2006 IEEE Congress on Evolutionary Computation
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

On the Design of Diploid Genetic Algorithms for Problem Optimization in Dynamic Environments

Shengxiang Yang, *Member, IEEE*

***Abstract*— Using diploidy and dominance is one method to enhance the performance of genetic algorithms in dynamic environments. For diploidy genetic algorithms, there are two key design factors: the cardinality of genotypic alleles and the uncertainty in the dominance scheme. This paper investigates the effect of these two factors on the performance of diploidy genetic algorithms in dynamic environments. A generalized diploidy and dominance scheme is proposed for diploidy genetic algorithms, where the cardinality of genotypic alleles and/or the uncertainty in the dominance scheme can be easily tuned and studied. The experimental results show the efficiency of increasing genotypic cardinality rather than introducing uncertainty in the dominance scheme.**

I. Introduction

Many optimization problems in the real world are subject to dynamic environments. For example, in the manufacturing industry, new jobs may arrive stochastically and machines may break down. The nature of dynamic environments challenges traditional optimization algorithms because it requires them to be able to adapt to the changing environment with time. For dynamic optimization problems (DOPs), genetic algorithms (GAs) are a good choice because they are inspired from the principles of biological evolution, which takes place in a dynamic environment in nature. However, when solving DOPs, traditional GAs face a big problem: the convergence problem. Convergence deprives GAs' adaptability to the changing environment: Once converged, they are unable to adapt to the new environment when change occurs. In order to enhance the performance of GAs for DOPs, several approaches have been developed [3], such as random immigrants [8], [24], hypermutation [4], memory [1], [15], [16], [19], [20], and multi-population schemes [2]. Many of these approaches are inspired from biological principles, e.g., the diploidy and dominance mechanisms.

In biology, most complicated organisms have a diploid or multiploid structure in their genetic representation and relevant dominance scheme for gene expression [10]. In the diploid structure, two set of *homologue chromosomes* are twisted together into a duplex structure. *Genes* are the smallest hereditary unit that contains genetic information and the alternative forms/values that genes occupying the same locus on homologue chromosomes can take are called *alleles*. The *genotype* of an individual consists of all the genes located on all the homologue chromosomes and those genes that are expressed form the *phenotype* of the individual. A *dominant* allele is always expressed while a *recessive* allele may be expressed only when both genes occupying the same locus of the pair of homologue chromosomes have the same recessive value. Whether an allele is dominant or recessive is determined by a *dominance scheme*.

Diploidy and dominance are the driving force for organisms to adapt to and survive in the ever-changing environment in nature. In diploidy organisms, it requires two genes for each biological trait and only the gene with dominant allele gets expressed. Though the other gene is not expressed and seems redundant, it helps keep the genetic diversity, which allows organisms to respond to environmental changes more quickly and efficiently and hence gives organisms a better chance to survive.

The mechanisms of diploidy and dominance in biology have been integrated into traditional GAs to enhance their performance for DOPs [7], [17], [18]. GAs with diploidy representation and dominance schemes are called *diploid genetic algorithms (DGAs)*. DGAs have proved advantageous to deal with dynamic environments. For DGAs, there are two important design factors: the *cardinality of genotypic representation*, i.e., the number of alleles a gene can take, and the dominance scheme, which maps genotype to phenotype.

This paper investigates the effect of the cardinality of genotypic representation and the existence of uncertainty in the dominance scheme for DGAs in dynamic environments. For this aim, a generalized genotype representation and dominance scheme is proposed. In this generalized scheme, it is convenient to tune the cardinality of genotypic representation and the uncertainty in the genotype to phenotype mapping for DGAs. Using the dynamic problem generator proposed in [23], [25], a series of DOPs are constructed as the dynamic test environments. And two sets of experiments are carried out: to investigate the performance of DGAs with different genotypic cardinalities and to investigate the performance of DGAs with deterministic and non-deterministic dominance schemes (to be described in Section III). The experimental results show the efficiency of increasing cardinality in the genotypic representation rather than introducing uncertainty in the genotype-to-phenotype mapping for DGAs in dynamic environments.

The rest of this paper is organized as follows. The next section briefly reviews the background on DGAs and dominance schemes. Section III details the proposed generalized genotype representation and dominance scheme. Section IV presents the experimental study, including the dynamic test environments, experimental design, experimental results and relevant analysis. Finally, Section V concludes the paper with some discussions.

Shengxiang Yang is with the Department of Computer Science, University of Leicester, University Road, Leicester LE1 7RH, United Kingdom (Tel: 0044-116-2515341; Fax: 0044-116-252 3915; Email: s.yang@mcs.le.ac.uk).

0-7803-9487-9/06/$20.00/©2006 IEEE

$t := 0$ and initialize population $P(0)$ randomly
evaluate population $P(0)$
repeat
  $P'(t) :=$ selectForReproduction$(P(t))$
  crossover$(P'(t), p_c)$ // p_c is the crossover probability
  mutate$(P'(t), p_m)$ // p_m is the mutation probability
  evaluate the interim population $P'(t)$
  if *dominance change used* and *condition holds* **then**
    change dominance scheme
until $terminated =$ **true** // *e.g.*, $t > t_{max}$

Fig. 1. Pseudo-code of general diploid genetic algorithms.

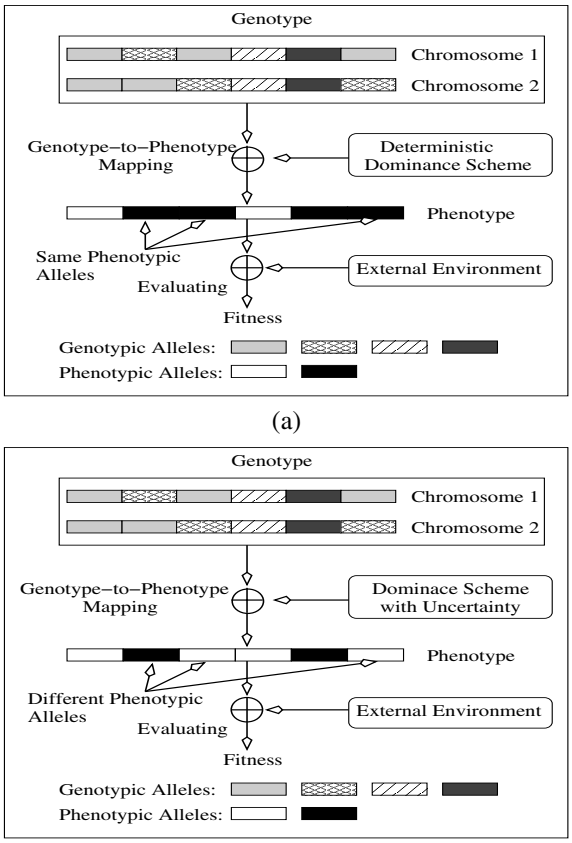


Fig. 2. Representation and evaluation of an individual for DGAs with (a) deterministic and (b) non-deterministic dominance scheme.

II. Relevant Research

A. Diploid Genetic Algorithms

As mentioned above, DGAs are inspired by the genetic encoding and expression mechanisms in biology. The pseudo-code of general DGAs is shown in Fig. 1, where dominance change may be integrated. DGAs differs from traditional GAs mainly in two aspects: the representation and evaluation scheme and the reproduction operations.

For the first aspect, each individual in DGAs has two genotypic chromosomes. In order to evaluate an individual we need first map the diploid genotype into a haploid phenotype according to some dominance mechanism. Then, the phenotype is evaluated according to the external environment to give the individual a fitness. The representation and evaluation scheme for DGAs is illustrated in Fig. 2. In DGAs, each gene in the genotype may have several alleles while in the phenotype it usually takes binary values 0 or 1. The black and white bars in Fig. 2 represents 0 and 1 in the phenotype respectively, vice versa.

For the second aspect, in DGAs crossover is divided into two steps. In the first step, two parents exchange their chromosomes randomly to create two temporary offsprings. Each offspring has one chromosome from each parent and hence the genotypic materials from the parents are mixed and propagated to the offsprings. In the second step, two chromosomes in each offspring undergo the crossover operation with a probability p_c, which is the same as in traditional GAs. After crossover, each of the two genotypic chromosomes of an offspring is independently subject to a bitwise mutation with a probability p_m. Since mutation operates on the genotype of an individual, a mutation of a gene may not change the phenotype of an individual. This is called *neutral mutation* in the biological terminology [10].

B. Genotype Representation and Dominance Schemes

As mentioned before, genotypic representation and dominance schemes play a key role in the performance of DGAs. Researchers have developed several genotypic representation and dominance schemes [5], [7], [9], [13], [22].

In [12], Hollstien proposed a tri-allelic dominance scheme where genotypic genes take alleles from $\{0, 1, 2\}$ and both genotypic alleles "2" and "1" map to phenotypic "1", but "2" dominates "0" and "0" dominates "1". Holland [11] later studied Hollstien's tri-allelic scheme and introduced the clearer 3-alphabet $\{0, 1_0, 1\}$ instead. Goldberg and Smith [7] compared the performance of a haploid GA, a diploid GA with fixed dominance map where 1's dominate 0's, and a diploid GA with Holstien-Holland tri-allelic dominance map on a non-stationary knapsack problem and concluded that Holstien-Holland's tri-allelic dominance is better than either fixed dominance or haploid structure.

In [17], Ng and Wong proposed a dominance scheme that uses four genotypic alleles: dominant "1" and "0", and recessive "i" and "o". The dominant allele is always expressed in the phenotype. If contention exists between two dominant or two recessive alleles, one is randomly chosen to be expressed. The occurrence of "1i" or "0o" is prohibited. If it occurs the recessive gene is promoted to be dominant. The genotype-to-phenotype mapping is shown in Fig. 3(a). Ng and Wong also incorporated a dominance change scheme: when the fitness of an individual drops by a threshold percentage 20% between successive evaluation cycles, the dominance values of all allele-pairs are inverted such that "11" becomes "ii", "00" becomes "oo", "1o" becomes "i0", and vice versa. The genotype-to-phenotype mapping keeps unchanged.

	0	o	1	i
0	0	0	0/1	0
o	0	0	1	0/1
1	0/1	1	1	1
i	0	0/1	1	1

(a)

	A	B	C	D
A	0	0	0	1
B	0	0	0	1
C	0	0	1	1
D	1	1	1	1

(b)

Fig. 3. Two dominance schemes: (a) Ng-Wong and (b) additive.

In [18], Ryan proposed an *additive dominance* scheme, where genotypic alleles are represented by ordered values that are combined using a pseudo-arithmetic to determine the phenotypic allele. Ryan used four genotypic alleles A, B, C, and D, and allocated the values 2, 3, 7, and 9 to them respectively. An addition is performed on the values allocated with the two genotypic alleles for each gene locus. If the sum is greater than 10, the phenotypic allele becomes 1; otherwise, it becomes 0. The resulting dominance map is shown in Fig. 3(b).

In [14], Lewis *et. al.* extended Ryan's scheme by adding a dominance change scheme where genotypic alleles are demoted or promoted by one grade. For example, demoting a "B" makes it a "A" whereas promoting it makes it a "C". Allele "A" cannot be demoted and "D" cannot be promoted. As in the Ng-Wong approach, dominance change occurs when the fitness of an individual drops by a threshold percentage 20% between successive evaluation cycles. If this happens, for each locus one of the two genotypic alleles is randomly chosen to undergo the following procedure:

- If the phenotypic allele at this locus is 1, then demote the chosen genotypic allele, unless it is an "A".
- If the phenotypic allele at this locus is 0, then promote the chosen genotypic allele, unless it is a "D".

Lewis *et. al.* [14] compared the Ng-Wong and the additive dominance schemes with or without dominance change on a dynamic knapsack problem and concluded that some form of dominance change is essential for DGAs to adapt to environmental changes as a diploid encoding is not enough in itself to allow flexible response to changes.

In [21], [22], Uyar and Harmanci proposed an adaptive dominance mechanism for DGAs where the dominance characteristics for each gene locus is adapted according to the current population through a statistics calculation.

III. Generalized Diploidy and Dominance Scheme

From above brief review, it can be seen that researchers have used different cardinalities in the genotypic representation. And in the developed dominance schemes there may exist uncertainty: giving two different genotypic alleles the phenotypic allele may be random. For example, in the Ng-Wong scheme when "0" meets "1" in the genotype at a locus, the phenotype can be either 0 or 1 for that locus, see Fig. 3 (a). In this paper, we call those dominance mechanisms that have no uncertainty in their genotype-to-phenotype mapping *deterministic* and those with uncertainty *non-deterministic*. For deterministic dominance mechanisms, when two different genotypic alleles meet several times in the genotype of an individual at different loci, the phenotypic alleles are always the same for those loci, as illustrated in Fig. 2 (a), while for non-deterministic dominance mechanisms the phenotypic alleles may differ at those loci, as illustrated in Fig. 2 (b).

In order to investigate the effect of the cardinality of the genotypic representation and the uncertainty in the dominance scheme, this paper proposes a generalized diploidy representation and dominance scheme for DGAs, which comes in two types: deterministic and non-deterministic. The generalized dominance scheme is described as follows.

Let $\vec{A}=\{A_1, \cdots, A_c\}$ denote the set of genotypic alleles with cardinality c. And let $\vec{C}^1 = \{C_1^1, \cdots, C_l^1\}$ and $\vec{C}^2 = \{C_1^2, \cdots, C_l^2\}$ denote the two chromosomes in the genotype of an individual in the population, where l is the encoding length, and $\vec{P}=\{P_1, \cdots, P_l\}$ be the corresponding phenotype of the individual. For locus i, we further assume $C_i^1 = A_m$ and $C_i^2 = A_n$, i.e., the first and second chromosomes have a genotypic allele A_m and A_n ($m, n \in \{1, \cdots, c\}$) for locus i respectively. For non-deterministic generalized dominance scheme, C_i^1 and C_i^2 are mapped to P_i as follows:

$$P_i = \begin{cases} 0, & m+n < c+1 \\ 1, & m+n > c+1 \\ 0/1, & m+n = c+1, \end{cases} \quad (1)$$

where $0/1$ means a 0 or 1 with equal probability.

For deterministic generalized dominance scheme, we further assume that the cardinality is a multiple of 4, i.e., $c = 4k$. Then, C_i^1 and C_i^2 are mapped to P_i as follows:

$$P_i = \begin{cases} 0, \; m+n < c+1 \\ 1, \; m+n > c+1 \\ 0, \; m+n = c+1 \;\&\; min\{m,n\} > c/4 = k \\ 1, \; m+n = c+1 \;\&\; min\{m,n\} <= c/4 = k \end{cases} \quad (2)$$

For example, Fig. 4 shows the deterministic and non-deterministic generalized dominance schemes with the set of genotypic alleles defined as $\vec{A} = \{A_1, A_2, A_3, A_4\} = \{A, B, C, D\}$. It can be seen that the generalized deterministic dominance scheme in Fig. 4(a) is the same as the additive dominance scheme shown in Fig. 3(b). The difference lies in that now there are no ordered values allocated to alleles.

	A	B	C	D
A	0	0	0	1
B	0	0	0	1
C	0	0	1	1
D	1	1	1	1

(a)

	A	B	C	D
A	0	0	0	0/1
B	0	0	0/1	1
C	0	0/1	1	1
D	0/1	1	1	1

(b)

Fig. 4. Two 4-cardinality generalized (a) deterministic and (b) non-deterministic dominance schemes with $\vec{A} = \{A, B, C, D\}$.

As in the Ng-Wong and additive dominance schemes, the generalized dominance scheme is unbiased toward 0 or 1

because the total probability of creating a phenotypic allele 0 or 1 is exactly 0.5. And as in the modified additive scheme [14], for the generalized dominance scheme we can integrate dominance change via demoting or promoting genotypic alleles by one grade as follows. If a dominance change is triggered by some environmental change detection scheme, for each gene locus one of the two genotypic alleles is randomly chosen to undergo the following procedure:

- If the phenotypic allele at this locus is 1, then demote the chosen genotypic allele A_i to A_{i-1}, unless it is A_1.
- If the phenotypic allele at this locus is 0, then promote the chosen genotypic allele A_i to A_{i+1}, unless it is A_c.

For the generalized dominance scheme, increasing the value of c will increase the diversity of genotypic alleles and changing the formula from Eq. (2) to Eq. (1) will increase the diversity of phenotypic alleles due to the uncertainty introduced in the dominance scheme. In the following section, we will investigate the effect of these two aspects on the performance of DGAs via experiments.

IV. Experimental Study

A. Dynamic Test Environments

The DOP generator proposed in [23], [25] is used to construct dynamic test environments. This generator can construct DOPs from any binary-encoded stationary function $f(\vec{x})$ as follows. Suppose the environment changes every τ generations. For each environment k, an XORing mask $\vec{M}(k)$ is incrementally generated as follows:

$$\vec{M}(k) = \vec{M}(k-1) \oplus \vec{T}(k), \tag{3}$$

where "$\oplus$" is a bitwise exclusive-or (XOR) operator and $\vec{T}(k)$ is an intermediate binary template for environment k. $\vec{T}(k)$ is created with $\rho \times l$ ($\rho \in (0.0, 1.0]$) random loci set to 1 while the remaining loci set to 0. For the first initial environment $k = 1$, $\vec{M}(1)$ is set to a zero vector.

An individual at generation t are evaluated as follows:

$$f(\vec{x}, t) = f(\vec{x} \oplus \vec{M}(k)), \tag{4}$$

where $k = \lceil t/\tau \rceil$ is the environmental index at time t. With this XOR generator, τ and ρ control the speed and severity of environmental changes respectively. Smaller τ means faster changes while bigger ρ means severer changes.

Three 100-bit binary functions are selected as base stationary functions to construct dynamic test environments. All the three functions have an optimum fitness of 100. The first is the well-known *OneMax* function that aims to maximize the number of ones in a binary string. The second one is a variant of Forrest and Mitchell's *Royal Road* function [6], which consists of 25 contiguous 4-bit building blocks (BBs). Each BB of *Royal Road* contributes 4 to the total fitness if all bits inside the BB have the allele of one; otherwise, it contributes 0. The third function also consists of 25 contiguous 4-bit building blocks. Each building block, BB_i, for the third function is fully *deceptive* and is defined as follows.

$$f(BB_i) = \begin{cases} 4, & u(BB_i) = 4 \\ 3 - u(BB_i), & u(BB_i) < 4, \end{cases} \tag{5}$$

where $u(BB_i)$ is the unitation function of BB_i, which returns the number of ones inside BB_i.

Dynamic environments are constructed from each of the three base functions using the aforementioned dynamic problem generator. For each dynamic environment, the landscape is periodically changed every τ generations during the run of a GA. In order to compare the performance of DGAs in different dynamic environments, the parameters τ is set to 10, 25 and 50 and ρ is set to 0.1, 0.2, 0.5, and 1.0 respectively. Hence, a series of 12 DOPs with 3 values of τ times 4 values of ρ are constructed from each stationary function.

B. Experimental Design

Two sets of experiments were carried out on the above constructed DOPs to investigate the effect of the cardinality of genotype alleles and uncertainty in the dominance scheme respectively. For the first set, we test DGAs with the non-deterministic generalized dominance scheme and set the cardinality c to 2, 3, 4, 6, and 8. The DGAs are denoted as *c-NDDGA* accordingly. For the second set, we fix c to 4 and 8 and test DGAs with non-deterministic and deterministic dominance schemes. The DGAs with the generalized deterministic dominance scheme are denoted as *c-DDDGA* accordingly.

For all DGAs, parameter settings are as follows: generational with population size $n = 100$, uniform crossover with $p_c = 0.6$, bit flip mutation with $p_m = 0.01$, standard tournament selection with tournament size of 2, and elitism of size 1. For all DGAs, the dominance change scheme is triggered whenever the environment changes.

For each experiment of a DGA on a DOP, 50 independent runs were executed with the same set of random seeds. For each run 50 environmental changes were allowed. For each run the best-of-generation fitness was recorded every generation. The overall performance of an algorithm on a DOP is defined as:

$$\overline{F}_{BOG} = \frac{1}{G} \sum_{i=1}^{G} \left(\frac{1}{50} \sum_{j=1}^{50} F_{BOG_{ij}} \right), \tag{6}$$

where $G = 50 * \tau$ is the total number of generations for a run and $F_{BOG_{ij}}$ is the best-of-generation fitness of generation i of run j. The off-line performance $\overline{F}_{BOG}$ is the best-of-generation fitness averaged over 50 runs and then averaged over the data gathering period.

C. Experimental Results and Analysis

The experimental results of the first and second set of experiments are plotted in Fig. 5 and 6 respectively. The corresponding statistical results of comparing DGAs by one-tailed t-test with 98 degrees of freedom at a 0.05 level of significance are given in Table I and II respectively. The t-test result regarding $Alg.\ 1 - Alg.\ 2$ is shown as "$+$", "$-$", "$s+$" or "$s-$" when $Alg.\ 1$ is better than, worse than,

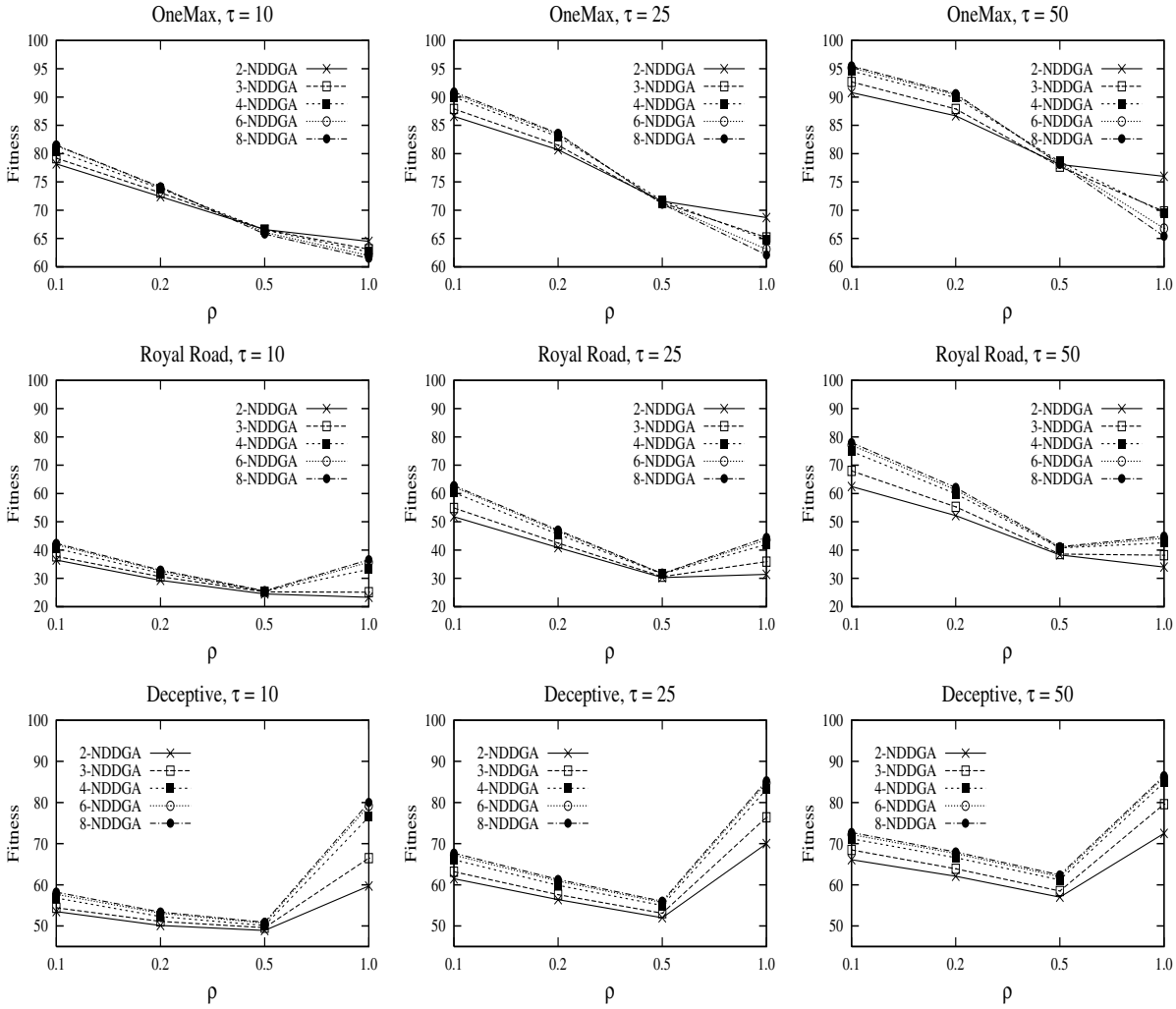


Fig. 5. Experimental results of comparing NDDGAs with different cardinalities on dynamic test problems.

TABLE I

THE t-TEST RESULTS OF COMPARING NDDGAS WITH DIFFERENT CARDINALITIES ON DOPS.

t-test Result	*OneMax*				*Royal Road*				*Deceptive*			
$\tau = 10$, $\rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
3-NDDGA − 2-NDDGA	$s+$	$s+$	$+$	$s-$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
4-NDDGA − 3-NDDGA	$s+$	$s+$	$s-$	$s-$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
6-NDDGA − 4-NDDGA	$s+$	$s+$	$s-$	$s-$	$s+$	$s+$	$+$	$s+$	$s+$	$s+$	$s+$	$s+$
8-NDDGA − 6-NDDGA	$+$	$+$	$s-$	$s-$	$s+$	$s+$	$+$	$s+$	$s+$	$s+$	$s+$	$s+$
$\tau = 25$, $\rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
3-NDDGA − 2-NDDGA	$s+$	$s+$	$s-$	$s-$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
4-NDDGA − 3-NDDGA	$s+$	$s+$	$s+$	$s-$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
6-NDDGA − 4-NDDGA	$s+$	$s+$	$s-$	$s-$	$s+$	$s+$	$+$	$s+$	$s+$	$s+$	$s+$	$s+$
8-NDDGA − 6-NDDGA	$s+$	$s+$	$s-$	$s-$	$s+$	$s+$	$-$	$s+$	$s+$	$s+$	$s+$	$s+$
$\tau = 50$, $\rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
3-NDDGA − 2-NDDGA	$s+$	$s+$	$s-$	$s-$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
4-NDDGA − 3-NDDGA	$s+$	$s+$	$s+$	$s-$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
6-NDDGA − 4-NDDGA	$s+$	$s+$	$s-$	$s-$	$s+$	$s+$	$+$	$s+$	$s+$	$s+$	$s+$	$s+$
8-NDDGA − 6-NDDGA	$s+$	$s+$	$s-$	$s-$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$

significantly better than or significantly worse than $Alg.$ 2 respectively. The dynamic behaviour of several DGAs with respect to best-of-generation fitness against generations on DOPs with $\tau = 50$ and $\rho = 0.1$ and $\rho = 1.0$ is plotted in Fig. 7 and 8 respectively. In Fig. 7 and 8, the first 10 environmental changes, i.e., 500 generations, are shown and the data were averaged over 50 runs. From the tables and figures, several results can be observed.

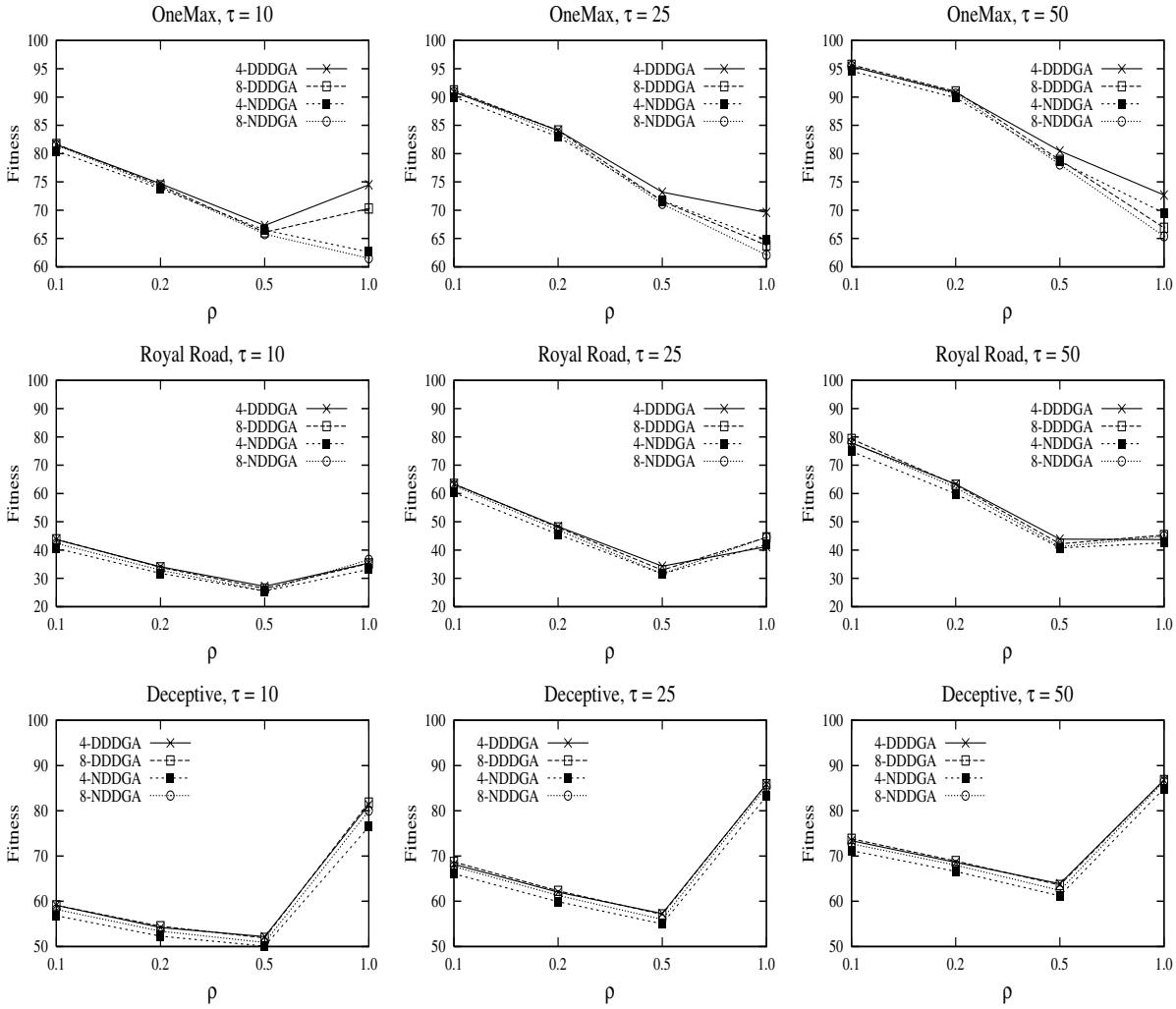
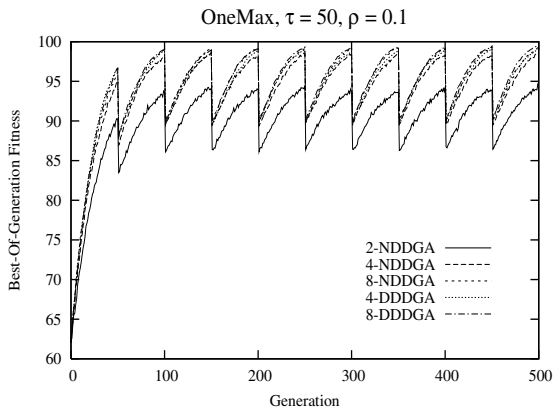


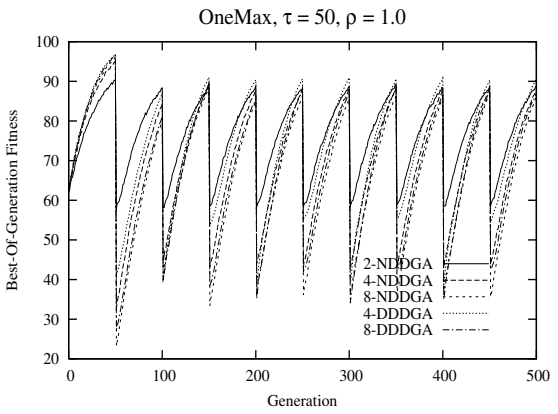
Fig. 6. Experimental results of comparing NDDGAs and DDDGAs on dynamic test problems.

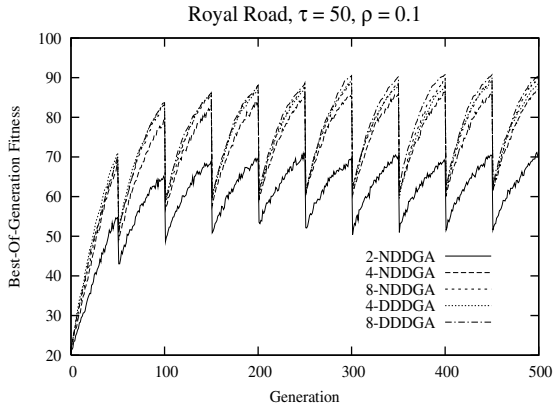
TABLE II
THE t-TEST RESULTS OF COMPARING NDDGAS AND DDDGAS ON DOPS.

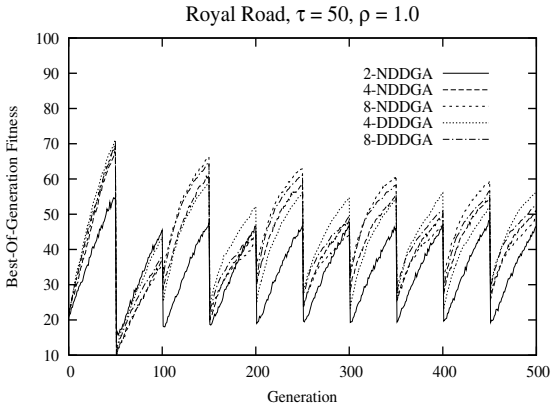
t-test Result	*OneMax*				*RoyalRoad*				*Deceptive*			
$\tau = 10$, $\rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
4-DDDGA − 4-NDDGA	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
8-DDDGA − 8-NDDGA	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s-$	$s+$	$s+$	$s+$	$s+$
8-DDDGA − 4-DDDGA	$+$	$s-$	$s-$	$s-$	$+$	$-$	$s-$	$-$	$+$	$s+$	$s-$	$s+$
$\tau = 25$, $\rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
4-DDDGA − 4-NDDGA	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s-$	$s+$	$s+$	$s+$	$s+$
8-DDDGA − 8-NDDGA	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$+$	$s+$	$s+$	$s+$	$s+$
8-DDDGA − 4-DDDGA	$s+$	$+$	$s-$	$s-$	$+$	$-$	$s-$	$s+$	$s+$	$s+$	$s-$	$+$
$\tau = 50$, $\rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
4-DDDGA − 4-NDDGA	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
8-DDDGA − 8-NDDGA	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
8-DDDGA − 4-DDDGA	$s+$	$s+$	$s-$	$s-$	$s+$	$-$	$s-$	$s+$	$s+$	$s+$	$s-$	$+$

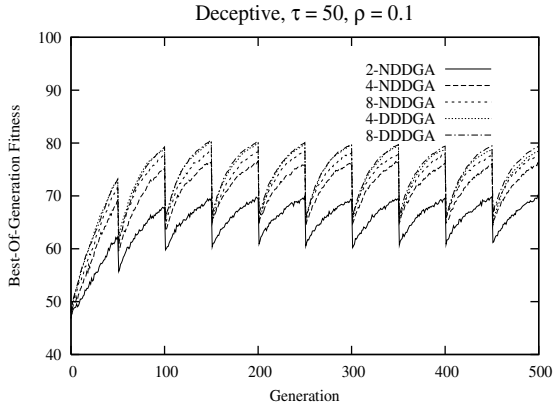
First, from Fig. 5 and Table I it can be seen that the performance of NDDGA rises when the value of c is increased on most DOPs. This result shows the efficiency of increasing the diversity in the genotypic representation. When the value of c is raised from 2 to 3 and from 3 to 4 the performance of NDDGA is significantly improved on almost all dynamic test problems except on dynamic OneMax problems with large values of ρ, see the t-test results regarding 3-NDDGA − 2-NDDGA and 4-NDDGA−3-NDDGA in Table I. For example, on the dynamic Royal Road function with $\tau = 50$ and $\rho = 0.1$, when c is raised from 2 to 3 to 4, the performance of NDDGA is improved from $\overline{F}_{BOG}$(2-NDDGA) $= 62.5$











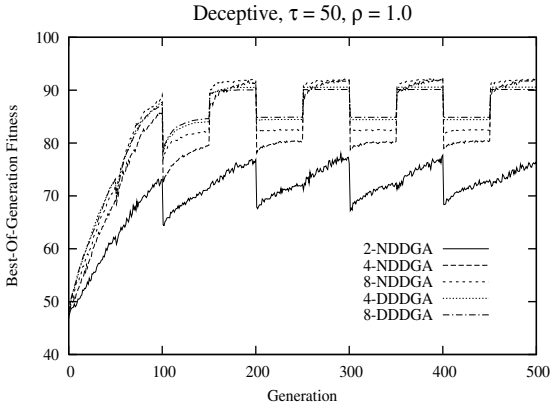


Fig. 7. Dynamic behaviour of DGAs on DOPs with $\tau = 50$ and $\rho = 0.1$.

Fig. 8. Dynamic behaviour of DGAs on DOPs with $\tau = 50$ and $\rho = 1.0$.

to $\overline{F}_{BOG}$(3-NDDGA) $= 68.0$ and to $\overline{F}_{BOG}$(4-NDDGA) $= 74.9$. When c is further raised to 6 and 8, the performance gain is much smaller, e.g., $\overline{F}_{BOG}$(6-NDDGA) $= 77.1$ and $\overline{F}_{BOG}$(8-NDDGA) $= 78.0$. The effect of the cardinality can be more clearly observed from the dynamic behaviour of NDDGAs, as shown in Fig. 7. In Fig. 7, every time the environment changes 4-NDDGA and 8-NDDGA can climb to a much higher fitness level than 2-NDDGA can do.

However, when $\rho = 1.0$, i.e., the environment oscillates between two opposite fitness landscape, the effect of c on the performance of NDDGAs is quite different on the dynamic OneMax problems than on the dynamic Royal Road and Deceptive problems. When the value of c increases, the performance of NDDGAs decreases on the dynamic OneMax problems while increases on the dynamic Royal Road and Deceptive problems, see the t-test results regarding the $\rho = 1.0$ column. This result can be further observed from the dynamic behaviour of NDDGAs in Fig. 8. In Fig. 8, it can be seen that on the dynamic OneMax problem, each time the environment changes the performance of 2-NDDGA drops much less than 4-NDDGA and 4-NDDGA drops much less than 8-NDDGA. In the contrast, on the dynamic Royal Road and Deceptive problems, 8-NDDGA manages to keep a higher fitness level than 4-NDDGA and 4-NDDGA keeps a higher fitness level than 2-NDDGA across the dynamic environments.

Second, from Fig. 6 and Table II it can be seen that DDDGAs significantly outperform corresponding NDDGAs with the same c on most DOPs, see the t-test results regarding 4-DDDGA−4-NDDGA and 8-DDDGA−8-NDDGA in Table II. This result shows that the existence of uncertainty in the dominance scheme is disadvantageous for the performance of DGAs. This is because for each given environment the allele for each locus in the optimal solution is deterministic for the test DOPs, which prefers deterministic phenotypic alleles. This result can be seen from the dynamic behaviour of DGAs in Fig. 7 and Fig. 8, where 4-DDDGA and 8-DDDGA can climb to or maintain a higher fitness level than 4-NDDGA and 8-NDDGA respectively during each dynamic environment.

Third, when comparing the performance of 8-DDDGA over 4-DDDGA, it can be seen that there is no clear winner between them on the DOPs. It seems 8-DDDGA performs better than 4-DDDGA on most DOPs with smaller ρ while worse than 4-DDDGA on most DOPs with bigger ρ. This result can be observed from the dynamic behaviour of 4-DDDGA and 8-DDDGA in Fig. 7 and Fig. 8 respectively.

Another relevant result is that the performance difference between 8-DDDGA and 4-DDDGA is much less than that between 4-DDDGA and 4-NDDGA.

V. Conclusions

Integrating diploidy representation and dominance mechanisms into traditional GAs has long been one approach studied by researchers to enhance their performance for dynamic optimization problems. This paper investigates the two important design factors for DGAs: the cardinality of genotypic representation and the existence of uncertainty in the genotype-to-phenotype mapping. For this purpose, a generalized dominance mechanism is proposed for DGAs in dynamic environments. With this generalized dominance scheme we can conveniently adjust the cardinality of genotypic representation and switch on or off the uncertainty in the dominance map. The effect of these two aspects for DGAs were experimentally studied based on a series of constructed dynamic test problems.

From the experimental results and relevant analysis, two major conclusions can be drawn on the dynamic test environments. First, increasing the cardinality of the genotypic representation improves the performance of DGAs in dynamic environments. It seems setting the cardinality of the genotypic representation in the range of $[4, 8]$ is a good choice for DGAs. Second, the existence of uncertainty in the dominance scheme significantly degrades the performance of DGAs in dynamic environments.

Generally speaking, our preliminary experiments indicate that the diversity game in designing DGAs for dynamic environments should be played in the genotypic level instead of in the dominance scheme. The result observed can be used to guide the design of new DGAs for dynamic environments. Combining the generalized dominance mechanism with other advanced dominance changing schemes is now under investigation. We also believe that combining the generalized dominance scheme with other memory schemes will further improve the performance of DGAs in dynamic environments.

References

[1] J. Branke. Memory enhanced evolutionary algorithms for changing optimization problems. *Proc. of the 1999 Congress on Evol. Comput.*, vol. 3, pp. 1875-1882, 1999.

[2] J. Branke, T. Kaußler, C. Schmidth, and H. Schmeck. A multi-population approach to dynamic optimization problems. *Proc. of the Adaptive Computing in Design and Manufacturing*, pp. 299–308, 2000.

[3] J. Branke. *Evolutionary Optimization in Dynamic Environments*. Kluwer Academic Publishers, 2002.

[4] H. G. Cobb and J. J. Grefenstette. Genetic algorithms for tracking changing environments. *Proc. of the 5th Int. Conf. on Genetic Algorithms*, pp. 523–530, 1993.

[5] E. Collingwood, D. Corne, and P. Ross. Useful diversity via multiploidy. *Proc. of the 1996 Int. Conf. on Evol. Computing*, 1996.

[6] S. Forrest and M. Mitchell. Relative building-block fitness and the building-block hyperthesis. *Foundations of Genetic Algorithms 2*,1993.

[7] D. E. Goldberg and R. E. Smith. Nonstationary function optimization using genetic algorithms with dominance and diploidy. *Proc. of the 2nd Int. Conf. on Genetic Algorithms*, pp. 59-68, 1987.

[8] J. J. Grefenstette. Genetic algorithms for changing environments. *Parallel Problem Solving from Nature II*, pp. 137-144, 1992.

[9] B. S. Hadad and C. F. Eick. Supporting poliploidy in genetic algorithms using dominance vectors. *Proc. of the 6th Int. Conf. on Evolutionary Programming*, pp. 223-234, 1997.

[10] D. L. Hartl and E. W. Jones. *Genetics: Principles and Analysis*. Jones and Bartllet Publishers, Inc., 1998.

[11] J. H. Holland. *Adaptation in Natural and Artificial Systems*, Ann Arbor, University of Michigan Press, 1975.

[12] R. B. Hollstien. *Artificial Genetic Aadaptation in Computer Control Systems*, Doctoral Dissertation, University of Michigan, 1971.

[13] Y. Kim, J. K. Kim, S. Lee, C. Cho and L. Hyung. Winner take all strategy for a diploid genetic algorithm. *Proc. of the 1st Asia-Pacific Conf. on Simulated Evolution and Learning*, 1996.

[14] 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.

[15] S. J. Louis and Z. Xu. Genetic algorithms for open shop scheduling and re-scheduling. *Proc. of the 11th ISCA Int. Conf. on Computers and their Applications*, pp. 99-102, 1996.

[16] N. Mori, H. Kita and Y. Nishikawa. Adaptation to changing environments by means of the memory based thermodynamical genetic algorithm. *Proc. of the 7th Int. Conf. on Genetic Algorithms*, pp. 299-306, 1997.

[17] 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.

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

[19] A. Simões and E. Costa. An immune system-based genetic algorithm to deal with dynamic environments: diversity and memory. *Proc. of the 6th Int. Conf. on Neural Networks and Genetic Algorithms*, pp. 168-174, 2003.

[20] K. Trojanowski and Z. Michalewicz. Searching for optima in non-stationary environments. *Proc. of the 1999 Congress on Evol. Comput.*, pp. 1843-1850, 1999.

[21] A. Ş. Uyar and A. E. Harmanci. Investigation of new operators for a diploid genetic algorithm. *Proc. of SPIE: Applications and Science of Neural Netwroks, Fuzzy Systems and Evolutionary Computation II*, 1999.

[22] A. Ş. Uyar and A. E. Harmanci. A new population based adaptive dominance change mechanism for diploid genetic algorithms in dynamic environments. *Soft Computing*, vol. 9, no. 11, pp. 803-815, 2005.

[23] S. Yang. Non-stationary problem optimization using the primal-dual genetic algorithm. *Proc. of the 2003 Congress on Evol. Comput.*, vol. 3, pp. 2246-2253, 2003.

[24] S. Yang. Memory-based immigrants for genetic algorithms in dynamic environments. *Proc. of the 2005 Genetic and Evol. Comput. Conference*, vol. 2, pp. 1115-1122, 2005.

[25] 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.