting the Survival of *Escherichia coli* O157:H7 NCTC 12900 in Homemade Eggplant Salad at Various Temperatures, pHs, and Oregano Essential Oil Concentrations. *Applied and Environmental Microbiology*, Vol. 66, No. 4, pp. 1646–1653.

Sugiyama, M.; Hirowatari, E.; Tsuiki, H. & Yamamoto, A. (2010). Learning figures with the hausdorff metric by fractals. In: *Proc. of the 21st international conference on Algorithmic learning theory*, Springer-Verlag Berlin, Heidelberg, pp. 315–329.

Syam, W. P. & Al-Harkan, I. M. (2010). Comparison of Three Meta Heuristics to Optimize Hybrid Flow Shop Scheduling Problem with Parallel Machines. *World Academy of Science, Engineering and Technology*, Vol. 62, pp. 271–278.

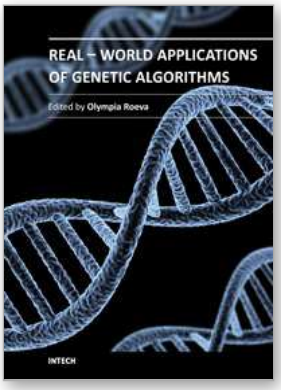
Tahouni, N.; Smith, R. & Panjeshahi, M. H. (2010). Comparison of Stochastic Methods with Respect to Performance and Reliability of Low-temperature Gas Separation Processes. *The Canadian Journal of Chemical Engineering*, Vol. 88, No. 2, pp. 256–267.

Umarani, R. & Selvi, V. (2010). Particle Swarm Optimization: Evolution, Overview and Applications. *Int J of Engineering Science and Technology*, Vol. 2, No. 7, pp. 2802–2806.

Viesturs, U.; Karklina, D. & Ciprovica, I. (2004). *Bioprocess and Bioengineering*. Jeglava.

Yedjour, H.; Meftah, B.; Yedjour, D. & Benyettou, A. (2011). Combining Spiking Neural Network with Hausdorff Distance Matching for Object Tracking. *Asian Journal of Applied Sciences*, Vol. 4, pp. 63–71.

Yusof, M.K. & Stapa, M.A. (2010). Achieving of Tabu Search Algorithm for Scheduling Technique in Grid Computing using GridSim Simulation Tool: Multiple Jobs on Limited Resource. *Int J of Grid and Distributed Computing*, Vol. 3, No. 4, pp. 19–31.



Real-World Applications of Genetic Algorithms
Edited by Dr. Olympia Roeva

ISBN 978-953-51-0146-8
Hard cover, 376 pages
Publisher InTech
Published online 07, March, 2012
Published in print edition March, 2012

The book addresses some of the most recent issues, with the theoretical and methodological aspects, of evolutionary multi-objective optimization problems and the various design challenges using different hybrid intelligent approaches. Multi-objective optimization has been available for about two decades, and its application in real-world problems is continuously increasing. Furthermore, many applications function more effectively using a hybrid systems approach. The book presents hybrid techniques based on Artificial Neural Network, Fuzzy Sets, Automata Theory, other metaheuristic or classical algorithms, etc. The book examines various examples of algorithms in different real-world application domains as graph growing problem, speech synthesis, traveling salesman problem, scheduling problems, antenna design, genes design, modeling of chemical and biochemical processes etc.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821