A MODIFIED GENETIC ALGORITHM FOR A PARAMETER IDENTIFICATION OF FERMENTATION PROCESSES

O. Roeva
Centre of Biomedical Engineering ”Prof. Ivan Daskalov”, BAS, Sofia, Bulgaria

ABSTRACT

A modified genetic algorithm is proposed for a parameter identification of an E. coli fed-batch fermentation model. In the simple genetic algorithm the best chromosome does not always keep on improving in each generation. Obtained good solutions could be destroyed by either crossover or mutation or both operations. The aim of the modified genetic algorithm is to prevent this disadvantage. In view of the fact that there is no general theory about tuning the genetic algorithm parameters, some adjustments of genetic parameters, according to the regarded problem, are done to improve the conventional genetic algorithm. The simulation results illustrate that the use of proposed modified genetic algorithm for a parameter identification of fermentation processes is highly efficient and effective. The functions and parameter adjustments, proposed here, enhance the algorithm performance. Still more, the implementation of the modified genetic algorithm leads to noticeable decreasing of the solution time.

Introduction

Genetic algorithms (GA) are global, parallel, stochastic search method. The algorithms are founded on Darwinian evolutionary principles – mechanics of natural selection and natural genetics (12, 16). Genetic algorithms are proved to be very suitable for the optimization of highly nonlinear problems and can find the global optimal solution in complex multidimensional search spaces. Recently, genetic algorithms have been used extensively in solving many optimization-searching problems including mathematical function optimization, very large scale integration chip layout, molecular docking, parameter fitting, scheduling, manufacturing, clustering, machine learning, etc. Genetic algorithms have been successfully applied in a variety of areas and are still finding increasing acceptance (4, 7, 10, 17). In this paper genetic algorithms are applied in the area of biotechnology, especially for a parameter estimation of a fermentation process model.

Fermentation processes are known to be complex and their modeling is a complicated and rather time consuming task. 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. The model must be regarded as a step to reach more easily the final aim – an optimization of a real biotechnological process. The model has to describe those aspects of the process that significantly affect the process performance.

Compared with conventional optimization methods, GA simultaneously evaluates many points in the parameter space. It is more probable to converge towards the global solution. A genetic algorithm does not assume that the space is differentiable or continuous. 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 does not require such problem-specific information, they are more flexible than most search methods. The genetic algorithms can solve hard problems, are noise tolerant, easy to interface to existing simulation models, and easy to hybridize. Therefore GA are suitable and more workable for a parameter identification of fermentation process models.

Many variations of the standard genetic algorithm, as presented by Goldberg (12), can be found in the literature. All these modifications and hybridizations have been motivated by a desire to improve the performance of the GA, and to adapt it to particular problem domains (8, 14).

As mentioned in (13) in simple genetic algorithm good solutions in old population may be destroyed by either crossover or mutation or both operations. It is known that in the simple genetic algorithm the best chromosome does not always keep on improving in each generation. The aim of this paper is to propose a modified genetic algorithm (MGA) that prevents this disadvantage. A simple modification of the standard genetic algorithm, namely the reproduction to be processed after performing of both the crossover and mutation, is here presented. In this way the destroying of reached good solution by either crossover or mutation or both operations could be prevented. In this report some adjustments of MGA operators and parameters are described. The proposed modified algorithm is applied for a parameter identification of a fed-batch fermentation model. The experimental data of *E. coli* fed-batch fermentation (1, 11) are used to illustrate a number of features of the modified genetic algorithm presented here. The experiment is carried out in the *Institute of Technical Chemistry, University of Hannover* during the teamwork of the *DFG project №113-9/2000.*

Materials and Methods

Process Description

The strain used for the fermentation process is *Escherichia coli MC4110.* The fermentation is performed in a 2l bioreactor, using a mineral medium (1). 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 ºC. The aeration rate is kept at 275 l/h air, stirrer speed starts at 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 measurements are performed using Yellow Springs Analyzer. For on-line glucose measurement a flow injection analysis system has been employed using two pumps for a continuous sample and carrier flow rate. The feeding rate in the considered process is presented in **Fig. 1**.

In *Escherichia coli* fermentation the rates of cell growth, glucose consumption, acetate formation and dissolved oxygen consumption are commonly described as follows according to the mass balance (3, 9):

$$\frac{dX}{dt} = \mu_{\max}\frac{S}{k_s + S}X - \frac{F}{V}X \qquad (1)$$

$$\frac{dS}{dt} = -\frac{1}{Y_{S/X}}\mu_{\max}\frac{S}{k_s + S}X + \frac{F}{V}(S_{in} - S) \qquad (2)$$

$$\frac{dA}{dt} = \frac{1}{Y_{A/X}}\mu_{\max}\frac{S}{k_s + S}X - \frac{F}{V}A \qquad (3)$$

$$\frac{dDO}{dt} = -\frac{1}{Y_{DO/X}}\mu_{\max}\frac{S}{k_s + S}X + k_L a(DO^* - DO) - \frac{F}{V}DO \qquad (4)$$

$$\frac{dV}{dt} = F, \qquad (5)$$

where: X is the concentration of biomass, [g/l]; S – concentration of substrate (glu-

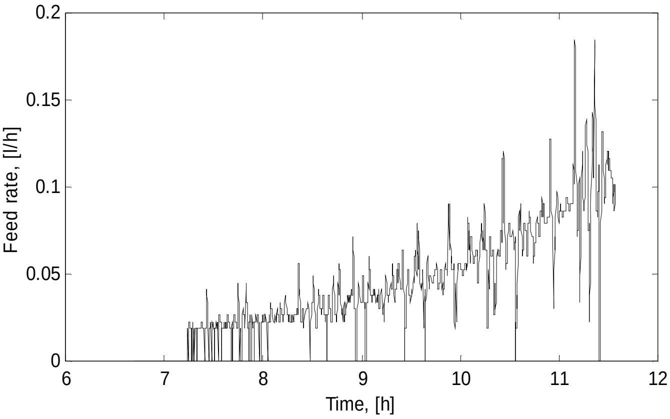

Fig. 1. Feeding rate as a function of time in a fed-batch fermentation of *Escherichia coli MC4110.*

cose), [g/l]; A – concentration of acetate, [g/l]; DO – concentration of dissolved oxygen, [%]; F – feeding rate, [l/h]; V – bioreactor volume, [l]; S_{in} – concentration of substrate in the feeding solution, [g/l]; μ_{max} – maximum growth rate, [h^{-1}]; k_S – saturation constant, [g/l]; $Y_{S/X}, Y_{A/X}, Y_{DO/X}$ – yield coefficients, [gg^{-1}]; $k_L a$ – volumetric oxygen transfer coefficient, [h^{-1}].

Modified Genetic Algorithm

The reproduction in simple genetic algorithm (SGA) is considered for determining which chromosomes will be chosen as the basis of the next generation. Generating populations from only two parents may cause loss of the best chromosome from the last population. Reached good solution may be destroyed by either crossover or mutation or both operations. Thereby, the best solution in SGA popped up from the new population may be inferior to the old generations. The aim of the modified genetic algorithm is to prevent this demerit. MGA possesses a structure similar to SGA. However, the MGA has been distinguished from the SGA in that the reproduction is processed after both the crossover and mutation have been performed. Thus the deterioration problem never happens since the best solution from the current generation will be superior to or at least the same with the past.

In the beginning the modified genetic algorithm creates an initial population. In the next step the algorithm evaluate the objective values (cost values) of the individuals in the current population. After that individuals are reproduced. During the reproduction, recombination (or crossover) first occurs. Genes from parents combine to form a whole new chromosome. The newly created offspring then mutates. Mutation means that the elements of chromosome are a bit changed. These changes are mainly caused by errors in copying genes from parents. Then MGA ranked individuals represented by their associated cost, to be “minimized”, and returns the corresponding individual fitnesses. Next the most fitted individuals from offspring are selected. Here the objective values of the individuals in the offspring are evaluated and re-insertion of offspring in population replacing parents is done. The MGA is terminated when some criteria are satisfied, e.g. a certain number of generations, a mean deviation in the population, or when a particular point in the search space is encountered.

The outline of the modified genetic algorithm can be presented as follows:

Step 1. [*Start*] Generate a random population of n chromosomes (suitable solutions for the problem).

Step 2. [*Population Objective function*] Evaluate the objective function of the generated population.

Step 3. [*Crossover*] With a crossover probability cross over the parents to form a new offspring (children).

Step 4. [*Mutation*] With a mutation probability produces new offspring.

Step 5. [*Fitness*] Evaluate the fitness of each chromosome in the new offspring.

Step 6. [*Selection*] Select chromosomes from the new offspring according to their fitness (better fitness, bigger chance to be selected).

Step 7. [*Offspring Objective function*] Evaluate the objective function of the new offspring.

Step 8. [*Insertion*] Insert the new offspring in the new population and use new generated population for a further run of the algorithm.

Step 9. [*Test*] If the end conditions are not satisfied, go to Step 2. Otherwise – stop and return the best solution in current population.

Results and Discussion

Choice and Adjustment of Genetic Algorithm Operators and Parameters

In the genetic algorithms, there are many operators, functions, parameters and settings that can be implemented differently in various problems. The inappropriate choice of operators and parameters in the evolutionary process makes GA susceptible to premature convergence. Primary choice of genetic operators and parameters depends on the chosen encoding, as well as on the problem.

The first decision to be taken is how to create chromosomes and what type of encoding to be choosen. The encoding depends mainly on the solved problem. A chromosome should in some way contain information about solution that it represents. Examples of chromosome representation are binary, integer and floating-point. Binary representation is the most common one, mainly because of its relative simplicity. Due to these facts a binary 20 bit representation is here considered.

The next question in the simple genetic algorithm is how to select parents for crossover. This can be done in many ways, but the main purpose is to select the best parents (best survivors) in the hope of that the better parents will produce better offspring. In some cases generating populations from only two parents may cause the best chromosome from the last population to be lost. Therefore in the presented here MGA the reproduction is processed after performing of both crossover and mutation.

Crossover operates on generated chromosomes and creates a new offspring. The simplest way is to choose randomly some crossover point and copy everything before this point from the first parent and then copy everything after the crossover point from the other parent. There are other ways how to make crossover, for example to be choosen more crossover points – double point crossover, multi-point crossover. Crossover can be quite complicated and depends (as well as the technique of mutation) mainly on the encoding of chromosomes. Specific crossover made for a specific problem can improve performance of the genetic algorithm. A double point crossover is used here. After a crossover is performed, mutation takes place. The mutation is intended to prevent falling of all solutions in the population into a local optimum of the solved problem. The mutation operation changes randomly the offspring resulted from crossover. In accepted encoding here a bit inversion mutation is used.

As it has been already shown in the MGA outline, chromosomes has to be selected from the population to become parents for crossover. The problem is how to select these chromosomes. According to Darwinian theory of evolution the best ones survive to create a new offspring. There are many methods for selection of

TABLE 1

Genetic operators and parameters in modified genetic algorithm

Operator	Type	Parameter	Value
encoding	binary	generation gap	0.97
crossover	double point	crossover probability	0.70
mutation	bit inversion	mutation probability	0.05
selection	RWS	precision	20
fitness	linear	number of individuals	100
function	ranking	number of generation	100

the best chromosomes. Examples are roulette wheel selection (RWS), Boltzman selection, tournament selection, rank selection, steady state selection, selection with stochastic universal sampling, elitism and etc. (5, 6, 15). The roulette wheel selection, which is the most popular selection method, is here used. Chromosomes are selected according to their fitness. The chromosomes with bigger fitness value will be selected more times.

There are some genetic parameters that should be determined depending on the regarded problem, i.e. generation gap, crossover and mutation probability, numbers of individuals and generations. Higher value of generation gap does not improve the performance of GA, especially concerning how fast the solution will be found (18). Crossover rate should be generally high, about 65%-95%. Mutation is randomly applied with low probability, typically in the range 0.01-0.1 (18). Particularly important parameters of GA are the population size and the number of generations. 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.

Several runs of MGA with different values of the mentioned above five genetic parameters have been performed. The adjustments of genetic algorithm parameters are done with a criterion of finding of the best solution in the shortest time, based on the simulated data. The initial values of the genetic parameters are as presented in (4). First tests are done with a large value of numbers of individuals and generations, exactly 1500 individuals and 500 generations. The researches show that above some limit (which depends mainly on the encoding and the problem) it is not useful to use very large populations. It does not solve the problem faster than moderate sized populations. After different runs 100 number of individuals and 100 numbers of generations are assumed. The experiments confirmed that the increase of number of individuals and generations do not lead to improvement of the decision. Large number of individuals and generations slow down the algorithm. The tests show that in considered problem a generation gap value of 0.97 is suitable. The best results are received with crossover rate of 70% and mutation probability equal to 0.05. The genetic algorithm operators and parameters for the parameter identification of the considered fermentation process are summarized in **Table 1**.

An objective function should be defined after determination of genetic algorithm operators and parameters. Let's assume that there are *n* training patterns. Each has a desired output y_i and predicted output y_i', 1 ≤ *i* ≤ *n*. The desired outputs correspond to experimental data for cell growth, glucose consumption, acetate formation and dissolved oxygen consumption. Then the objective function *f* can be defined as the mean square deviation between the model output y_i' and the corresponding data y_i obtained during the fermentation:

$$f(s_j) = \sum_{i=1}^{n} (y_i - y_i')^2, \tag{6}$$

where s_j represents the j$^{\text{th}}$ chromosome.

The objective function is used to provide a measure of how individuals have performed in the problem domain. In this case, of a minimization problem, the most fitted

TABLE 2

Estimated values of model parameters

Parameter	μ_{max}	k_s	$Y_{S/X}$	$Y_{A/X}$	$Y_{DO/X}$	$k_L a$
Estimated value	0.49 h^{-1}	0.02 g/l	0.50 gg^{-1}	0.014 gg^{-1}	0.021gg^{-1}	151.49 h^{-1}

individuals should have the lowest numerical value of the associated objective function. This raw of measure of fitness is usually used only as an intermediate stage in determining the relative performance of individuals in a genetic algorithm. The fitness function is normally used to transform the objective function value into a measure of relative fitness. In this work a linear ranking of Backer (2) is used.

Parameter Identification Using Modified Genetic Algorithm

The parameter identification of *E. coli* fermentation model, described with Eq. (1) – Eq. (5), is fulfilled in *Matlab* 5.3 environment. The proposed modification – to process the reproduction after performing of both crossover and mutation, is implemented on the genetic algorithm available in the *Genetic Algorithm Toolbox* (15). Based on the accepted genetic functions and parameters (**Table 1**) and the optimization criterion (Eq. (6)) a parameter identification is carried out. The results show that the chosen genetic parameters are appropriate for considered here problem. The MGA find the solution around the 80-90-th iteration. The population size of 100 individuals is proved to be satisfactory.

The estimated values of model parameters are listed in **Table 2**. The specific growth rate μ is generally found to be a function of three parameters: the concentration of limiting substrate *S*, the maximum growth rate μ_{max} and the substrate-specific constant k_S. The value for k_S is generally very low. Maximum specific growth rates are of considerable industrial importance. The specific growth rates typical of *E. coli* vary between 0.09-0.61 h^{-1} (9, 19). According to Monod kinetics, residual substrate should decrease as dilution rate decreases resulting in an increase in the cell concentration. Over most of the range of μ which will operate in fed-batch culture, S_{in} will be much higher than k_S, so that, for all practical purpose, the change in residual substrate concentration would be extremely small and may be considered as zero (19). The estimated values of model parameters are closed to the values presented in (1, 11). The proposed modified genetic algorithm achieves a value of the optimization criterion of 7.312, which is better than result achieved by the standard genetic algorithm (7.936).

Both the experimental fermentation trajectories and the simulated from the model ones for the fed-batch culture of *E. coli* are presented in **Fig. 2**.

The simulation results on **Fig. 2** show that the developed model is adequate and predicts successfully the variation of process variables during the fed-batch fermentation of *E. coli*. The results are obtained in shorter time – for 73.4220 s, in comparison with a simple genetic algorithm – 282.063 s.

Conclusions

The modified genetic algorithm is proposed for a parameter identification of a fermentation process. The experimental data of *E. coli* fed-batch fermentation are used to illustrate the advantages of the presented genetic algorithm. The aim of the modified genetic algorithm is to prevent some disadvantages of simple genetic algorithms. The reproduction in SGA is considered for determining which chromosomes will be chosen as the basis of the next generation. Although elite preserving strategy could be used to guarantee the survival of the most fitted chromosome from population into the

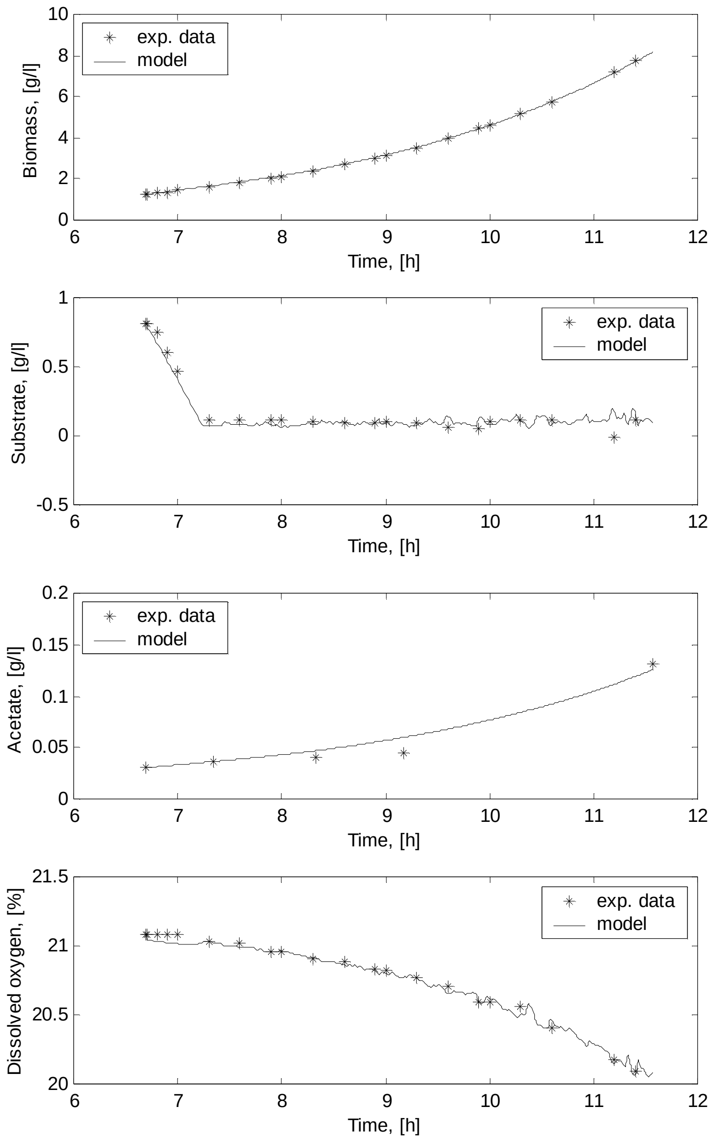

Fig. 2. Measured and simulated model data for process variables: biomass, substrate, acetate and dissolved oxygen.

matting pool, it is possible to become deterioration of the result. In MGA, the problem never happens since the best solution from the current generation will be superior to or at least the same with the past.

The implementation of modified genetic algorithm for parameter identification of fermentation processes is highly efficient and effective. The proposed modification and adjustments of modified genetic algorithm parameters, i.e. generation gap, crossover and mutation probability, numbers of individuals and generations, improve the optimization capability and the decision time of the algorithm.

Acknowledgements

This work is partially supported from National Science Fund Project № MI – 1505/2005

REFERENCES

1. **Arndt M., Hitzmann B.** (2001) In: Proceedings of the 8th IFAC Int. Conf. on Comp. Appl. in Biotechn., Quebec City, Canada, 425-429.

2. **Backer J.E.** (1985) In: Proc. ICGA 1, 101-111.

3. **Bastian A.** (2000) Fuzzy Sets and Systems, **113**, 333-350.

4. **Carrillo-Ureta G.E., Roberts P.D., Becerra V.M.** (2001) In: Proceedings of the 2001 IEEE International Symposium on Intelligent Control, Mexico City, Mexico, 391-396.

5. **Chipperfield A.J., Fleming P.J.** (1995) The Matlab Genetic Algorithm Toolbox.

6. **Chipperfield A.J., Fleming P.J., Pohlheim H., Fonseca C.M.** (1994). Genetic Algorithm Toolbox for Use with MATLAB. User's Guide. Version 1.2. Dept. of Automatic Control and System Engineering, University of Sheffield, U.K.

7. **Cockshott A.R., Bogle I.D.L.** (1999) Bioprocess Engineering, **20**, 83-90.

8. **Cordon O., Herrera F.** (2001) Fuzzy Sets and Systems, **118**, 235-255.

9. **Crueger W., Grueger D.** (1984) Biotechnology. A Textbook of Industrial Microbiology, Sci. Tech. Inc.

10. **Fleming P.J., Purshouse R.C.** (2001) In: Research Report No. 789., available at http://www.lania.mx/~ccoello/EMOO/ EMOOtechreports.html

11. **Georgieva O., Arndt M., Hitzmann B.** (2001) International Symposium "Bioprocess Systems'2001 -BioPS'2001", 1-3.X, Sofia, Bulgaria, I.61-I.64.

12. **Goldberg D.** (1989) Genetic algorithms in search, optimization and machine learning. Addison-Wesly Publishing Company, Massachusetts.

13. **Huang Y.-P., Wang S.-F.** (2000) Fuzzy Sets and Systems, **113**, 367-379.

14. **Kuo R.J., Chen C.H., Hwang Y.C.** (2001) Fuzzy Sets and Systems, **118**, 21-45.

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

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

17. **Na J.-G., Chang Y.K., Chung B.H., Lim H.C.** (2002) Bioprocess and Biosystems Engineering, **24**, 299-308.

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

19. **Standbary P.F., Whitaker A.** (1984) Principles of Fermentation Technology. Pergamon Press.