Volume 3, Issue 10, October 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

An Improved Genetic Algorithm with Variable Probability and Immune Genetic Algorithm

Abhishek*
Student ICE(EEE)&MDU
India

Md.Fahim Ansari
Prof & HOD EEE in BRCM College Bhal
India

Abstract— Genetic algorithms are computer based search techniques patterned after the genetic mechanism of biological organisms. This paper presents an improved genetic algorithm, which combines the concept of immunity, variable probabilities and multi objective optimization. PID parameter optimization is an important problem in control field. This improved genetic algorithm is used in PID parameter optimization. The algorithms are simulated with MATLAB programming. The simulation result shows that the PID controller with improved genetic algorithm shows better performance in comparison with canonical GAs

Keywords— Genetic Algorithms, Immune Genetic Algorithms, Improved Genetic Algorithms, Parameter Optimization, PID Parameter Tuning, Variable Probability

I. INTRODUCTION

Genetic Algorithms or in short GAs are stochastic algorithm based on principles of natural selection and genetics [1, 2]. These robust genetic algorithms have been successfully applied to problems in a variety of fields of study, and their popularity continues to increase due to their effectiveness, their applicability and their ease of use [2, 3]. However it is seen that GA does not always guarantee an optimum solution [8,9]. Keeping this fact in mind in this paper, an improved version of canonical genetic algorithm is introduced in order to avoid the uncertainty in the final solution. Concept of immunity with elitism [9], variable probability functions [10] and multi objective functions [3] are integrated to make GA work faster and better. Concept of immunity helps in convergence to optimum solution. Due to variable probabilities as GA moves towards a better solution, number of crossovers and number of mutations are decremented to reduce the simulation time. A multi-objective fitness function which simultaneously maximizes different objectives is included to give better solution than other performance indices. Simulation results indicate that the improved GA overcomes many of the difficulties associated with canonical GA and conventional tuning methods.

II. IMPROVED GENETIC ALGORITHMS

In order to solve the problem of not converging to an optimum solution, the advantages of immune genetic algorithm (to converge to optimum or near optimum), variable probabilities (to make genetic faster) and multi-objective function (to solve various objectives) are combined to make some refinements in GA

A. Immune Genetic Algorithms

The idea of immunity is mainly realized through two steps i.e. antibody selection and antibody injection. Each individual is compared with antibody and if it does not satisfy the criterion antibody injection is done.
Assume that v = {v1, v2… vn} is the antibody and w = {w1, w2… wn} is any individual in a given y population. Their root mean square (rms) difference is represented by d (v, w) and their fitness values are represented by fv and fw respectively. Assume that ε is a small positive constant. If

$$J(v,w) = \left(d(v,w) + \alpha \left| f_v - f_w \right|\right) \leq \in \tag{1}$$

is satisfied, it means that the individual has passed the immune test clearly and that can participate in next generation .α is a tunable parameter that controls the relative importance of the fitness difference $|fv-fw|$ versus the rms difference d (v, w) in J(v, w). d (v, w) reflects the similarity of the two antibodies in their structures and $|fv-fw|$ reflects the similarity of the two antibodies in their performance or qualities. The rms difference between the antibodies v and w is defined as

$$d(v,w) = \sqrt{\left\{\sum_{i=1}^{n}\left(v_i - w_i\right)^2\right\}/n} \tag{2}$$

Where n is the number of bits or genes in the individual

B. Variable Probabilities

Crossover rate Pc and mutation rate Pm decide the performance of GA. In the conventional mechanism, the crossover rate and mutation rate are held constants. We use the adaptive probabilities of crossover and mutation mechanism.

$$P_c = P_{c_1}(1-\frac{m}{M}) \quad (3)$$

$$P_m = P_{m_1}(1-\frac{m}{M}) \quad (4)$$

Where, m is evolution generation, M is the total generation, Pc1 is 0.6- 0.8. & Pm1 is 0.1.

The advantage of this method is that initially as we are starting from a random point, we need to cover a large search space but as number of generation is increasing and we are moving towards a better solution we can decrease probabilities and hence number of crossovers and mutations to make genetic search faster and better

C. Multi-objective Fitness Function

For the genetic implementation of a PID controller settling time, rise time and peak over shoot can also be a performance criterion [19]. So we can a make an objective function which takes into account these three objectives.

$$M.O = w_1 * m_p + w_2 * t_s + w_3 * t_R \quad (5)$$

Where mp is the maximum overshoot ts is the settling time, tr is the rising time, w1, w2 and w3 are the corresponding weight such that $w_i \in [0,1]$ and $\sum_{i=1}^{k} w_i = 1$.

Fitness function will be

$$fitness\ \ function = \frac{1}{MO} \quad (6)$$

III. DESIGNING OF PID USING IMPROVED GENETIC ALGORITHM

A PID Controller is designed using Improved GA in this section. The P, I and D gains are used to create a PID controller according to the equation below.

$$C_{PID} = \frac{K_D s^2 + K_p s + K_I}{s} \quad (7)$$

Fig 1 below shows the block diagram of the controlled system

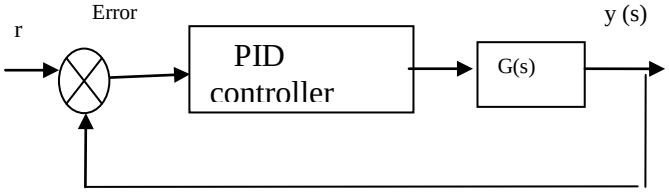

Fig 1. Block diagram of the controlled system

Complete algorithm

Step 1: Define the population size m, the maximum iteration number M, number of variables, number of bits, crossover probability, mutation probability etc;

Step 2: Based on the property and application condition of the given controlled object, determine the ranges and numerical precisions of Kp, Ki, and Kd, respectively.

Step 3: Randomly generate m individual to constitute the initial population

Step 4: Set the iteration counter t=1.

Step 5: For each individual k (k=1-m):

(a) Determine the PID gains Kp, Ki, and Kd,

(b) Send these gains into the PID controller and perform a simulation experiment on the controlled system using the unit step signal as the system input, and then calculate fitness.

(c) Determine the individual which has the largest fitness in the population and save it as antibody.

Step 6: Calculate average fitness and hence relative and cumulative fitness.

Step 7: Do the roulette wheel selection

Step 8: Perform crossover & mutation operations based on Probability function eq. (3) & (4).

Step 9: For each individual k repeat step 5 and Determine the individual which has the largest fitness in the population and save it as present antibody if this antibody fitness is larger than previous antibody replace the antibody.

Step 10: Replace population with this new population
Step 11: t←t+1. If t≤ M, return to Step 5; otherwise, output the optimal PID gains Kp, Ki, and Kd
Step 12: Plot step response of plant with optimum value of Kp, Ki, and Kd.

IV. SIMULATION & RESULTS

Improved genetic algorithm is tested for a third order system. System taken for analysis is

$$G(s) = \frac{1}{[s(s+1)(s+5)]} \tag{8}$$

Tool used is MATLAB 7.0.4. Parameters taken for analysis are listed in Table 1

Table 1. List of system parameters

Parameter	Value	Parameter	Value
Population size	30	Lower limits of gain	25
Maximum no. of generation	50	$\in$	10
Mutation probability (Pm1)	0.1	α	1
Crossover probability (Pc1)	0.66	w1	0.6
Number of bits per variable	5	w2	0.3
Upper limit of gains	25	w3	0.1

A. Step Response

Fig 3 shows the Z-N response , Fig 4 shows the response with classical GA using MSE criterion Fig 5. shows the step response with improved GA and Fig 6 shows the graph of Kp, Ki, and Kd

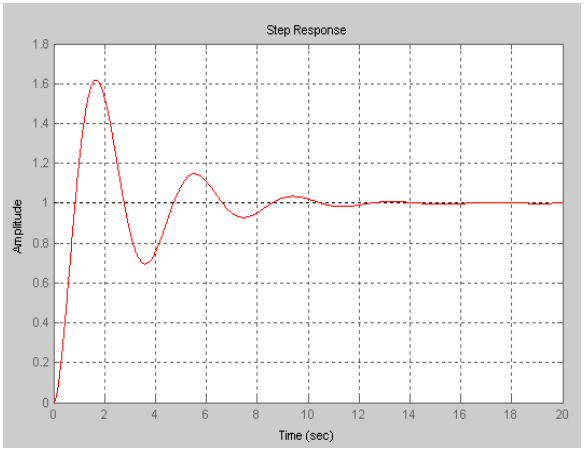

Fig 3. Z-N response

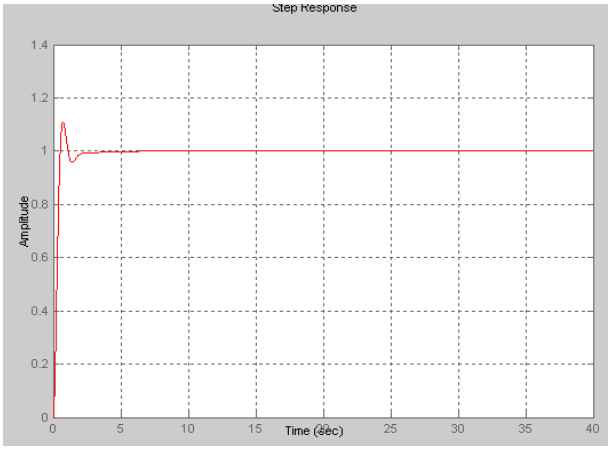

Fig 4. Step Response with canonical GA

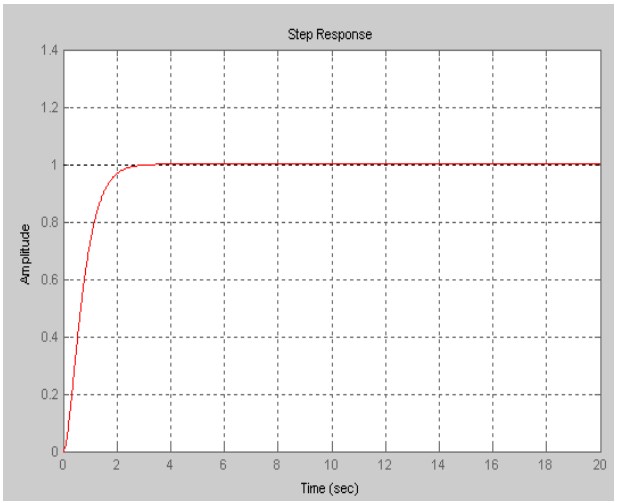

Fig 5. Step Response with improved GA

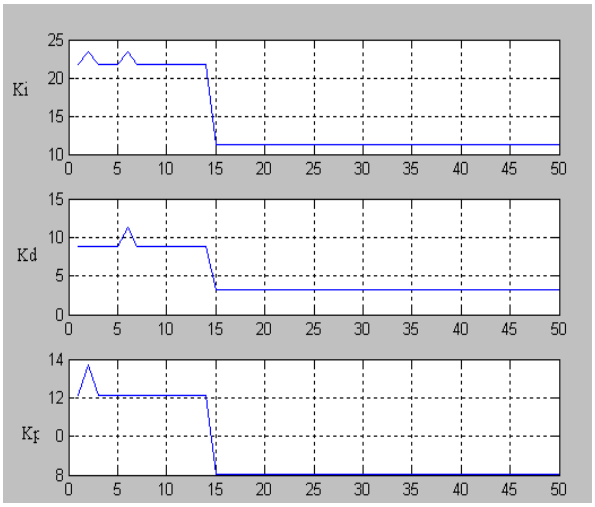

Fig 6. Plot of variables (Kp, Ki and Kd)

Table 2 shows the results for these simulations.

Table 2 : Simulation results

Method	Mp	Ts	Tr	Kp	Kd	Ki
ZN	1.62	10	.577	6.322	17.99	12.8
GA (MSE)	1.11	1.85	.442	1.85	2.5	0.1
Improved GA(MO)	0	1.2	1.46	6.451	7.258	0.1

B. Calculation of computation time

Improved GA because of its variable probabilities takes comparatively lesser simulation time. Table 3 shows the comparison data of the computation time of canonical GA and improved GA.

Table 3. Comparison of computation time

Algorithms	Average CPU time per iteration (sec)
Canonical GA	2.5
Improved GA	1.5

Improved GA reduces the computation time by 40%, reason being reduction in number of crossovers and number of mutation with increase in iteration counts.

C. Results

- Improved GA is finally converging to optimum or near optimum solutions.
- With improved GA peak overshoot as compared to classical GA is reduced to zero.
- Settling time is also reduced to zero.
- Rise time can be reduced by adjusting the weight criterion

V. CONCLUSION

This paper presents an improved genetic algorithm for PID control and its inherent advantages. The responses as shown in previous section show that the designed PID with improved GA has much better response than using the canonical GA method. The classical method is good for giving us as the starting point of what are the PID values. The improved GA designed PID is much better in terms of the overshoot and settling time and with the concept of immunity it does converge to an optimum or near optimum solution. The concept of variable probabilities also decreases the processing time. This improved GA is tested for a third order system and a time delay system. It is seen that it gives optimum or near optimum values for both system. Consequently, the improved genetic algorithms are better than simple genetic algorithms and they are feasible and effective optimization algorithms

REFERENCES

[1] Holland, J.J. Adaptation in Natural and Artificial Systems, University of Michigan Press. I975. Goldberg, David E., Genetic Algorithms in Search, Optimization and Machine Learning. Addison- Wesley Pub. Co. 1989.
[2] Michalewicz Z , Genetic algorithms + Data structure = Evolution Programs., Springer Publications., 1999
[3] Ogata K, Modern Control Engineering, 4th edition, Pearson publications,2002
[4] Visioli Antonio, Practical PID Control, SPRINGER Publication, 2006.
[5] Kiam Heong Ang, Gregory Chong , and Yun Li, "PID Control System Analysis, Design, and Technology" IEEE Transactions On Control Systems Technology, Vol. 13, No. 4, July 2005
[6] Kwok D.P., Wang P. "Fine tuning of classical PID Controller" Emerging Technologies and Factory Automation, IEEE, 1992.
[7] O' Mohanty, T., Downing,C.J. and Klaudiuz, F., 'Genetic Algorithms for PID Parameter Optimisation: Minimising Error Criteria' [online], URL: http://www.pwr.wroc.pl/~i-8zas/kf_glas00.pdf
[8] Licheng Jiao ,"A novel genetic algorithm based on immunity", IEEE Trans. on systems, man, and cybernetics—part a: systems and humans, vol. 30, no. 5, September 2000.
[9] Guanzheng Tan, Bin Jiang, Liming Yang. "A Novel Immune Genetic Algorithm-Based PID Controller Design and Its Application to CIP-I Intelligent Leg" Third International Conference on Natural Computation (ICNC 2007).IEEE
[10] Xiangzhong Meng, Baoye Song. "Fast Genetic Algorithms Used for PID Parameter Optimization",Proceedings of the IEEE International Conference on Automation and Logistics, 2007, China
[11] Andri Mirzal, Shinichiro Yoshii, Masashi Furukawa, "PID Parameters Optimization by Using Genetic Algorithm", ISTE Journal Of Science And Technology Policy, 2012
[12] Toledo,C.F.M ,Lima,J.M.G;da silva Arantes." A multi-population genetic algorithm approach for PID controller auto-tuning" Emerging Technologies & Factory Automation (ETFA), IEEE 17th Conference ,2012,Krakow

