Evolutionary Algorithms for Optimal Control in Fed-batch Fermentation Processes

Miguel Rocha[1], José Neves[1], Isabel Rocha[2], and Eugénio C. Ferreira[2]

[1] Departamento de Informática - Universidade do Minho
Campus de Gualtar, 4710-057 Braga - PORTUGAL
mrocha@di.uminho.pt jneves@di.uminho.pt
[2] Centro de Engenharia Biológica - Universidade do Minho
Campus de Gualtar, 4710-057 Braga - PORTUGAL
irocha@deb.uminho.pt ecferreira@deb.uminho.pt

Abstract. In this work, *Evolutionary Algorithms (EAs)* are used to achieve optimal feedforward control in a recombinant bacterial fed-batch fermentation process, that aims at producing a bio-pharmaceutical product. Three different aspects are the target of the optimization procedure: the feeding trajectory (the amount of substrate introduced in a bioreactor per time unit), the duration of the fermentation and the initial conditions of the process. A novel *EA* with variable size chromosomes and using real-valued representations is proposed that is capable of simultaneously optimizing the aforementioned aspects. Outstanding productivity levels were achieved and the results are validated by practice.

Keywords: Bio-engineering processes, Fed-batch fermentation optimization, Variable size chromosomes, Real-valued representations

1 Introduction

A number of valuable products such as recombinant proteins, antibiotics and amino-acids are produced using fermentation techniques and thus there is an enormous economic incentive to optimize such processes. However, these are typically very complex, involving different transport phenomena, microbial components and biochemical reactions. Furthermore, the nonlinear behavior and time-varying properties make bioreactors difficult to control with traditional techniques. Under this context, there is the need to consider quantitative mathematical models, capable of describing the process dynamics and the interrelation among relevant variables. Aditionally, robust optimization techniques must deal with the model's complexity, the environment constraints and the inherent noise of the experimental process.

Several optimization methods have been applied in this task. It has been shown that, for simple bioreactor systems, the problem can be solved analytically, from the Hamiltonian function, by applying the minimum principle of Pontryagin. However, besides having a problem of singular control, those methodologies become too complex when the number of state variables increases [14].

G.R. Raidl et al. (Eds.): EvoWorkshop 2004, LNCS 3005, pp. 84-93, 2004.
© Springer-Verlag Berlin Heidelber 2004

Numerical methods make a distinct approach to dynamic optimization. The gradient algorithms are based on the local sensitivities of the objective function for changes in the values of the control variables, which are used to adjust the control trajectories in order to iteratively improve the objective function [3]. In contrast, dynamic programming methods discretize both time and control variables to a predefined number of values. A systematic backward search method in combination with the simulation of the system model equations is used to find the optimal path through the defined grid. However, in order to achieve a global minimum the computational burden is very high [13].

An alternative approach comes from the use of *Evolutionary Algorithms (EAs)*, which have been used in the past to optimize nonlinear problems with a large number of variables. Previous work has obtained interesting results in the optimization of feeding or temperature trajectories [8][1]. Comparisons with traditional methods have been addressed with favorable results [12][4]. Recent work [6] focused on the feed optimization in the fed-batch culture of insect cells, but uses *EAs* based on binary representations and only optimizes the maximum densities of a pre-defined set of nutrients.

In this work, a fed-batch recombinant *E. coli* fermentation process was studied, aimed at producing a bio-pharmaceutical product [10]. In fed-batch fermentations there is an addition of certain nutrients along the process, in order to prevent the accumulation of toxic products, allowing the achievement of higher product concentrations. The purpose is to apply real-valued representation based *EAs*, with novel features such as variable sized chromosomes, in order to optimize not only some of the fermentation's variables, namely the substrate feeding trajectory and the initial conditions, but also the duration of the fermentation.

2 The fed-batch fermentation process

Since the advent of the recombinant DNA technology, the bacterium *E. coli* is the microorganism of choice for the production of the majority of the valuable biopharmaceuticals, usually grown under fed-batch mode due to the effect of acetic acid, which is produced when glucose is present above certain concentrations.

During this process the system states change considerably, from a low initial to a very high biomass and product concentration. This dynamic behavior motivates the development of optimization methods to find the optimal input feeding trajectories in order to improve the process. The typical input in this process is the substrate inflow rate as a function of time. One way to evaluate the process performance is the productivity, defined as the units of product (recombinant protein) formed per unit of time. In this case, it is usually related with the final biomass obtained, when the duration of the process is pre-defined.

For the proper optimization of the process, a white box mathematical model was developed, based on differential equations that represent the mass balances of the relevant state variables. The general dynamical model [2] is accepted as representing the dynamics of an n components and m reactions bioprocess.

During the aerobic growth of the bacterium, with glucose as the only added substrate, the microorganism can follow three different metabolic pathways:

– Oxidative growth on glucose:

$$k_1 S + k_5 O \xrightarrow{\mu_1} X + k_8 C \tag{1}$$

– Fermentative growth on glucose:

$$k_2 S + k_6 O \xrightarrow{\mu_2} X + k_9 C + k_3 A \tag{2}$$

– Oxidative growth on acetic acid:

$$k_4 A + k_7 O \xrightarrow{\mu_3} X + k_{10} C \tag{3}$$

where S, O, X, C, A represent glucose, dissolved oxygen, biomass, dissolved carbon dioxide and acetate components, respectively. In the sequel, the same symbols are used to represent the state variables concentrations (in g/kg); μ_1 to μ_3 are time variant specific growth rates that nonlinearly depend on the state variables, and k_i are constant yield coefficients.

The associated dynamical model can be described by the following equations:

$$\frac{dX}{dt} = (\mu_1 + \mu_2 + \mu_3)X - DX \tag{4}$$

$$\frac{dS}{dt} = (-k_1\mu_1 - k_2\mu_2)X + \frac{F_{in,S}S_{in}}{W} - DS \tag{5}$$

$$\frac{dA}{dt} = (k_3\mu_2 - k_4\mu_3)X - DA \tag{6}$$

$$\frac{dO}{dt} = (-k_5\mu_1 - k_6\mu_2 - k_7\mu_3)X + OTR - DO \tag{7}$$

$$\frac{dC}{dt} = (k_8\mu_1 + k_9\mu_2 + k_{10}\mu_3)X - CTR - DC \tag{8}$$

$$\frac{dW}{dt} \simeq F_{in,S} \tag{9}$$

being D the dilution rate, $F_{in,S}$ the substrate feeding rate (in kg/h), W the fermentation weight (in kg), OTR the oxygen transfer rate and CTR the carbon dioxide transfer rate.

Real experiments served as the basis for the model derivation, being conducted in a fermentation laboratory with a 5-L bioreactor, being the experimental results consistent with the model [9]. This model was thus used for the optimization of several relevant features of the process. For optimization purposes, the model simulation was performed, by using the Euler numerical integration method, with a small step size d and a given duration for the process (T_f) measured in hours. Typical values of T_f and d were 25 and 0.005. The kinetic behavior, expressed in the rates μ_1 to μ_3, was given by a specific algorithm based on the state variables, that is out of the scope of the present work but can be found in [9].

3 Evolutionary Algorithms for Feeding Trajectory Optimization

The first approach to the problem was to develop an *EA* capable of optimizing the feeding trajectory, i.e., to determine the amount of substrate (glucose) to be fed into the bioreactor per time unit ($F_{in,S}$). Real-valued representations were used in order to encode the feeding amounts, since these have proven to be more appropriate than the classical binary ones, in tasks where the purpose is to optimize real valued parameters [5][7].

Thus, each gene will encode the amount of substrate to be introduced into the bioreactor in a given time unit and the genome will be given by the temporal sequence of such values. In this case, the size of the genome is determined based on the final time of the process (T_f) and the discretization step (d) considered in the simulation, being given by the expression: $\frac{T_f}{d}$.

However, this could produce a very large genome (a typical value would be 5000), which would difficult the *EA*'s convergence. Thus, feeding values are defined only at certain equally spaced points, and the remaining values are linearly interpolated. The size of the genome becomes $\frac{T_f}{dp} + 1$, where p stands for the number of points within each interpolation interval. The values used in the experiments described in this section are: $T_f = 25$, $d = 0.005$ and $p = 200$.

There are physical constraints on the amount of substrate that can be introduced per time unit, due to limitations in the feeding pump capacity. Thus, there is the need to impose limits in the gene values, in this particular case defined in the range $[0.0; 0.4]$. In the initial population, each individual is assigned, for each of its genes, a random value in the appropriate range.

The evaluation process, for each individual in the population, measures the quality of the feeding trajectory in terms of the fermentation's productivity. This calculation is achieved by firstly running a simulation of the defined model, given as input the feeding values in the genome. In each simulation, the relevant state variables are initialized with the following values: $X_0 = 5$, $S_0 = 0$, $A_0 = 0$, $W_0 = 3$. The fitness value is then calculated from the final and initial values of the state variables X and W, by the expression $X_f * W_f - X_0 * W_0$ (measured in g).

Regarding the recombination step, both mutation and crossover operators were taken into account. Two mutation operators were used, namely:

- *Random Mutation*, which replaces one gene by a new randomly generated value, within the range $[min_i, max_i]$ [7]; and
- *Gaussian Mutation*, which adds to a given gene a value taken from a gaussian distribution, with a zero mean and a standard deviation given by $\frac{max_i - min_i}{4}$ (i.e., small perturbations will be preferred over larger ones).

where $[min_i; max_i]$ is the range of values allowed for gene i. In both cases, an innovation is introduced: the mutation operators are applied to a variable number of genes (a value that is randomly set between 1 and 10 in each application). This operator obtained interesting results in the training of *Artificial Neural Networks* [11], and its good performance in this context was verified empirically

On the other hand, the following crossover operators were chosen:

- *Two-Point* and *Uniform*, two standard *Genetic Algorithm* operators [7], applied in the traditional way;
- *Arithmetical*, each gene in the offspring will be a linear combination of the values in the ancestors' chromosomes [7];
- *Sum*, inspired in *Differential Evolution* [4], where the offspring genes denote the sum/ subtraction of the ones in the parents.

All operators were constrained to respect the limits of the gene's values, so when a value would be out of range it was replaced by the nearest boundary.

In terms of the *EAs* setup, the population size was set to 200. The selection procedure is done by converting the fitness value into a linear ranking in the population, and then applying a roulette wheel scheme. In each generation, 50% of the individuals are kept from the previous generation, and 50% are bred by the application of the genetic operators.

The implementation of the proposed *EA* was based on a general purpose package, developed by the authors in the *Java* programming language. All experiments reported were run on a *PC* with a *Pentium IV 2.4 GHz* processor.

A first set of experiments was conducted in order to find the best set of genetic operators to tackle this problem. All possible combinations of a crossover and a mutation operator were tested, as well as alternatives with one single mutation operator. It must be noticed that, in the proposed *EA*, genetic operators are selected whenever a new individual is created, based on a given probability. In this case, both crossover and mutation operators were considered to have equal oportunities of breeding offspring. Each run of the *EA* is stopped after 500 iterations and the results are given in terms of the mean of thirty runs, with the associated 95% confidence intervals. The results are given in Table 1, where the most promising method seems to be the combination of the arithmetical crossover with a random mutation, although there are alternatives with overlapping confidence intervals.

Table 1. Comparison of *EAs* with different genetic operators on the problem of feeding trajectory optimization.

Crossover	**Gaussian Mutation**	**Random Mutation**
None	111.6 ± 2.6	128.4 ± 6.1
Two-Point	184.4 ± 4.4	185.3 ± 2.3
Uniform	182.3 ± 2.3	198.3 ± 3.6
Arithmetical	191.3 ± 4.3	200.6 ± 3.1
Sum	103.7 ± 3.2	100.2 ± 3.3

An alternative that contemplates the use of all genetic operators described above was also attempted. In this case each crossover operator is responsible for breeding 12.5% of the offspring and each mutation operator 25%. Testing

this alternative a result of 210.4 ± 1.9 was obtained, which is better than all the previous attempts, being the difference statiscally significant. To further evaluate this alternative, the number of generations was increased to 3000, being obtained a new result of 224.4 ± 1.4.

These results are similar to those obtained by two different approaches. The first uses a gradient based algorithm, implemented by the MATLAB optimization toolbox function *fmincon* (version 2.1); the latter is based on the genetic algorithm toolbox for MATLAB (version 1.7), developed by Polheim at University of Sheffield. However, the computational time taken to achieve results in both cases is clearly superior (between 10 to 100 times) than the ones of the proposed Java-based *EAs*. As a consequence, more experiments were made possible and statistical significance could be reached. Since that is not the case with the other approaches the results are not given here. Furthermore, it was possible to develop new algorithms to tackle other kinds of , such as the ones described in the following.

The best of the thirty runs will be analyzed in detail, to get an insight of the *EA*'s behavior. The fitness value was 228.0, obtained with the values of $X_f = 50.1$ and $W_f = 4.85$. It is known that a value of $W = 5$ is a physical limit (maximum weight allowed by the fermentation vessel), so the optimal result should set the value of W_f to 5. The value obtained is near, but is still sub-optimal, which can be due to an optimization insufficiency, but can also mean that the initial conditions and/or the duration of the process are not correctly set.

4 Optimization of Initial Conditions

The initial conditions of the experiments were set based on the practicioner's experience and wisdom. However, there is no guarantee that the initial values of the state variables are optimal. So, it was decided to incorporate the initial values of significant state variables in the optimization procedure.

Once each variable has different physical constraints it was necessary to define a genome where the limits are distinct in each position. The variables chosen to be optimized, aditionally to the feeding trajectory, were the initial values of X, W, S and A, with their range of variability given by $X_0 \in [1; 5]$, $W_0 \in [2; 4]$, $S_0 \in [0; 5]$ and $A_0 \in [0; 5]$. In the experiment performed to evaluate this approach, the fitness function, the genetic operators and the *EA*'s setup was kept from the previous section and each run of the *EA* was stopped after 3000 generations.

The mean of the results obtained was 234.4 (± 1.4), which means a considerable improvement over the previous results. As before the best run will be examined in detail to illustrate the *EAs* behavior, being the fitness obtained 236.5, with $X_f = 50.3$ and $W_f = 5.0$. As explained before this value for W is highly desirable and an important condition to reach the optimality of the process. The initial values of the state variables, in the best run, were $X_i = 5.0$, $S_i = 0.45$, $A_i = 0.01$ and $W_i = 3.15$. It is believed that the slight difference of values in the W initial's value is determinant to the result obtained, since it is almost unchanged in the other runs.

5 Optimization of the Final Time

The duration of the fermentation is not imposed by any theoretical result, yet is chosen by empirical knowledge, making it possible to optimize its value. In this section, an *EA* will be proposedfor this task, based on the ones defined above, but considering variable size chromosomes and new genetic operators.

The genetic operators defined in Section 3 were kept: the mutations were unchanged and the crossovers were updated to cope with parents of different sizes. In this case, each of the offspring keeps the size of one parent and for the genes where only one parent is defined (the one with greater length), their value is passed into the corresponding offspring. In the creation of the initial population the individuals are given chromosomes with distinct sizes, randomly selected in a range defined by two parameters: a minimum and a maximum size. Furthermore, two novel mutation operators were defined, in order to allow for the change of the size of individuals during the evolution process:

- *Grow*: consists in the introduction of a new gene into the genome, in a random position, being its value the average of the values of the two neighboring genes.
- *Shrink*: a randomly selected gene is removed from the genome.

Both operators are only applied when the maximum and minimum size constraints are obbeyed. With the introduction of the new genetic operators, the probabilities used in the experiments are the following: each of the four crossover operators has a probability value of 10%, the random and gaussian mutation keep their probabilities of 25% and the new mutation operators have a probability of 5% each.

Two different experiments were conducted: in the first, only the final time and feeding trajectory are optimized, being the genome made out of the feeding trajectory; in the latter, the initial conditions are also considered a target of optimization, being the initial parameters encoded into the first group of four genes (fixed size), as before, and the remaining of the genome used to code the feeding trajectory (variable size).

The fitness function also has to suffer a slight modification. Indeed, once the time is variable it makes sense to evaluate the productivity of the process per time unit. The fitness of an individual is thus given by the following expression:

$$\frac{X_f * W_f - X_0 * W_0}{T_f}$$

The minimum and maximum duration of the fermentation are set to 20 and 50 hours, respectively. The remaining parameters of the *EA* are kept unchanged. The results obtained are displayed in Table 2 and compared with the previous ones. In the table, the first column indicates the purpose of the *EA*, where F stands for feeding trajectory, I for initial conditions and T for final time optimization. The results are given in terms of the newly defined fitness, in order to make a comparison possible, being shown, in the second column, the

mean of the thirty runs and the confidence interval and in the third column the best result obtained over all the runs.

Table 2. Comparison of the results obtained by the *EAs* for feeding trajectory, initial conditions and duration optimization.

Optimization aim	Mean and conf.interval	Best result
F	8.98 ± 0.06	9.12
F+I	9.38 ± 0.06	9.46
F+T	9.16 ± 0.09	9.32
F+T+I	9.44 ± 0.05	9.50

From the results, it is possible to conclude that the final time optimization implies some significant improvement over the results of the feeding trajectory optimization. The mean of duration obtained was 26.2 hours, a value slightly higher than the value of 25 (set empirically). The best run results in a process duration of 26 hours, obtaining the values of $X_f = 51.7$ and $W_f = 4.98$. So, comparing with previous results, the final biomass is increased, taking advantage of the extra hour and the final value of W is very near its physical limit.

A further and more significant improvement is obtained when the initial parameter optimization is added. The duration of the process obtained in all runs was 25 hours, the default value, showing the correctness of the practicioner's choice. Analyzing the results of the best run, the values of X_f and W_f are 50.7 and 5.0, respectively. The initial parameters took values similar to those obtained in the previous section, being worth to notice that $W_i = 3.15$. The feeding trajectory obtained is depicted in Figure 1. It can be noticed the smoothness of the result obtained, without resorting to specially tailored genetic algorithms, such as the ones proposed in [12]. The smoothness is important since it makes easier the physical implementation of the proposed solution.

Making a comparison of the different *EAs*, their computational performance is quite similar, although the *EAs* for final time optimization can be dependent on the minimum and maximum values defined. The convergence of the different algorithms is plotted in Figure 3, where it is visible that good results are obtained in about 1000 generations, which means about 15 minutes of computation time.

6 Conclusions and further work

In this work an *EA*, based on real-valued representations and variable size chromosomes was proposed in order to optimize both the feeding trajectory, the initial conditions and the duration of a fermentation process. The results, although based on a simulation model, show that the *EA* is capable of simultaneously optimizing all the aforementioned aspects, a result that is never been obtained in any study known by the authors. The settings reached by the *EAs* are quite

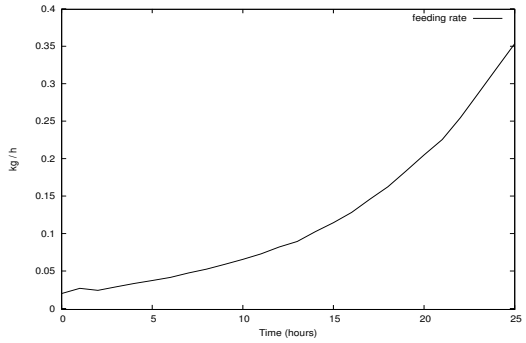

Fig. 1. Feeding trajectory obtained by the *EA* which combines feeding, initial conditions and final time optimization (best run).

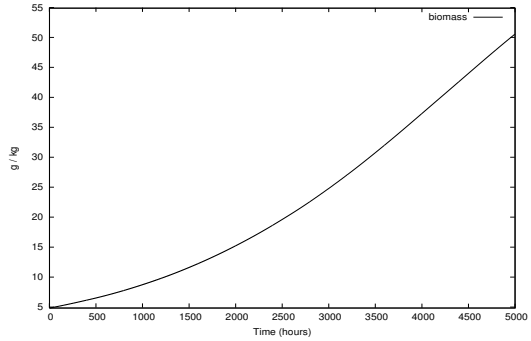

Fig. 2. Biomass trajectory obtained by the *EA* which combines feeding, initial conditions and final time optimization (best run).

near the values used in real experiments, being the result of both theoretical knowledge and years of practice. It is remarkable that the *EAs* are capable of setting, in tens of minutes, a number of parameters that takes years for a practicioner to learn. Therefore, the results of the *EA* are quite encouraging and its application to these kind of bioprocesses highly recommended.

The quantitative model that serves as a base for the simulations done in this work is based on differential equations. Other types of models have been proposed in literature, namely *Fuzzy Rules* or *Artificial Neural Networks* [13]. The testing of the proposed *EAs* in these settings is desirable. Another area of future research is the consideration of on-line adaptation, being the model of the process updated during the fermentation process, a task that can be also performed by *EAs*. In this case, the good computational performance of the proposed *EAs* are a benefit, if there is the need to re-optimize the feeding given a new model and values for the state variables measured on-line.

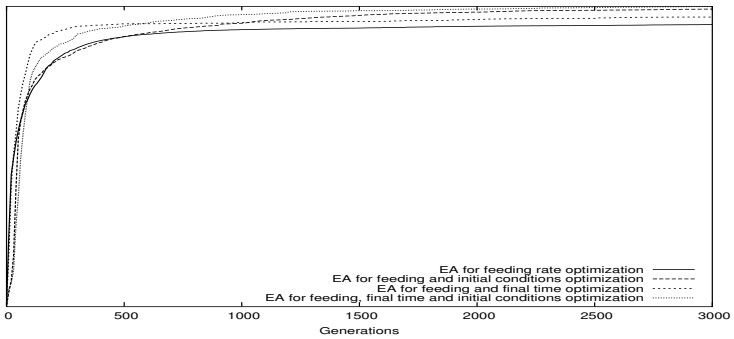

Fig. 3. Plot of the convergence of the different *EAs*.

References

1. P. Angelov and R. Guthke. A Genetic-Algorithm-based Approach to Optimization of Bioprocesses Described by Fuzzy Rules. *Bioprocess Engin.*, 16:299–303, 1997.
2. G. Bastin and D. Dochain. *On-line Estimation and Adaptive Control of Bioreactors*. Elsevier Science Publishers, Amsterdam, 1990.
3. A.E. Bryson and Y.C. Ho. *Applied Optimal Control - Optimization, Estimation and Control*. Hemisphere Publication Company, New York, 1975.
4. J.P. Chiou and F.S. Wang. Hybrid Method of Evolutionary Algorithms for Static and Dynamic Optimization Problems with Application to a Fed-batch Fermentation Process. *Computers & Chemical Engineering*, 23:1277–1291, 1999.
5. F. Herrera, M. Lozano, and J. Verdegay. Tackling Real-Coded GAs: Operators and Tools for Behavioral Analysis. *Artif. Intel. Review*, 12:265–319, 1998.
6. R. Marteijn, O. Jurrius, J. Dhont, C. de Gooijer, J. Tramper, and D. Martens. Optimization of a Feed Medium for Fed-Batch Culture of Insect Cells Using a Genetic Algorithm. *Biotechnol Bioeng*, 81:269–278, 2003.
7. Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, USA, third edition, 1996.
8. H. Moriyama and K. Shimizu. On-line Optimization of Culture Temperature for Ethanol Fermentation Using a GA. *Journ. Chem. Tec. Biotec.*, 66:217–222, 1996.
9. I. Rocha. *Model-based strategies for computer-aided operation of recombinant E. coli fermentation*. PhD thesis, Universidade do Minho, 2003.
10. I. Rocha and E.C. Ferreira. On-line Simultaneous Monitoring of Glucose and Acetate with FIA During High Cell Density Fermentation of Recombinant E. coli. *Analytica Chimica Acta*, 462(2):293–304, 2002.
11. M. Rocha, P. Cortez, and J. Neves. Evolutionary Approaches to Neural Network Learning. In *Proc. Portug. Conf. Artif. Intel.-EPIA'2003*. Springer, 2003.
12. J.A. Roubos, G. van Straten, and A.J. van Boxtel. An Evolutionary Strategy for Fed-batch Bioreactor Optimization: Concepts and Performance. *Journal of Biotechnology*, 67:173–187, 1999.
13. A. Tholudur and W.F. Ramirez. Optimization of Fed-batch Bioreactors Using Neural Network Parameters. *Biotechnology Progress*, 12:302–309, 1996.
14. V. van Breusegem and G. Bastin. Optimal Control of Biomass Growth in a Mixed Culture. *Biotechnology and Bioengineering*, 35:349–355, 1990.