RETAINING DIVERSITY OF GENETIC ALGORITHMS FOR MULTIVARIABLE OPTIMIZATION AND NEURAL NETWORK LEARNING

Yoshiaki Ichikawa and Yoshikazu Ishii
Energy Research Laboratory, Hitachi, Ltd.
7-2-1 Omika-cho, Hitachi-shi, Ibaraki-ken, 319-12 Japan
Tel: 81-294-53-3111, Fax: 81-294-53-2830
E-mail: yoshi@hrl.hitachi.co.jp

Abstract—This paper presents methods to retain diversity of the allele distribution in the search for Genetic Algorithms (GA). GA is a search algorithm employing multiple concurrent search points and is well fitted to optimization problems where mathematical analyses are not applicable; i.e., combinatorial optimization, unsupervised learning, and some ill-structured problems. One of the crucial issues in applying GA has been "premature convergence" which deteriorates diversity of the search points distribution and causes trapping into local optima. The purpose of this study is to prevent premature convergence and refine the performance of GA for use in multivariable optimization and unsupervised learning of neural networks. We define an integer string representation for chromosomes, which is well fitted to this usage. The diversity of each locus and Rareness of a chromosome are evaluated based on the distribution of alleles in a population. The fitness of a chromosome is adjusted with the Rareness so that rare chromosomes will be likely to survive. In addition, Mutation width is introduced to adjust the effect of mutation which can generate rare chromosomes. By dynamically changing Mutation width at each locus according to the diversity, prematurity can be avoided while conserving effective convergence. Case studies with problems of neural network pattern matching and unsupervised learning of a neural network which controls an inverted pendulum are discussed.

1. INTRODUCTION

Growing number of attempts have been made to apply Genetic Algorithms (GA) to various problem areas[1]-[4]. Among them, we proposed a method for unsupervised learning of neural networks[1]-[2]. Despite some successful results, many applications still confront difficulties which restrict the range of their uses. One of the largest difficulties lies in the loss of diversity of alleles which is referred to as premature convergence[5]. In this paper, we discuss retaining the diversity of the alleles for integer string representation which facilitates GA applications to multivariable problems. In order to keep diversity, the following means are needed;
(1) to evaluate diversity,
(2) to generate diversity, and
(3) to prevent the loss of diversity.
While some considerations in this regard have been presented for binary string representations[6]-[7], no method have been established generally applicable to integer or numerical string representations. The methods we propose are not simply a translation of those of binary representations, but rather more effective ones exploiting specific features of the integer string representation.

2. INTEGER STRING REPRESENTATION

We define the problem as the optimization of numerical variables x_1 - x_N, in terms of a fitness evaluation function $F(x_1, x_2, .., x_N)$. Conventional coding uses binary representations of the variables and packs them together in a row to obtain a bit string, a chromosome, in which a locus corresponds to a bit of one of the variables and alleles correspond to 1 or 0. By contrast, the integer representation in our case codes a variable into an integer ranging from -10 to 10 and forms a chromosome with an array x[i] (i = 1 to N) in which a locus indexed by i corresponds to a variable and alleles correspond to integers from -10 1to 10. A population of chromosomes is represented by a two dimensional array x[q][i] where q indexes an individual. Uniform crossover is used for mating two individuals; N/2 loci are randomly chosen to exchange alleles between the two.

0-7803-0999-5/93/$03.00 ©1993 IEEE

3. Evaluating Diversity of Alleles and Rareness of an Individual

A measure of diversity can be defined as an extension of the "number of alleles lost" which was originally defined for the binary string representation[5]. We, however, need some extra measures related to the definition. *Histogram*[n][i] expresses the alleles count of the locus i within a population which falls into the seven divided intervals indexed by n; [-10,-8], [-7,-5], ..., [-1,1], ..., [8,10]. This is defined as follows:

$$Histogram[n][i] = \sum_{q=1}^{Qmax} \sigma(q,n,i) \quad (1)$$

$$\text{where } \sigma(q,n,i) = \begin{cases} 1 \text{ if } 3n\text{-}13 \le x[q][i] \le 3n\text{-}11 \\ 0 \text{ otherwise} \end{cases}$$

Qmax used in the expression stands for the size of the population, the total number of chromosomes. Using the measure above, AL[i], the number of alleles lost, is defined with regard to a locus, i, as follows:

$$AL[i] = \sum_{n=1}^{7} \{ \text{ number of zeros in } Histogram[n][i] \} \quad (2)$$

Then the total number of alleles lost is defined as

$$TAL = \sum_{i=1}^{N} AL[i] \quad (3)$$

In relation to the *Histogram*, the *Rareness* of an individual can be evaluated as follows:

$$Rareness[q] = \sum_{i=1}^{N} Histogram[n(q,i)][i] \quad (4)$$

$$\text{where } n(q,i) = \{ \; n \mid 3n\text{-}13 \le x[q][i] \le 3n\text{-}11 \; \}$$

The above equation means the sum of the counts in *Histogram* with regard to an individual. Thus the *Rareness* is small when an individual is rare; alleles of all the loci of the individual fall in rare regions. This measure can be used to prevent the loss of diversity.

4. Retaining Diversity

Although the effect of mutation for generating random alleles to increase diversity has been pointed out, it has not been utilized due to its side effect. For a binary string, the mutation is performed by bit inversion. This however causes a dramatic change in the decoded variable if the inversion takes place on a significant bit. A too large random effect is not desirable in terms of convergence efficiency and thus the mutation probability has been assigned with a very small value less than 1/100. By contrast, for the integer string representation, we can adopt a 100% mutation probability with a special mutational operation defined as follows:

> Select a random integer, *d*, within the range of [-Dw, Dw] and add *d* to x[q][i]. Repeat this for i=1 to N when an individual, q, undergoes a mutation.

The *Mutation width*, Dw, can be used to adjust the randomization effect. When we use a large Dw this can be equivalent to the binary case whereas a moderate Dw can suppress the undesirable effect while still being effective for generating rare alleles. To balance the trade-off between quick convergence and retaining diversity, we adjust the *Mutation width* on each locus according to the following empirical equation:

$$Dw = 3AL[i] \quad (5)$$

This equation intends to relate the *Mutation width* and the number of alleles lost so that the mutation takes effect when alleles begin to be lost.

In addition to the *Mutation width*, we propose a fitness scaling method as follows:

$$\{\text{New fitness of q}\} = \{\text{fitness of q}\} \,/\, Rareness[q] \quad (6)$$

This fitness scaling is effective to prevent the loss of rare individuals after being generated by mutation.

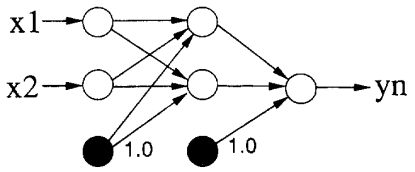
5. Case Studies

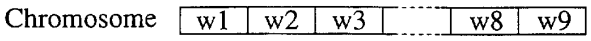
As examples, we have applied our methods to neural network problems including two-input-XOR pattern learning and control of a non-linear dynamic

system. Fig. 1(a) illustrates a perceptron-type neural network structure used in the XOR problem. The network is three layered and composed of 7 neuron units. The shaded circles shown in the figure are constant units which always output 1.0 to serve threshold values. The other units in the first layer simply output the same values as inputs while each unshaded unit in the second and third layer performs a sigmoidal transform on the sum of inputs as follows:

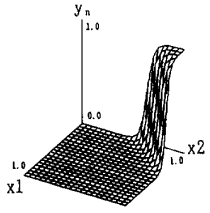
$$o = \frac{1-\exp(-i)}{1+\exp(-i)} \qquad (7)$$

where o and i stand for the output and the sum of inputs of a neuron unit. In this case, there are 9 connection weights including connections from the constant units to adjust threshold values. Those weights are directly coded in a chromosome as shown in the figure. The linear inseparable nature of this problem provides a form of local optimum as illustrated in Fig. 1(b) while a global optimum, shown in Fig. 1.(c) as an example, is attained if a chromosome can escape from the local optimum.

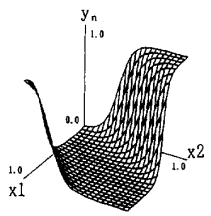




(a) Setup for XOR learning



(b) Local optimum



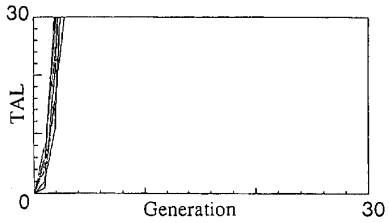
(c) Global optimum

Fig. 1. Exclusive OR pattern learning

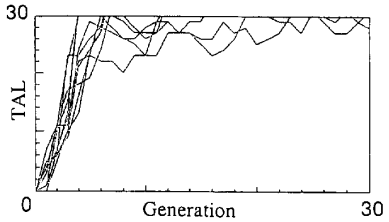
For GA implementation, the following evaluation is conducted as to each chromosome to get its value V:

$$V=\left\{ \sum_{\text{four patterns}} | yn - XOR(x1,x2) | \right\}^{-1} \qquad (8)$$

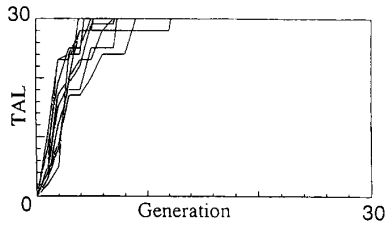
Where "four patterns" means the four input cases, (1,1), (1,0), (0,0), and (0,1). The fitness values are first evaluated by directly reflecting V as above, and then adjusted according to Eq. (6). The results with ten independent runs are exhibited in Fig. 2 which are demonstrating the effect of the mutation and fitness scaling in TAL(total number of alleles lost). In case of no means are used to retain diversity (Fig. 2(a)), TAL has increased rapidly which suggests that all chromosomes are caught in local optima and none of the 10 runs has escaped from it. When mutation is applied (Fig. 2(b)), the TAL burst is prevented while it has yet increased to a high level. The effect of the fitness scaling is shown in Fig. 2(c) where the speed of TAL increase is suppressed compared with Fig. 2(a). A significant synergism of the fitness scaling and mutation is demonstrated in Fig. 2(d), in which TAL is kept small and 7 cases in the 10 runs have escaped from local optima. Fig. 2(e) exhibits an effect of the *dynamic mutation width* using Eq. (3). TAL is suppressed more uniformly than Fig. 2(d) and the highest escape rate from local optima, 80%, is achieved.



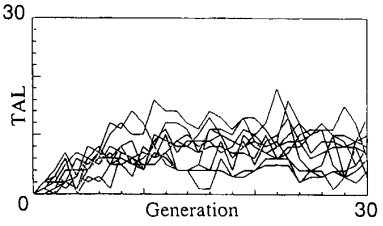
(a) No mutation, no fitness scaling.
Local optima escape rate: 0%



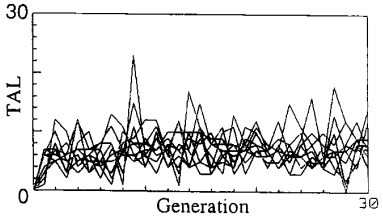
(b) Mutation alone (Dw=3).
Local optima escape rate: 50%



(c) Fitness scaling alone.
Local optima escape rate: 0%



(d) Mutation (Dw=3) and fitness scaling.
Local optima escape rate: 70%



(e) Mutation (Dw=3AL[i]) and fitness scaling.
Local optima escape rate: 80%

Fig. 2. Effect of retaining diversity

The other case study is conducted by simulation in controlling an inverted pendulum as shown in Fig. 3. Four state variables of an inverted pendulum are input to a three layered network with 25 connection weights which are coded in a chromosome as shown in the figure. The output of the network is digitized to become a nonlinear control input; i.e., u=5 if the output is positive and u=-5 otherwise. The equation of motion and parameters are as follows:

$$(M+m)\ddot{r}+ml\ddot{\theta}\cos\theta-ml\dot{\theta}^2\sin\theta=u$$
$$\frac{4}{3}ml^2\ddot{\theta}+ml\ddot{r}\cos\theta-mgl\sin\theta=0 \qquad (9)$$

where M=1kg, m=1kg, l=1m, g=9.8m/s^2

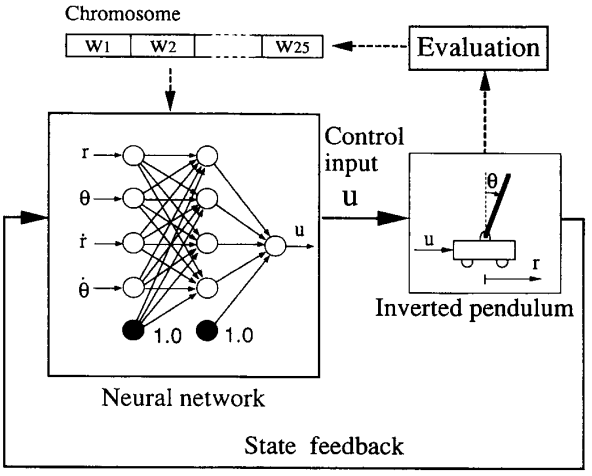


Fig. 3. Nonlinear control

The evaluation of each chromosome is conducted as follows:

(1) Start simulation with initial conditions, (r=0, q=0.1, $\dot{r}=\dot{\theta}=0$) and (r=0, q= -0.1, $\dot{r}=\dot{\theta}=0$). Then a simulator employing the 4th order Runge-Kutta method returns the final simulation time, Tf, defined as:

$$Tf = \{\text{time t when } |q(t)| \text{ exceeds } |q(0)|\} \qquad (10)$$

(2) If Tf is smaller than 5s the value of a response, v, is evaluated as

$$v = Tf \qquad (11)$$

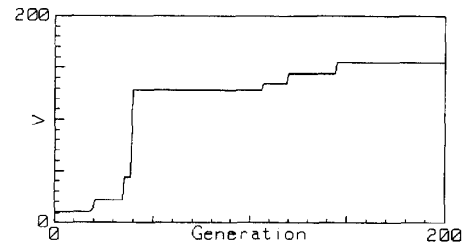
(3) If Tf is larger than 5s, v is evaluated as

$$v = Tf + \left[\frac{1}{5}\int_0^5 \left(r(t)^2+\theta(t)^2+\dot{r}(t)^2+\dot{\theta}(t)^2\right)\,dt\right]^{-1} \qquad (12)$$

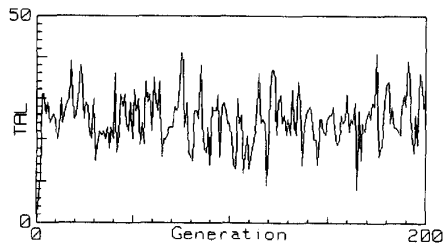
(4) Take the minimum of v obtained for the two initial conditions and make the value V of a chromosome.

The value V is converted to fitness after the scaling, and the *dynamic mutation width* is also used to retain diversity. Convergence behaviors with 20 population is shown in Fig. 4(a) and (b). It is demonstrated that

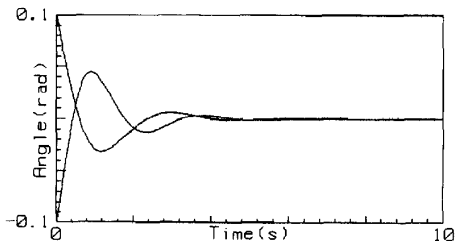
TAL is kept constant even in the period during which population-best value is stagnating. The best response curves from the two initials are shown in Fig. 4(c), which indicate a stable regulator capability.



(a) Population-best value



(b) Diversity



(c) Population-best responses with two initial conditions at 200 generations.

Fig. 4. Control results

6. CONCLUSION

Rareness of an individual and an integer-string version of the *number of alleles lost* are defined for evaluating diversity. The *Mutation width* and scaling of fitness according to *Rareness* are proposed to generate diversity and prevent its loss. In addition, a dynamic adjustment method of the *Mutation width* of each locus is proposed to balance convergence and diversity. Case studies with problems of neural network learning have demonstrated the utility of the methods.

REFERENCES

[1] Y. Ichikawa, "Evolution of Neural Networks and Application to Motion Control", in Proc. IEEE Int. Conf. Intel. Motion Control, pp.239-245, 1990

[2] Y. Ichikawa and T. Sawa, "Neural Network Application for Direct Feedback Controllers", IEEE Trans. on Neural Networks, vol.3, no.2, pp.224-231, 1992

[3] L. Davis,"Handbook of Genetic Algorithms", Van Nostrand Reinhold, 1991

[4] R. K. Belew and L B. Booker (ed.) ,"Genetic Algorithms", Proceedings of the 4th ICGA, 1991

[5] D. E. Goldberg,"Genetic Algorithms in Search, Optimization, and Machine Learning", Addison-Wesley, 1989

[6] L. J. Eshelman and J. D. Schaffer "Preventing Premature Convergence in Genetic Algorithms by Preventing Incest", in Proc. of 4th ICGA, pp.115-122, 1991

[7] Y. Nishikawa and H. Tamaki,"A Neighborhood Model of the Genetic Algorithm and Its Application to the Jobshop Scheduling", in Proc. 34th Japan Joint Automatic Control Conf., pp.345-346, 1991