A Biologically Inspired Intelligent PID Controller Tuning for AVR Systems

Dong Hwa Kim and Jae Hoon Cho

Abstract: This paper proposes a hybrid approach involving Genetic Algorithm (GA) and Bacterial Foraging (BF) for tuning the PID controller of an AVR. Recently the social foraging behavior of E. coli bacteria has been used to solve optimization problems. We first illustrate the proposed method using four test functions and the performance of the algorithm is studied with an emphasis on mutation, crossover, variation of step sizes, chemotactic steps, and the life time of the bacteria. Further, the proposed algorithm is used for tuning the PID controller of an AVR. Simulation results are very encouraging and this approach provides us a novel hybrid model based on foraging behavior with a possible new connection between evolutionary forces in social foraging and distributed non-gradient optimization algorithm design for global optimization over noisy surfaces.

Keywords: Bacterial foraging optimization, genetic algorithm, hybrid system, optimal algorithm.

1. INTRODUCTION

In the last decade, Genetic Algorithm (GA) based approaches have received increased attention from the engineers dealing with problems that could not be solved using conventional problem solving techniques [1-7]. A typical task of a GA in this context is to find the best values of a predefined set of free parameters associated with either a process model or a control vector. One of the active areas of research in GA approaches is for system identification [8-12]. A recent survey of evolutionary algorithms for the evaluation of improved learning algorithm and control system engineering can be found in [8,12,13]. GA has also been used to optimize nonlinear systems. Among them, a large amount of research is focused on the design of fuzzy controllers using evolutionary algorithm approaches. GAs could be used for developing the knowledge base about the controlled process in the form of linguistic rules and the fine tuning of fuzzy membership function [13].

A possible solution to a specific problem can be encoded as an individual (or a chromosome), which consists of group of genes. Each individual represents a point in the search space and a possible solution to the problem can be formulated. A population consists of a finite number of individuals and each individual is decided by an evaluating mechanism to obtain its fitness value. Using this fitness value and genetic operators, a new population is generated iteratively, and that is referred to as a generation. The GA uses the basic reproduction operators such as crossover and mutation to produce the genetic composition of a population. The crossover operator produces two offspring's (new candidate solutions) by means of recombining the information from two parents. As mutation operation is a random alteration of some gene values in an individual, the allele of each gene is a candidate for mutation, and its function is determined by the mutation probability. Many efforts for the enhancement of traditional GAs have been proposed [14-16]. Among them, one category focuses on modifying the structure of the population or on the individual's role [11]. Some examples are distributed GA [11], cellular GA [7] and symbiotic GA. Another category is focused on modification/efficient control of the basic operations, such as crossover or mutation, of traditional GAs [17].

On the other hand, as natural selection tends to eliminate animals with poor foraging strategies through methods for locating, handling, and ingesting food, and to favor the propagation of genes of those animals that have successful foraging strategies, they are more likely to apply reproductive success to have an optimal solution [18,19]. After many generations, poor foraging strategies are either eliminated or shaped into good ones. Logically, such evolutionary principles have led scientists in the field of foraging

Manuscript received November 17, 2005; accepted June 26, 2006. Recommended by Editorial Board member Eun Tai Kim under the direction of Editor Jin Young Choi.

Dong Hwa Kim is with the School of Electrical, Electronic, Control & Instrumentation Engineering, Hanbat National University, San 16-1, Duckmyoung-dong, Yuseong-gu, Daejeon 305-716, Korea (e-mail: kimdh@hanbat.ac.kr).

Jae Hoon Cho is with the School of Electrical & Computer Engineering, Chungbuk National University, 12 Gaeshin-dong, Heungduk-gu, Chungbuk 361-763, Korea (e-mail: jhcho@hanbat.ac.kr).

theory to hypothesize that it is appropriate to model the activity of foraging as an optimization process. Since a foraging animal takes actions to maximize the energy obtained per unit time spent foraging, considering all the constraints presented by its own physiology such as, sensing and cognitive capabilities and environment (e.g., density of prey, risks from predators, and physical characteristics of the search area), evolution could lead to optimization and essentially could be applied to complex problem solving. These optimization models could provide a social foraging environment where groups of parameters communicate cooperatively for finding solutions to engineering problems.

Section 2 gives a brief literature overview of the area of bacterial foraging followed by the proposed approach based on BA (Bacterial Foraging) and GA (Genetic Algorithm). The proposed algorithm is validated using four test functions in Section 3 and the algorithm is illustrated for PID controller tuning in Section 4. Some conclusions are provided at the end.

2. HYBRID SYSTEM CONSISTING OF GENETIC ALGORITHM AND BACTERIA FORAGING

2.1. Genetic algorithms

In nature, evolution is mostly determined by natural selection or different individuals competing for resources in the environment. Superior individuals are more likely to survive and propagate their genetic material. The encoding for genetic information (genome) is done in a way that admits asexual reproduction which results in offspring that are genetically identical to the parent. Sexual reproduction allows some exchange and re-ordering of chromosomes, producing offspring that contain a combination of information from each parent. This is the recombination operation, which is often referred to as crossover because of the way strands of chromosomes cross over during the exchange. The diversity in the population is achieved by mutation. Genetic algorithms are ubiquitous nowadays, having been successfully applied to numerous problems from different domains, including optimization, automatic programming, machine learning, operations research, bioinformatics, and social systems [20]. A population of candidate solutions (for the optimization task to be solved) is initialized. New solutions are created by applying reproduction operators (mutation and/or crossover). The fitness (how good the solutions are) of the resulting solutions are evaluated and suitable selection strategy is then applied to determine which solutions will be maintained into the next generation. The procedure is then iterated.

2.2. Bacteria foraging algorithm

Search and optimal foraging decision-making of animals can be used for solving engineering problems. To perform social foraging an animal needs communication capabilities and it gains advantages that can exploit essentially the sensing capabilities of the group, so that the group can gang-up on larger prey, individuals can obtain protection from predators while in a group, and in a certain sense the group can forage a type of collective intelligence.

2.2.1 Over view of chemotactic behavior of E. coli.

This paper considers the foraging behavior of E. coli, which is a common type of bacteria [18,19]. Its behavior and movement comes from a set of six rigid spinning (100–200 r.p.s) flagella, each driven as a biological motor. An E. coli bacterium alternates through running and tumbling. Running speed is 10–20 $\mu m/s$, but they are unable to swim straight. We modeled the chemotactic actions of the bacteria as follows:

In a neutral medium, if it tumbles and runs in an alternating fashion, its action could be similar to search.

If swimming up a nutrient gradient (or out of noxious substances), or swimming for a longer period of time (climb up nutrient gradient or down noxious gradient),. its behavior seeks increasingly favorable environments.

If swimming down a nutrient gradient (or up noxious substance gradient), then the search action is avoiding unfavorable environments.

Subsequently, it can climb up nutrient hills and at the same time avoid noxious substances. The sensors it needs for optimal resolution are receptor proteins that are very sensitive and possess high gain. That is, a small change in the concentration of nutrients can cause a significant change in behavior. This is probably the best-understood sensory and decision-making system in biology.

Mutations in E. coli affect the reproductive efficiency at different temperatures, and occur at a rate of about 10^{-7} per gene per generation. E. coli occasionally engages in a conjugation that affects the characteristics of a population of bacteria. There are many types of taxis that are used in bacteria such as, aerotaxis (attracted to oxygen), phototaxis (light), thermotaxis (temperature), magnetotaxis (magnetic lines of flux) and some bacteria can change their shape and number of flagella based on the medium to reconfigure in order to ensure efficient foraging in a variety of media. Bacteria can form intricate stable spatio-temporal patterns in certain semisolid nutrient substances and they can radially eat their way through a medium if placed together initially at its center. Moreover, under certain conditions, they will secrete cell-to-cell attractant signals in order to group and protect each other.

2.2.2 Optimization function for the Hybrid GA-BF algorithm

The main goal of the Hybrid GA-BF based algorithm is to apply and find the minimum of $P(\phi)$, $\phi \in R^n$, not in the gradient $\nabla P(\phi)$. Here, when ϕ is the position of a bacterium, $P(\phi)$ is an attractant-repellant profile. That is, where nutrients and noxious substances are located, $P < 0$, $P=0$, $P > 0$ represent the presence of nutrients. A neutral medium, and the presence of noxious substances, respectively can be defined by

$$H(j,k,l) = \{\phi^x(j,k,l) \mid x = 1,2,...,N\}. \quad (1)$$

(1) represents the position of each member in the population of the N bacteria at the jth chemotactic step, kth reproduction step, and lth elimination-dispersal event. Let $P(x, j, k, l)$ denote the cost at the location of the ith bacterium $\phi^x(i,j,k) \in R^n$, and

$$\phi^x = (i+1,j,k) = \phi^x(i,j,k) + C((x)\varphi(i), \quad (2)$$

so that $C(x)>0$ is the size of the step taken in the random direction specified by the tumble. If at $\phi^x(i+1,j,k)$ the cost $P(x, j+1, k, l)$ is lower than at $\phi^x(i,j,k)$, then another chemotactic step of size $C(x)$ in this same direction will be taken and repeated up to a maximum number of steps Ns. Ns is the length of the lifetime of the bacteria measured by the number of chemotactic steps. Function $P_c^i(\phi)$, i=1, 2, ... , S, to model the cell-to-cell signaling via an attractant and a repellant is represented by [17-19,21]

$$\begin{aligned} P_c(\phi) &= \sum_{i=1}^{N} P_{cc}^i \quad (3) \\ &= \sum_{i=1}^{N}\left[-L_{attract} \quad \exp\left(-\delta_{attract}\sum(\phi_j - \phi_j^i)^2\right)\right] \\ &\quad + \sum_{i=1}^{N}\left[-K_{repeelarnt} \quad \exp\left(-\delta_{attract}\sum(\phi_j - \phi_j^i)^2\right)\right], \end{aligned}$$

where $\phi = [\phi_{1,...,}\phi_p]^T$ is a point on the optimization domain, Lattract is the depth of the attractant released by the cell and $\delta_{attract}$ is a measure of the width of the attractant signal. $K_{repellant} = L_{attract}$ is the height of the repellant effect magnitude, and $\delta_{attract}$ is a measure of the width of the repellant. The expression of $P_c(\phi)$ means that its value does not depend on the nutrient concentration at position ϕ. That is, a bacterium with high nutrient concentration secretes stronger attractant than one with low nutrient concentration. The model uses the function $P_{ar}(\phi)$ to represent the environment-dependent cell-to-cell signaling as

$$P_{ar}(\phi) = \exp\left(T - P(\phi)\right)P_c(\phi), \quad (3a)$$

where T is a tunable parameter. By considering minimization of $P(i, j, k, l) + P_{ar}\left(\phi^i(j,k,l)\right)$, the cells try to find nutrients, avoid noxious substances, and at the same time try to move toward other cells, but not too close to them. The function $P_{ar}\left(\phi^i(j,k,l)\right)$ implies that, with M being constant, the smaller the $P(\phi)$, the larger the $P_{ar}(\phi)$ and thus the stronger attraction, which is intuitively reasonable. In tuning the parameter M, it is normally found that, when M is very large, $P_{ar}(\phi)$ is much larger than $J(\phi)$, and thus the profile of the search space is dominated by the chemical attractant secreted by E. coli.

On the other hand, if T is very small, then Par (ϕ) is much smaller than $P(\phi)$, and it is the effect of the nutrients that dominates. In $P_{ar}(\phi)$, the scaling factor of $P_c(\phi)$ is given as in exponential form.

The algorithm to search optimal values of parameters is described as follows:

Step 1: Initialize parameters n, N, NC, NS, Nre, Ned, Ped, $C(i)(i=1,2,\ldots,N)$, ϕ^i,

where

n: Dimension of the search space

N: The number of bacteria in the population

NC: chemotactic steps

Nre: The number of reproduction steps

Ned: the number of elimination-dispersal events

Ped: elimination-dispersal with probability

$C(i)$: the size of the step taken in the random direction specified by the tumble.

Step 2: Elimination-dispersal loop: l=l+1

Step 3: Reproduction loop: k=k+1

Step 4: Chemotaxis loop: j=j+1

substep a: For i =1,2,…,N, take a chemotactic step for bacterium i as follows.

substep b: Compute fitness function, ITSE (i,j,k,l).

substep c: Let ITSElast=ITSE (i,j,k,l) to save this value since we may find a better cost via a run.

substep d: Tumble: generate a random vector $\Delta(i) \in R^n$ with each element $\Delta_m(i), m = 1,2,...,p$, a random number on [-1, 1].

substep e: Move: Let

$$\phi^x(i+1,j,k) = \phi^x(i,j,k) + C(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}.$$

This results in a step of size $C(i)$ in the direction of the tumble for bacterium *i*.

substep f: Compute ITSE $(i, j+1, k, l)$.

substep g: Swim

i) Let *m*=0 (counter for swim length).

ii) While $m < N_s$ (if not climbed down too long).

- Let *m*=*m*+1.
- If ITSE$(i, j+1, k, l)$ < ITSElast(if doing better), let ITSElast=ITSE$(i, j+1, k, l)$ and let

$$\phi^x(i+1, j, k) = \phi^x(i+1, j, k) + C(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$$

and use this $\phi^x(i+1, j, k)$ to compute the new ITSE$(i, j+1, k, l)$ as we did in substep f.

- Else, let m= N_s. This is the end of the while statement.

substep h: Go to next bacterium (*i*, 1) if $i \neq N$ (i.e., go to substep b to process the next bacterium).

Step 5: If $j < N_C$, go to step 3. In this case, continue chemotaxis, since the life of the bacteria is not over.

Step 6: Reproduction:

substep a: For the given *k* and l, and for each $i = 1, 2, ..., N$, let

$$ITSE^i_{health} = \sum_{j=1}^{N_c+1} ITSE(i, j, k, l)$$

be the health of bacterium i (a measure of how many nutrients it got over its lifetime and how successful it was at avoiding noxious substances). Sort bacteria and chemotactic parameters $C(i)$ in order of ascending cost $ITSE_{health}$ (higher cost means lower health).

substep b: The S_r bacteria with the highest $ITSE_{health}$ values die and the other S_r bacteria with the best values split (this process is performed by the copies that are at the same location as their parent).

Step 7: If $k < N_{re}$, go to step 3. In this case, we have not reached the number of specified reproduction steps, so we start the next generation in the chemotactic loop.

Step 8: Elimination-dispersal: For $i = 1, 2..., N$, with probability P_{ed}, eliminate and disperse each bacterium, and this results in maintaining the number of bacteria in the population constant.

To do this, if a bacterium is eliminated, simply disperse one to a random location on the optimization domain. If $l < N_{ed}$, then go to step 2 otherwise end.

3. SIMULATION USING TEST FUNCTIONS

This section illustrates some comparisons between the proposed GA-BF (Genetic Algorithms-Bacteria Foraging algorithm) and the conventional SGA (Simple Genetic Algorithm) using some test functions.

3.1. Mutation operation in GA-BF

Dynamic mutation [22] is used in the proposed GA-BF algorithm

$$x_j = \begin{cases} \tilde{x}_j + \Delta(k, x_j^{(U)} - \tilde{x}_j), & \tau = 0 \\ \tilde{x}_j - \Delta(k, \tilde{x}_j - x_j^{(L)}), & \tau = 1, \end{cases} \quad (4)$$

where random constant, τ becomes 0 or 1 and $\Delta(k, y)$ is given as

$$\Delta(k, y) = y \cdot \eta \cdot (1 - \frac{k}{z})^A. \quad (5)$$

Here, η has 0 or 1 randomly and z is the maximum number of generations as defined by the user.

3.2. Crossover operation in GA-BF

A modified simple crossover [22] is employed for the BF-GA algorithm using

$$\begin{aligned} \tilde{x}_j^u &= \lambda \bar{x}_j^v + (1-\lambda)\bar{x}_j^u, \\ \tilde{x}_j^v &= \lambda \bar{x}_j^u + (1-\lambda)\bar{x}_j^v, \end{aligned} \quad (6)$$

where $\bar{x}_j^u$, $\bar{x}_j^v$ refers to the parent's generations, $\tilde{x}_j^u$, $\tilde{x}_j^v$ refers to the offspring's generations, *j* is the chromosome of *j* th and λ is the multiplier.

3.3. Performance variation for different step sizes

Step size here refers to the moving distance per step of the bacteria. For performance comparison test function F_1 is used

$$F_1(x) = \sum_{i=1}^{3} x_i^2, \quad -5.12 \le x_1, x_2, x_3 \le 5.11. \quad (7)$$

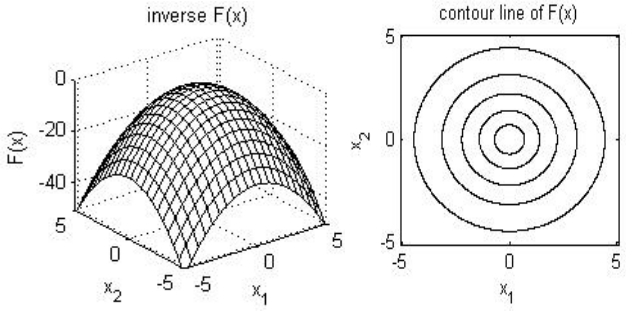
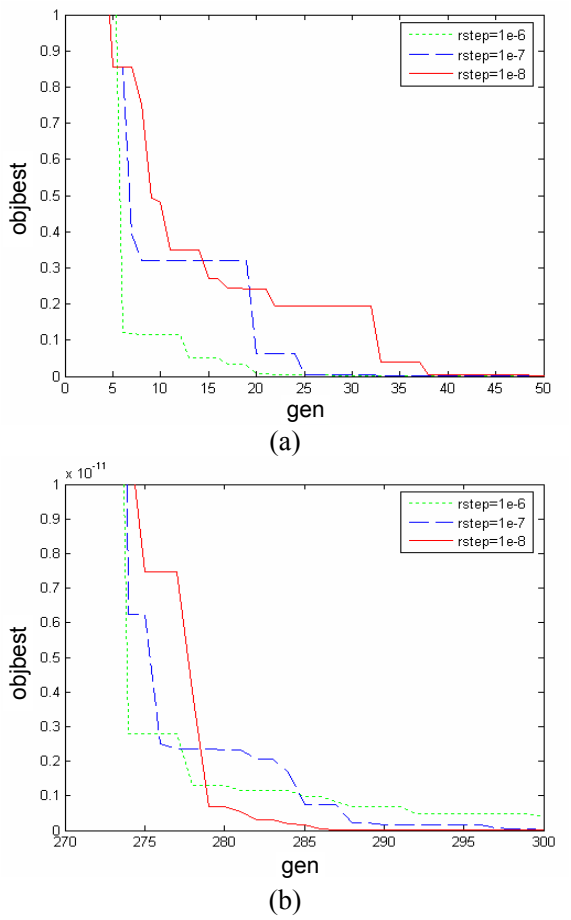


Fig. 1. Contour of test function F_1.



(a)

(b)

Fig. 2. (a) Performance value for the three different step sizes for the first 50 generations. (b) Performance value for the three different step sizes for generations 270-300.

Table 1. Parameter values for various step sizes.

Step size	x1	x2	x3	Optimal objective function	Average objective function
1.0e-6	3.87E-13	6.60E-13	2.92E-07	-5.43E-07	-8.98E-08
1.0e-7	2.85E-14	2.34E-13	-5.52E-08	1.50E-07	-5.45E-08
1.0e-8	5.01E-16	1.43E-15	-1.70E-08	-1.44E-08	-2.31E-09

Fig. 2(a) and (b) and Table 1 illustrate the performance of the GA-BF algorithm for the 300 generations. As evident from the results for bigger step size, the convergence is faster. Table 1 illustrates parameters of variables obtained by the step size of Fig. 5(a).

3.4. Performance for different chemotactic steps of GA-BF

Fig. 3 and Table 2 illustrate the relationship between objective function and the number of generations for different chemotactic steps. When the chemotactic step is smaller, the objective function converges faster.

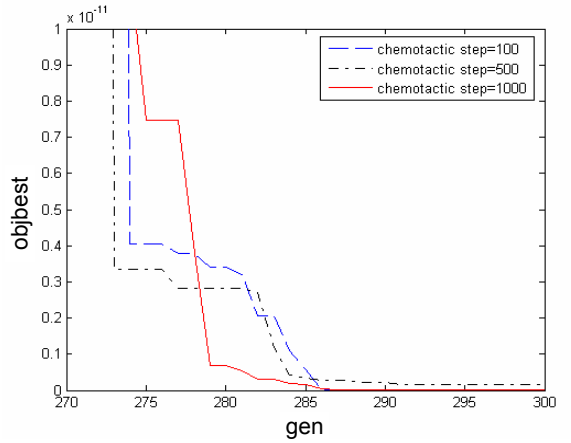


Fig. 3. Performance value for different chemotactic step for generations 270-300.

Table 2. Variation of objective function for different chemotactic steps.

Chemo. Step	x1	x2	x3	Optimal objective function	Average objective function
100	-9.32E-08	3.78E-07	-8.57E-09	1.52E-13	1.59E-13
500	2.97E-08	1.92E-08	2.32E-08	1.79E-15	3.26E-15
1000	-1.70E-08	-1.44E-08	-2.31E-09	5.01E-16	1.43E-15

3.5. Performance for different life time N

Fig. 4(a) and (b) illustrate the characteristics between objective function and the number of generations for different life time N of bacteria. Table 3 depicts some empirical results for a few more test functions showing the initial condition's variation of objective values, parameter values, chemotactic steps, total number of chemotactic reaction of bacteria, step sizes, basic unit for movement of bacteria, the number of critical reaction (N), the number of bacteria (S), generations (G), mutation (Mu), and crossover (Cr).

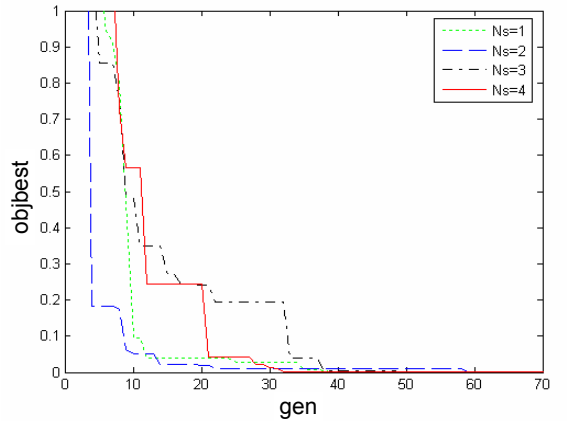
3.6. Performance of GA-BF for test functions

3.6.1 Test function: F_1

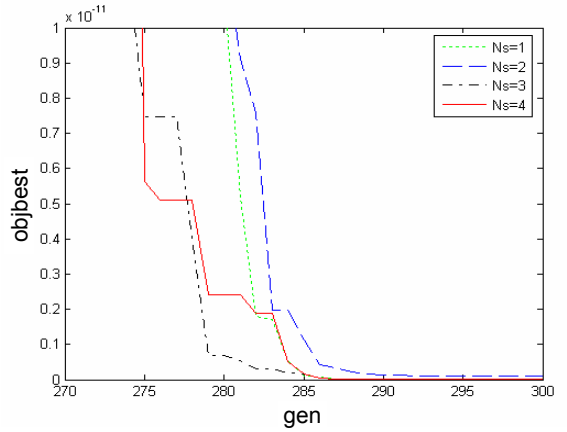
Fig. 5(a) and (c) show the performance comparison of GA and GA-BF with stepsize=1×10^{-5} for generations 70, 300. As evident from Fig. 5(a) and (c) the hybrid GA-BF approach could search the optimal solutions earlier (10 generations) compared to GA. Fig. 5(b) reveals that the GA-BF could converge faster than GA during the final few iterations.

Fig. 5(d) depicts how the parameters are optimized during the 27-300 generations by the characteristics of GA and GA-BF with different step size (stepsize =1×10^{-5}). Table 4 depicts the final parameter values obtained using GA and GA-BF algorithms. Fig. 5(e) represents the characteristic of optimal approach of variables on different 100 generations.

3.6.2 Test function: F_2



(a)



(b)

Fig. 4. (a) Performance value for different lifetime N for the first 70 generations. (b) Performance value for different lifetime N for generations 270-300.

Function $F_2(x)=100\left(x_1^2-x_2\right)^2+\left(1-x_1\right)^2$ is used to illustrate the performance of GA and GA-FA. Fig. 6(a) illustrates the contour of this function at $x=\begin{bmatrix}1 & 1\end{bmatrix}^T$. Fig. 6(b) represents the performance characteristics of the conventional GA and the GA-BF algorithm.

From these figures, it is evident that the proposed GA-BF algorithm converges to the optimal solution much faster than the conventional GA approach. Table 5 illustrates the various empirical results obtained using GA and GA-BF approaches.

3.6.3 Test function: F_3

Function $F_3=\sum_{i=1}^{5}[x_i]$ is applied to compare the performance of GA and GA-BF. This function has minimum -30 at

$$x=\begin{bmatrix}-5.12, & -5.12, & -5.12, & -5.12, & -5.12, \end{bmatrix}.$$

Fig. 7(a) illustrates the contour map for this function. Fig. 7(b)-(d) represent the various results obtained on the test function f_3 and Table 6 is for the values of variables.

3.6.4 Test function: F_4

Function $F_4=\sum_{i=1}^{30}ix_i^4+N(0,\ 1)$ is used to compare the conventional GA and the proposed system GA-BF. Fig. 8(a) illustrates the contour map of this function. Fig. 8(b)-(c), depict the performance by the GA and GA-BF method on different generation. Fig. 8(b), (c) illustrate that the proposed method converges faster than the conventional GA.

4. INTELLIGENT TUNING OF PID CONTROLLER FOR AUTOMATIC VOLTAGE REGULATOR (AVR) USING GA-BF APPROACH

The transfer function of the PID controller in the AVR system is given by

$$PID(s) = k_p+\frac{k_i}{s}+k_d s, \tag{8}$$

Table 3. Initial conditions of test function and variation of different parameters.

Test function	Range		Genetic Algorithm Parameters				Bacteria Foraging Parameters			
	$x_i^{(L)}$	$x_i^{(U)}$		G	Mu	Cr	CS	Step size	Ns	S
$F_1(x)=\sum_{i=1}^{3}x_i^2$	-5.12	5.11	20	300	0.9	0.1	1000	1e-007	3	10
$F_2(x)=100(x_1^2-x_2)^2+(1-x_1)^2$	-2.048	2.047	20	600	0.9	0.1	1000	1e-007	3	10
$F_3=\sum_{i=1}^{5}[x_i]$	-5.12	5.12	20	180	0.9	0.1	1000	1e-007	3	10
$F_4=\sum_{i=1}^{30}ix_i^4+N(0,1)$	-1.28	1.27	20	300	0.9	0.1	1000	1e-007	3	10

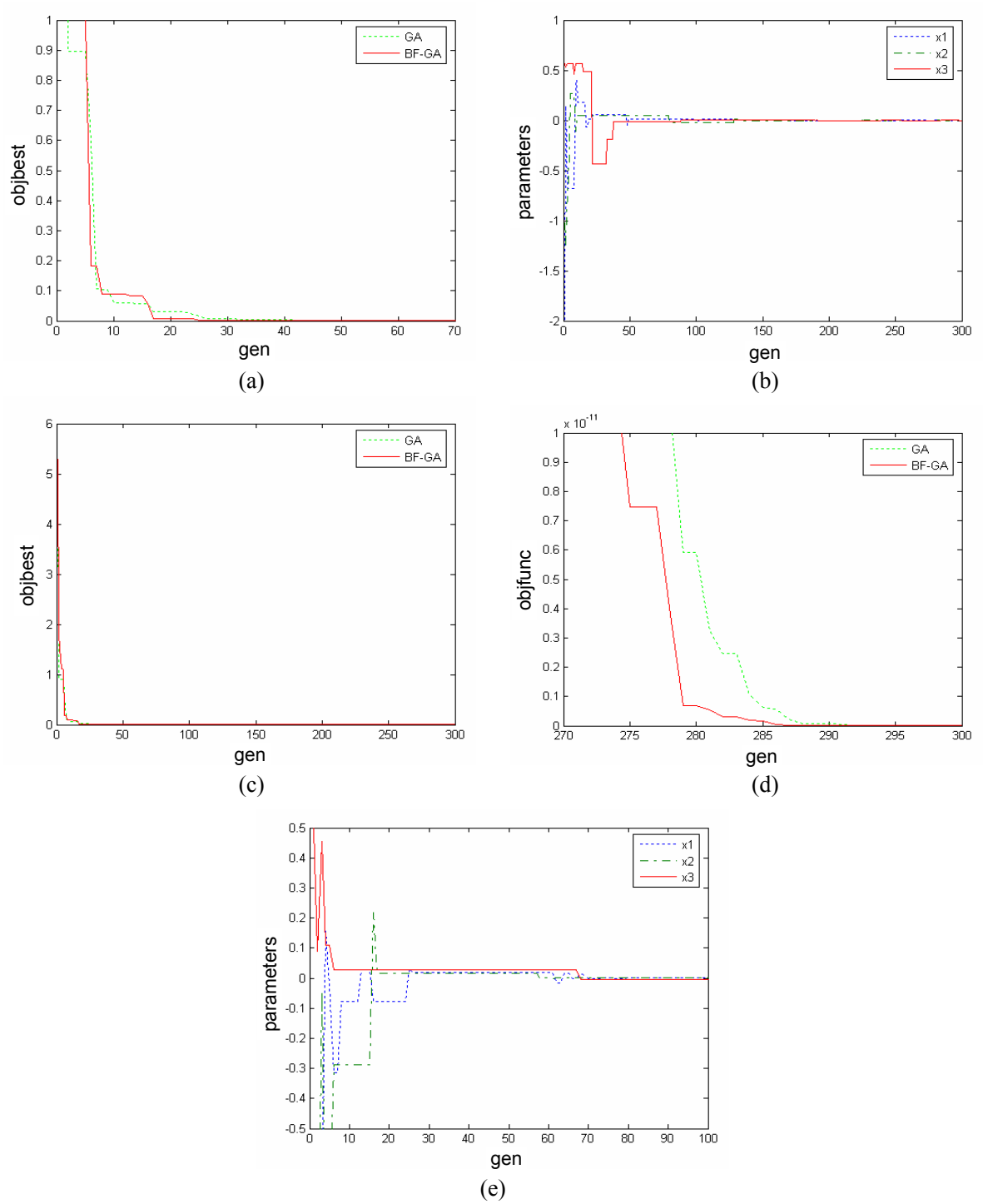


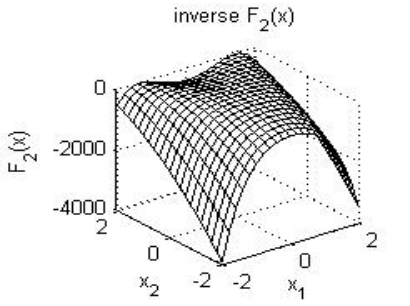
Fig. 5. (a) Convergence of GA and GA-BF for stepsize=1×10-5 during the first 70 generations. (b) Tuning of parameters during 70 generations. (c) Convergence of GA and GA-BF for stepsize=1×10-5 during 300 generations. (d) Performance of GA and GA-BF for stepsize=1×10-5 during generations 270-300. (e) Tuning of parameters for stepsize=1×10-5 during 100 generations.

Table 4. Performance of GA and GA-BF function *F1*.

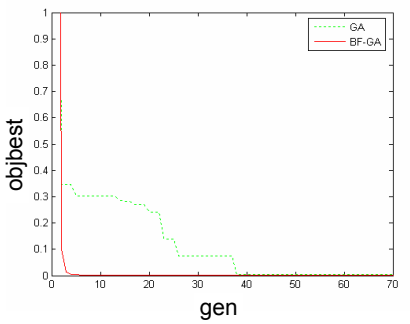
Method	x1	x2	x3	Optimal objective function	Average objective function
GA	7.22E-08	5.07E-08	-9.43E-09	7.87E-15	8.03E-15
GA-BF	-1.70E-08	-1.44E-08	-2.31E-09	5.01E-16	1.43E-15

Table 5. GA and GA-BF performance for function F_2.

Method	x1	x2	Optimal objective function	Average objective function
GA	0.001967	0.001967	1.0443267	1.0907699
BF-GA	5.12E-09	5.17E-09	0.9999285	0.9998567



(a)



(b)

Fig. 6. (a) Contour of test function (F2). (b) Performance of GA and GA-BF during the first 70 generations on test function F_2.

and the block diagram of the AVR system is shown in Fig. 9. The performance index of the control response is defined by

$$\begin{aligned}
&\min F\left(k_p, k_i, k_d\right) \\
&= \frac{e^{-\beta . t_s} / \max(t)}{\left(1-e^{-\beta}\right)\left|1-t_r / \max(t)\right|} + e^{-\beta} . M_o + ess \\
&= \frac{e^{-\beta}\left(t_s + \alpha_2 . \left\|1-t_r / \max(t)\right| . M_o \right|\right)}{\left(1-e^{-\beta}\right)\left|1-t_r / \max(t)\right|} + ess \\
&= \frac{e^{-\beta}\left(t_s / \max(t) + \alpha . Mo\right)}{\alpha} + ess,
\end{aligned} \tag{9}$$

$$\alpha = \left(1-e^{-\beta}\right) \cdot \left|1-t_r / \max(t)\right|,$$

k_p, k_i, k_d : Parameter of PID controller,

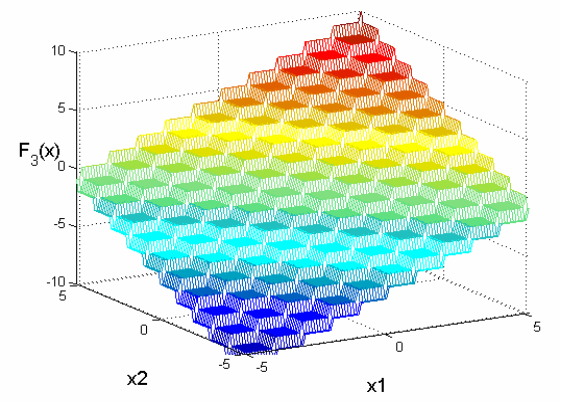
β : Weighting factor,

Mo : Overshoot,

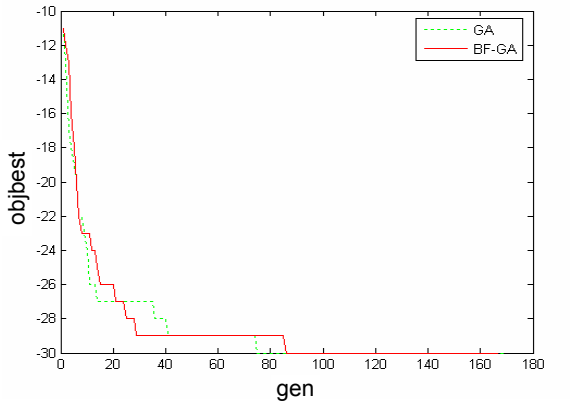
t_s : Settling time (2%),

ess : Steady-state error,

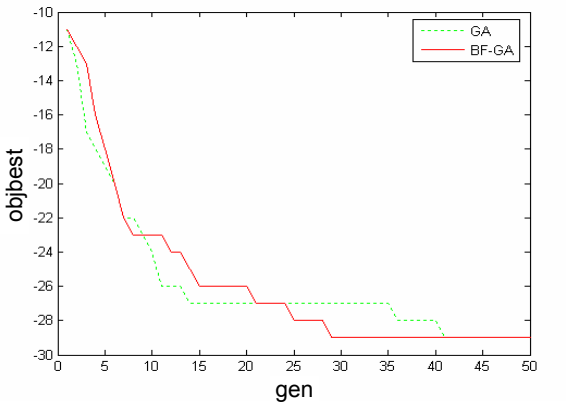
t : Desired settling time.



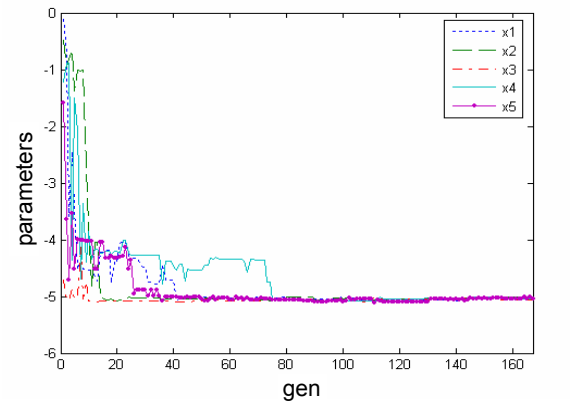
(a)



(b)



(c)



(d)

Fig. 7. (a) Contour map of test function F_3. (b) Performance of GA and GA-BF during the first 180 generations on test function F_3. (c) Performance of GA and GA-BF during the first 70 generations on test function F_3. (d) Tuning of parameters during 160 generations on test function F_3.

Table 6. Performance of GA and GA-BF for test function F_3.

Method	x1	x2	x3	x4	x5	Optimal objective function	Average objective function
GA	-5.024811	-5.015523	-5.059941	-5.03529	-5.03527	-30	-29.4
BF-GA	-5.111186	-5.097807	-5.089435	-5.06529	-5.06891	-30	-29.95

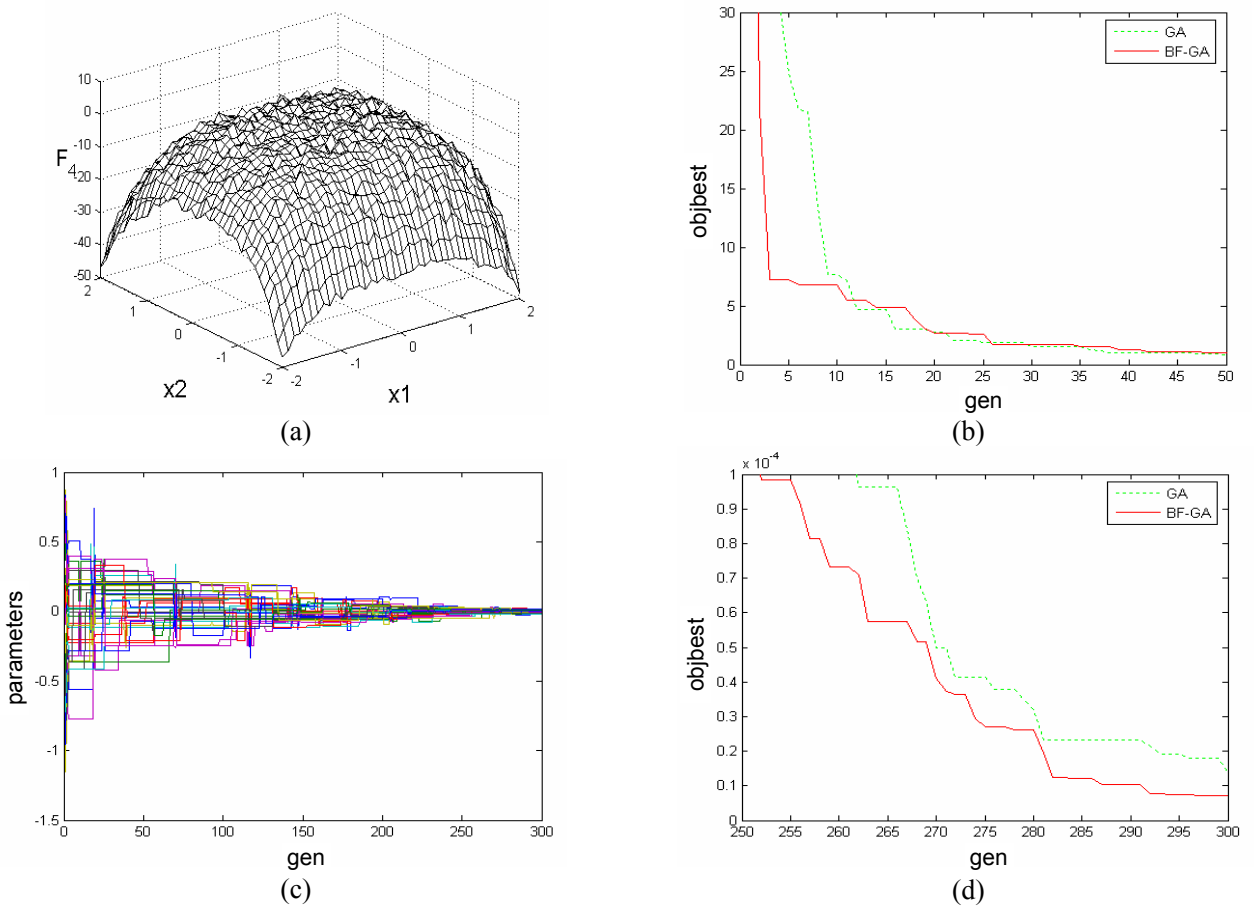


Fig. 8. (a) Contour map of test function F_4. (b) Performance of GA and GA-BF during the first 50 generations on test function F_4. (c) Performance of GA and GA-BF during generations 250-300 on test function. (d) Tuning of parameters during 300 generations on test function F_4.

In (9), if the weighting factor β, increases, rising time of response curve is small, and when β decreases, rising time is big. Performance criterion is defined as $Mo = 5061\%$, $ess = 0.0909$, $t_r = 0.2693(s)$, $t_s = 6.9834(s)$. Initial values of PID Controller and the GA-BF algorithm are depicted in Table 8 and Table 9,

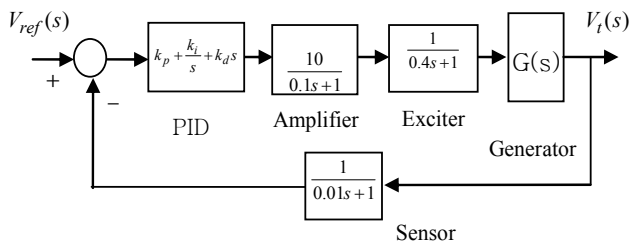


Fig. 9. Block diagram of an AVR system.

respectively. For comparison purposes, we also used a Particle Swarm Optimization (PSO) approach and a Hybrid GA - PSO approach [22-29].

The Particle Swarm Optimization (PSO) algorithm is mainly inspired by social behavior patterns of organisms that live and interact within large groups [25]. The standard PSO model consists of a swarm of particles, which are initialized with a population of random candidate solutions. They move iteratively through the d-dimension problem space to search the new solutions, where the fitness, f, can be calculated as certain qualities are measured. Each particle has a position represented by a position-vector $\vec{x}_i$ (i is the index of the particle), and a velocity represented by a velocity-vector $\vec{v}_i$. Each particle remembers its own best position so far in a vector $\vec{x}_i^{\#}$, and the j-th

Table 8. Range of PID parameters.

PID parameters	Range	
	Min	Max
k_p	0	1.5
k_i	0	1
k_d	0	1

Table 9. Parameters of BF-GA algorithm.

parameters	Values
Stepsize	0.08
Ns	4
Pc	0.9
Pm	0.65

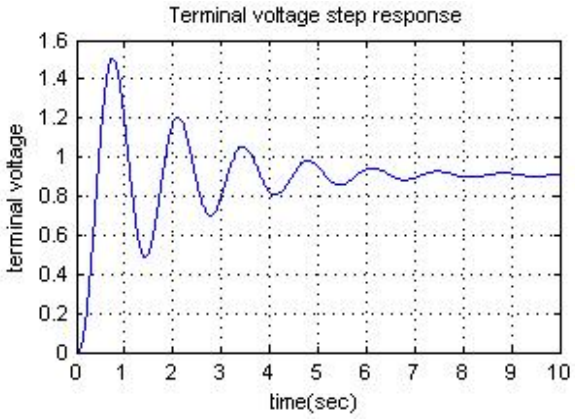


Fig. 10. Step response of terminal voltage in an AVR system without controller.

dimensional value of the vector $\vec{x}_i^{\#}$ is $x_{i,j}^{\#}$. The best position-vector among the swarm so far is then stored in a vector $\vec{x}^*$, and the j-th dimensional value of the vector $\vec{x}^*$ is x_j^*. During the iteration time t, the update of the velocity from the previous velocity to the new velocity is determined and then the new position is determined by the sum of the previous position and the new velocity. The conventional PSO algorithm was used for controlling the mutation process of the genetic algorithm (GA), as an attempt to improve the GA learning efficiency. The architecture and flow chart of the proposed method are given in [26]. Euclidean distance is used for selecting crossover parents (in the hybrid GA-PSO approach) to avoid local optima and to obtain fast solutions.

Fig. 10 illustrates the response of terminal voltage to a step input in the control system. Figs. 11-14 represent results obtained by the GA and GA-BF algorithm for the variation of β for 200 generations as per (9). Empirical results show satisfactory learning. Figs. 15-17 illustrate the search process for optimal parameters for the variation of β (=0.5, 1.0, and 1.5)

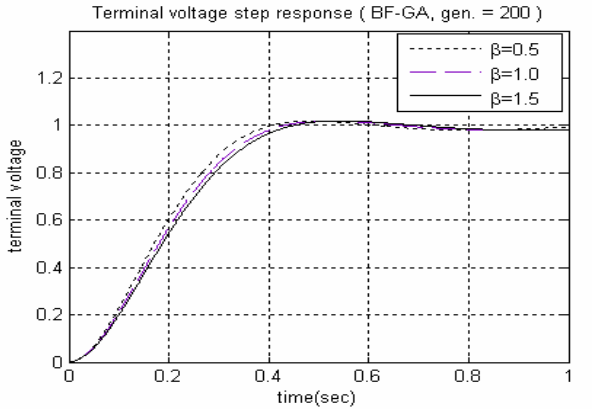


Fig. 11. Terminal voltage step response of an AVR system using BF-GA algorithm.

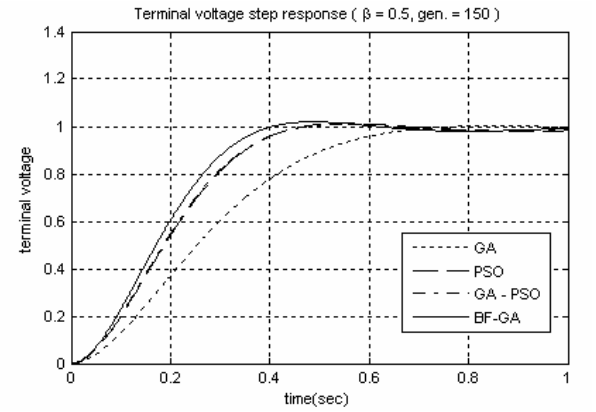


Fig. 12. Terminal voltage step response of an AVR system with different controllers ($\beta = 0.5$, generations=200).

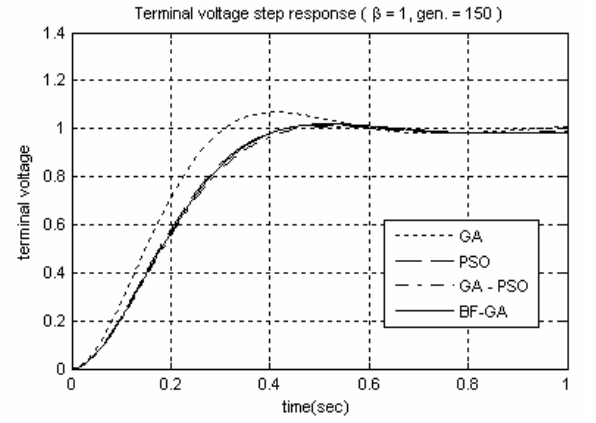


Fig. 13. Terminal voltage step response of an AVR system with different controllers ($\beta = 1.0$, generations=200).

by GA-BF approach. Table 10 depicts the best solution using BF-GA controller for different β values and Table 11 illustrates a performance comparison of the values evaluated using different methods (β =1.5, 200 generations).

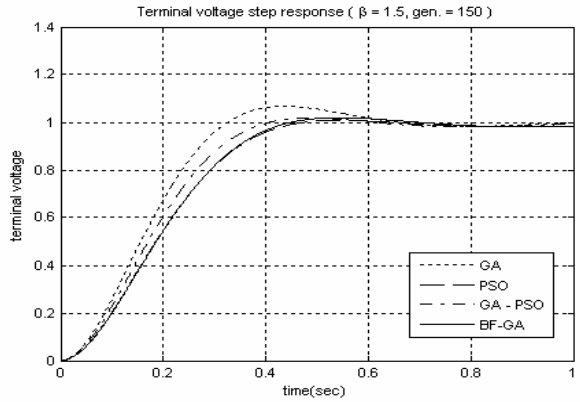


Fig. 14. Terminal voltage step response of an AVR system with different controllers ($\beta = 1.5$, generations=200).

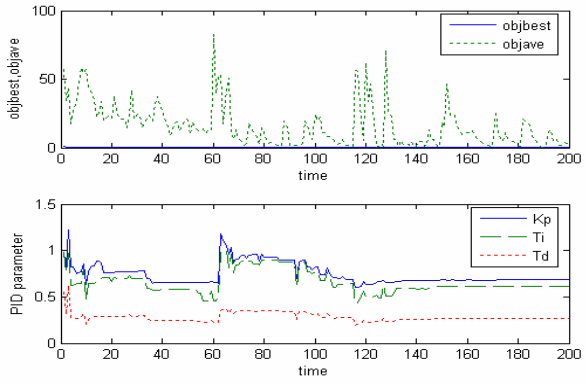


Fig. 16. Search process for optimal parameter values of an AVR system by GA-BF method for $\beta = 1.0$.

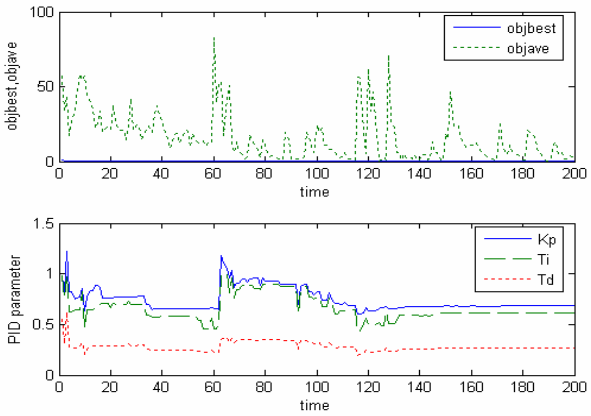


Fig. 15. Search process for optimal parameter values of an AVR system by the GA-BF method for $\beta = 0.5$.

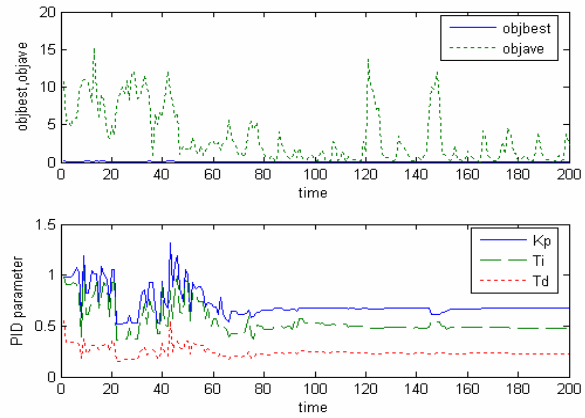


Fig. 17. Search process for optimal parameter values of an AVR system by GA-BF method for $\beta = 1.5$.

Table 10. Best solution obtained using BF-GA controller with different β values.

β	Number of generation	k_p	k_i	k_d	Mo (%)	ess	t_s	t_r	Evaluation value
0.5	200	0.68233	0.6138	0.26782	1.94	0.0171	0.3770	0.2522	0.3614
1	200	0.68002	0.52212	0.24401	1.97	0.0067	0.4010	0.2684	0.1487
1.5	200	0.67278	0.47869	0.22987	1.97	0.0014	0.4180	0.2795	0.07562

Table 11. Comparison of the objective value using different methods (β =1.5, generation=200).

β	Methods	k_p	k_i	k_d	Mo (%)	ess	t_s	t_r	Evaluation value
1.5	GA	0.8282	0.7143	0.3010	6.7122	0.0112	0.5950	0.2156	0.0135
	PSO	0.6445	0.5043	0.2348	0.8399	0.0084	0.4300	0.2827	0.0073
	GA-PSO	0.6794	0.6167	0.2681	1.8540	0.0178	0.8000	0.2526	0.0071
	BF-GA	0.6728	0.4787	0.2299	1.97	0.0014	0.4180	0.2795	0.0756

5. CONCLUSIONS

Recently, many variants of genetic algorithms for improving the learning related to control engineering have been investigated. The general problem of evolutionary algorithm based engineering system design has been tackled in various ways mainly from a convergence perspective and finding local or suboptimal solutions. The GA has also been used to optimize nonlinear system strategies. Among some

other methods, a large amount of research is focused on the design of fuzzy controllers using evolutionary algorithm approaches. The GA could be used for developing the knowledge base in the form of linguistic rules and the fine tuning of fuzzy membership functions, fuzzy operators, etc. In all these situations, there could be problems with local optimization or suboptimal solutions.

This paper proposed a novel hybrid approach consisting of GA (Genetic Algorithm) and BF (Bacterial Foraging) and the performance is illustrated using various test functions. Also, the proposed GA-BF algorithm is used for tuning a PID controller of the AVR system. From Figs. 2-8 of test functions $F_1 - F_4$, the suggested hybrid system GA-BF has the better performance obtaining the optimal parameter simultaneously. We applied the suggested hybrid system GA-BF to the AVR system of Fig. 9, and the resulting Figs. 11-14 show the comparison of the performance obtained by each approach.

The proposed approach has the potential to be useful in other practical optimization problems (e.g., engineering design, online distributed optimization in distributed computing and cooperative control) as social foraging models work very well for distributed non-gradient optimization methods. Other species of bacteria or biological based computing approaches could be studied but it depends on how practically useful these optimization algorithms are for engineering optimization problems, because they depend on the theoretical properties of the algorithm, theoretical and empirical comparisons to other methods, and extensive evaluation on many benchmark problems and real-world problems.

REFERENCES

[1] C.-L. Lin and H.-W. Su, "Intelligent control theory in guidance and control system design: An overview," *Proc. of Natul. Sci., Counc. ROC(A)*, vol. 24, no. 1, pp. 15-30, 2000.

[2] P. J. Fleming and R. C. Purshouse, "Evolutionary algorithms in control system engineering: A survey," *Control Eng. Practice*, vol. 10, pp. 1223-1241, 2002.

[3] M. Dotoli, G. Maione, D. Naso, and E. B. Turchiano, "Genetic identification of dynamical systems with static nonlinearities," *Proc. of IEEE SMCia/01, Mountain Workshop Soft Computing Industrial Applications*, Blacksburg, VA, pp. 65-70, June 25-27, 2001.

[4] G. J. Gray, D. J. Murray-Smith, Y. Li, K. C. Sharman, and T. Weinbrenner, "Nonlinear model structure identification using genetic programming," *Contr. Eng. Practice*, vol. 6, no. 11, pp. 1341-1352, 1998.

[5] K. Kristinnson and G. A. Dumont, "System identification and control using genetic algorithms," *IEEE Trans. System, Man, Cybern.*, vol. 22, pp. 1033-1046, Sept.-Oct. 1992.

[6] B. Maione, D. Naso, and B. Turchiano, "GARA: A genetic algorithm with resolution adaptation for solving system identification problems," *Proc. of Eur. Control Conf.*, Porto, Portugal, pp. 3570-3575, Sept. 4-7, 2001.

[7] C. M. Fonseca and P. J. Fleming, "Multi-objective optimization and multiple constraint handling with evolutionary algorithms-Part I: A unified formulation", "-Part II: Application example," *IEEE Trans. System, Man, Cybern. A: Systems and Humans*, vol. 28, no. 1, pp. 26-47, Jan. 1998.

[8] Z. Michalewicz, *Genetic Algorithms+Data Structures=Evolution Programs*, Springer-Verlag, New York, 1999.

[9] J. Arabas, Z. Michalewicz, and J. Mulawka, "GAVaPS-A genetic algorithm with varying population size," *Proc. of IEEE Int. Conf. on Evolutionary Computation*, Orlando, pp. 73-78, 1994.

[10] R. Tanese, "Distributed genetic algorithm," *Proc. of Int. Conf. Genetic Algorithms*, pp. 434-439, 1989.

[11] R. J. Collins and D. R. Jefferson, "Selection in massively parallel genetic algorithms," *Proc. of the Fourth Intl. Conf. on Genetic Algorithms*, pp. 249-256, 1991.

[12] S. Tsutsui and D. E. Goldberg, "Simplex crossover and linkage identification: Single-stage evolution vs. multi-stage evolution," *Proc. of IEEE Int. Conf. Evolutionary Computation*, pp. 974-979, 2002.

[13] H. Yoshida, K. Kawata, and Y. Fukuyama, "A particle swarm optimization for reactive power and voltage control considering voltage security assessment," *IEEE Trans. Power Syst.*, vol. 15, pp. 1232-1239, November 2000.

[14] J. Alcock, *Animal Behavior: An Evolutionary Approach*, Sinauer Associates, Sunderland, Massachusetts, 1998.

[15] W. J. Bell, *Searching Behavior: The Behavioral Ecology of Finding Resources*, Chapman and Hall, London, England, 1991.

[16] D. Grunbaum, "Schooling as a strategy for taxis in a noisy environment," *Evolutionary Ecology*, vol. 12, pp. 503-522, 1998.

[17] C.-F. Juang, "A hybrid of genetic algorithm and particle swarm optimization for recurrent network design," *IEEE Trans. on Systems, Man and Cybernetics, Part B*, vol. 34, no. 2, pp. 997-1006, April 2004.

[18] K. M. Passino, *Biomimicry of Bacterial Foraging for Distributed Optimization*, University Press, Princeton, New Jersey, 2001.

[19] K. M. Passino, "Biomimicry of bacterial forag-

ing for distributed optimization and control," *IEEE Control Systems Magazine*, vol. 22, no. 3, pp. 52-67, 2002.
[20] D. H. Kim, "Intelligent PID controller tuning of AVR system using GA and PSO," *Proc. of IEEE Intelligent Computing, Lecture Notes in Computer Science Proceeding of Springer*, Hefei, China, Aug. 23-26, 2005.
[21] D. W. Stephens and J. R. Krebs, *Foraging Theory*, Princeton University Press, Princeton, New Jersey, 1986.
[22] D. H. Kim and J. I. Park, "Loss minimization control of induction motor using GA-PSO," *Lecture Notes in Computer Science Proceeding of Springer* (*KES 2005*), Melbourne, Australia, Sep. 12-15, 2005.
[23] D. H. Kim and J. I. Park, "Intelligent tuning of PID controller for AVR system using a hybrid GA-PSO approach," *Lecture Notes in Computer Science Proceeding of Springer*, Atlanta, May 12-15, 2005.
[24] D. H. Kim and J. I. Park, "Improvement of genetic algorithm using PSO and euclidean data distance," *Proc. of IEEE Intelligent Computing, Lecture Notes in Computer Science Proceeding of Springer*, Aug. 23-26, China.
[25] D. H. Kim and J. H. Cho, "Intelligent control of AVR system using GA-BF," *Lecture Notes in Computer Science Proceeding of Springer Melbourne, Australia* (*KES 2005*), June 12-15, 2005.
[26] J.-O. Krah and J. Holtz, "High performance current regulation and efficient PWM implementation for low-inductance servo motors," *IEEE Trans. Ind. App*., vol. 35, pp. 1039-1049, Sep./Oct. 1999.
[27] D. H. Kim, "Robust tuning of PID controllers with disturbance rejection using bacterial foraging based optimization," *WSEAS Trans. on Systems*, vol. 3, no. 9, pp. 2834-2840, 2004.
[28] D. H. Kim, "Robust tuning of embedded intelligent PID controller for induction motor using bacterial foraging based optimization," *Lecture Notes in Computer Science Proceeding of Springer, ICESS2004*, Zhejiang University, Hanzhou, China, Dec. 9-10, 2004.
[29] D. H. Kim and J. H. Cho, "Adaptive tuning of PID controller for multivariable system using bacterial foraging based optimization," *Lecture Notes in Computer Science Proceeding of Springer*, Poland, June 12-15, 2005.

Dong Hwa Kim received the Ph.D. degrees in Electronic Engineering from Ahjou University in 1991, Korea and TIT (Tokyo Institute of Technology), Japan. He is now with Hanbat National University in Daejeon and is charging of Science Culture Research Institute of Korea Science Foundation, Seoul in Korea. His work history and experience include KAERI (Korea Atomic Energy Research Institute, 1977-1991) and AECL (Canada, 1985.11-1986.11) in computer aided multivariable control system design. He also has experience in ANL (1988. 9-1988.12) and with the University of Alberta, Canada (2000.3-2001.3) as a Visiting Professor. He is a member of IEEE (Fuzzy, Neural network, Evolutionary, Industrial application, Control technology, Automatic control, System and man, cybernetic), ISA, IEEJ, JROS, SOFT, INNS, SICE. ITEP, KASAS, ICASE, KES2005 (Special session), Sep. 12-15, Australia. IASTED2005-ACIT (Special session), He has been part of many committees for conferences including ANNIE04, ISCIA 2004, AFSS2004, ISCIIA2004, ICS2004. June 20-24, Russia, and IASTED2005, and ISCIIA2007. He was awarded best paper in KES2004, New Zealand and won the best research prize in 2006 at Hanbat National University. He is currently interested in intelligent control in automatic systems, artificial intelligence control, neural network control, fuzzy systems and control, artificial immune control and its application, computational intelligence, natural computing intelligence, bacteria foraging, particle swarm optimization, hybrid intelligent system, and intelligence based industrial control.

Jae-Hoon Cho received the M.S. degree in Control & Instrumentation Engineering, Hanbat National University, Daejeon, Korea. He is currently working toward a Ph.D. degree in Electrical & Computer Engineering. His interests are evolutionary computation, swarm intelligence, pattern recognition, neural network, and bioinformatics.