

A Hybrid Improved Genetic Algorithm and Its Application in Dynamic Optimization Problems of Chemical Processes^{*}

SUN Fan (孙帆), DU Wenli (杜文莉), QI Rongbin (祁荣宾), QIAN Feng (钱锋)^{**} and ZHONG Weimin (钟伟民)

Key Laboratory of Advanced Control and Optimization for Chemical Processes, Ministry of Education, East China University of Science and Technology, Shanghai 200237, China

Abstract The solutions of dynamic optimization problems are usually very difficult due to their highly nonlinear and multidimensional nature. Genetic algorithm (GA) has been proved to be a feasible method when the gradient is difficult to calculate. Its advantage is that the control profiles at all time stages are optimized simultaneously, but its convergence is very slow in the later period of evolution and it is easily trapped in the local optimum. In this study, a hybrid improved genetic algorithm (HIGA) for solving dynamic optimization problems is proposed to overcome these defects. Simplex method (SM) is used to perform the local search in the neighborhood of the optimal solution. By using SM, the ideal searching direction of global optimal solution could be found as soon as possible and the convergence speed of the algorithm is improved. The hybrid algorithm presents some improvements, such as protecting the best individual, accepting immigrations, as well as employing adaptive crossover and Gaussian mutation operators. The efficiency of the proposed algorithm is demonstrated by solving several dynamic optimization problems. At last, HIGA is applied to the optimal production of secreted protein in a fed batch reactor and the optimal feed-rate found by HIGA is effective and relatively stable.

Keywords genetic algorithm, simplex method, dynamic optimization, chemical process

1 INTRODUCTION

Chemical processes are often described by a group of differential algebraic equations (DAEs). Dynamic optimization problems are encountered in a large number of chemical engineering applications aiming to optimize a predefined performance index, such as profitability or productivity, subject to safety or environmental specifications over a time interval by controlling operation variables [1–3].

A typical dynamic optimization problem for a continuous process is described as

$$\min J(u) = \Phi[x(t_f), t_f] + \int_0^{t_f} \Psi[x(t), u(t), t] dt \quad (1)$$

subject to the dynamics of the process, described by a set of DAEs

$$\begin{cases} \frac{dx}{dt} = \phi[x(t), u(t), t] \\ 0 = \varphi[x(t), u(t), t] \end{cases} \quad (2)$$

where $x(t)$ is the state variable, $u(t)$ denotes the control variable, J is the performance index, t_0 is the initial time, t_f is the final time, Φ is a penalty function for the final state $x(t_f)$ at t_f , Ψ is a function to weigh the state path traveled and the control sequence used, ϕ is the differential equation constraints, and φ is the algebraic equation constraints, assumed to be index one. Initial conditions are specified in order to assure the solvability of the DAEs: $x(t_0) = x_0$.

Several methods to solve the dynamic optimization problems have been reported in literature [4–11]. Gradient algorithms based on the Hamiltonian function, such as the conjugate gradient method and the steepest descent method, have been proved to be reliable. However, when solving dynamic optimization problems in chemical processes, the gradient information is sometimes unavailable. Dynamic programming (DP) and some intelligent optimization algorithms, including genetic algorithm (GA), ant-colony algorithm (ACA) and particle swarm algorithm (PSA) have been proved to be feasible methods when the gradient is difficult to calculate [12–21]. DP methods rely on Bellman's optimality conditions. It is a successful method for solving dynamic optimization problems, except for the dimension curse, which impairs the applicability of this method. To overcome this drawback, iterative DP (IDP) has been proposed by Mekarapiruk and Luus [8]. IDP reduces the expansion of dimension and promotes the reliability of the solution through iterative computation, but the high computational cost for systems involving a large number of DAEs restricts its use to problems on a small scale. Rajesh *et al.* [13] utilizes the ant colony framework to solve dynamic optimization problem. The framework is characterized by six dynamic benchmark examples with varying degrees of complexity. The method is easier to implement than IDP, but at the cost of solution accuracy. In the iterative ACA (IACA) proposed by Zhang *et al.* [17], the basic principle is the iterative execution of the ACA and the gradual approximation of the

Received 2011-03-31, accepted 2012-09-21.

^{*} Supported by Major State Basic Research Development Program of China (2012CB720500), National Natural Science Foundation of China (Key Program: U1162202), National Science Fund for Outstanding Young Scholars (61222303), National Natural Science Foundation of China (21276078, 21206037) and the Fundamental Research Funds for the Central Universities.

^{**} To whom correspondence should be addressed. E-mail: fqian@ecust.edu.cn

optimal control profile. In a new charged ACA proposed by Tfaili and Siarry [18] for continuous dynamic optimization, the diversification in the ant population is maintained by the repulsive electrostatic charge of each ant, which allows these ants to keep some distance from one another.

When GA is used for solving dynamic optimization problems, its advantage is that with crossover and mutation operators, control profiles at all time stages are optimized simultaneously. The major problem of GA is that it may spend a lot of time to find the optimum value as dynamic optimization problems are usually more complex than static optimization problems, and it may be trapped in the local optimum. In this paper, a hybrid improved genetic algorithm (HIGA) is proposed to overcome these defects. In HIGA, simplex method (SM) is introduced to do local search for the best chromosome in the evolution process of GA. Some improvements, such as protecting the best individual, accepting immigrations, as well as employing adaptive crossover and Gaussian mutation operators, are also presented to overcome the slow and premature convergence of GA. HIGA is applied to some benchmarks to demonstrate its efficiency and effectiveness for dealing with dynamic optimization problems.

2 THE SIMPLEX METHOD

In the later period of evolution, GA has a very slow convergence speed. To overcome the defects in local search by GA, SM is implemented to improve the search speed. SM is a classical powerful local descent algorithm, without the use of objective function derivatives [22]. SM is a local search technique that uses the evaluation of the current dataset to determine the promising search direction [23–26]. A simplex is defined by a number of points equal to one more than the number of dimensions of the space. SM searches for an optimal point by evaluating a set of points forming a simplex and continually forming new simplices by replacing the worst point in the simplex over the centroid of the remaining points.

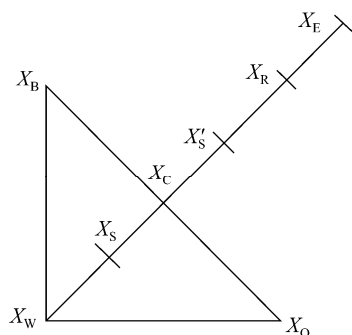


Figure 1 The vertex in simplex method

Step 1. The simplex begins by choosing $d+1$ vertexes in d -dimensional space.

Step 2. Let X_W denote the vertex with the worst evaluation and computes the centroid X_C of the remaining vertexes. Let X_B denote the vertex with the best evaluation and X_O denote other vertexes. Thus $f(X_B) < f(X_O) < f(X_W)$.

Step 3. Through a sequence of elementary geometric transformations (reflection, contraction, expansion and multi-contraction), the initial simplex moves, expands or contracts (see Fig. 1). After each transformation, the current worst vertex is replaced by a better one.

The worst vertex X_W is reflected across X_C to obtain a new point X_R .

$$\text{Reflection: } X_R = X_W + \alpha(X_C - X_W), \quad \alpha > 1 \quad (3)$$

$$\text{Expansion: } X_E = X_W + \beta(X_C - X_W), \quad \beta > \alpha \quad (4)$$

$$\text{Contraction 1: } X_S = X_W + \gamma(X_C - X_W), \quad \gamma < 1 \quad (5)$$

$$\text{Contraction 2: } X'_S = X_W + \lambda(X_C - X_W), \quad 1 < \lambda < \alpha \quad (6)$$

If $f(X_R) < f(X_B)$, an extension is made to point X_E . We then keep X_R or X_E as a replacement for X_W that depends on which function value is lower.

If $f(X_R) > f(X_B)$, a contraction is made to point X_S or X'_S depending on which is lower, $f(X_W)$ or $f(X_E)$.

If $f(X_S)$ or $f(X'_S)$ is higher than $f(X_B)$, the contraction fails and then we perform shrinkage operation. The shrinkage operation reduces the size of the simplex by moving all but the best point X_B halfway towards the best point X_B . The algorithm then evaluates function values at each vertex and returns to the reflection step to start a new iteration.

In HIGA, another reflection point X'_R is added to reduce the number of contraction. And this can increase the opportunities of the worst vertex X_W being replaced by the point on the other side of centroid X_C :

$$X'_R = X_W + \eta(X_C - X_W), \quad \alpha < \eta < \beta \quad (7)$$

Step 4. This cycle iterates, until the step size becomes less than a predetermined value or the simplex finds a better vertex than the worst vertex.

The performance of local search is illustrated by four case studies in Section 4.5.

3 HYBRID IMPROVED GENETIC ALGORITHM

Referring to Eq. (1), the objective is to find the profile of the control variable $u(t)$, which optimizes the objective function. The “unknowns” in this optimization problem are differential state variables $x(t)$ and control variables $u(t)$. In this work, the time interval $[t_0, t_f]$ is uniformly divided into n stages, so the i th time stage is $[t_{i-1}, t_i]$ ($i=1, 2, \dots, n$), where $t_0 = t_0$ and $t_n = t_f$. At the i th stage $[t_{i-1}, t_i]$ ($i=1, 2, \dots, n$), control variable $u(t)$ is approximate to a constant u_i ($i=1, 2, \dots, n$), so the control profile is $u_1, u_2, \dots, u_n$. GA is to search the best control profile of this discrete time

system as an approximation of the continuous problem.

The construction of a typical control profile is illustrated with an example. For a control profile with only one control variable at $n = 5$, the typical chromosome is shown in Fig. 2. For more control variables, the chromosome will be more complicated. For example, for a control profile with two control variables at $n = 5$, the typical chromosome is as shown in Fig. 3.

2.33	4.69	3.58	1.09	0.96
------	------	------	------	------

Figure 2 A typical chromosome of one control variable at $n = 5$

2.33	4.69	3.58	1.09	0.96	6.46	8.64	7.49	9.52	5.78
------	------	------	------	------	------	------	------	------	------

Figure 3 A typical chromosome of two control variables at $n = 5$

In order to improve the local fine tuning capability of GA and quicken the convergence rate, GA is combined with SM and cultural algorithm (CA) to form a hybrid optimization algorithm. The detailed process of our HIGA is as follows.

3.1 Normalization for population

Combined with the normative knowledge in CA, some constraint conditions are set to make sure no invalid solutions in the population. The normative knowledge regulates the search region and ensures the evolution in the optimal regions. At the k th step of iteration, the searching region of the control variable u is denoted by $[u_{i\min}^{(k)}, u_{i\max}^{(k)}]$ in the i th time stage ($i = 1, 2, \dots, n$). At the beginning of iteration, that is,

$$\text{When } k = 0, \quad u_{i\min}^{(0)} = u_{\min}, \quad i = 1, 2, \dots, n \quad (8)$$

$$u_{i\max}^{(0)} = u_{\max}, \quad i = 1, 2, \dots, n \quad (9)$$

And the initial search width of $u(i)$ will be

$$w_i^{(0)} = u_{i\max}^{(0)} - u_{i\min}^{(0)}, \quad i = 1, 2, \dots, n \quad (10)$$

During iterations, $u_{i\min}^{(k)}$ and $u_{i\max}^{(k)}$ will change and the search width $w_i^{(k)}$ will decrease.

The immune operators in the artificial immune algorithm (AIA) are introduced in HIGA to make the individuals search in feasible region. Immune operators include vaccination and immune selection. (1) Vaccination. Some genes in the individuals are modified according to the normative knowledge. When the gene is out of the searching region, it is forced to the border of the search space. (2) Immune selection. Test the vaccinated individual to find out if the fitness of the vaccinated individual increases. When the fitness of the vaccinated individual is larger than the fitness of the original individual, replace the original individual with the vaccinated one.

3.2 The elitist strategy

Elitism reservation strategy is adopted by transferring the best chromosome of each generation to the next generation. And a new elitist strategy is presented, which leads in a new individual in generation. First, we calculate individual fitness in the current population and find the optimal and the worst individuals. Then, a new individual is added to ensure that it has much difference from the optimal individual. The new individual is obtained by a mutation of the best individual. At last, if the fitness of the new individual is larger than the worst one, the worst individual is replaced by the new one. The optimal individual and the new lead-in individual are added in the population after the selection, crossover and mutation.

3.3 Adjustment of population

Considering that the changing trend of the optimal control profiles is monotonic in many actual chemical processes, some monotone increasing and monotone decreasing individual are added in the population. If the fitness of a monotone increasing or decreasing individual is larger than the average fitness of the population, this individual can be reserved. Thus there is great probability to find better solution in the early evolution, and it will improve the average fitness of the population.

The adjustment of population is only carried out in the early evolution to speed the searching. The monotone increasing and monotone decreasing individual are obtained from the best individual by bubble sort method. Taking monotone increasing as example, bubble sort method is operated: former next, then compare two adjacent genes on the chromosome. If the former is larger than the latter, the two genes are exchanged, and then continue later operation. If the fitness of a monotone increasing or decreasing individual is larger than the fitness of the worst individual, this individual is reserved and added in the population after the selection, crossover and mutation.

If there is no exchange of gene location in a trip to the sorting, the bubble sort can be ended after this trip. Thus we introduce a variable "exchange". Before every trip to the sorting, the variable "exchange" is "False"; if exchange occurs in the sorting, it becomes "True". After each trip, we check the variable "exchange", and if it is "False", the bubble sort is over.

3.4 Genetic operations

3.4.1 Roulette selection

The roulette selection approach is applied for the selection operation in HIGA. A random number is generated, and the individual whose segment spans the random number is selected. The process is repeated until the desired number of individuals is obtained. The probability of the selection of an individual depends

on the relative fitness of individual. The individuals with higher fitness have a greater probability of survival. The probability of individual j being selected is

$$P_{sj} = \frac{f_j}{\sum_{i=1}^M f_i}, \quad j=1, 2, \dots, M \quad (11)$$

where f_j is the fitness value of individual j . And the fitness value f is calculated in HIGA according to

$$f = \begin{cases} J, & \text{if the objective is to maximize } J \\ 1/J, & \text{if the objective is to minimize } J \end{cases} \quad (12)$$

3.4.2 Adaptive crossover

Crossover is used to recombine genetic material in two parent chromosomes to produce two child chromosomes that share the characteristics of their parents. In order to expand the search region in the early evolution, a good crossover operator should make the father and the offspring individuals have much difference. With the evolutionary generation increasing, the differences should be gradually narrowed. An adaptive crossover operator is designed to enhance the global search capability of GA in our algorithm:

$$P_c = P_{c0} \sqrt{1 - k/G} \quad (13)$$

Thus we crossover more frequently at the beginning but crossover less later.

3.4.3 Gaussian mutation

A kind of Gaussian mutation operator is used to enhance the local search ability around every peak value, which makes the individuals searched towards the best fitness. In the mutation operations, a random number accorded with normal distribution is used to replace the original genes. The random number Q accorded with normal distribution $N(m, s^2)$ is obtained by a random number $v_i (i=1, 2, \dots, 10)$ accorded with uniform distribution:

$$Q = m + s \left(\sum_{i=1}^{10} v_i - 5 \right) \quad (14)$$

When the chromosome $X = X_1 X_2 \dots X_i \dots X_n$ becomes $X = X_1 X_2 \dots X'_i \dots X_n$ by Gaussian mutation and the search region of gene X_i is $[u_{i\min}^{(k)}, u_{i\max}^{(k)}]$, we have

$$m = \frac{u_{i\min}^{(k)} + u_{i\max}^{(k)}}{2} \quad (15)$$

$$s = \frac{u_{i\max}^{(k)} - u_{i\min}^{(k)}}{5} \quad (16)$$

The new gene X'_i will be

$$Q = \frac{u_{i\min}^{(k)} + u_{i\max}^{(k)}}{2} + \frac{u_{i\max}^{(k)} - u_{i\min}^{(k)}}{5} \left(\sum_{i=1}^{10} v_i - 5 \right) \quad (17)$$

3.5 Search region reduction

The search region of the control variable $u(i)$ is denoted by $[u_{i\min}^{(k)}, u_{i\max}^{(k)}]$ ($i=1, 2, \dots, n$). In this study, after each generation, the baseline for the search will be moved to the optimal value obtained in the previous search. Then the region reduction strategy is employed to reduce the search region:

$$w_i^{(k+1)} = r w_i^{(k)}, \quad i=1, 2, \dots, n \quad (18)$$

$$u_{i\min}^{(k+1)} = \frac{u_{ib}^{(k)} - w_i^{(k+1)}}{2}, \quad i=1, 2, \dots, n \quad (19)$$

$$u_{i\max}^{(k+1)} = \frac{u_{ib}^{(k)} + w_i^{(k+1)}}{2}, \quad i=1, 2, \dots, n \quad (20)$$

where r is the reduction ratio, $0.5 < r < 1$; and $u_{ib}^{(k)}$ is the optimal control variable value obtained in the previous search.

3.6 Steps of HIGA

Figure 4 shows the flow chart of the hybrid genetic algorithm. The detailed steps of HIGA are as follows.

Step 1. Initialize the population. The original population, including M individuals, is generated randomly ($M=50-200$). The current generation is denoted by k and $k=0$ initially. The maximum evolutionary generation is denoted by G .

Step 2. Normalize the population. The normative knowledge regulates the search region. At the k th step of iteration, the feasible search space of the chromosome $X = X_1 X_2 \dots X_n$ is $\{u_{\min 1}^{(k)}, u_{\max 1}^{(k)}, u_{\min 2}^{(k)}, u_{\max 2}^{(k)}; \dots; u_{\min n}^{(k)}, u_{\max n}^{(k)}\}$.

Step 3. Calculate objective function and individual fitness.

Step 4. Execute the elitist strategy. The optimal individual and the worst individual are found according to the individual fitness in the current population. Then, a new individual is added in the population by a mutation of the best individual, of which the mutation probability is 0.6. If the fitness of the new individual is larger than the worst one, the worst individual is replaced by the new one in Step 7.

Step 5. Execute the adjustment operation in Section 3.3 when $k < 0.05G$.

Step 6. Execute genetic operation including roulette selection, adaptive crossover and Gaussian mutation. The initial crossover probability is P_{c0} and the mutation probability is P_m .

Step 7. Add the optimal individual and new individual reserved in Step 5 in the population after the selection, crossover and mutation.

Step 8. Do the local search. Local search by SM is carried out every l iteration time ($l=0.1G$ and $l=0$ at first). The initial simplex is composed of the best q

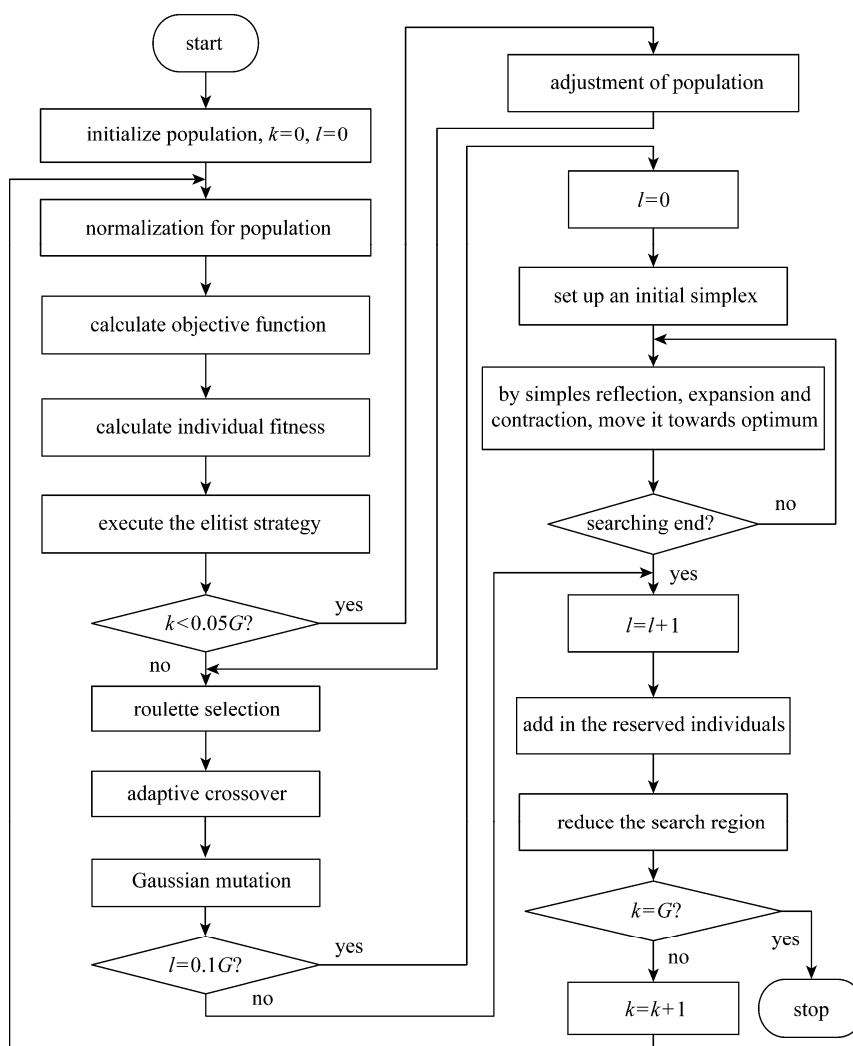


Figure 4 The flow chart of the hybrid genetic algorithm

individuals ($q = 0.1M$). SM is carried out for p times at most ($p = 10\text{--}30$).

Step 9. Reduce search region and judge whether it meets the loop termination conditions. If it is not the termination, turn to Step 2.

4 CASE STUDIES FOR HIGA

HIGA is applied to several cases to demonstrate its efficiency and effectiveness for dealing with dynamic optimization problems. For each case, 10 runs are taken, and the parameters of GA are $M = 50$, $P_{c0} = 0.5$, and $P_m = 0.1$. The parameters of SM are $\alpha = 2$, $\eta = 2.5$, $\beta = 3$, $\gamma = 0.5$, and $\lambda = 1.5$. The maximum evolutionary generations G and the reduction ratio r vary in different cases.

4.1 Case 1

The math model of case 1 is described as

$$\min J(u) = \frac{1}{2} \int_0^1 (x^2 + u^2) dt \quad (21)$$

$$\text{s.t. } \frac{dx}{dt} = -x + u, \quad x(0) = 10,$$

$$u^*(t) = 0.10(\sqrt{2} + 1)\exp(\sqrt{2}t) - 9.9(\sqrt{2} - 1)\exp(-\sqrt{2}t) \quad (22)$$

Eq. (22) is the optimal control profile obtained by Gong [27] on the basis of minimum theory, and the optimal value of objective is 19.2910. The optimal control profile is shown by dotted lines in Fig. 5. With iterative genetic algorithm (IGA), Zhang and Chen [15] obtained the values of 19.3869 ($n = 10$), 19.3869 ($n = 20$) and 19.3016 ($n = 30$). We use HIGA to solve the problem and obtain the minimal values 19.2983 ($n = 10$) and 19.2930 ($n = 20$). The optimal control profile is shown in Fig. 5.

4.2 Case 2

Case 2 is the temperature control for consecutive

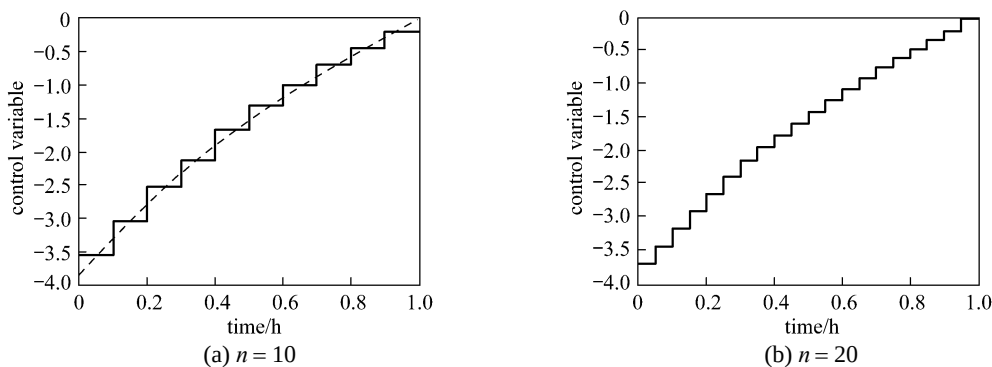


Figure 5 Optimal control profiles for case 1

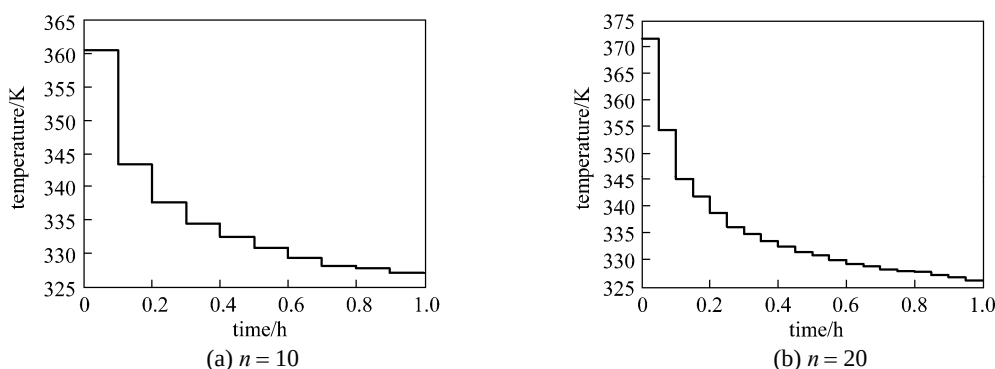


Figure 6 Optimal control profiles for case 2

reaction problem. The objective is to obtain the optimal temperature profile that maximizes the yield of the intermediate product B at the end of the operation in a batch reactor, where the reaction $A \rightarrow B \rightarrow C$ occurs. The math model is

$$\max J(T) = x_2(t_f) \quad (23)$$

$$\text{s.t.} \quad \frac{dx_1}{dt} = -4000 \exp\left(\frac{-2500}{T}\right) x_1^2, \quad x_1(0) = 1,$$

$$\frac{dx_2}{dt} = 4000 \exp\left(\frac{-2500}{T}\right) x_1^2 -$$

$$620000 \exp\left(\frac{-5000}{T}\right) x_2, \quad x_2(0) = 0$$

with $298 \leq T \leq 398$ and $t_f = 1$, where x_1 is the concentration of A, x_2 is the concentration of B, and T is the temperature (K).

A value of 0.610775 for the optimum has been reported. With complex ant colony algorithm (CACA), Rajesh *et al.* [13] obtained a value of 0.61045. With IACA, Zhang *et al.* [17] obtained the values of 0.6100 ($n = 10$) and 0.6104 ($n = 20$). We use HIGA to solve the problem and the maximal values 0.61007 ($n = 10$) and 0.61046 ($n = 20$). The optimal control profile is shown in Fig. 6.

4.3 Case 3

Case 3 is a nonlinear unconstrained four-state variable mathematical system. The math model is

$$\min J(\mathbf{u}) = \mathbf{x}_4(t_f) \quad (24)$$

$$\text{s.t.} \quad \frac{dx_1}{dt} = x_2, \quad \frac{dx_2}{dt} = -x_3 u + 16t - 8, \quad \frac{dx_3}{dt} = u,$$

$$\frac{dx_4}{dt} = x_1^2 + x_2^2 + 0.0005(x_2 + 16t - 8 - 0.1x_3 u^2)^2,$$

$$\mathbf{x}(0) = (0, -1, -\sqrt{5}, 0)^T, \quad -4 \leq u \leq 10$$

where $t_f = 1$, $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4$ are state vectors, and $\mathbf{u}(t)$ is the control vector.

A value of 0.12011 has been reported to be the optimum. With CACA, Rajesh *et al.* [13] obtained a value of 0.12904. With IACA, Zhang *et al.* [17] obtained a value of 0.1235 at $n = 16$. We use HIGA to solve the problem and obtain the minimal values 0.12157 ($n = 10$) and 0.12092 ($n = 20$). The optimal control profile is shown in Fig. 7. At $n = 16$, a minimal value 0.12114 is obtained by HIGA, better than that with IACA.

4.4 Case 4

Case 4 is an unconstrained two-state variable

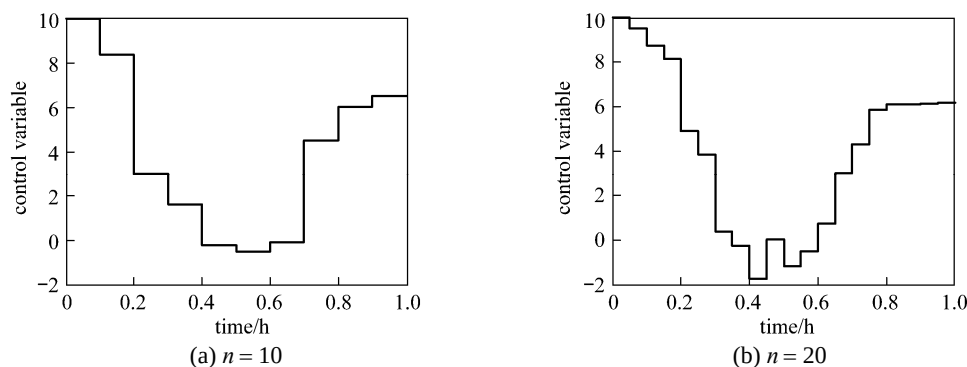


Figure 7 Optimal control profiles for case 3

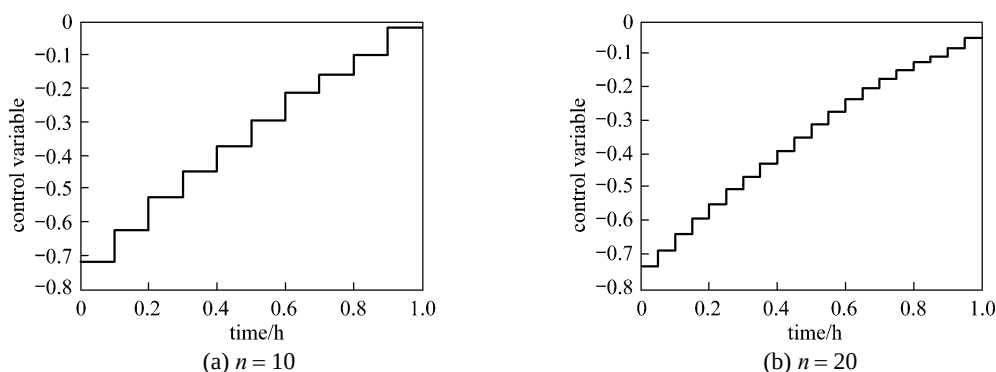


Figure 8 Optimal control profiles for case 4

mathematical system. The math model is

$$\min J(\mathbf{u}) = \mathbf{x}_2(t_f) \quad (25)$$

$$\text{s.t.} \quad \frac{d\mathbf{x}_1}{dt} = \mathbf{u}, \quad \frac{d\mathbf{x}_2}{dt} = \mathbf{x}_1^2 + \mathbf{u}^2, \quad \mathbf{x}(0) = (1, 0)^T,$$

where $t_f = 1$, $\mathbf{x}_1$, $\mathbf{x}_2$ are state vectors, and $\mathbf{u}(t)$ is the control vector.

For this case, a value of 0.76159 for the optimum has been reported. With CACA, Rajesh *et al.* [13] obtained a value of 0.76238. We use HIGA to solve the problem and obtain the minimal values 0.76215 ($n=10$) and 0.76191 ($n=20$). The optimal control profile is shown in Fig. 8.

All results for four cases are given in Table 1. The HIGA gives good results at an acceptable time interval partition number. At $n=10$ or 20, the results are accurate enough, so the computation time is reduced.

4.5 Discussion

4.5.1 Performance analysis of local search

The most efficient part in HIGA is the local search by SM. The performance of local search is illustrated with four cases. In each case, simple GA, HIGA with and without local search are used to search for the optimal value at $n=10$. All the algorithms use the same parameter settings.

In case 1, at $n=10$, the computing time of IGA reported is 12 s [17]. By HIGA without local search, it

Table 1 Summary of results for four case studies

Case	Best	Method	Global optimum	Relative error/%
1	19.2910	IGA [17]	19.3869 ($n=10$)	5.21
		IGA [17]	19.3869 ($n=20$)	5.21
		IGA [17]	19.3016 ($n=30$)	0.055
		HIGA	19.2983 ($n=10$)	0.038
		HIGA	19.2930 ($n=20$)	0.010
2	0.610775	CACA [13]	0.61045	0.053
		IACA [17]	0.6100 ($n=10$)	0.127
		IACA [17]	0.6104 ($n=20$)	0.061
		HIGA	0.61007 ($n=10$)	0.115
		HIGA	0.61046 ($n=20$)	0.052
3	0.12011	CACA [13]	0.12904	7.43
		IACA [17]	0.1235 ($n=16$)	2.82
		HIGA	0.12157 ($n=10$)	1.22
		HIGA	0.12114 ($n=16$)	0.86
		HIGA	0.12092 ($n=20$)	0.67
4	0.76159	CACA [13]	0.76238	0.104
		HIGA	0.76215 ($n=10$)	0.074
		HIGA	0.76191 ($n=20$)	0.042

Table 2 The performance of local search for case 1

Method	Optimum	Relative error/%	Time/s	Max function evaluations
IGA [17]	19.3869	5.21	12	16000
simple GA	19.3179	0.139	19	25000
HIGA without local search	19.3085	0.091	12	16000
HIGA with local search	19.2983	0.038	8	9000

Table 3 The performance of local search for cases 2, 3, and 4

Case	Method	Optimum	Relative error/%	Time/s	Max function evaluations
2	simple GA	0.60949	0.211	36	40000
	HIGA without local search	0.60995	0.135	24	28000
	HIGA with local search	0.61007	0.115	19	17000
3	simple GA	0.12341	2.75	88	100000
	HIGA without local search	0.12269	1.80	54	58000
	HIGA with local search	0.12157	1.22	36	39000
4	simple GA	0.76249	0.118	67	75000
	HIGA without local search	0.76239	0.105	44	46000
	HIGA with local search	0.76215	0.074	27	28000

also takes 12 s to obtain a value of 19.3024. By HIGA with local search, it takes 8 s to obtain the value of 19.2983, which shows that HIGA provides better result at faster speed. The results obtained by different algorithms for case 1 are summarized in Table 2. Fig. 9 shows the evolution process of the best objective value for three different algorithms. It is clear that the proposed hybrid algorithm is more precise than the simple GA.

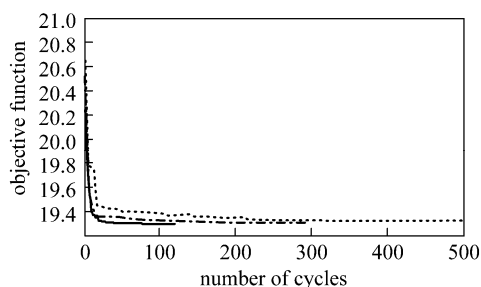


Figure 9 Objective functions with the number of generation for case 1
 simple GA; - - - - HIGA without local search; — HIGA with local search

From the results above, in simple GA, the genetic process is slow and only small changes of the fitting error are observed between 250 and 500 cycles. In HIGA, when the loss of diversity is observed, the search process is continued by SM. Thus HIGA has much faster convergence speed and lower relative error. It is also clearly to see the advantage of local search from case 2 to case 4 in Table 3.

4.5.2 Performance analysis on the adjustment of population

To verify the performance of the “adjustment of

population” in Section 3.3, the four cases are solved by HIGA with and without “adjustment of population”. In each case, $n = 10$ is used. For case 1, the max function evaluations are 9000 and 12000, respectively. For case 2, they are 17000 and 22000, respectively. For case 3, they are both 39000. The changing trend of the optimal control profile in case 3 is not monotonic, so the “adjustment of population” has no effect. For case 4, the max function evaluations are 28000 and 34000 with and without “adjustment of population”, respectively. Therefore, for problems that the changing trend of the optimal control profile is monotonic, the operation of the “adjustment of population” can reduce the function evaluations.

4.5.3 Analysis of the improvements in HIGA

In HIGA, SM is introduced to do local search for the best chromosome in the evolution process of GA. SM can enhance the search efficiency of GA by compressing the search space and improve the local search ability. As shown in Section 4.5.1, the local search effectively improves the search speed of the algorithm and reduces the function evaluations.

Elitism reservation strategy, adaptive crossover and Gaussian mutation are the improvements of the basic GA. In the new elitism reservation strategy, optimal individual is reserved to ensure global convergence. The new individual is added to prevent premature and maintain the population diversity. Larger initial probability of crossover operator is the main way to get new individuals in the early evolution. If the probability of crossover operator is great in all generations, the good genes may be damaged at the later period of evolution, increasing the possibility of random search. A new adaptive crossover operator is designed in HIGA. Gaussian mutation operator is used to enhance the local search ability around every peak value,

which makes the individuals searched towards the best fitness.

With the evolution process, search region should be focused on the advantage area. The normative knowledge is used to regulate the search region in HIGA. At the beginning of the search, all control variables in the whole time interval have the same search region. However, in different time stages, the optimal control variables differ from one another. If the search region remains unchanged, it will increase the difficulty of finding the optimal solution, and the computational efficiency will decrease. In order to improve the convergence rate, the search region is reduced in HIGA.

5 APPLICATION TO FED-BATCH REACTOR

A model for production of secreted protein in fed-batch bioreactor was reported by Park and Ramirez [28]. This problem deals with the optimal production of secreted protein. It shows very low sensitivity of the control profile on the performance index, often leading to computational difficulties, in particular when gradient-based algorithms are used. The objective here is to maximize the secreted heterologous protein by a yeast strain in a fed-batch culture. The mathematical statement is as follows.

Mass balance equations are $\frac{d(P_M V)}{dt} = A(S)(P_T - P_M)V$, $\frac{d(P_T V)}{dt} = B(S)XV$, $\frac{d(XV)}{dt} = C(S)XV$, $\frac{d(SV)}{dt} = -YC(S)XV + FS_F$, and $\frac{dV}{dt} = F$.

The kinetic model is $A(S) = \frac{4.73C(S)}{0.12 + C(S)}$, $B(S) = \frac{S \exp(-5S)}{0.1 + S}$, $C(S) = \frac{21.87S}{(0.4 + S)(62.5 + S)}$, and $Y = 7.3$.

Initial and operating conditions are $S_F = 20$, $P_M(0) = 0$, $P_T(0) = 0$, $X(0) = 1$, $S(0) = 5$, $V(0) = 1$, $0 \leq F \leq 2.5$, and $t_f = 15$ h.

The objective function is $\max J = P_M(t_f)V(t_f)$.

In above equations, P_M is the concentration of the secreted protein ($\text{g} \cdot \text{L}^{-1}$), P_T is the total protein ($\text{g} \cdot \text{L}^{-1}$), X is the culture cell density ($\text{g} \cdot \text{L}^{-1}$), S is the substrate concentration ($\text{g} \cdot \text{L}^{-1}$), V is the holdup volume (L), F is the nutrient feed-rate ($\text{L} \cdot \text{h}^{-1}$), S_F is the concentration of the substrate in the feed stream ($\text{g} \cdot \text{L}^{-1}$), and J is the mass of protein produced (g).

Tholudur and Ramirez [29] used a modified IDP (FIDP) to obtain the value 32.31–32.41 ($n = 50$) and 32.40–32.47 ($n = 200$). Balsa-Canto *et al.* [30] used second order information (SOI) to obtain the value of 32.028 ($n = 15$). And with the restricted second order method (RSOM), Balsa-Canto *et al.* [31] obtained values of 32.388 ($n = 40$) and 32.480 ($n = 320$). Sarkar and Modak [32] used a hybrid artificial neural

network/simulated annealing algorithm (ANNNA) and obtained 30.894534 ($n = 7$), 32.359190 ($n = 10$) and 32.414095 ($n = 13$). With Chaos PSO (CPSO), Mo *et al.* [33] obtained 32.4249–32.4541 ($n = 20$).

Time interval partition number of some of these algorithms is very large, which makes the control operation complex and difficult. In actual industrial productions, operational variables should not change frequently. However, if time interval partition number is too small, the optimal value may be unsatisfactory. Taking into account the efficiency and accuracy, we divide the time interval into 20 equal parts. HIGA is carried out for 10 times, with the worst value of 32.4368 and the best optimal value of 32.4559. Fig. 10 shows the optimal feed-rate found by HIGA.

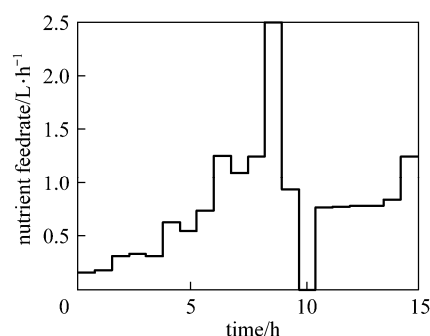


Figure 10 The optimal nutrient feed-rate

The number of function evaluations reported for this problem is 652001 to 6734001 for ANNNA and 8000000 for CPSO, which is very large. The number of objective function evaluations of HIGA is about 500000. Table 4 shows that HIGA has advantages in calculation and computational cost.

Table 4 Results for fed-batch reactor problem

Method	Optimal value	Max function evaluations
FIDP [29]	32.31–32.41 ($n = 50$)	not reported
FIDP [29]	32.40–32.47 ($n = 200$)	not reported
SOI [30]	32.028 ($n = 15$)	not reported
RSOM [31]	32.388 ($n = 40$)	not reported
RSOM [31]	32.480 ($n = 320$)	not reported
ANNNA [32]	30.894534 ($n = 7$)	674801
ANNNA [32]	32.359190 ($n = 10$)	652001
ANNNA [32]	32.414095 ($n = 13$)	6734001
CPSO [33]	32.4249–32.4541 ($n = 20$)	8000000
HIGA	32.4368–32.4559 ($n = 20$)	500000

6 CONCLUSIONS

A hybrid improved genetic algorithm, HIGA, for solving dynamic optimization problems is proposed in this paper. The approach is a reliable and useful

optimization tool for solving such problems. Local search strategy is added into GA, so that the search speed and precision is enhanced. This improves the convergence of the algorithm. The efficiency of HIGA is demonstrated with several dynamic optimization problems. The results show that our proposed algorithm is robust and efficient. At last, HIGA is used to maximize the production of secreted protein in fed-batch bioreactor and the optimal feed-rate found by HIGA is effective and relatively stable.

NOMENCLATURE

d	d -dimensional space
i	i th time stage
f	fitness of individuals
G	the maximum evolutionary generation
J	performance index
j	j th individual
k	current generation
l	iteration times to do local search
M	number of individuals in original population
m	expectations of the normal distribution
n	predefined number of time interval divided
P_c	crossover probability
P_m	mutation probability
P_s	probability of individuals being selected
p	times to carry out the simplex method
Q	a random number
q	number of individuals in the initial simplex
r	reduction ratio
s	standard deviation of the normal distribution
t_0	initial time
t_f	final time
u	control variables
u_{ib}	optimal control variable value obtained in the previous searching
u_{imax}	upper limit of feasible region
u_{imin}	lower limit of feasible region
v	a random number
w	search width
X_B	vertex with the best evaluation
X_C	centroid of vertexes
X_E	expansion vertex
X_O	other vertexes except the best vertex
X_R	reflection vertex
X_S	contraction vertex
X_W	vertex with the worst evaluation
x	state variables
α	constant
β	constant
γ	constant
η	constant
λ	constant
Φ	penalty function
ϕ	differential equation constraints
φ	algebraic equation constraints
ψ	function

REFERENCES

- Srinivasan, B., Palanki, S., Bonvin, D., "Dynamic optimization of batch processes I: Characterization of the nominal solution", *Compu. Chem. Eng.*, **27**, 1–26 (2003).
- Irizarry, R., "A generalized framework for solving dynamic optimization problems using the artificial chemical process paradigm: Applications to particulate processes and discrete dynamic systems", *Chem. Eng. Sci.*, **60**, 5663–5681 (2005).
- Zhang, B., Yu, H.J., Chen, D.Z., "Sequential optimization of chemical dynamic problems by ant-colony algorithm", *J. Chem. Eng. Chin. Univ.*, **20** (1), 120–125 (2006). (in Chinese)
- Biegler, L.T., Cervantes, A.M., Wachter, A., "Advances in simultaneous strategies for dynamic process optimization", *Chem. Eng. Sci.*, **57**, 575–593 (2002).
- Eva, B.C., Vassilios, S.V., Julio, R. B., "Dynamic optimization of single- and multi-stage systems using a hybrid stochastic-deterministic method", *Ind. Eng. Chem. Res.*, **44**, 1514–1523 (2005).
- Julio, R. B., Eva, B.C., Carmen, G.M., Antonio, A.A., "Dynamic optimization of bioprocesses: Efficient and robust numerical strategies", *J. Biotechnol.*, **117**, 407–419 (2005).
- Faber, R., Jockenhovel, T., Tsatsaronis, G., "Dynamic optimization with simulated annealing", *Compu. Chem. Eng.*, **29**, 273–290 (2005).
- Mekarapiruk, W., Luus, R., "Optimal control by iterative dynamic programming with deterministic and random candidates for control", *Ind. Eng. Chem. Res.*, **39**, 84–91 (2000).
- Hong, W.R., Tan, P.C., Wang, S.A.Q., Pu, L., "A comparison of arithmetic operations for dynamic process optimization approach", *Chin. J. Chem. Eng.*, **18** (1), 80–85 (2010).
- Tebhani, S., Dumur, D., Hafidi, G., "Open-loop optimization and trajectory tracking of a fed-batch bioreactor", *Chem. Eng. Proce.*, **47**, 1933–1941 (2008).
- Jose, A.E., Eva, B.C., Garcia, M.S., Julio, R.B., "Dynamic optimization of nonlinear processes with an enhanced scatter search method", *Ind. Eng. Chem. Res.*, **48**, 4388–4401 (2009).
- He, Y.J., Yu, H.J., Cheng, B., Chen, D.Z., "Multi-objective particle swarm optimization approach to solution of fed-batch bioreactor dynamic multi-objective optimization", *J. Chem. Ind. Eng.*, **58** (5), 1262–1270 (2007). (in Chinese)
- Rajesh, J., Gupta, K., Kusumakar, H.S., Jayaraman, V.K., Kulkarni, B.D., "Dynamic optimization of chemical processes using ant colony framework", *Compu. Chem.*, **25**, 583–595 (2001).
- Sarkar, D., Modak, J.M., "Optimisation of fed-batch bioreactors using genetic algorithms", *Chem. Eng. Sci.*, **58**, 2283–2296 (2003).
- Zhang, B., Chen, D.Z., "Iterative genetic algorithm and its application to policies optimization for bioreactor", *J. Chem. Ind. Eng.*, **56** (1), 100–104 (2005). (in Chinese)
- He, Y.J., Chen, D.Z., "Immune mechanism based multi-objective ant colony algorithm approach to batch reactor constrained dynamic multi-objective optimization problems", *J. Chem. Eng. Chin. Univ.*, **23** (2), 326–332 (2009). (in Chinese)
- Zhang, B., Chen, D.Z., Zhao, W.X., "Iterative ant-colony algorithm and its application to dynamic optimization of chemical process", *Compu. Chem. Eng.*, **29**, 2078–2086 (2005).
- Tfaily, W., Siarry, P., "A new charged ant colony algorithm for continuous dynamic optimization", *Appl. Math. Compu.*, **197**, 604–613 (2008).
- Asgari, S.A., Mahmoud, R.P., "Dynamic optimization in chemical processes using region reduction strategy and control vector parameterization with an ant colony optimization algorithm", *Chem. Eng. Technol.*, **31** (4), 507–512 (2008).
- Shelokar, P.S., Jayaraman, V.K., Kulkarni, B.D., "Multicanonical jump walk annealing assisted by tabu for dynamic optimization of chemical engineering processes", *Eur. J. Ope. Res.*, **185**, 1213–1229 (2008).
- Jorge, E.J., Ines, M.S., Isidoro, G.G., "Optimization of biotechnological processes. The acetic acid fermentation. Part III: Dynamic optimization", *Biochem. Eng. J.*, **45**, 22–29 (2009).
- Kasprzyk, G.P., Jasku, M., "Application of the hybrid genetic-simplex algorithm for deconvolution of electrochemical responses in SDLSV method", *J. Electra. Chem.*, **567**, 39–66 (2004).

- 23 Fan, S.K., Zahara, E., "A hybrid simplex search and particle swarm optimization for unconstrained optimization", *Eur. J. Ope. Res.*, **181**, 527–548 (2007).
- 24 Ren, Z.W., San, Y., Chen, J.F., "Hybrid simplex-improved genetic algorithm for global numerical optimization", *Acta Automatica Sinica*, **33**, 91–95 (2007).
- 25 Zahara, E., Kao, Y.T., "Hybrid Nelder-Mead simplex search and particle swarm optimization for constrained engineering design problems", *Exp. Sys. Appl.*, **36**, 3880–3886 (2009).
- 26 Amel, K., Mohamed, J., Moncef, C., "The use of the simplex method and its derivatives to the on-line optimization of the parameters of an injection molding process", *Chem. Intell. Lab. Sys.*, **96**, 117–122 (2009).
- 27 Gong, X.F., *Computation Method for Optimal Control Problem*, Science Press, Beijing (1979). (in Chinese)
- 28 Park, S., Ramirez, W.F., "Optimal production of secreted protein in fed-batch reactors", *AIChE J.*, **34** (8), 1550–1558 (1988).
- 29 Tholudur, A., Ramirez, W.F., "Obtaining smoother singular arc policies using a modified iterative dynamic programming algorithm", *Inter. J. Control.*, **8** (5), 1115–1128 (1997).
- 30 Balsa-Canto, E., Banga, J.R., "Efficient optimal control of bioprocesses using second-order information", *Ind. Eng. Chem. Res.*, **39** (11), 4287–4295 (2000).
- 31 Balsa-Canto, E., Banga, J.R., Alonso, A.A., Vassiliadis, V.S., "Dynamic optimization of chemical and biochemical processes using restricted second-order information", *Compu. Chem. Eng.*, **25**, 539–546 (2001).
- 32 Sarkar, D., Modak, J.M., "ANNSA: A hybrid artificial neural network/simulated annealing algorithm for optimal control problems", *Chem. Eng. Sci.*, **58**, 3131–3142 (2003).
- 33 Mo, Y.B., Chen, D.Z., Hu, S.X., "Chaos particle swarm optimization algorithm and its application in biochemical process dynamic optimization", *J. Chem. Ind. Eng.*, **57** (9), 2123–2127 (2006). (in Chinese)