Dominance Learning in Diploid Genetic Algorithms for Dynamic Optimization Problems

Shengxiang Yang
Department of Computer Science, University of Leicester
University Road, Leicester LE1 7RH, United Kingdom
s.yang@mcs.le.ac.uk

ABSTRACT

This paper proposes an adaptive dominance mechanism for diploidy genetic algorithms in dynamic environments. In this scheme, the genotype to phenotype mapping in each gene locus is controlled by a dominance probability, which is learned adaptively during the searching progress and hence is adapted to the dynamic environment. Using a series of dynamic test problems, the proposed dominance scheme is compared to two other dominance schemes for diploidy genetic algorithms. The experimental results validate the efficiency of the proposed dominance learning scheme.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Heuristic methods*

General Terms

Algorithms

Keywords

Diploid genetic algorithms, dominance change scheme, dominance learning, dynamic optimization problems

1. INTRODUCTION

Dynamic optimization problems (DOPs) present serious challenges to traditional genetic algorithms (GAs) due to the convergence problem: once converged, it is hard for GAs to adapt to the changing environment. Researchers have developed several approaches into GAs to address DOPs. For example, the diploid GAs (DGAs) integrate diploidy and dominance mechanisms in biology into GAs for DOPs. DGAs differ from traditional GAs in two major aspects. The first one lies in the representation and evaluation scheme. For DGAs, each individual has a pair of chromosomes and its genotype and phenotype are separated. When evaluating an individual, the diploid genotype is first mapped into a haploid phenotype by some dominance mechanism. Then, the phenotype is evaluated to obtain a fitness. The second difference lies in the reproduction operations. For DGAs, crossover takes two steps. First, two parents exchange their chromosomes randomly to create two temporary offsprings. Second, each offspring then undergoes the normal crossover operation within its own two chromosomes with a probability. For DGAs, mutation performs on each of the two genotypic chromosomes of an individual independently with the same mutation probability for each locus.

As in biology, the dominance scheme controls how genes are expressed in the phenotype and plays a key role in the performance of DGAs. Several dominance mechanisms have been developed [1, 2, 3]. In [2], Ng and Wong proposed a diploid representation with four genotypic alleles: dominant "1" and "0", and recessive "i" and "o". The dominant alleles are always expressed in the phenotype. If contention exists between two dominant or two recessive alleles, one is randomly expressed. Ng and Wong also incorporated a dominance change scheme when the fitness of an individual drops by a preset percentage (20%) between successive evaluation cycles. In [3], Ryan proposed an *additive dominance*, where genotypic alleles are represented by ordered values that are combined using a pseudo-arithmetic to determine the phenotypic allele. Lewis *et. al.* [1] extended Ryan's scheme by adding a dominance change mechanism where genotypic alleles are demoted or promoted by one grade, which is similar to the Ng-Wong dominance change scheme.

2. DOMINANCE LEARNING SCHEME

In biology the dominance mechanism may change with the environment. This inspires the *dominance learning mechanism* proposed in this paper for DGA, denoted *DLDGA*. DLDGA uses a *dominance probability vector* where each element is a *dominance probability* that represents the probability a genotypic allele can be expressed in the phenotype in a locus. For di-allelic encoding, without loss of generality, we can define the dominance probability vector at generation t $\vec{p}_d(t) = \{p_d(i,t), \cdots, p_d(l,t)\}$ (l is the encoding length) in terms of expressing allele 1 in the phenotype. That is, $p_d(i,t)$ denotes the probability that allele 1 will dominate allele 0 and appear in locus i of the phenotype of an individual if the two genotypic alleles of the individual do not agree in locus i. Let $\vec{C}_1(t) = \{C_1(1,t), \cdots, C_1(l,t)\}$ and $\vec{C}_2(t) = \{C_2(1,t), \cdots, C_2(l,t)\}$ be the two chromosomes in the genotype of an individual in the population at time t, then $\vec{C}_1(t)$ and $\vec{C}_2(t)$ are mapped to the phenotype $\vec{P}(t) = \{P(1,t), \cdots, P(l,t)\}$ of the individual as follows:

$$P(i,t) = \begin{cases} 1, & C_1(i,t) = C_2(i,t) = 1 \\ 0, & C_1(i,t) = C_2(i,t) = 0 \\ 1, & C_1(i,t) \neq C_2(i,t) \ \& \ r < p_d(i,t) \\ 0, & C_1(i,t) \neq C_2(i,t) \ \& \ r >= p_d(i,t), \end{cases} \quad (1)$$

where $r = rand(0.0, 1.0)$ is a random number.

GECCO'06, July 8–12, 2006, Seattle, Washington, USA.
ACM 1-59593-186-4/06/0007.

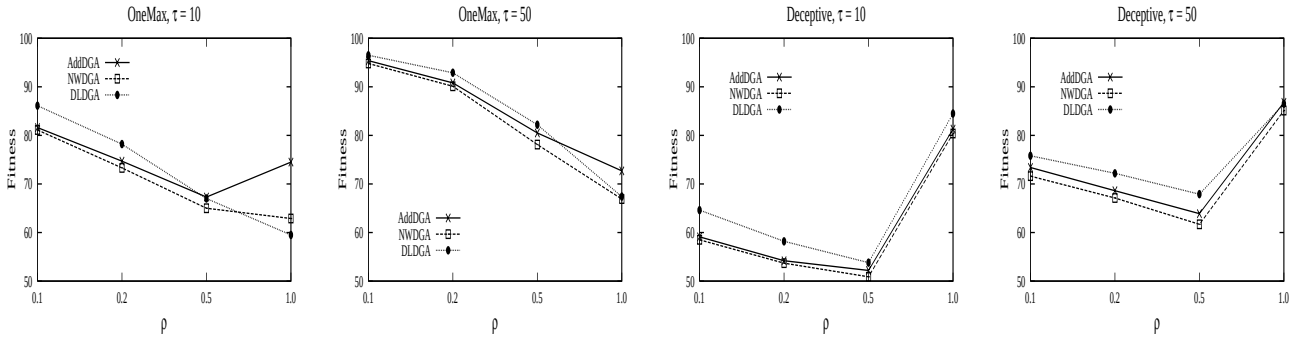

Figure 1: Experimental results of DGAs on the dynamic test problems.

The dominance probability vector is evolved by the following learning scheme. Originally, it starts from the *neutral dominance probability vector*, $\vec{p}_d(0) = \vec{0.5}$. Then, $\vec{p}_d$ is updated every generation according to the best individual in the population. Let $\vec{P}_B(t) = \{P_B(1,t), \cdots, P_B(l,t)\}$ denote the phenotype of the best individual at generation t. Then, $\vec{p}_d$ is learned toward $\vec{P}_B(t)$ as follows:

$$p_d(i, t+1) := p_d(i,t) * (1-\alpha) + \alpha * P_B(i,t), \qquad (2)$$

where $i \in \{1, \cdots, l\}$ and α is the learning rate.

3. EXPERIMENTAL STUDY

The DOP generator proposed in [4] is used to construct random dynamic test environments. With this XOR generator, parameters τ and ρ control the speed and severity of environmental changes respectively. Smaller τ means faster environmental changes while bigger ρ means more severe changes. Two 100-bit binary functions are selected as base stationary functions to construct DOPs. The first is the *OneMax* function. The second one consists of 25 contiguous 4-bit deceptive building blocks. Both functions have an optimum fitness of 100. For each DOP, the landscape is periodically changed every τ generations during the run of a GA. We change the environmental dynamics by setting τ to 10 and 50 and ρ to 0.1, 0.2, 0.5, and 1.0 respectively.

Based on the above dynamic environments, experiments are carried out to compare DLDGA against two DGAs with the Ng-Wong [2] and the additive [3] dominance schemes with dominance change [1], denoted *NWDGA* and *AddDGA* respectively. For both NWDGA and AddDGA, we switched off the environmental change detection scheme and instead, whenever the environment changes, the dominance change scheme in NWDGA and AddDGA starts to work immediately. All DGAs have parameters set as follows: generational, uniform crossover with the crossover probability 0.6, bit mutation with probability 0.01, binary tournament selection, and elitism of size 1. The population size n is set to 100. For DLDGA, the learning rate $\alpha = 0.5$.

For each experiment of a DGA on a DOP, 50 independent runs were executed with the same set of random seeds. For each run of a DGA on a DOP, 50 environmental changes were allowed and the best-of-generation fitness was recorded every generation. The overall performance of a DGA on a DOP is defined as the best-of-generation fitness averaged over 50 runs and then averaged over the data gathering period. The experimental results are plotted in Figure 1, where several results can be observed on the DOPs.

First, DLDGA significantly outperform both NWDGA and AddDGA on most dynamic test problems except on the OneMax DOP with $\rho = 1.0$. This result validates the efficiency of introducing dominance learning scheme over fixed ones into DGA for dynamic environments. When $\rho = 1.0$, i.e., the environment oscillates between two opposite fitness landscape, DLDGA is beaten by AddDGA on the OneMax problem. This happens because the dominance change scheme in AddDGA is in fact more directly oriented toward extreme environmental changes.

Second, the additive dominance scheme in AddDGA is significantly better than the Ng-Wong scheme for DGAs on the dynamic test environments. The existence of uncertainty in the dominance mapping in the Ng-Wong scheme gives it a disadvantage over the additive dominance scheme.

4. CONCLUSIONS AND FUTURE WORK

This paper proposes a dominance learning scheme that uses a dominance probability vector to map the genotype to phenotype of individuals for DGAs in dynamic environments. This dominance probability vector is learned adaptively toward the current environment and hence can adapt the DLDGA more efficiently in the changing environment. The experimental results based on a series of dynamic problems validate the efficiency of the dominance learning scheme.

Comparing the dominance learning scheme with other advanced ones is now under investigation. It is also interesting to compare it with advanced explicit memory schemes for GAs for DOPs. We also believe that combining it with other explicit memory schemes will further improve the performance of GAs in dynamic environments.

5. REFERENCES

[1] E. H. J. Lewis and G. Ritchie. A comparison of dominance mechanisms and simple mutation on non-stationary problems. *Parallel Problem Solving from Nature V*, pp. 139-148, 1998.

[2] K. P. Ng and K. C. Wong. A new diploid scheme and dominance change mechanism for non-stationary function optimisation. *Proc. of the 6th Int. Conf. on Genetic Algorithms*, 1995.

[3] C. Ryan. The degree of oneness. *Proc. of the 1994 ECAI Workshop on Genetic Algorithms*, 1994.

[4] S. Yang and X. Yao. Experimental study on population-based incremental learning algorithms for dynamic optimization problems. *Soft Computing*, vol. 9, no. 11, pp. 815-834, 2005.