A Variable-Based Genetic Algorithm

Kuang-Tsang Jean^{1,2} and Yung-Yaw Chen¹

¹Correspondence Address: Department of Electric Engineering, National Taiwan University, Taipei, Taiwan, R.O.C.. E-mail: jean@ipmc.ee.ntu.edu.tw

²Telecommunication Laboratories, Ministry of Transportation and Communications, Taiwan, R.O.C.

ABSTRACT

Genetic algorithms are very powerful search algorithms based on the mechanics of natural selection and natural genetics. As well known, one of differences from many other conventional search algorithms is that genetic algorithms require the natural parameter set of the optimization problem to be coded as a finite-length string. However, the encoding and decoding processes waste many computation time and lose the accuracy of the parameters. In this paper, a novel variable-based genetic algorithm is proposed. The algorithm processes the parameters themselves without coding. It can save the coding processing time and get more accurate values of the parameters. Finally, the system identification problem has been used to demonstrate the power of the algorithm.

1. INTRODUCTION

In the last few years research devoted to search and optimization has significantly grown. Genetic algorithms (GAs), developed by Holland, are very powerful and popular search algorithms based on the mechanics of natural selection and natural genetics. Recently GAs have been successfully used in many fancy areas such as system identification [2], fuzzy controller design [3], and neural network training [4].

GAs are different from many conventional search algorithms. Firstly, GAs search from a population of points in the search space simultaneously, not a single point, and therefore likely converge toward the global optimal solution. And a GA uses only payoff (objective function) information, no need for derivatives or other auxiliary knowledge. Thus, it is very flexible to solve many real-world complex

problems. Unlike many methods, GAs use probabilistic transition rules to guide their search. But, they are not simple random walks through the search space. GAs use random choice efficiently by employing prior knowledge to rapidly locate optimal solutions. Finally, as well known, GAs require the natural parameter set of the optimization problem to be coded as a finite-length string over some finite alphabet. Almost the variables to be optimized in the application, like neural network's weighting, are continuous, not digital. One must face a trade-off problem between the length of coded string and the resolution of the parameter value. In other words, under a large search space, the length of the string will be very long if a more accurate solution is desired. The coding process and the long code string will waste the computation time and slow down the convergent rate.

In this paper, a novel variable-based genetic algorithm (VGA) is provided. It is not required to encode or digitize the parameters to finite code string. The basic operations of VGA are based on the parameters themselves. It can save the coding processing time and get more accurate values of the parameters.

The organization of the paper is as follows. The basic operations of VGA are briefly introduced in section 2. The design example of the algorithm applying in system identification is

demonstrated in section 3. Finally, the conclusion is given in the last section.

2. A VARIABLE-BASED GENETIC ALGORITHM

The basic element processed by a VGA is the string formed by the real value of each parameter to be searched. The string can be view as a vector including all parameters. For example, consider the problem of maximizing the function $f(x_1, x_2, \dots, x_n)$, the string is a vector $X = (x_1, x_2, \dots, x_n) \in R^n$. Thus, the search space of the parameters can be extended to all real domain. A simple VGA is also composed of three basic operators: (1) reproduction, (2) crossover, and (3) mutation. But the processes of the operators is a little different from those in simple GA.

Reproduction is a process based on the principle of survival of the fittest. Individual strings are copied according to their fitness (objective) function values. The strings with a higher fitness value have a higher probability of contributing one or more offspring in the next generation. To achieve this, firstly, we must define the fitness function, which can be any nonlinear, discontinuous, positive function. Then individual fitness is calculated and normalized with the sum of all fitness values. The normalized fitness of the string is its survival probability in next generation.

After reproduction, simple crossover may processed in the following steps. First, members of the newly reproduced strings in the mating pool are mated at random. Second, each pair of the strings undergoing crossing over as follows: two real random number $r1$, $r2$ uniformly distributed from -1 to 1 are selected. The new strings are created by adding the difference between the two strings multiplied $r1$ and $r2$ respectively. For example, consider the pair of the string $X=(x_1, x_2, \dots, x_n)$ and $Y=(y_1, y_2, \dots, y_n)$, the difference between the two string $D = X - Y$. The resulting crossover yields two new string:

$$\begin{aligned} X' &= X + r1 \cdot D \\ Y' &= Y + r2 \cdot D \end{aligned}$$

Finally, the mutation operator allows new genetic string to be introduction. It just plays a secondary role in the simple VGA. After crossover, each variable of the string in the new population undergoes a random change with a probability equal to the mutation rate. The mutation operator serves to prevent any variable converged to a local optimal solution. However, a high level of mutation yields an essentially random search. The autoscaling mutation rate method [5] can used to fine-tune the optimal solution.

3. A DESIGN EXAMPLE

Consider a system described by an ARMAX model

$$A(q^{-1})y(t) = B(q^{-1})u(t-d) + C(q^{-1})e(t) \quad (1)$$

where

$$\begin{aligned} A(q^{-1}) &= 1.0 - 1.5q^{-1}, \\ B(q^{-1}) &= 1.0 + 0.5q^{-1}, \\ C(q^{-1}) &= 1.0 - 1.0q^{-1} + 0.2q^{-2}, \end{aligned}$$

$d=1$, and $y(t)$, $u(t)$ and q^{-1} are the output, input, and backward shift operator respectively. The $e(t)$ is a normal distributed random noise with zero mean and unit variance. The system has been used in [2] to show GA applying in system identification. Now the objective is to estimate the parameters of the polynomials $A(q^{-1})$, $B(q^{-1})$ and the delay d , given some the input and output pairs of system (1). The estimation system can write as follows.

$$\hat{A}(q^{-1})\hat{y}(t) = \hat{B}(q^{-1})u(t-\hat{d}) \quad (2)$$

where

$$\begin{aligned} \hat{A}(q^{-1}) &= 1.0 + a_1q^{-1} + a_2q^{-2}, \\ \hat{B}(q^{-1}) &= b_0 + b_1q^{-1}, \end{aligned}$$

$\hat{y}(t)$ is the output and a_1, a_2, b_0, b_1 , and a are the parameters to be estimated. So that the string is a vector $X = [a_1, a_2, b_0, b_1, d]$ and the fitness function is defined as

$$F(t) = M - \sum_{i=1}^w (y(t-i) - \hat{y}(t-i))^2 \quad (3)$$

where M is a bias term to ensure a positive fitness and w is the number of time steps. In this example, M and w is chosen as 2000 and 30

respectively. There are 100 population for every generation and the mutation rate is 0.05. The I/O pairs of system (1) are showed on Fig. 1. And the estimation results for each generation are showed on Fig. 2. This example is just used to test the capability of a simple VGA without any improving modification. We can find that the estimation parameter is very close to the real value.

4. CONCLUSIONS

In this paper, a novel variable-based genetic algorithm (VGA) is provided. It is not required to encode or digitize the parameters to finite code string. The basic element processed by a VGA is the string formed by the real value of each parameter to be search. The string can be viewed as a parameter vector. VGAs can save the coding processing time and get more accurate values of the parameters. It is very eligible to be used to obtain a real optimal solution, like fuzzy controller design, system identification and so on. Additional, some ways to speed up the genetic algorithm in [5] are also suitable to VGA.

References

- [1] D. E. Goldberg, *Genetic Algorithm in Search, Optimization and Machine Learning*, Reading, MA: Addison-Wesley, 1989.

- [2] K. Kristinsson and G. A. Dumont, "System Identification and Control Using Genetic Algorithm," *IEEE Trans. Syst., Man, Cyber.* Vol. 22, No. 5, pp. 1033-1046, Sept. 1992.
- [3] C.L. Karr and E. J. Gentry, "Fuzzy Control of pH Using Genetic Algorithms," *IEEE Trans. Fuzzy Systems*, vol. 1, no. 1, pp. 46 - 53, Feb. 1993.
- [4] M. Caudill, "Evolutionary Neural Networks," *AI Expert*, pp. 28-32, March 1991.
- [5] S.A. Kennedy, "Five Ways to a Smarter Genetic Algorithm," *AI Expert*, pp. 35-38, Dec. 1993.

Fig. 1 The inputs and outputs of the system (1)

Fig.2 Parameter estimation using a simple VG