

# A New Genotype to Phenotype Mapping Approach for Diploid Genetic Algorithms

A. Şima Uyar, A.Emre Harmancı

Istanbul Technical University, Computer Engineering Department Maslak Istanbul TR-80626 TURKEY  
{[etaner.harmanci](mailto:etaner.harmanci@cs.itu.edu.tr)}@cs.itu.edu.tr

**Abstract.** This study involves genetic algorithms in which a diploid representation of individuals is used. In conformance with the diploid representation, a reproductive scheme which models the meiotic cell division for gamete formation in diploid organisms in nature is employed. A domination strategy is applied for mapping an individual's genotype onto its phenotype. The domination factor for each allele at each location is determined by way of a statistical scan of the population in the previous generation. The effectiveness of this domination mechanism is shown by comparing it with two other major approaches found in literature.

## 1 Introduction

In 1831, Charles Darwin embarked on a five-year voyage on the HMS Beagle as a naturalist. On his return, he published his book *The Origin of Species by Means of Natural Selection* in 1859 [1], in which he related his journeys and observations and documented his theory, 'The Theory of Evolution'. The key idea behind this theory [2] is that all species have descended from other species. His work shows evidence that evolution has actually taken place and he correctly outlines the mechanisms by which it occurred.

Evolution is a two-stage process. In the first stage, random variations among individuals take place. Some of these variations may be inherited and they may either be useful or detrimental to the organism. The second stage is natural selection which is an interaction between an organism and its environment. As a result of this interaction, the organisms which adapt better to their environments leave more offspring than others. Over a very long period of time, evolution leads to diversities between groups of organisms living under different conditions. Evolution is viewed as one of the most important principles in biology and related sciences.

In an optimization problem, the aim is to find the optimal or in some cases the near optimal solution to a problem. Genetic algorithms, which are a class of stochastic, global optimization methods, model some of the principles of natural evolution in solving the problem at hand. In nature, organisms that have some characteristics which make them better than others in some way are more likely to survive and reproduce, thus leave offspring that may inherit the useful characteristic which made the organism more suitable to its environment. This may be seen as an optimization process where nature is searching for the optimal individual for a certain environment. In this sense, some mechanisms found in nature may be modeled in solving artificial optimization problems. For some classes of problems, it may be sufficient to model a simple subset of these natural principles and processes. However, for some other classes of problems that have special requirements and characteristics, a broader subset may lead to better results.

## 2 The Diploid Genetic Algorithm

In nature most complex organisms have a diploid structure, i.e. for each characteristic, the organism has two alleles located on two homolog chromosomes. Even though this seems like a redundancy, it is nature's way of keeping a genetic memory and introducing genetic diversity. In this way, genetic information which may be useful in the future is shielded against the selection process by way of a domination mechanism which masks the recessive allele until a future time when it may become useful for the organism.

There has been some study on diploidy and dominance in genetic algorithms. Most of these are summarized and discussed in [3]. More recent work can be found in [4], [5], [6], [7] and [8]. However, there has been a recent rising interest in the genetic algorithms community to work with dynamic environments. One of the methods a genetic algorithm deals with such environments is using diploidy which may be termed as implicit memory [9]. When diploidy is used, some type of genotype to phenotype mapping is necessary.

## 2.1 The Diploid Algorithm

In the implementation of diploidy in this study, each individual is represented, as seen in Fig. 1, by two chromosome arrays which make up the genotype of the organism, an array to represent the phenotype of the organism, a fitness value and an age indicator which shows for how many generations the individual has survived.



Fig. 1. The Individual.

The algorithm uses the basic concepts employed by the simple genetic algorithm explained in [3] and introduces some new operators and a new genotype to phenotype mapping scheme as given in the following sections. The pseudocode of the diploid algorithm is given in Fig. 2. More detail about the implementation of each step can be found in [10].

```

begin
initialize;
do
  select mating pool;
  form gametes;
  mate;
  mutate;
  for each dead parent
  begin
    form new random individual;
  end;
  select next generation:
  calculate domination array;
until stopping_criteria;

```

Fig. 2. The diploid GA.

## 2.2 Phenotype to Genotype Mapping and Domination

The phenotype of the individual is used in calculating its fitness. Since the individual is diploid, its genotype and phenotype are different. Thus a mechanism is needed to map the genotype onto the phenotype. This mechanism, called domination, is a very important part of diploid genetic algorithms. There has been some research done in this area. These are summarized in [3].

In this study, the approach used involves a statistical scan of each generation. When determining the phenotype, the genotype elements corresponding to that location may either be equal or different. Using  $c_{1i}$  and  $c_{2i}$  to represent the  $i$ -th location on chromosome 1 and chromosome 2 respectively and  $p_i$  to represent the corresponding  $i$ -th location on the phenotype,

if  $c_{1i}=0$  and  $c_{2i}=0$  then  $p_i=0$

if  $c_{1i}=1$  and  $c_{2i}=1$  then  $p_i=1$

In the above cases where the two alleles for the genes on homologue chromosomes are the same, the corresponding phenotype equals that allele but in the case where they are different, i.e. where ( $c_{1i}=0$  and  $c_{2i}=1$ ) or ( $c_{1i}=1$  and  $c_{2i}=0$ ), a method to determine the phenotypic value is needed. In natural organisms, the allele to be seen in the phenotype is the dominant one, so an artificial mechanism to simulate this in artificial systems must be designed. There are several approaches to genotype to phenotype mapping in literature and they are summarized in [3].

In this implementation, a variable, global domination array composed of real numbers in  $[0.0,1.0]$  is used. The length of the array is the same as the chromosome length with each value showing the dominance factor of the allele 1 over the allele 0 corresponding to the same location on the chromosomes. For example, if the alleles on the two chromosomes are different for the  $i$ -th location and if the  $i$ -th entry in the domination array is  $dom_i=0.8$ , the phenotypic value for that location will be 1 with probability 0.8 and 0 with probability 0.2.

The domination array is evaluated at the end of each generation, so in a way it follows the evolution of the individuals. It is calculated using Equation 1.

$$dom_i = \frac{\sum_j p_{ij} * f_j}{\sum_j f_j}, i = 1, 2, \dots, length, j = 1, 2, \dots, size \quad (1)$$

where  $p_{ij}$  is the phenotypic value of the  $j$ -th individual at the  $i$ -th location,  $f_j$  is the fitness value of the  $j$ -th individual,  $length$  is the chromosome length and  $size$  is the population size (the total number of individuals in the population). The  $dom_i$  value will be higher if individuals with the allele 1 in the  $i$ -th location have higher fitnesses compared to those that have allele 0. Since the domination array is one of the driving forces of the population, it is expected that the values corresponding to locations on the phenotype that should be 1 in the optimal solution, should approach 1.0 and 0.0 for the case where the optimal value should be 0. But in the case of a dynamic fitness function, the domination value for each gene location should also follow the change in the optimal solution.

### 3 Test Functions and Results

The diploid algorithm has been tested in a previous study by the same authors [10] against the simple algorithm which uses a haploid representation. Two test functions were used in the comparisons. The first function was the one-max problem which is considered easy even for the simple genetic algorithm. The second function was of the type where the optimum oscillated between two peaks. This second function is a type of a dynamic environment where the fitness function changes and it is considered hard for the simple genetic algorithm. The results obtained in that study showed that the proposed diploid algorithm performed as well as the simple genetic algorithm in the first test case and better in the second test case.

In this study, to determine the effects of the new domination mechanism, a comparison between two genotype to phenotype mapping approaches and the proposed approach is made.

The test function used for the comparisons is the second one used in the previous study [10]. The chromosomes are made up of 32 genes. The fitness function oscillates every 30 generations between trying to maximize the decimal value represented by the chromosome and trying to minimize it. A 32-bit chromosome string is taken as a binary number and its decimal equivalent is calculated. When trying to maximize this value, the decimal number itself becomes the fitness and when trying to minimize it, the decimal value is subtracted from the largest decimal number that can be represented by 32 bits and this becomes the fitness value of the individual.

In order to determine the effect of the proposed domination mechanism, all steps of the algorithm, except for the genotype to phenotype mapping phase is kept the same in all test cases. Three different domination approaches are integrated into the diploid algorithm at each testing case and the results are compared based on each of the three implementations' online and offline performances [3].

### 3.1 Test Cases

In the following sections, the results of these comparisons will be given. In each test case, the programs are run 100 times and the average of the results are given over 100 runs. The programs are run each time with the same set of parameters but with different initial populations. The parameters chosen for the programs are given in Table 1.

**Table 1.** Parameters used in all test cases.

Parameters	Value
Number of Generations	1000
Population Size	250
Cross-Over Probability	0.9
Mutation Probability	0.009
Aging and Dying Factor (k)	0.04

#### 3.1.1 Test Case 1: Fixed, Global Dominance Map

In this test case, genotype to phenotype mapping is done via a fixed, global dominance map. Here the 1s in the genotype dominate over the 0s for all loci, and thus in the case where the individual is heterozygous for a locus, the allele 1 is always expressed in the phenotype. This dominance map is kept constant throughout all generations.

#### 3.1.2 Test Case 2: Hollstien-Holland Triallelic Dominance Map

In this test case, genotype to phenotype mapping is done via Hollstien-Holland triallelic dominance map. This triallelic scheme covers both dominance map and allele information at a single location. Each gene may take on one of three alleles: a 2 means a dominant 1, a 1 means a recessive 1 and a 0 means a 0. In this light, the single locus, triallelic dominance map seen in Table 2 is used in determining an individual's phenotype.

**Table 2.** Single locus, triallelic dominance map.

	0	1	2
0	0	0	1
1	0	1	1
2	1	1	1

#### 3.1.3 Test Case 3: Proposed Dominance Mechanism

In this test case, genotype to phenotype mapping is done via the mechanism explained in the previous sections. A global dominance mapping array, which varies along with the population, is used.

### 3.2 Test Results

In the following sections online and offline performances of each test case are given. In each plotting the x-axis represents the number of generations ranging from 0 to 1000 generations and the y-axis represents the fitness value.

#### 3.2.1 Offline Performance

The offline performance of an algorithm as defined by DeJong and explained in Goldberg's book [3], is designed to measure convergence. In an offline application, a simulation of the system may be used and the algorithm may be run on the simulation to achieve the best results and then these best results can be applied to the real system.

The offline performances of the three approaches are given in Fig. 3. In the figure, the better plot line belongs to the proposed dominance mechanism. The second better line belongs to the triallelic dominance map approach and the worst one to the fixed dominance map method.

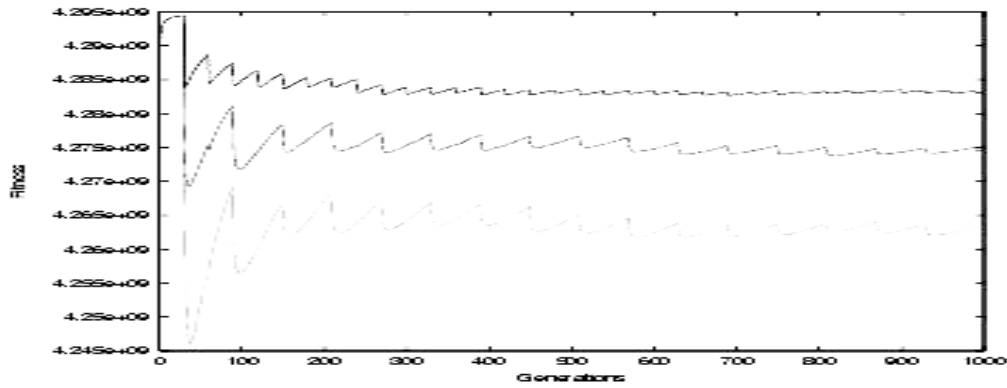


Fig. 3. Offline performances.

### 3.2.2 Online Performance

The online performance of an algorithm as defined by DeJong and explained in Goldberg's book [3], is designed to measure ongoing performance of the algorithm. In an online application, the results of function evaluations are the results of actual experimentation on the real system, so in such applications, the time it takes to reach an acceptable solution becomes more important than getting the best solution.

The online performances of the three approaches are given in Fig. 4. In the figure, the better plot line belongs to the proposed approach. The online performances for the fixed dominance map and the triallelic dominance method are almost the same.

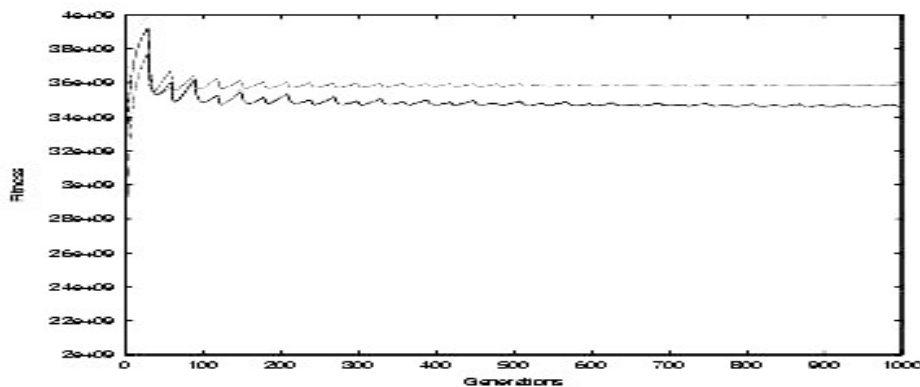


Fig. 4. Online performances.

### 3.2.3 Recovery After Change

In the test problem used, the fitness function changes every 30 generations. The time it takes for the algorithm to recover and find the new maximum after a change is an important factor for determining how well it performs. The maximums found in each generation for the fixed dominance map method, triallelic dominance scheme and the proposed dominance map approach are given in Fig. 5, Fig. 6 and Fig. 7 respectively.

It can be seen from the figures that after a change, the program using the proposed dominance map approach recovers and finds the new maximum in fewer steps than the others.

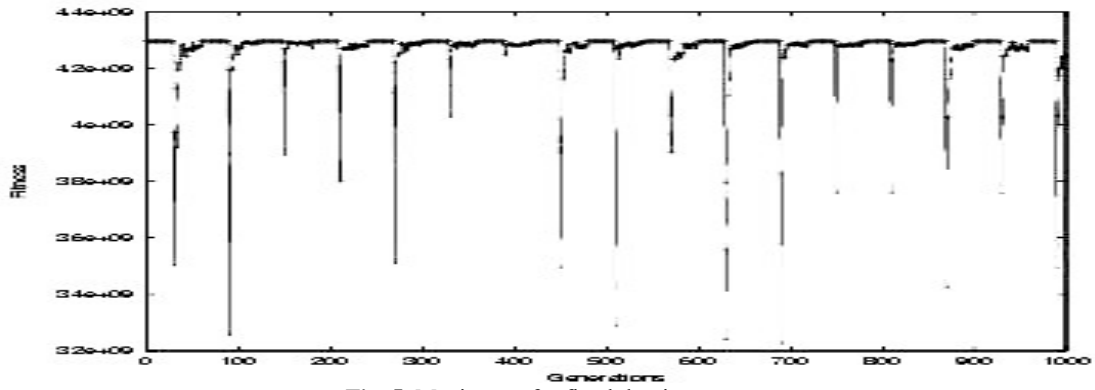


Fig. 5. Maximums for fixed dominance map.

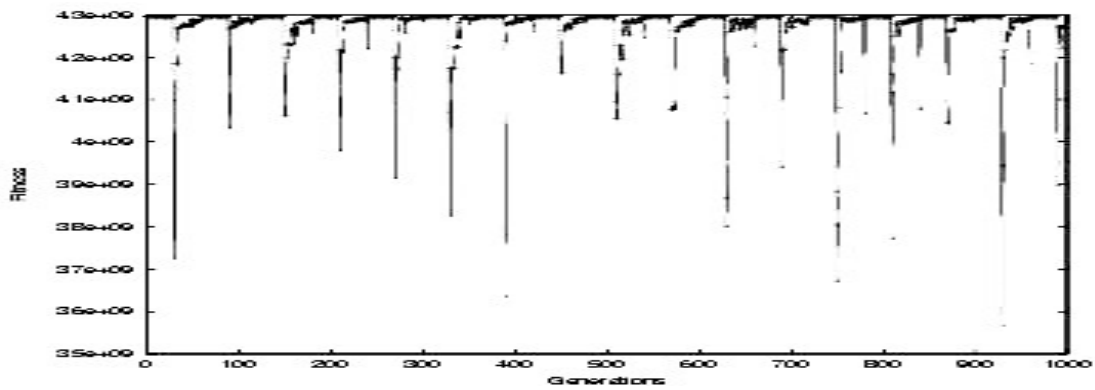


Fig. 6. Maximums for triallelic dominance.

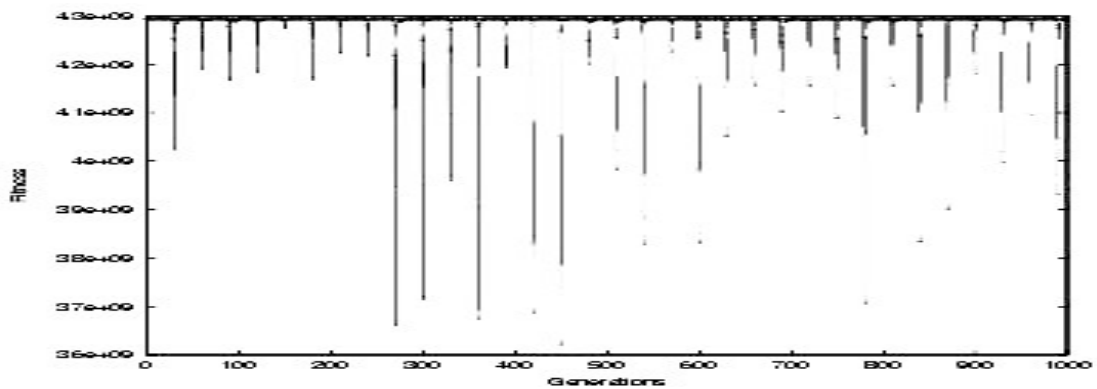


Fig. 7. Maximums for the proposed dominance map.

### 3.2.4 Comments

The results given in the previous sections are to be expected. In the proposed approach, the dominance map for each location evolves along with the population of individuals, thus it follows the change in the environment. If the maximums obtained at each generation are also observed, it is noted that the program using the proposed approach recovers and finds the new maximum much faster than the other two approaches right after a change in the fitness function occurs. Since in dynamic optimization problems the fitness function changes in time, a method that also follows this change and adapts to it quickly improves the performance of the genetic algorithm used making it more suitable for this class of problems.

## 4 Conclusion and Future Work

There is however still work to be done. The results obtained in this study and in the previously mentioned study by the same authors [10], first show that the proposed diploid algorithm performs better than the simple haploid one and second shows that the proposed dominance mechanism works better for the implemented diploid algorithm. However in order to determine the effects of each mechanism in the proposed diploid algorithm, several other test runs have to be performed to examine and test these separately.

## References

- [1] Darwin, Charles. *The Origin of Species*. Penguin Books Ltd. 1985.
- [2] Curtis, Helena. Barnes, N. Sue. *Invitation to Biology*, 3rd ed.. Worth Publishers Inc. 1981.
- [3] Goldberg, David E.. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley. 1989.
- [4] Mitchell, Melanie. Forrest, Stephanie. Holland, John H.. "The Royal Road for Genetic Algorithms: Fitness Landscapes and GA Performance", in *Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*. MIT Press. 1991.
- [5] Ryan, Conor. "The Degree of Oneness", in *Proceedings of the 1994 ECAI Workshop on Genetic Algorithms*. Springer Verlag. 1994.
- [6] Collingwood, Emma. Corne, David. Ross, Peter. "Useful Diversity via Multiploidy", *AISB Workshop on Evolutionary Computation*. 1996.
- [7] Greene, F.. "A Method for Utilizing Diploid/Dominance in Genetic Search", in *Proceedings of the First IEEE Conference on Evolutionary Computation*. 1994.
- [8] Kim, Young-il. Kim, JongKyou. Lee, Seung-Soo. Cho, Choong-Ho. Lee-Kwang, Hyung. "Winner Take All Strategy for a Diploid Genetic Algorithm", in *Proceedings of the First Asia-Pacific Conference on Simulated Evolution and Learning*. 1996.
- [9] Branke, J.. *Evolutionary Algorithms for Dynamic Optimization Problems - A Survey*, Forschungsbericht 387, Institut fuer Angewandte Informatik und Formale Beshreibungsverfahren, Universitaet Karlsruhe. Februar 1999.
- [10] Uyar, Şima. Harmancı, Emre. "Investigation of New Operators for a Diploid Genetic Algorithm" in *Proceedings of SPIE's Applications and Science of Neural Networks, Fuzzy Systems and Evolutionary Computation II*, Volume 3812.1999.