PDGA: the Primal-Dual Genetic Algorithm

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

Abstract. Genetic algorithms (GAs) are a class of search algorithms based on principles of natural evolution. Hence, incorporating mechanisms used in nature may improve the performance of GAs. In this paper inspired by the mechanisms of complementarity and dominance that broadly exist in nature, we present a new genetic algorithm — Primal-Dual Genetic Algorithm (PDGA). PDGA operates on a pair of chromosomes that are primal-dual to each other through the primal-dual mapping, which maps one to the other with a maximum distance away in a given distance space in genotype. The primal-dual mapping improves the exploration capacity of PDGA and thus its searching efficiency in the search space. To test the performance of PDGA, experiments were carried out to compare PDGA over traditional simple GA (SGA) and a peer GA, called Dual Genetic Algorithm (DGA), over a typical set of test problems. The experimental results demonstrate that PDGA outperforms both SGA and DGA on the test set. The results show that PDGA is a good candidate genetic algorithm.

1 Introduction

Genetic algorithms (GAs) are a class of search algorithms based on principles of natural selection and population genetics. They are widely used for optimization problem solving and machine learning. To realize their task, GAs maintain a population of individuals, usually encoded as fixed length binary strings. Each individual is associated a fitness value according to the problem being solved, usually called objective function. GAs iteratively generate new population by selecting individuals with relatively higher fitness from the present population for reproduction and then performing recombination and mutation operations on these selected individuals. Based on Holland's SGA [8], there have been many variations and extensions developed, involving GA's macro-structure and micro-structure [5].

Most GAs studied so far are haploidy-based, i.e., they operate on a set of single-stranded chromosomes. However, haploid genotype is the simplest genotype found in nature. In nature, most organisms have a genotype form of diploid, i.e., a set of double-stranded chromosomes. As the genetic material that is propagated from generation to generation, deoxyribonucleic acid (DNA) molecules consist of two long chains twisted around one another in a double-stranded helix [7]. In eukaryotes DNA combines with proteins to form chromosomes. The two chains in DNA are held together by base pairs. DNA consists of four kinds of bases joined to a sugar-phosphate backbone. The four bases are *adenine* (A), *guanine* (G), *thymine* (T) and *cytosine* (C). They are paired in DNA according to a *complementary pairing rule*: A pairs with T and G pairs with C. This pairing rule results in two complementary strands in DNA.

When the double-stranded chromosomes are exposed to the environment of the organism, dominance mechanism (an important genotype-to-phenotype mapping mechanism) comes to effect by expressing dominant genes (segments of DNA) while repressing recessive genes.

In this paper, inspired by the phenomena of complementarity and dominance mechanisms that broadly exist in nature, we propose a new genetic algorithm, called primal-dual genetic algorithm (PDGA). Within PDGA, each chromosome is associated with a dual chromosome that is of maximum distance away from it in genotype in a given distance space, e.g., the Hamming distance space. When a new population is created, a set of individuals is selected to evaluate their dual chromosomes to give their dual chromosomes that are superior chances to be expressed into the next generation. Through the primal-dual mapping between the primal-dual chromosomes, PDGA's exploration capacity in the search space is improved and thus its searching efficiency as a whole is improved.

The rest of this paper first describes the framework of PDGA, next compares PDGA with Collard and his co-workers' Dual Genetic Algorithm (DGA) [2, 3], then provides experimental results comparing PDGA with SGA and DGA on a typical test suite, and finally presents conclusions as well as discussions on potential future works relevant to PDGA.

2 Primal-Dual Genetic Algorithm

2.1 Definitions

Definition 1. *A chromosome that is explicitly recorded in the population of a GA is called a* primal chromosome. *Given a distance space and relevant distance measure, the chromosome that has the maximum distance to a primal chromosome is called its* dual chromosome. *The transformation function from a primal chromosome to its dual is called* primal-dual mapping.

Given a primal chromosome x, its dual is denoted by $x' = dual(x)$ where $dual(\cdot)$ is the primal-dual mapping function. Note that x and x' are primal-dual to each other with respect to the given distance space, i.e., $x = dual(x') = dual(dual(x))$. For GAs with binary-encoded representation of genotype naturally the Hamming distance, i.e., the number of locations at which corresponding bits of two chromosome differ, can be used as the distance measure between two chromosomes. A pair of chromosomes is then said to be primal-dual to each other if their Hamming distance is the maximum (equal to their length) in the search space. In other words, given a chromosome $x = (x_1, x_2, \cdots, x_L) \in I = \{0, 1\}^L$ of fixed length L, its dual is defined as its complementary chromosome, i.e., $x' = \bar{x} = (\bar{x}_1, \bar{x}_2, \cdots, \bar{x}_L) \in I$ where $\bar{x}_i = 1 - x_i$ $(i = 1, \cdots, L)$. In this case we can say that x is mapped to its dual x' by the Hamming distance mapping, vice versa. In this paper we will deal with binary-encoded GAs and naturally use the Hamming distance as the primal-dual mapping function.

Definition 2. *If the two chromosomes of a primal-dual pair have different fitness, the chromosome with higher fitness is called a* superior chromosome *while the one with lower fitness is an* inferior chromosome. *If the primal-dual chromosomes have equal fitness, they are called* tie chromosomes *or they are said to form a* tie pair.

Here we use the definition of superior and inferior to deal with competition between primal-dual chromosomes. It is different from the natural gene expression mechanism where

```
begin
  parameterize($N$, $P_c$, $P_m$);
  $t := 0$;
  initializePopulation($P(0)$);
  evaluatePopulation($P(0)$);
  $D(0)$ := selectForDualEvaluation($P(0)$);
  for each individual $x$ in $D(0)$ do    {evaluate $D(0)$}
    evaluateDualChromosome($x'$);    {$x' = dual(x)$}
    if $f(x') > f(x)$ then $x := x'$;    {replace $x$ with its dual $x'$}
  endfor;
  repeat
    $P'(t)$ := selectForReproduction($P(t)$);
    recombine($P'(t)$);
    mutate($P'(t)$);
    evaluatePopulation($P'(t)$);
    $D(t)$ := selectForDualEvaluation($P'(t)$);
    for each individual $x$ in $D(t)$ do    {evaluate $D(t)$}
      evaluateDualChromosome($x'$);    {$x' = dual(x)$}
      if $f(x') > f(x)$ then $x := x'$;    {replace $x$ with its dual $x'$}
    endfor;
    $t := t + 1$;
  until $terminated = \mathtt{true}$;    {e.g., $t > t_{max}$ (maximum generation)}
end;
```

Figure 1: Pseudocode for PDGA.

the concept of dominant and recessive genes is applied. In nature the double-stranded chromosomes compete at gene level. When a dominant gene and a recessive gene meet and compete together the dominant gene dominates the recessive one and gets expressed in phenotype while the recessive gene is not expressed. In the proposed PDGA primal-dual chromosomes compete at the chromosome level and the superior chromosome of a primal-dual pair is not always expressed in the population since not all the primal chromosomes are subject to primal-dual mapping operation or dual evaluation.

Definition 3. *A primal-dual mapping operation or dual evaluation is called* valid *if the obtained dual chromosome is superior to the primal chromosome; otherwise, it is called* invalid.

Obviously, valid primal-dual mapping operations or dual evaluations are expected to be beneficial to GA's performance. This is just what PDGA pursues.

2.2 Framework of PDGA

With above definitions, we can now give out the framework of PDGA in the form of pseudocode in Figure 1, where N, P_c, P_m are the population size, crossover probability and mutation probability respectively, and $f(x)$ denotes the fitness of an individual x. From Figure 1, it can be seen that PDGA differs from traditional SGA in that PDGA introduces into SGA a process of selecting primal chromosomes for dual evaluations and replacing selected inferior chromosomes with their superior duals. Except for this, all the genetic operations including reproduction selection, crossover and mutation are the same for SGA and PDGA.

Within PDGA, when a new population $P'(t)$ at generation t has just been created and evaluated and before the next generation starts, a set $D(t)$ of primal individuals from $P'(t)$ are selected to evaluate their duals. For a selected primal chromosome $x \in D(t)$, if its dual $x' = dual(x)$ is evaluated to be better than x, x is replaced with its dual x'; otherwise, x will stay intact and keep expressed into the next generation. That is, within PDGA only valid primal-dual mappings take effect to give dual chromosomes that are superior chances to be expressed. This is similar to the dominance mechanism used in nature and is reasonable to improve the average fitness of the population and protect superior primal chromosomes found so far. Now what is left unsolved with PDGA is how to select individuals from $P'(t)$ to form the set $D(t)$ for dual chromosome evaluation. This is analyzed and described below.

2.3 Selection Scheme for Dual Chromosome Evaluation

From above discussion, it is clear that the scheme of selecting primal chromosomes for dual evaluation should try to maximize valid primal-dual mappings. With this goal associated, the selection scheme should concern two questions: 1. Which primal chromosomes in a population should be selected? and 2. How many primal chromosomes should be selected?

The first question is relatively easy to answer. Since only valid primal-dual mappings take effect and performing primal-dual mapping on chromosomes with low fitness is more likely to be valid, we can select primal chromosomes with low fitness from $P'(t)$ to form $D(t)$.

To answer the second question, let us briefly look at the dynamic behavior of traditional genetic algorithms. Holland [8] first proposed the notation of *schema* to describe a set of binary strings of fixed length that have similarities at certain positions. Holland worked out the *schema theorem* for GAs that use the fitness proportionate selection, 1-point crossover and bit mutation. The schema theorem states that short, low-order, better than average schemas receive an exponentially increasing number of trials in the subsequent generations. Stephens and Waelbroeck [10] have derived a new schema theorem based on the concept of *effective fitness* showing that schemas with higher than average effective fitness receive an exponentially increasing number of trials over time.

Both schema theorems also indicate that schemas or strings with less than average fitness or average effective fitness receive an exponentially decreasing number of trials over time. This means that inferior primal chromosomes in the population will decrease at an exponential rate since they usually have less than average fitness or average effective fitness. To test this, experiments were carried out running SGA on a typical test suite, to be described in Section 4 later on. The operator and parameter settings of SGA are described in Section 5.1. On each test problem, 100 runs of SGA were executed[1]. Let $n_{inf}(t)$ denote the actual number of inferior primal chromosomes in the population $P'(t)$ at generation t and $r_{inf}(t) = n_{inf}(t)/N$ denote the ratio of inferior primal chromosomes in $P'(t)$. For each run $r_{inf}(t)$ was recorded over generation t. The experimental results, averaged over 100 runs, are shown in Figure 2. From Figure 2 it can be seen that $r_{inf}(t)$ decreases approximately exponentially over time to

[1]In order to help analyzing the main experimental results given in Section 5.2 the same 100 random seeds as in the main experiments were used here to create initial populations.

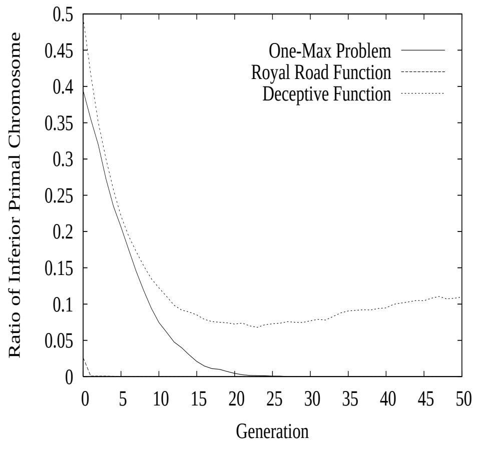

Figure 2: Ratio of inferior primal chromosome in the population against generations of SGA on the test set.

0 or an approximately stable value.

The schema theorems and our preliminary experiments indicate that an exponentially decreasing scheme should be used to decide the number of primal individuals to be selected for dual evaluation. Let $n_d(t)$ denote the actual number of primal chromosomes selected from $P'(t)$ for dual evaluation at generation t, i.e., $n_d(t) = |D(t)|$. Ideally $n_d(t)$ should be equal to $n_{inf}(t)$. It is difficult to achieve this since $n_{inf}(t)$ is unknown in advance. However, it is clear that $n_d(t)$ should decrease exponentially over time t.

Given the above discussions, we summarize the scheme of selecting primal chromosomes from the population $P'(t)$ for dual evaluation in pseudocode in Figure 3. In Figure 3, G is the generational gap to update the value of variable $n_d(t)$, α and β $(0 < \alpha, \beta < 1)$ control the initial value of $n_d(t)$ and its decreasing speed respectively, γ $(0 < \gamma < N)$ is the minimal number of primal chromosomes to be selected for dual evaluation, and $\lceil \cdot \rceil$ is the ceiling function. With this selection scheme, the value of $n_d(t)$ starts from an initial value, decreases by a factor of β every G generations until a preset minimum number γ is reached, and thereafter keeps unchanged. For each generation t, $n_d(t)$ least fit primal chromosomes are selected for dual evaluations. Incorporating Figure 3 into Figure 1 completes the framework of PDGA.

3 Collard and Co-workers' Dual Genetic Algorithm (DGA)

Collard and his co-workers have proposed a genetic algorithm, first called Double-based Genetic Algorithm [2] and then renamed Dual Genetic Algorithm [3]. DGA also manipulates pairs of twins in the population The initial aim of DGA is to improve the performance of a GA by adding one single meta-bit in front of the regular bits. This single meta-bit is similar to introns (i.e. non-coding genes) in DNA molecule that can influence the expression of exons (regular coding genes). This meta-bit in DGA alters the phenotype of the overall chromosome. If the meta-bit is activated ("1") all regular bits are translated to their complement for fitness evaluation, otherwise they keep their original value for fitness evaluation. Consequently, there may exist complementary individuals in the population that represent the same phenotype while have fundamentally different genotype. The added meta-bit undergoes the same genetic operations in DGA as other regular bits do.

From the above descriptions, it can be seen that both PDGA and DGA are inspired by the

```
**Procedure** selectForDualEvaluation($P'(t)$):

**begin**
  **if** $t = 0$ **then**
    $n_d(t) := \lceil \alpha * N \rceil$;    $\{\alpha$ controls $n_d(t)$'s initial value, i.e., $n_d(0)\}$
  **else if** $(t \% G) = 0$     $\{$update $n_d(t)$ every $G$ generations$\}$
    $n_d(t) :=$ max $\{\lceil \beta * n_d(t - G) \rceil, \gamma\}$;   $\{\beta$ controls $n_d(t)$'s decreasing speed$\}$
  **endif**;
  $P''(t) :=$ sortPopulation($P'(t)$);  $\{$sort population in increasing fitness order$\}$
  truncate the first $n_d(t)$ individuals in $P''(t)$ to form $D(t)$;
**end**;
```

Figure 3: Pseudocode for selecting primal chromosomes for dual evaluation.

complementary mechanism in DNA duplex structure. In DGA mutating the meta-bit enables an individual to make a long jump to its complement in the search space while in PDGA when an inferior primal chromosome is selected into the set $D(t)$, it will get the chance to jump to its superior dual with a maximum distance away in the search space with a chosen distance measure. Although both PDGA and DGA make use of the complementary mechanism in nature, physically they are encoded as a single-stranded string instead of as a double-stranded chromosome as in DNA molecule. Hence both can be called *pseudo-diploid* and work on a *pseudo-pair* of complementary chromosomes. Though inspired from similar natural mechanism, PDGA and DGA do have different properties. There are two main differences. First, in DGA the jumping between complementary chromosomes is driven by mutation and hence by chance. It uses no dominance mechanism and is blind in the sense of applying complement mechanism. PDGA is dominance-based using fitness as its dominance mechanism that works at the chromosome level. This is reflected in the selection of low fit chromosomes for evaluating their dual chromosomes and in the fact of only replacing inferior primal chromosome with its superior dual. Second, DGA doubles the size of the search space in genotype by adding a meta-bit while with PDGA the size of the genotypical space remains unchanged.

4 The Test Suite

1. **One-Max Problem**: This problem simply aims to maximize ones in a binary string. The fitness of a string is the number of ones it contains. A string length of 100 bits is used for this study. And the optimal solution has a fitness of 100.

2. **Royal Road Function**: This function is the same as Mitchell, Forrest and Holland's Royal Road function $R1$ [9] that was devised to investigate GA's performance with respect to schema processing and recombination. It is defined on a sixty-four bit string consisting of eight contiguous building blocks of eight bits, each of which contributes $c_i = 8$ ($i = 1, \cdots, 8$) to the total fitness if all of the eight bits are set to one. The fitness of a bit string x is computed by summing the coefficients c_i corresponding to each of the given building blocks s_i of which x is an instance. We denote by $x \in s_i$ the situation that x is an instance of s_i. That is, the Royal Road function is defined as follows: $f(x) = \sum_{i=1}^{i=8} c_i \delta_i(x)$ where

$\delta_i(x) = \{1, \text{if } x \in s_i; 0, \text{otherwise}\}$. The optimal solution $x^* = 11...1$ has a fitness of 64.

3. **Deceptive Function**: Deceptive functions are a family of functions where there exist low-order building blocks that do not combine to form higher-order building blocks: instead they form building blocks resulting in a deceptive solution that is sub-optimal itself or near a sub-optimal solution. Deceptive functions are devised as difficult test functions for GAs. It is even claimed that the only challenging problems for GAs are problems that involve some degree of deception [11]. By explicitly calculating and comparing all schema fitness values, Goldberg [6] devised an order-3 minimum fully deceptive problem as follows:

```
f(000) = 28  f(001) = 26  f(010) = 22  f(011) = 0
f(100) = 14  f(101) = 0   f(110) = 0   f(111) = 30
```

where all the order-1 and order-2 building blocks (e.g., “0**” and “*00”) in the search space are deceptive and will lead the genetic search away from the global optimum “111” and instead toward the local optimum “000”. In this study, we constructed a deceptive function that consists of 10 copies of the above order-3 minimum deceptive subproblem. This function has an optimum fitness of 300.

5 Experimental Study

5.1 Design of Experiment

Experiments were carried out to compare PDGA with traditional SGA and Collard and Aurand’s DGA. All the GAs were generational and used typical genetic operator and parameter settings: 1-point crossover with a fixed crossover probability $p_c = 0.6$, traditional bit mutation with mutation probability $p_m = 0.001$ recommended by De Jong [4], and fitness proportionate selection with the Stochastic Universal Sampling (SUS) [1] and elitist model [4]. The population size N was set to 128 for all the GAs. PDGA-specific parameters were set as follows: $G = 1$, $\alpha = \beta = 0.5$ and $\gamma = 1$. With this setting, the value of $n_d(t)$ starts from $N/2$ and is halved every generation until $n_d(7) = 1$ at generation 7.

For each experiment of combining different GA and test problem, 100 independent runs were executed with the same 100 different random seeds to generate initial populations. For each run, the best-so-far fitness was recorded every 100 evaluations[2] as well as the mean fitness over every 100 evaluations and the maximum allowable number of evaluations was set to 20000. Each experimental result was averaged over 100 independent runs.

5.2 Experimental Result and Analysis

The experimental results on different test functions are shown in Figure 4 through Figure 6 respectively. From these figures, it can be seen that in general PDGA performs better than SGA and DGA on the test problems.

On the One-Max problem, PDGA and SGA perform as well as each other but both perform better than DGA. During early stage of searching, within around 2500 evaluations the mean fitness of the population with PDGA is a little higher than that with SGA. This is because with SGA on One-Max the ratio of inferior chromosomes in the population $r_{inf}(t)$

[2]Here, only those primal chromosomes changed by crossover and mutation operations were evaluated and counted into the number of evaluations. With PDGA all dual evaluations, valid or not, were also counted into the total number of evaluations.

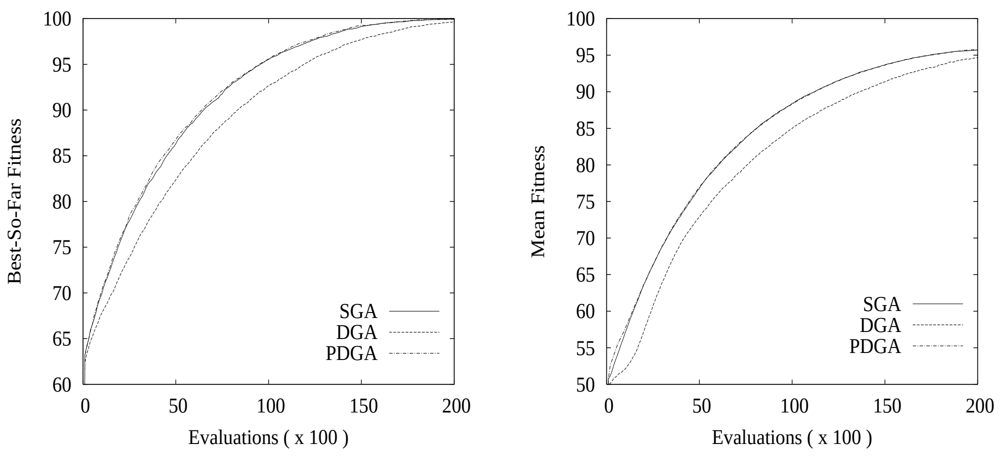

Figure 4: Best-so-far (*Left*) and mean (*Right*) fitness against evaluations of GAs on the One-Max problem.

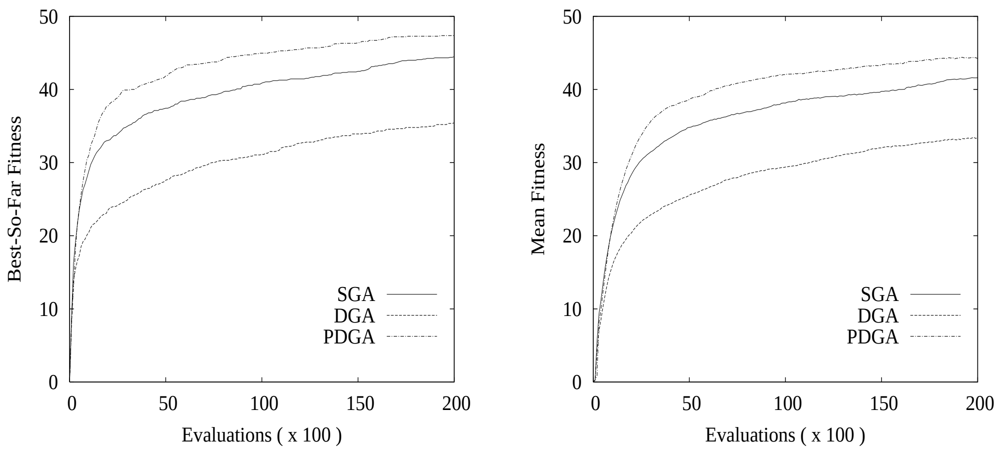

Figure 5: Best-so-far (*Left*) and mean (*Right*) fitness against evaluations of GAs on the Royal Road Function.

decreases from 0.394 at the initial generation to 0.001 at generation 25 (see Figure 2). With PDGA, taking into account that $N = 128$, that we only evaluate primal chromosomes changed by genetic operation and count them together with all dual evaluations into the total number of evaluations, and that the effect of valid primal-dual mappings during early generations, the number of valid primal-dual mappings decreases to near zero after about 2500 evaluations. However, the valid primal-dual mappings with PDGA seem to have no contribution to the best-so-far fitness of the population. This is because the basic building block of One-Max is only one bit (i.e., order-1), which makes it easy for the SGA to find each building block (i.e., “1” at each locus). DGA performs worse than SGA and PDGA on both the best-so-far and mean fitness performances due to its blindness in mutating the meta-bit.

On the Royal Road function, PDGA outperforms both SGA and DGA while SGA outperforms DGA. And the performance difference between GAs is now much bigger than that on the One-Max problem. This is because the basic building blocks are now of order-8 instead of order-1 as in the One-Max problem. The increased size of basic building blocks makes it much harder for SGA to search them and hence makes the valid primal-dual mappings

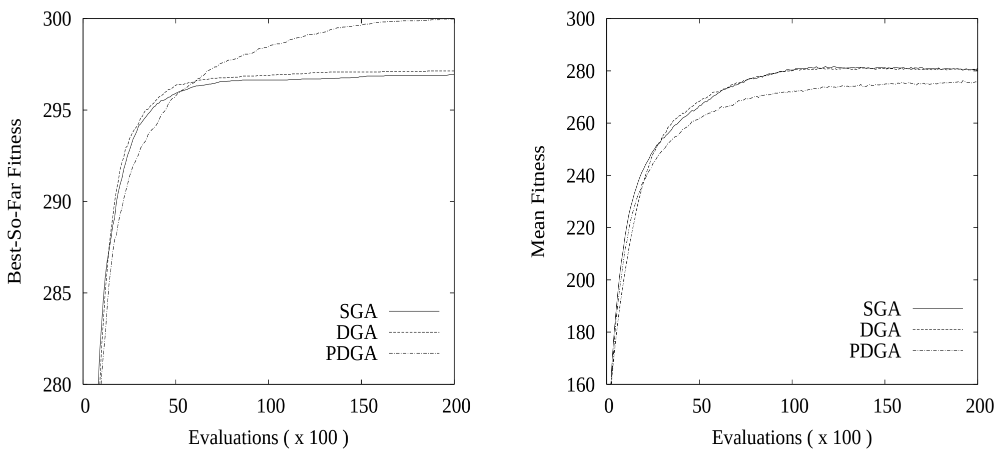

Figure 6: Best-so-far (*Left*) and mean (*Right*) fitness against evaluations of GAs on the Deceptive Function.

(though the ratio $r_{inf}(t)$ is quite small, see Figure 2) with PDGA during the early search stage more precious. For example to achieve the fitness level of 40 (i.e., 5 building blocks), it took PDGA about 2800 and 6300 evaluations while it took SGA about 8700 and 16300 evaluations with respect to best-so-far and mean fitness respectively. For the same reason as on the One-Max problem, DGA is beaten again on Royal Road function but with a heavier degree. For example, it took DGA about 11000 and 14800 evaluations to find only four building blocks (equivalent to the fitness value of 32) with respect to best-so-far fitness and mean fitness respectively.

On the Deceptive function, the situation seems quite different. DGA slightly outperforms SGA while both are beaten by PDGA. Within 20000 evaluations PDGA found the optimal solution in 93 out of the 100 runs, while DGA and SGA only achieved the optimal solution in 16 and 19 out of the 100 runs respectively. One thing to note is that now PDGA seems working better during late stage of searching instead of during early stage as it did on One-Max and Royal Road problems. Another notable thing is that during late searching stage the mean fitness of the population with PDGA is less than that with SGA and DGA. The reason to these two observations is as follows: with PDGA during early stage valid primal-dual mappings from basic units “011”, “101”, “110” to their duals slow down the growth of building blocks “000” and “111”, hence the best-so-far fitness grows a little slower. When certain amount of building blocks “000” and “111” have been built up, valid primal-dual mappings (though a few in number, due to only one dual evaluation per generation), which convert strings with more “000” units to strings with more “111” units, work quite efficiently, pushing the best individual towards optimum faster than SGA and DGA. Meanwhile, these valid mappings give crossover more chances to create non “000” or “111” units on certain loci and hence lower the mean fitness of the population a little.

6 Conclusions and Future Works

Inspired by the complementarity and dominance mechanisms in nature a new variation of genetic algorithm, the primal-dual genetic algorithm, is proposed. PDGA operates on a pseudo-pair of chromosomes, which are primal-dual to each other in the sense of maximum distance in genotype in a given distance space, e.g., the Hamming distance used for the primal-dual mapping in this paper. In PDGA just before entering next generation a set of relatively low fit individuals is selected to give them chance to jump to their superior dual chromosomes.

Experiments were carried out to compare PDGA with traditional SGA and Collard and co-workers' DGA using typical genetic operator and parameter settings on a set of benchmark test problems. The experimental results demonstrate that PDGA outperforms both SGA and DGA. When solving non-deceptive functions PDGA works efficiently during early search stage, while on deceptive functions PDGA works more efficiently when the GA has built up proper deceptive building blocks since they are usually complementary to optimal building blocks [11]. DGA was beaten by PDGA due to its blindness in applying complement mechanism. In general, our experiments indicate that PDGA is a good candidate GA.

In this paper the Hamming distance was used as the primal-dual mapping function. This is quite natural for binary-encoded GAs. However, the principle of PDGA doesn't exclude other distance measures. Here the key idea is to develop a prima-dual mapping policy that maps a primal chromosome to its dual with a maximum distance away in a distance space specific to the problem representation. This way through the primal-dual mapping of two chromosomes, PDGA's exploration capacity in the search space is improved and hence its searching efficiency may be improved. Expanding the prima-dual mapping mechanism to real-encoded or permutation-encoded GAs is thus one future work on PDGA.

Another key point with PDGA is the scheme of selecting primal chromosomes for dual evaluation. In this study, a simple deterministic policy that selects an exponentially decreasing number of least fit chromosomes was applied. A selection scheme that can adaptively adjust the value of $n_d(t)$ over time t may further improve the performace of PDGA, which falls in another future work on PDGA.

References

[1] J. E. Baker (1987). Reducing Bias and Inefficiency in the Selection Algorithms. In J. J. Grefenstelle (ed.), *Proc. 2nd Int. Conf. on Genetic Algorithms*, 14-21.

[2] P. Collard and J. P. Aurand (1994). DGA: An Efficient Genetic Algorithm. In A. Cohn (ed.), *Proc. of the 11th European Conf. on Artificial Intelligence*, 487-491.

[3] P. Collard and C. Escazut (1996). Fitness Distance Correlation in a Dual Genetic Algorithm. In W. Wahlster (ed.), *Proc. of the 12th European Conf. on Artificial Intelligence*, 218-222.

[4] K. A. De Jong (1975). *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD Thesis, University of Michigan, Ann Abor.

[5] D. E. Goldberg (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley.

[6] D. E. Goldberg (1989). Genetic Algorithms and Walsh Functions: Part I, a Gentle Introduction. *Complex Systems*, **3**: 129-152.

[7] D. L. Hartl and E. W. Jones (2001). Genetics: Analysis of Genes and Genomes, 5th ed. Sudbury, MA: Jones & Bartlett Publishers.

[8] J. H. Holland (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press.

[9] M. Mitchell, S. Forrest and J. H. Holland (1992). The Royal Road for Genetic Algorithms: Fitness Landscapes and GA Performance. *Proc. of the 1st European Conference on Artificial Life*, 245-254.

[10] C. Stephens and H. Waelbroeck (1999). Schemata Evolution and Building Blocks. *Evolutionary Computation*, **7**(2): 109-128.

[11] L. D. Whitley (1991). Fundamental Principles of Deception in Genetic Search. In G. J. E. Rawlins (ed.), *Foundations of Genetic Algorithms 1*, 221-241.