ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

USING SYMBIOTIC POPULATIONS FOR LEARNING DOMINANCE IN DIPLOID POPULATIONS FOR GENETIC ALGORITHMS

M.Sc. THESIS

Canan BATUR

Department of Computer Engineering

Computer Engineering Programme

Thesis Advisor:Asst.Prof. Dr. Ömer Sinan SARAÇ

JANUARY 2014

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

USING SYMBIOTIC POPULATIONS FOR LEARNING DOMINANCE IN DIPLOID POPULATIONS FOR GENETIC ALGORITHMS

M.Sc. THESIS

Canan BATUR
(504101545)

Department of Computer Engineering

Computer Engineering Programme

Thesis Advisor:Asst.Prof. Dr. Ömer Sinan SARAÇ

JANUARY 2014

Canan BATUR , a **M.Sc.** student of ITU **Graduate School of Science Engineering and Technology** student ID **504101545,** successfully defended the **thesis** entitled **"USING SYMBIOTIC POPULATIONS FOR LEARNING DOMINANCE IN DIPLOID POPULATIONS FOR GENETIC ALGORITHMS"** which she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor: **Asst. Prof. Dr. Ömer Sinan SARAÇ** .....
Istanbul Technical University

Jury Members : **Assoc. Prof. Dr. Şima UYAR** .....
Istanbul Technical University

Assoc. Prof. Dr. Arzucan ÖZGÜR .....
Boğaziçi University

Date of Submission: 16 December 2013
Date of Defense: 20 January 2014

To Father and Mother,

FOREWORD

I would like to express my deepest appreciation to my supervisor Asist. Prof. Dr. Ömer Sinan SARAÇ, of being great help during the development of this thesis and for giving me valuable advice and support always when needed. His valuable insights and directions gave me needful guidance to complete the research and write this thesis.

I also would like to thank Assoc. Prof. Dr. Şima ETANER UYAR for sharing knowledge and the best available information.

Finally, thanks to my father Halil BATUR, my mother Emine BATUR and my sister Özlem BATUR, who endured this long process with me, always offering support and love.

January 2014

Canan BATUR
(Computer Engineer)

TABLE OF CONTENTS

Sayfa

ABBREVIATIONS

GA	**:** Genetic Algorithm(s)
DomGA	**:** Dominance Genetic Algorithm(s)
GEA	**:** Genetic and Evolutionary Algorithm(s)
EC	**:** Evolutionary Computation(s)
EA	**:** Evolutionary Algorithm(s)
ES	**:** Evolutionary Strategies
DOP	**:** Dynamic Optimization Problems
XOR	**:** Bitwise Exclusive OR Operator
PDGA	**:** Primal Dual Genetic Algorithms
PDM	**:** Primal Dual Mapping

LIST OF FIGURES

Page

LIST OF TABLES

Page

USING SYMBIOTIC POPULATIONS FOR LEARNING DOMINANCE IN DIPLOID POPULATIONS FOR GENETIC ALGORITHMS

SUMMARY

The universe in which we live is nonstationary, so optimization problems are often assessed in manner of time varying. This makes optimization more difficult than it would be otherwise. For that reason, recently, there has been a significant increase in the number of works, which are applying Genetic Algorithms (GAs) in dynamic environments.Genetic algorithms are rooted in Darwin's theory of natural selection and evolution. They provide an alternative to traditional optimization methods by using powerful search techniques to locate optimal solutions in complex landscapes. The popularity of genetic algorithms is reflected in the ever increasing mass of literature devoted to theoretical works and real-world applications on various subjects such as financial portfolio management, strategy planning, design of equipment, and so on.

Genetic Algorithms (GAs) belong to the type of Evolutionary Algorithm that is the population based optimization algorithms and it was initially conceived by Holland as a means of studying adaptive behavior and performs an adaptive search by maintaining a population of candidate solutions that are allocated dynamically to promising regions of the search space. They generally considered as function optimization methods. The distributed nature of the genetic search provides a natural source of power for searching in changing environments. As long as sufficient diversity remains in the population, the genetic algorithm may respond to a changing response surface by reallocating future trials. However, the tendency of genetic algorithms to converge rapidly reduces their ability to identify regions of the search space that might suddenly become more attractive as the environment changes. Inspired by the diploidy and dominance mechanisms in nature, a model of dominance mechanisms has been proposed for dynamic optimization problems (DOPs). In this thesis, an important mechanism in Diploidy GA is dominance scheme, and further investigated to improve the robustness and adaptability of conventional GA in dynamic environments. The most computer-based GAs is haploid. Haploid GAs involves the use of a single stranded chromosome to represent the solution to a problem. Haploid GAs has been shown to be useful for a variety of difficult optimization problems. However, the most complex biological organisms have two strands of chromosomes. This fact makes it natural to investigate the use of diploid GAs (double stranded chromosomes) for optimization problems.

In this thesis, we accept the learning dominance mechanism from symbiotic population as a combinatorial optimisation problem. So we propose a new population type which called Symbiotic population. We carefully investigate the effect of the symbiotic population to the diploid genetic algorithm with constructing two different symbiotic population based models. After that, both Symbiotic models are applied on bit match benchmark problem for comparing with DomGA (Dominance Genetic Algorithm) and adaptive PDGA (adaptive Primal-Dual Genetic Algorithm) algorithms.

The aim of this work is to illustrate applications of one of the evolutionary computation type which is Genetic Algorithm (GA) to problems in the biological sciences, with particular emphasis on problems in optimization. Therefore; this thesis

represents the unification of DomGA, adaptive PDGA and SymGA-I genetic algorithm (SymGA-I GA) and SymGA-II genetic algorithm (SymGA-II GA) with evolution as a diploidy and dominance theme tested on bit match benchmark problem and then compared.All of those models are offered for the optimization in need of real world problems in order to improve robustness of Genetic Algorithm in dynamic environments.

This thesis mainly discusses the modification of the conventional genetic algorithm that is designed to maintain the diversity required to track a changing response surface. Therefore, this thesis investigates the idea of different dominance mechanism models for combinatorial optimization problems in dynamic environments. Symbiotic genetic algorithm schemes based on the important factor are introduced: we built dominance mechanism as a separate haploid population (dominance population). Then original diploid population and the population representing the dominance information (haploid) are co-evolved with different approach in both Symbiotic approach. At each iteration, one of the populations is fixed and the other is allowed to produce a new generation.Therefore, the main objective of the thesis is the study on these model in order to properly balance between the exploration and exploitation. First ideas were analyzed to create mechanism based on the dynamical system and then associated with balanced form in term of exploration and exploitation in order to adapt more suitable characteristics of diploidy genetic algorithms. Then, the most suitable parameters were selected and combined to create different design alternatives.

The design alternatives are evaluated according to certain criteria. These criteria are as following: genetic encoding of solutions, initial population of solutions, evolution of the fitness of solution, genetic operators for the generation of new solutions and parameters such as population size, probabilities of crossover and mutation, replacement scheme and number of generations. The criteria are given quotients according to their overall importance and each design alternative is applied with the objective function in order to improving optimization in dynamic system. After this evaluation process,these parameter alternatives give best optimizing quality behaviour of diploidy Genetic algorithm. Later, for controlling changing process in this thesis we use dynamic bit changing process, so we used dynamic bit matching benchmark based on different levels of change severities and change frequencies for applied to these builded models.

As a result of this thesis; a study is carried out to show the way to trying to design effective genotype-to-phenotype mapping in dynamic environments.When we compare the SymGA-I GA,SymGA-II GA ,domGA and adaptive PDGA based on the bitmatching benchmark problem, we cretated mechanisms by means of systematic approach in Dynamic Optimization Problems.

SİMBİYOTİK POPULASYON KULLANIMI İLE DİPLOİD GENETİC ALGORİTMALARDA BASKINLIK MEKANİZMALARININ ÖĞRENİLMESİ

ÖZET

İçinde yaşadığımız evren durağan olmadığından optimizasyon problemleri genellikle değişen zaman açısından değerlendirilir. Bu da optimizasyonu olduğundan daha zor duruma getirmektedir.Bu nedenden dolayı ,dinamik ortamlardaki Genetik Algoritma uygulamaları hakkında yapılan çalışmaların sayısı son zamanlarda önemli ölçüde artmıştır. Genetik algoritma (GA) raslantısal arama tekniklerini kullanarak çözüm bulmaya çalışan,parametre kodlama esasına dayanan sezgisel bir arama tekniğidir.(Goldberg, 1989).

Genetik algoritmalar Darwin'in doğal seleksiyon ve evrim teorisinin temeline dayanmaktadır.Darwin'in "en iyi olan yaşar (survival of the fittest) " prensibine dayalı olarak bir popülasyonu oluşturan bireylerin rekabet etmelerini ve rekabet sonucu elemelerini sağlayan evrimsel süreci simüle eden Genetik Algoritmalar,ilk olarak John Holland, meslektaşları ve Michigan Üniversitesindeki öğrencileri taafınfan ortaya atılmıştır (Holland, 1975). Bu Algoritma, standart optimizasyon metodlarında güçlü arama tekniklerini kullanarak optimal çözümleri komplex yer uzayına yerleştimek için alternatifler sunar.Genetik algoritmaların popularitesi güçlü arama algoritması ve çeşitli alanlardaki teorik çalışmalara ve gerçek dünya uygulamalarına başarıyla uygulanabilmesi nedeniyle yankı bulmuştur.

Genetik algoritmalar evrimsel hesaplamaların bir türü olup populasyona dayalı bir optimizasyon algoritmasıdır. İlk olarak Holland tarafından tasarlanmış olan adaptif davranış çalışması anlamına gelen bu çalışma populasyonun aday çözümlerinin arama uzayının ümit verici bölgelerinde dinamik bir şekilde adaptif yerleşmesini gerçekleştirir. Genetik aramanın dağıtık doğası değişen ortamlarda arama işlemlerinin yerine getirilmesi için gereken gücü doğal kaynağından elde etmektedir. Genetik algoritmalar kendilerini güçlü arama algoritmaları olarak ispatlamış ve çoğunlukla bu durum optimizasyon fonksiyonunun metodları olarak da düşünülmüştür. Populasyon içerisindeki yeterli çeşitlilik bulunduğu sürece genetic algoritmalar değişen ortamı gelecek örnekleri yeniden yerleştirerek yerine getirir. Buna rağmen, genetik algoritmaların hızlı bir şekilde yakınsayarak arama alanlarını tanımlama yeteneklerini azaltma eğilimleri, bir anda ortam değişikliklerinin meydana gelmesini daha cazip hale getirir.

Standart genetik algoritmalar tek kromozomlu yapıdadır.Tek kromozom yapılı Genetik algoritmaları tek sarmal kollu kromozomlar ile probleme ait çözümü temsil ederler.Tek kromozom yapılı genetik algoritmaların komplex optimizasyon problemleri için yararlı olduğu gösterilmiştir. Ancak, daha komplex biyolojik organizmalar çift sarmal kollu kromozom yapısındadır.Başka bir deyişle, doğada bulunan organizmaların çoğu diploid kromozom yapısına sahiptir ve çift sarmal kollu kromozom üzerinde yer alan iki alel tarafından temsil edilmektedir.

Organizmanın dıştan görünen özelliği ise bu iki alelin baskın olanı tarafından belirlenmekte,çekinik alel ise organizmanın genotipinde saklı kalmaktadır.Her özelliğin iki alel tarafından temsil edilmesi fazlalık olarak görünmenin aksine genetik belleğin tutulması açısıdan yararlı bir metodtur.Bu gerçek ,Optimizasyon problemleri için diploid genetic algoritmalarının (çift sarmal kollu kromozom yapılı)

kullanımının incelenmesini doğal olarak ortaya koyar.Bu çalışmadaki amaç evrimsel hesaplamalardan biri olan genetik algoritmanın uygulamalarını biyoloji bilimindeki problemlerinden özellikle optimizasyon problemlerini vurgulayarak örneklemektir.

Doğadan esinlenen diploid ve baskınlık mekanizmalarıyla dinamik optimizasyon problemleri için diploid popülasyonlar kullanılarak modellenen bir baskınlık mekanizması önerilmiştir.Diploid genetik algoritmaların baskınlık tasarımı tümleşik optimizasyon problemi olarak değerlendirilmiştir.Bu tezde yeni popülasyon çeşidi kullanılarak diploid genetik algoritmaların baskınlık mekanizmalarının öğrenilmesi hedef alınmıştır.Bu popülasyon çeşidi bu tezde Simbiyotik popülasyon olarak adlandırılmıştır. Geleneksel genetik algoritmaların dinamik ortam içerisindeki dayanıklılık ve adaptabilitesini artırmak amacıyla baskınlık mekanizmalarının öğrenilmesi bu tezde önemle incelenmiştir.

Bu yüzden, simbiyotik popülasyonların diploid genetik algoritmaya etkisi iki farklı modelle incelenmiştir.Simbiyotik popülasyon kullanılarak tasarlanan her iki model, domGA (Dominance Genetik Algoritması) ve adaptive PDGA(adaptive Primal-Dual Genetic Algorithm) algoritmalarıyla bitmatching problem üzerine uygulanmıştır ve test edilmiştir, sonrasında da domGA (Dominance Genetik Algoritması) ve adaptive PDGA(adaptive Primal-Dual Genetic Algorithm) algoritmalarıyla kıyaslanmıştır.

Günümüzde dinamik ortamlarda optimizasyon problemlerinin genetik algoritmalarla çözümü yoğun ilgi görmektedir.Bu nedenle ,bu tezin konusu geleneksel genetik algoritmanın modifiye edilerek gerekli çeşitliliğin korunmasının uygun değişken yüzeyin izlenmesi esas alınarak dikkatle tasarlanmıştır. Bu tez yaygın olarak gerçek dünya üzerinde uygulanan dinamik ortamdaki tümleşik optimizasyon problemleri için modellenen farklı baskınlık mekanizmalarını inceler. Dominance genetik algoritması (DomGA), Adaptive Primal-Dual genetik algoritması (adaptive PDGA) ve Simbiyotik Versiyon-I genetik algoritması (Symbiotic Versiyon-I GA) ve Simbiyotik Versiyon-II genetik algoritması (Symbiotic Versiyon-II GA) diploid ve baskınlık temaları çerçevesinde bitmatching problemi üzerinde test edilerek gösterilmiştir.

Genetik algoritmanın dinamik ortamda tasarlanması Symbiotic genetik algoritması olarak isimlendirilmiştir. Symbiotic genetik algoritmasının baskınlık mekanizması populasyon tabanlı bir yapı halinde inşa edilmiştir ve her bireyin baskınlık kromozomu genotipindeki diploid kromozomun (çift kromozomlu yapı) aksine (tek kromozomlu yapı) olarak modellenmiştir. Daha sonra orjinal (çift kromozomlu yapı) ve baskınlık bilgilerini gösteren (tek kromozomlu yapı) biraraya getirilerek beklenti maksimizasyonu yaklaşımı mantığıyla geliştirilmiştir. Her bir jenerasyonda, populasyonlardan biri sabit tutularak diger populasyonun yeni jenerasyonunun oluşturulmasına izin verilir.Bu nedenle, bu tezin başlıca amacı bu modeler üzerine çalışarak keşif ve sömürme arasında uygun bir denge sağlamaktır.Karşılaşılabilecek problemleri engellemek amacıyla yeni haritalama uygunluk tasarımı kullanılarak problemlerin üstesinden gelmeye çalışılmıştır. Bireyin başarım değeri fenotipi kullanılarak hesaplandığından her aşama sonunda bireylerin genotiplerinden fenotiplerinin belirlenmesi için bir baskınlık mekanizmasının inşa edilmesi modellerin tasarımının temel fonksiyonudur.İstenen sayıda jenerasyon tamamlandığında algoritma sonlanır. Tüm jenerasyonlar boyunca tutulan, o jenerasyon içinde en yüksek başarım değerine sahip bireylerin en iyisi, aranan optimal çözüm olarak belirlenir. Böylece, pek çok nesil aracılığıyla iyi özellikler

populasyon içerisinde yayılırlar ve genetik işlemler aracılığıyla da diğer iyi özelliklerle bütünleşirler.

Durağan değerlendirme problemleri tarafından düzenlenen dinamik test ortamlarına dayandırılarak oluşturulan testler önerilen Symbiotik genetik algoritmayı onaylamaktadır.Test sonuçları dinamik ortamdaki Symbiotik genetik algoritmanın yeterli olduğunu göstermektedir.

Symbiotik genetik algoritmada , inşa edilen mekanizma dinamik sistem tabanlı olup sonradan keşif ve sömürme dengesiyle birleştirilerek genetik algoritma karakteristiklerinin adaptesi daha uygun hale getirilmiştir.Sonrasında en uygun parametreler seçilerek farklı tasarım alternatifleri oluşturmak için birleştirilmiştir.Tasarım alternatifleri bazı kriterlere göre değerlendirilmiştir.Bu kriterler, mesela çözümlerin genetik kodlaması, çözümlerin ilk populasyonları, bireyin başarım değeri, jenerasyonun yeni çözümleri için genetic işlemciler ve parametreler, mesela populasyon büyüklüğü, çaprazlama ve mutasyon olasılıkları, Yeniden yerleştirme planı, jenerasyon numarası. Bu kriterler her bölümün önemlilik derecelerine ve her biri tasarım alternatifleri hedef fonksiyonları üzerine uygulanarak dinamik ortamdaki optimizasyonu artırmak amacına göre önemsenmiştir. Daha sonra , değişen sürecin kontrolü için bu tez de dinamik bit değişim süreci kullanılmıştır,Bu yüzden dinamik bit eşleştirme kriteri farklı seviyelerdeki değişim şiddetleri ve değişim frekansları temeliyle oluşturularak modellere uygulanmıştır.

Sonuç olarak bu tezde, dinamik ortamdaki tümleşik problemlerin çözümü için önemli kriterler göz önünde bulunularak inşa edilen verimli Genotip-Penotip haritalamasının (baskınlık mekanizmasının) tasarımı için denenen yollar gösterilmiştir. Arama uzaylarının büyük ve karmaşık olduğu, geleneksel genetik algoritmadan istenen sonuçlar alınmadığı , arama uzayı içerisinde mevcut optimum çözüm bulunamadığı ve matematiksel olarak modellenemeyen problemlerin çözümünde etkili olmak amacıyla tasarımlara ihtiyaç duyulmuştur.Böylelikle diploid genetik algoritma için önerilen baskınlık mekanizmalarının tanımı yapılmış ve Symbiotik genetik algoritmalar test edilmiş sonuçların ve algoritmanın başarımının DomGA ve Adaptive Primal-Dual genetik algoritması ile karşılaştırması yapılmıştır ve sonrasında mekanizmaların dinamik optimizasyon problemlerinin sistematik yaklaşımla tasarlandığı ortaya konmuştur.

1. INTRODUCTION

Genetic Algorithms (Holland, 1975) (GAs) are stochastic and global optimization methods and they are robust structures fundamentally constructed from the model of biological principles of Darwin's theory and Mendelian principles of inheritance. These algorithms are widely used as an effective optimization technique for real-world problem such as, scheduling, design, ill-behaved objective functions, and highly complex combinatorial problems. When we try to solve a specific problem with GA, an initially population of individuals or candidate solutions are generated randomly. This candidate population of solution is evolved by means of the principles of variation, selection, and inheritance.

This model depends largely on the careful design and set-up of the algorithm components, mechanisms and parameters. This includes encoding mechanism, creation of a population of chromosomes, the definition of a fitness function, genetic manipulation of the chromosomes, genetic operators for the generation of new solutions and parameters such as population size, probabilities of crossover and mutation, the replacement scheme and the number of generations. The initial population of individuals is usually generated randomly with fixed number of individuals for each generation. In the design of genetic algorithm, to decide for population size is crucial because increasing population size increases its diversity and reduces the probability of a premature converge to a local optimum. Individuals of generations are obtained from the previous generation through the following procedure: individuals are randomly selected from the current populations with preffered selection scheme. In that selection scheme individual pairs are selected then these parents are submitted to the any type of the crossover operation then a mutation operation with a mutation probability *pm* is applied; and then fitness function is determined.

In Darwinian evolution each individuals of the population use an evoluation function in order to determine each individual fitness. That function plays key role in the environmental pressure which means low fitness individuals are less likely to be

selected than high-fitness individuals. That environment pressure determines which survival individual will be parents of the next generations. In order to prevent premature converge, we use selection scheme. One of the most commonly one is tournament selection, in which set of individual are selected randomly. Then individuals with the highest fitness of the tournament competitors will be winner of the tournament. Then, that winner is incorporated in the mating pool and that individual produces cause the selection pressure because of its highest fitness. Those factors improve the fitness of each generation as a result. Effect of the genetic operators which are Crossover and mutation are different in the generation process. This means, these genetic operators allow the creation of new chromosomes during the reproduction phase. Another important step in the generation process is the Elitism ; substitution of "worst" individuals of the current population by the "best" individuals; the algorithm stops after a predefined number of generations has been created. An alternative stopping mechanism is a limit on computing time.

1.1 Research Objective

The primary objective of our research is to propose and build a framework of a GA, which tries to learn about the dominance mechanism from symbiotic population in order to improve the robustness and adaptability of genetic algorithms especially in dynamic environments.

Primary objective can be further divided into 4 parts:

1. To conduct an extensive survey of GAs in the context of diploid GAs

2. To focus on solving dynamic fitness problems with using diploid structured individuals in dynamic environments

3. To conduct research on various ways to construct the Dominance Change Mechanism in GAs

4. To show performance of the proposed dominance change mechanisms based on the Symbiotic population for optimization problems and then compare the results with that of other dominance change mechanism.

1.2 Outline Of the Thesis

The thesis is divided into 6 chapters.

Chapter 2. Genetic Algorithms

Chapter 2 describes Genetic Algorithms. It starts by describing GAs then continue with the brief description of genetic representations. This part shows a comprehensive overview on the influence of problem representation on Genetic Algorithm performance. It further explains genetic alphabet and genes, genotypes and phenotypes, the characteristics of fitness landscape and then shows more significant characteristics of the GA these are: Genetic Algorithms in Feature Selection, the basic algorithm of GAs, Feature Selection, Creating an Initial Population, Building a Mating Population, Parental Selection, Reproduction and Inheritance , Process Offspring, Updating the Population. Then with motivating, each of these GA characteristics describes the general workflow of GA.

Chapter 3. Diploid Genetic Algorithm in Dynamic Environment

Conventional GA was modeled differently with building domination mechanisms to characterize population of individuals in diploidy manner. This section starts by describing Diploid Genetic Algorithm Scheme in Dynamic environment, then we continue with Diploidy and Dominance. Chapter 3 reviews the Models of Adaptive Domination Change Mechanism, then continues with introduction of DomGA (Dominance Genetic Algorithm), adaptive PDGA(Adaptive Primal-Dual Genetic Algorithm and SYMbiotic Genetic Algorithm. More focus is given in the Research on performance which shows improved results of the Research in more efficient way to modelize Diploid GA parameters and Performance enhancement using different GA techniques.

Chapter 4. Results and Discussions

Chapter 4 presents Discussion of Problem and Tests Results about the Dynamic performance of Symbiotic-Iv1, Symbiotic-Iv2, Symbiotic-IIv1, Symbiotic-IIv2, domGA and adaptive PDGA on Test Environment.

Chapter 5. Conclusions

Chapter 5 presents some of the important contributions made by our research study and highlights the thesis's general conclusion. Then it outlines some of the immediate future works that may be of significance to the development of more effective Genetic algorithms in dynamic environments.

2. CONVENTIONAL GENETIC ALGORITHM

2.1 Genetic Representations

The starting point of the genetic algorithm, which differs one algorithm from others, is finding suitable schemes, which used to represent chromosomes. The semantic of the genetic operators, the design, the implementation and the measures used to evaluate fitness of the genetic algorithm directly affected by using of the encoding scheme.

In this section, we briefly describe the structure of chromosomes and how their fitness can be evaluated. The first stage of building any Genetic algorithm is to decide on a genetic representation of a candidate solution to the problem. In a GA, a solution to the problem is encoded as a set of values, x={x1, x2,....,xn}.Each representing solution is composed of a string of genes. The string of values is known as a chromosome. The genetic encoding of a real or artificial organism is contained within their chromosomes. Each chromosome consists of a large number of genes, each uniquely located on the chromosome. Each gene in turn is composed of several alleles. In artificial organism, i.e. genetic algorithms, an allele is encoded with the discrete values. Depending on the application or upon the problem type, a bit, real or integer string can be used for the chromosome. Once choosing a suitable representation is important to enable a solution to be encoded with the "right" representation for the problem being solved. Getting the representation right is one of the most difficult parts of designing a good genetic algorithm. The large variety of data types, encoding and crossover options allow users to solve a wide range of search and optimization problems using Genetic algorithm. Often this is only decided with practicing of a wide range of optimization problems and a good knowledge of the application domain. Some commonly used representations types are binary representations, Integer representations, real -valued or floating- point representation, permutation representations.

In this thesis, we will mainly be concerned with bit-integer chromosomes. The number of genes in the chromosome is known as *chromosome* length and will be defined by n. Each solution x, has an extra value associated with it known as its fitness value, which measures the goodness of that solution.

The fitness value is calculated from given optimization criteria that are modeled in the form of a function known as *fitness function* f(x). For example, Figure 2.1 shows a bit-string chromosome with chromosome length n=6.Here, the fitness function is simply the sum of all the bits in the chromosome.

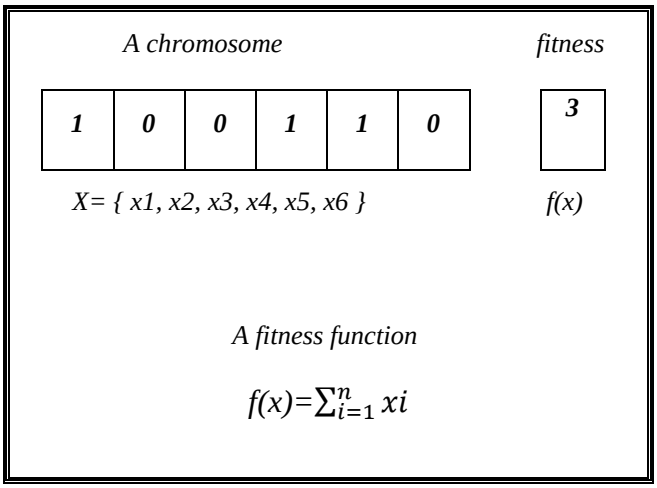


Figure 2.1: A 6-bit long chromosome and its fitness function (Shakya 2006)

The set of all possible solutions is known as the *search space*. For the 6-bit long chromosome shown in Figure 2.1, the search space consist of 2^6 solutions (Shakya 2006).

2.1.1 Genetic alphabet and genes

Genetic methods work based on the population of solutions, which are represented by a string of values. A small number of vectors is used to store each solution of the problem in order to represent the chromosomes of the organisms, which consist of a series of genes. The coding scheme, and the representation of the values, have an effect on the form of the mating and mutation operators. Three different coding schemes will be denoted *gene based*, *node based* and *delta coding*.

In a gene based coding, each value in the string represents an independent variable (Leardi,2003).There is a one-to-one correspondence between the gene number and a particular spectral intensity or a particular descriptor used to describe a set of

molecules. This is the most common coding scheme, but is limited to only certain types of problems.

In a node-based scheme, the string represents a path, schedule or route. This coding scheme is necessary for the travelling salesperson problem (TSP).In the TSP; each city can only be included once in a route. This means that the values of the genetic string are not independent.

Delta coding can only be used for problems that can also accept gene based coding or used less because it can only be applied to certain types of problem. The fitness of a solution is determined by applying the string of deltas to the template values, obtaining a solution, and calculating the accuracy of that solution.

The form of the genetic alphabet is dependent upon the problem and coding scheme. So, the problem should determine which genetic alphabet and which coding scheme can be used (Leardi, 2003).

2.1.2 Genotypes to phenotypes

Mendel recognized that the nature stores the complete genetic information for an individual in pair wise alleles (Mendel 1866). A number of strings store the genetic information that determines the properties, appearance, and shape of an individual. Later, it was discovered that a double string of four nucleotides, called DNA, forms the genetic information.

Mendel realized that the nature distinguishes between the genetic code of an individual and its outward appearance. The genotype represents all the information stored in the chromosomes and allows us to describe an individual on the level of genes. The phenotype describes the outward appearance of an individual, like its hair color or eye size, which is determined by one, or more alleles; then these alleles together are denoted to be a gene. A gene is a region on a chromosome that must be interpreted together and which is responsible for a specific phenotypic property. A transformation exists (a genotype-phenotype mapping or a representation) that uses the genotypic information to construct the phenotype.

When talking about individuals in a population, we must carefully distinguish between genotypes and phenotypes. The phenotypic appearance of an individual determines its success in life. Therefore, when comparing the abilities of different

individuals, they should be evaluated on the level of the phenotype. However, when it comes to reproduction we must view individuals on the level of the genotype. During sexual reproduction, the offspring does not inherit the phenotypic properties of its parents, but only the genotypic information regarding the phenotypic properties. The offspring inherits genetic material from both parents. Therefore, genetic operators work on the level of the genotype, whereas the evaluation of the individuals is performed on the level of the phenotype (Rothlauf and Spahr 2006).

2.1.3 Characteristics of fitness landscape

The majority of the EC applications to date has had problem domains in which the fitness landscape is time-invariant and the fitness of individuals can be computed independently from other members of the current population (Grefenstette 1999). In genetic algorithms on fixed fitness landscapes, the usual practice is to scale the fitness by, assume that, increasing the baseline value of the objective function against which fitness is measured. This is necessary to maintain selective pressure as the population converges toward high fitness regions (Grefenstette, 1986). In a dynamic fitness landscape, such baseline scaling may lead to instabilities since the mean fitness of the population may vary dramatically as the landscape shifts (Grefenstette 1999). It seems even more interesting and open-ended if we attack problem classes in which the fitness landscape varies over time.

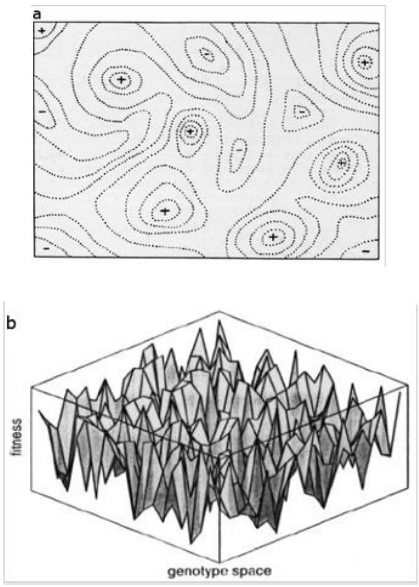


Figure 2.2: Wright's adaptive landscapes, then and now (Johnson 2008).

2.2 Genetic Algorithms in Feature Selection

Starting from the original algorithm, several changes made genetic algorithms a powerful tool in feature selection. Feature selection is the problem of selecting a subset of d features from a set of D features based on certain optimization criterion. The primary purpose of feature selection is to design a more compact classifier with as little performance degradation as possible. GA is naturally applicable to feature selection since the problem has an exponential search space. The pioneering work by Siedleckiand Sklansky demonstrated evidence for the superiority of the GA compared to representative classical algorithms (Oh,Lee and Moon 2004).

2.2.1 About feature selection

The procedure of feature selection,appearently so simple ,is indeed very dangerous and needs a very careful validation,to avoid the risk of over estimating the predictive ability of the selected model;in such cases,when using it on new data,one can be deceived,discovering that it has no predictive ability at all (Lanteri,1992).

The only way to be sure of selecting the best subset of variables (of course without taking into account the problem of random correlations) would be to compute all the possible models. The only limitation to this approach is the fact that with n variables, 2^{n-1} combinations are possible. Consequently, this method becomes not applicable when the variables are more than just a few. To give an idea, with 30 variables (a rather small data set) more than one bilion combinations are possible (computing one model per second, it would take 34 years!).

GA, in their favour, always allows the exploration of the completely experimental space; due to the occurrence of the mutations, each possible combination can occur at any moment (Deviller, 1996).

2.2.2 Creating an initial population

In creating an initial population, two decisions are required; the size of the population and the source of the initial guesses at the solution.

If the genetic operators have a strong focusing effect, it may be advisable to use a rather large population size. This will improve the chances of at least one initial

solution having a relatively high fitness and having many values in common with the optimal solution. If the initial population is too small, relatively good initial solution may only have many values in common with a suboptimal solution and very likely not the optimal one but this solution will be the one found. Conversely, if the size of the population is too large, an initial solution with many values in common with the optimal one may overcome by the sheer number of other solutions and may not be able to have an impact on the overall population. In other words, studies have shown that if good solutions are too 'diluted' by less fit ones, the good solutions may actually disappear after several generations (Cecconi and parisi, 1994).

If the genetic operator has a non-confusing effect, a smaller population may suffice. This will also depend upon how much similar to the parent(s) the offspring is. If this similarity is high and the offspring displaces other solutions in the population, it is possible that the population will converge on a solution that is close (in search space) to an initial solution. Here, the population size should also be large so that there is a better chance that one of the initial solutions will be in vicinity of the optimal one.

In general, these initial solutions may come from purely random guesses that satisfy some minimal fitness criteria, or from the results of other calculations or studies, just as for population size. There is no single best method for generating initial populations for all problems .Different methods will have to be tried with each initial population size and set of genetic operators to see which method yields the best results (Deviller, 1996).

2.2.3 Building a mating population

In many applications, of genetic methods presented in the literature, a *selection* process occurs where members of the current population are placed in a mating population, with very fit members being placed more often than less fit ones. This mating population is then used to create offspring.

Several selection strategies are available, and some will be described here. The first is to simply use some linear ramping functions. For example, if the population size was selected to be 100,the best solution can be copied to the mating population 6 times, the second best 5 times, the third best 4 times, the fourth best 3 times and the fifth best 2 times. Since the mating population now contains 20 solutions, member 6

through 85(assuming they are preordered by fitness) can be copied once to the mating population. This ramping can be more severe such that fewer different solutions are placed in the mating population and better ones are placed more often.

Another selection procedure that is heavily used is called *tournament selection*. In this procedure, the user has to select a tournament size. If this size is 2,for example,two members of the current population are selected at random. The solution with the highest fitness of the two is placed in the mating population. If the tournament size is increased from 1 to the population size ,the mating population will change from one that is likely to resemble the original population to one that it completely composed of the most fit,or *dominant*,solution.

It should be realized that selecting a mating population is a focusing process. If selection is used, it should be taken into account that the dominant solution in the initial population does not contribute considerably to all future offspring. This can be done by not biasing the selection process too heavily towards the most fit solutions and/or allowing mutations to play a large role in generating offspring (Deviller, 1996).

2.2.4 Parental selection

Similar issues arise with respect to choosing which parents will produce offspring. Although to bias the selection too strongly towards the best individual's leads to limit the search focus, to bias too little brings out a lack of focus needed. Current methods include uniform random selection, rank- proportional selection, and fitness-proportional selection.

We understand these selection strategies in isolation quite well (Back, 1995; Blickle and Thiele, 1995). However, it is clear that parental selection and individual deletion strategies must complement each other in terms of the overall effect they have on the exploration/exploitation balance. We can mention certain theories here for particular cases such as Holland's "optimal allocation of trials" characterization of traditional GAs (Holland, 1975) but much stronger results are required (Menon, 2004).

2.2.5 Reproduction and inheritance

In addition to these selection processes, the mechanisms used for reproduction also affect the balance between exploration and exploitation.

The EC community has focused primarily on two reproductive mechanisms, which fall in between these two extremes: 1-parent reproduction with mutation and 2-parent reproduction with recombination and mutation. Historically, the EP and ES communities have emphasized the former while the GA community has emphasized the latter.

Beginning with Holland's initial work, the GA community has been analyzed in a considerable detail as well as the role of crossover and mutation (De Jong, 1975; Goldberg, 1989; Voseand Liepins, 1991; Booker, 1992 Spears, 1998). The ES community has developed theoretical models for optimal mutation rates with respect to convergence and convergence rates in the context of function optimization (Schwefel, 1975).

One of the important and complicated issues is the benefit of adaptive reproductive operators. There are now a variety of empirical studies that show the effectiveness of adaptive mutation rates (Fogarty, 1989), (Back and Schwefel, 1993) or (Fogel, 1995) as well as adaptive recombination mechanisms (Schaffer and Morishima, 1987) or (Davis, 1989) (Menon, 2004).

2.2.6 Process offspring

There is no reason to require that the number of offspring must equal the number of parents, or that all offspring should be placed into the population. There are any number of options and only some of them will be described here.

1-Use mating (with or without mutation and maturation) to create as many offspring as desired from two parents. From this group of offspring, keep the best one or more. Please realize that if only the best solution is kept, or if a solution and its complement are not kept together, this mating will be strongly focusing. If a solution and its complement are kept together ,this pair spans the same search space dimensionality as their parents and will not result in immediate focusing. Conversely, if a large

population size is used, it may be advantageous to focus onto good areas as quickly as possible. Keeping only the best offspring will help this.

2- Use a mutation operator to generate multiple offspring from a single parent (with or without maturation) and only keep the best one. This is non-focusing, and the number of values changed and the magnitude of the change caused by mutation can control the similarity between the parent and the offspring.

3-Use a mutation operator (with or without maturation) to generate 100/N offspring from N% of the parents. For example, a new population can be created by generating 5 offspring from the best %20 of the solutions through mutation (Nolfi and Parisi, 1991; Parisi et al., 1991; Nolfi et al., 1994a, b; calabretta et al., 1995).

When one or more offspring are selected, they can either be added to the current population by displacing an existing member, or can be placed in a new population. If an offspring is placed into the current population, it can be either

1. Displace the weakest member of the population

2. Displace a parent. This could be the only parent if single parents generate a single selected offspring through mating.

3. Displace the most similar member of the current population, or a subset of this population (Deviller, 1996).

2.2.7 Updating the population

When the selected offspring are placed in a separate population and the size of this population reaches a predetermined size, which will be called KPOP (usually KPOP=NPOP, but this does not have to be the case) the generation of offspring stops. At this point, some mechanism has to be implemented to create a new 'current' population. As with all the other operators discussed so far, there are many alternative options.

1. Take the NPOP best solutions from the new population and make this the current population.

2. Combine the NPOP-1 best solutions from the new population and the best solution from the current population to make the next current population. Including the best current solution in the next generation is called the *elitist strategy*.

3. Combine the current and new populations to make a population of size NPOP+KPOP. Take the NPOP best solutions from this combined population.

4. Combine the current and new populations to make a population of size NPOP+KPOP. At this point, use a selection strategy to select NPOP members of this combined population. This can be performed with or without forcing the elitist strategy.

5. Combine the current and new populations and select the fit n unique solutions to produce focus points. Then, in a cyclic fashion, choose the solution from the combined population that is most similar to focus point.

As a final point in this section, the choices presented above can be cast in the (μ , λ) and (μ + λ) notation used in much of the literature on genetic methods. In this notation, μ parents (i.e., NPOP) create λ offspring (i.e., KPOP) .In a (μ , λ) method, λ must be greater than or equal to μ individuals needed to build the new population are taken solely from the λ offspring. In a (μ + λ) method, there are no restrictions on the size of either parameter. The μ solutions are simply chosen from the combined population, μ + λ ,using a selection procedure. Therefore, choices 1 and 2 are (μ , λ) methods(without and with elitist strategy) and choices 3 to 5 are (μ + λ) methods (Deviller ,1996).

3. DIPLOID GENETIC ALGORITHM IN DYNAMIC ENVIRONMENT

3.1 Diploid Genetic Algorithm Scheme in Dynamic Environment

Genetic Algorithm is stochastic and global optimization method, which is based on the population of solutions. Fundamental algorithm structure is constructed from the model of biological principles of Darwin's theory and Mendelian principles of inheritance Schaefer (2006). In the last four decades, the use of Genetic Algorithms (GAs) has developed into a powerful optimisation and problem-solving technique in a diverse range of fields. Originally, GA was modeled on the biological process of evolution by natural selection: the variation in individuals results in a range of reproductive fitnesses, which turns to fitter individuals having a greater genetic representation in future generations. Because primitive GA was inspired by biological genetics, the reproductive functions used to guide propagation in most algorithms also borrowed from a biological repertoire, and the process of crossing over, mutation, and asexual reproduction became instantiated into mainstream evolutionary algorithm strategies. At that point, the increasing availability of computer power and an unending store of potential problems led GA into many different fields, often relying on the same catalog of crossover, mutation and reproduction with slight variations in algorithm metastructure.

This model depends largely on the careful design and set-up of the algorithm components and parameters. Algorithm components of the mechanism are genetic encoding of solutions, the initial population of solutions, the evaluation of the fitness of solutions and genetic operators for the generation of new solutions. Parameters are population size, the probabilities of crossover and the probabilities of mutation, the replacement scheme and the number of generations.

The standard genetic algorithm is based on the haploid structure and each individual's genome structure constituting from one chromosome, but diploid genetic algorithm is based on diploid structure and each individual's genome structure consititutes of haploid genome structure. Also, each individual constitutes of diploid genome structure.

The main topic of this thesis is creating models of the adaptive domination change mechanism in terms of the diploid genome structure.Diploidy and dominance is

used as a powerful tool in creating model in order to improve performance of the nature inspired algorithms during processes of the evolution computations based on the balanced form between exploration and exploitation with combining of diploidy and the adaptive domination. In this thesis we proposed SYMbiotic model. We create population-based dominance mechanism.Each dominance chromosome consititutes from haploid genome structure.And also each individual constitutes from diploid genome structure.

Creating dominance mechanism of the diploid genetic algorithm requires a specific algorithm structure because of each individual in this model has paired genotypes. When there is heterozygosity (two different alleles) in a locus, dominance mechanisms are used to determine the final phenotype of the individual.

The main advantage of diploid model is that the recessive genes are conserved and in case of a change in the environment genetic diversity is preserved. We model our dominance mechanism as a separate haploid population (dominance population). Then original diploid population and the population representing the dominance information (haploid) are co-evolved similar to an Expectation-Maximization approach. At each iteration, one of the populations is fixed and the other is allowed to produce a new generation.

Identifying regions of the search space can be changed;it means that the sufficient diversity can remain in the population or the rapidly convergence reduces the ability of identifying regions of the search space. So, the population loses its genetic diversity, that causes premature converging to solutions. For that reason, traditional algorithm doesn't perform well in that cases and diversity is an important factor for performance. At this point, we introduce Diploid genetic algorithm for improving performance. The basis of this approach is the modelling of population-based dominance mechanism in order to determine the phenotype of each individual and in turn these phenotypes are used to calculate the fitness of individuals. In other words, we need to use that mechanism for mapping genotype to phenotype .That mapping process indicates a very critical part of the diploidy genetic algorithm. For each individual phenotypes which expresses a set of characteristics of individuals and also fitness which represents the ability of the candidate solutions in order to express the quality of the chromosome determined (Uyar and Harmanci (2002)). When determining the phenotype, the genotype elements corresponding to that

location may either be equal or different in the 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 a method to determine the phenotypic value is needed. Several studies that have addressed the used diploidy Genetic algorithm, Goldberg and Smith (1987) first proposed a diploidy-based GA with a tri-allelic dominance scheme for the time-varying knapsack problem. Thereafter, Ng and Wong (1995) investigated a dominance scheme with four possible alleles for a diploid GA and reported a better performance than the tri-allelic scheme. Hadad and Eick (1997) used multiploidy and a dominance vector as an additional part of an individual that breaks the ties whenever there are an equal amount of 0's and 1's at a specific gene location. (Ryan, 1997) used an additive multiploidy, where the genes determining one trait are added to determine the phenotypic trait. The phenotypic trait becomes 1 if a certain threshold is exceeded, or 0, otherwise. (Lewis et al. , 1998) compared several multiploid approaches and observed some interesting results. For example, a simple dominance scheme is not sufficient to track the changing optimum well, but much better results can be obtained if the method is extended with a dominance change mechanism.

Recently, Uyar and Harmanci (2005) proposed an adaptive dominance change mechanism for diploid GAs, where the dominance characteristics for each locus are dynamically adjusted via the feedback from the current population. (Wang and Yang, 2009).

3.1.1 Characteristics of dynamic environment generator

In many real-world optimization problems in order to compare the performance of different GAs in dynamic environments, a wide range of uncertainties have to be taken into account. These uncertainties use dynamic problem generators in the evolutionary optimization, which should meet some common properties.

Some of these properties are listed as follows:

-It should be possible to vary environmental parameters related to different facets of the problem being solved;

-It should be simple to realize different dynamics, such as frequency of change, severity of change, cyclic or not;

-It should be convenient to adjust the complexity and difficulty of dynamic problems;

-It should be computationally efficient to realize required dynamic environments; . It should be easy to carry out formal analysis. (Zhu , Luo and Li 2011).

3.1.2 Classification of dynamic environment generator

In general, through changing (the parameters of) the stationary problem(s) different dynamic environments can be constructed.The environmental dynamics can be easily tuned by two parameters: the speed of change T and the severity of change determined by p, the ratio of ones in the bigger the value of p, the severer the environmental change and the bigger the challenge to GAS. If p = 0.0, the environment stays stationary while if p = 1.0 the environment undergoes extreme changes in the sense of Hamming space. Changing mechanisms dynamic problem generators can be roughly divided into four types, as described below.

1) Switching Fitness Landscapes : In this type of dynamic environment generators the environment is just switched between two or more stationary problems or between two or more states of one stationary problem. For example, time varying knapsack problem where the total weight capacity of the knapsack changes over time, usually oscillating between two or more fixed values ,the dynamic bit-matching problem aims to maximize the number of bits in a string that matches a given template and the template varies over time. For this type of generator, the dynamics of environmental changes is mainly characterized by the speed of environmental changes. It can be fast or slow relative to EA time and is usually measured in EA generations.

2) Drifting Fitness Landscapes : In this type, the dynamic problem generator starts from a fitness landscape f(x), defined in n-dimensional real space ($x \in \mathbf{R}^n$). This fitness landscape is drifted along one or more axes over time while its overall shape (morphology) keeps unchanged.

3) Reshaping Fitness Landscapes In this type, the dynamic environment generators starts from a predefined fitness landscape, each of which can change independently, defined in n-dimensional real space (x Ɛ $\mathbf{R}^n$). Each component has its own morphology with such parameters as peak height, peak slope and peak location. And the center of the highest peak is the optimum of the landscape. (Zhu , Luo and Li 2011).

3.1.3 Dominance and diploidy

Despite the maintanence of diversity is an essential requirement in genetic algorithm, conventional genetic algorithm cannot adapt well in dynamic environments. The loss of the diversity in gene pool reduces the adaptation ability of the genetic algorithm. Organisms must adapt to change and diversity in order to survive in dynamic environments. For that reason, diploidy is considered in GA. In diploidy genetic algorithm, each individual has two chromosomes, each character is represented by two genes located on these chromosomes. This implies that twice amount of genotypic information are added to gene pool. Alleles in both genomes might be recessive alleles or dominant alleles. Only the dominant one of these genes appears in the phenotype wheras the other allelle which is recessive one is not lost.In this way phenotype is formed with a dominance change mechanism.Eventually, individual has a chance of being fitter individuals in the next generations by variation in individuals results in a range of reproductive fitnesses, which depends on their phenotypic expression.

3.2 Models of Adaptive Domination Change Mechanism

3.2.1 domGA

In the model, the most important feature is proposing of the adaptive dominance, mechanism for diploid genetic algorithms in dynamic environment, which is different from non-domination, based genetic algorithm. The basic principles of the dominance genetic algortihm (domGA) is very similar to conventional genetic algorithms except two main differences.One of the main difference is representation and evaluation scheme and the other is reproduction operation.

The genotype and phenotype structure of an individual for Diploid Genetic algorithm is shown in Fig. 1. In that figure,the global domination map is represented for diploidy genotypes and also dominance scheme is proposed for each gene locus on genotypes in order to map phenotypes.

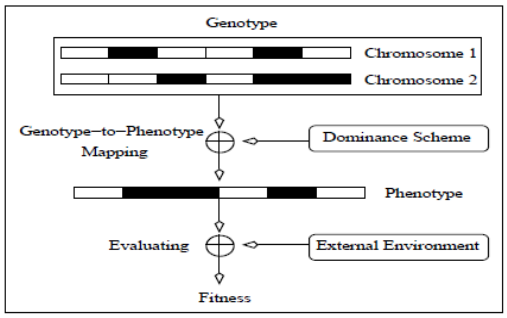


Figure 3.1 : Representation and evaluation of an individual for DGAs. (Uyar and Harmanci 2005).

The general pseudocode of domGA is given in Figure 3.2.

domGA is formed from some basic criteria.The first criteria for domGA is the initialization of the individual parameters with random values.The second criteria is the selection operation , whic is performed with using the tournament selection for the mating pool.The third criteria is reproduction step in domGA, which consists of three phases: Crossover, Formation of offspring and Mutation.

In crossover step two parents (each parent give one chromosome from their's pair of chromosomes) exchange their chromosomes based on uniform crossover method.Then their chromosomes randomly create one temporary offspring.That offspring has one chromosomes from each parent and hence the genotype materials from the parents are mixed and propagated to the offspring.Then that produced offspring goes to mutation phase, with using same mutation probablity pm, for each locus, mutation independently applied to each of the two genotype chromosomes. The fourth criteria is updating_dominance_characteristics.

```
algorithm_domGA

begin
  initialize_population;
  repeat
  select_mating_pool;
  perform_reproduction;
  update_dominance_characteristics;
  until end_of_generations;
end.
```

Figure 3.2 : domGA pseudocode (Uyar and Harmanci, 2005).

In domGA, individuals are represented in terms of the a di-allelic encoding form as in Fig. 3.3.

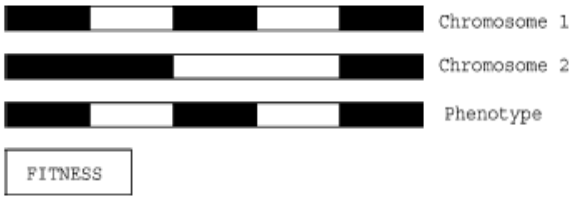


Figure 3.3 : The representation of an individual (Uyar and Harmanci, 2005).

Dominance was determined in diploid populations based on the genotype elements which corresponding to that location may either be equal or different.

if c1i =0 and c2i = 0 then pi =0

if c1i =1 and c2i =1 then pi =1

This means that the probability of expressing a 1 or 0 in the phenotype on each locus is equal when two genotypic alleles of a locus are same. But when the probablity of expressing a 1 or 0 in the phenotype on each locus is not equal then, the dominance probability vector is updated every generation according to the current population.

Calculation of the domination value domi for the ith location is given in Eqs. 3.1

$$Savg[\mathrm{i}] = \frac{\sum_j Pij * fi}{ni} \; and \; Pavg = \frac{\sum_j fj}{n} \qquad \textbf{(3.1)}$$

Defining of the dominance probability vector is determined in terms of expressing alleles in the phenotype. Originally the dominance probability vector is initialized with the value 0,5 for each locus. The one which represents the alleles 1 and 0 are equally dominant on the phenotype expression. Then new domination array

computed for determining the phenotypes of the individuals, which driving forces in the next generation.In equation 3.1,where pij is the phenotypic value of the jth individual at the ith location, fj is the fitness value of the jth individual.At the end of each generation the domination values are recalculated.

At the end of each generation a generational replacement strategy with simple elitism is used for that model.Then with combining XOR operators in the dynamic environment manner we try to solve Dynamic optimisation problems.

3.2.2 Adaptive primal-dual genetic algorithm

Adaptive primal-dual genetic algorithm is constructed based on the inspiration of complementary mechanism in nature.

Because of the tradional genetic algorithm cannot adapt well on changing environment,adaptive PDGA genetic algorithm builded in order to improve genetic algortihm's robustness and adaptability in dynamic environments.

Improving of the robustness and adaptability is achieved by contributing improved schemes.Which are primal-dual mapping (PDM) schemes, two different probablities based PDM operators,lamarckian learning mechanism and Adaptive dominant replacement scheme.

Wang and Yang (2009) In PDGA, each primal chromosome which is explicitly recorded in the population, has its dual chromosome,which is calculated using a Primal –Dual Mapping (PDM) operator.At each generation,a set of chromosomes in the population is selected to evaluate their duals before the next generation starts ,and a dominant replacement scheme is used to decide whether the duals can replace the selected primal chromosomes.

Designing of the PDM operator is constructed as the maximum Hamming distance between a pair of primal-dual chromosomes.Each allele on the primal chromosome translated to its dual based on the PDM operator.In this model ,PDM scheme is applied by using the statistical information of the distribution of the alleles in a gene locus over the population.

General outline of the PDGA is shown in Fig 3.4. The PDGA starts by creating initial population.Genetic parameter pop_size initialize with the genetic operators probablity crossover and probablity mutation at the initialization step.Then for each subsequent generation, chromosomes proportionaly are selected from the current

population in order to evaluate with crossover and mutation operation.Selecting D(t) population is setted in order to performed PDM operation before the next generation starts.

```
Procedure general PDGA
begin
  parameterize(pop_size,pc,pm);
  t := 0;
  initializePopulation(P(0));
  evaluatePopulation(P(0));
  D(0):=selectForDualEvaluation(P(0));
  for each chromosome x in D(0) do
    executePDMOperation(x);
  endfor
  repeat
    P'(t) := selectForRecombination(P(t));
    P''(t) := crossover(P'(t));
    mutate(P''(t));
    evaluatePopulation(P''(t));
    P(t+1) :=selectForSurvial(P(t)+P''(t));
    D(t+1):=selectForDualEvaluation(P(t+1));
    for each chromosome x in D(t+1) do
      executePDMOperation(x);
    endfor
    t := t+1;
  until a termination condition is met;
end
```

Figure 3.4 : Pseudocode of the framework of PDGA (Wang and Yang, 2009).

Pseudocode of a general PDM operator is shown in Fig. 3.5. PDM operator is a function between a pair of primal – dual chromosomes, which create dual chromosome from primal chromosome. It means that, each allele in the gene locus of primal chromosome is translated to its complement by producing dual chromosome.

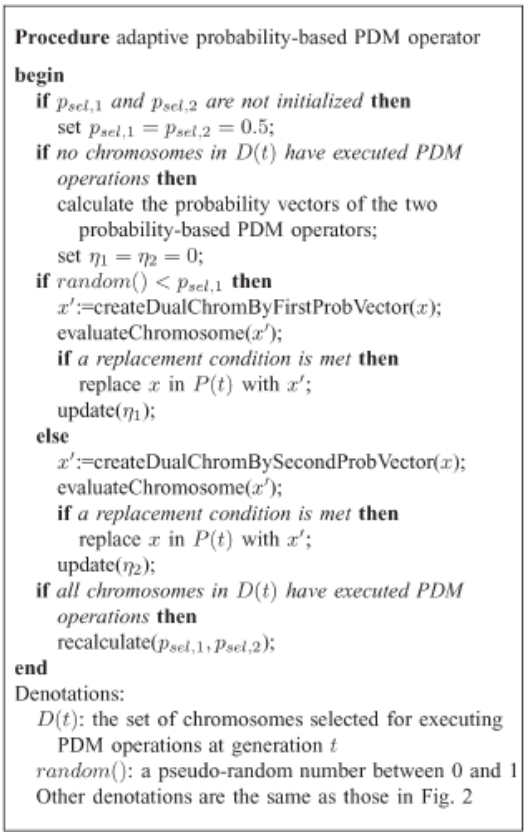
```
Procedure general PDM operator
begin
  x'=createDualChromosome(x);
  evaluateChromosome(x');
  if a replacement condition is met then
    replace x in P(t) with x';
end
Denotations:
  P(t): the population at generation t
  x: the selected primal chromosome in P(t)
  x': the generated dual chromosome after
    executing the PDM operation for x
```

Figure 3.5 : Pseudocode of a general PDM operator (Wang and Yang, 2009).

In order to improve this PDM operator , two different probablity–based mapping scheme is used for each of a gene locus.Based on the statistical information over the

population two different mapping probablities are calculated.

In addition to this probablity-based PDM operator, an adaptive dominant replacement approach is developed.In Fig.3.6. Pseudocode for the adaptive probablity-based PDM oparator is shown.



Procedure adaptive probability-based PDM operator
begin
  if $p_{sel,1}$ *and* $p_{sel,2}$ *are not initialized* **then**
    set $p_{sel,1} = p_{sel,2} = 0.5$;
  if *no chromosomes in* $D(t)$ *have executed PDM operations* **then**
    calculate the probability vectors of the two probability-based PDM operators;
    set $\eta_1 = \eta_2 = 0$;
  if $random() < p_{sel,1}$ **then**
    x':=createDualChromByFirstProbVector(x);
    evaluateChromosome(x');
    if *a replacement condition is met* **then**
      replace x in $P(t)$ with x';
    update(η_1);
  else
    x':=createDualChromBySecondProbVector(x);
    evaluateChromosome(x');
    if *a replacement condition is met* **then**
      replace x in $P(t)$ with x';
    update(η_2);
  if *all chromosomes in* $D(t)$ *have executed PDM operations* **then**
    recalculate($p_{sel,1}, p_{sel,2}$);
end
Denotations:
  $D(t)$: the set of chromosomes selected for executing PDM operations at generation t
  $random()$: a pseudo-random number between 0 and 1
  Other denotations are the same as those in Fig. 2

Figure 3.6 : Pseudocode for the adaptive probablity-based PDM operator (Wang and Yang, 2009).

In this improvement scheme two different probability-based PDM operators are according to an Adaptive Lamarckian learning mechanism.Then effectively the adaptive dominant replacement scheme is contributed to that model..

This replacement mechanism enables algorithm to be highly explorative by accepting an inferior dual chromosome with replacing primal chromosome in the current population with its complement.

3.2.3 Proposed Symbiotic population genetic algorithms

In this thesis, we accept the learning dominance mechanism from symbiotic populations as another combinatorial optimization problem.Inspired by the diploidy and dominance mechanism in nature, we try to modelling and learning different

Dominance mechanisms from Symbiotic populations in Genetic Algorithms (GAs) for dynamic optimisation Problems (DOPs). So, we propose a new population type which called Symbiotic population.In the Symbiotic population based on genetic algorithm, an important factor is using dominance population as a separate haploid population and using genotype population as a diploid population.Then population which represent the dominance information (haploid) and original genotype information (diploid) are co-evolved with each other.

In SymGA-I genetic algorithm approach, evolution starts from a separate haploid population (dominance population) which initially constitutes of randomly generated individuals. In each generation, the fitness of every individual in the haploid population is evaluated. After producing offsprings, the best individual is selected as the dominance to be used in diploid population. In this way, we use the best dominance individual of the haploid population in order to create phenotype of each individual in the diploid population. The fitness values of the haploid population are never used for evaluating the performance of SymGA-I genetic algorithm.In other words, we just use the fitness value of the diploid populations during each step of the tests. At each iteration, one of the populations is fixed and the other is allowed to produce a new generation.

```
Procedure SymGA-I
begin
parameterize (pop size, pc, pm )
t:=0
initialize Population H(t);
initialize Population D(t);
repeat
H'(t):=SelectForRecombination(H(t));
H''(t)=Crossover(H'(t));
Mutate(H''(t));
evaluatePopulation(H''(t));
parameterize( bestdom );
for each diploid chromosome x in D(t)
  use bestdom to evaluate phenotype ;
end for
repeat
D'(t):=SelectForRecombination(D(t));
D''(t)=Crossover(D'(t));
Mutate(D''(t));
evaluatePopulation(D''(t));
parameterize( bestdip);
until a termination condition is met;
end

Denotations :
H(t):= the dominance population at generation t
D(t):=the population at generation t
bestdom:= best individual of the H(t)
bestdip:= best individual of the D(t)
```

Figure 3.7: Pseudocode for the Symbitoic genetic Algorithm-I.

We construct version-1 of the SymGA-I , with setting population sizes to 60 for diploid population and 60 for haploid population.Which called as SymGA-Iv1. In the same manner we try to create version-II of the SymGA-I, with setting population sizes to 100 for diploid population and 20 for haploid population. Which called as SymGA-Iv2.In SymGA-II genetic algorithm approach, we start with assigning a random value to best dominance. Initially we use a random individual from haploid population as the best dominance to generate the phenotype of the diploid population. After producing offsprings,the fitness of each individual in diploid population is evaluated for obtaining best indvidual as a best diploid. Then, the best diploid individual is used in separate haploid population in order to create phenotype for each individual in the haploid population. That best dominance indvidual is then evaluated as a best dominance. Thus, we update best dominance and best diploid inidividual for each successive generation.

```
Procedure SymGA-II
begin
parameterize (pop size, pc, pm )
set bestdom=random();
t:=0
initialize Population D(t);
initialize Population H(t);
repeat
for each diploid chromosome x in D(t)
  use bestdom to evaluate phenotype ;
end for
repeat
D'(t):=SelectForRecombination(D(t));
D''(t)=Crossover(D'(t));
Mutate(D''(t));
evaluatePopulation(P''(t));
parameterize(bestdip);
for each haploid chromosome x in H(t)
 use bestdip to evaluate phenotype ;
end for
repeat
H'(t):=SelectForRecombination(H(t));
H''(t)=Crossover(H'(t));
Mutate(H''(t));
evaluatePopulation(H''(t));
parameterize( bestdom );
until a termination condition is met;
end
Denotations :
H(t):= the dominance population at generation t;
D(t):=the population at generation t;
bestdom:= best individual of the H(t);
bestdip:=best individual of the P(t);
random(): a pseudo random number between 0 and
popsize ;
```

Figure 3.8: Pseudocode for the Symbitoic genetic Algorithm-II.

The fitness values of the haploid population are never used for evaluating the performance of SymGA-II genetic algorithm.In other words, we just use the fitness value of the diploid populations during each step of the tests. At each iteration, one of the populations is fixed and the other is allowed to produce a new generation.

We construct version-1 of the SymGA-II , with setting population sizes to 60 for diploid population and 60 for haploid population. Which is called as SymGA-IIv1. As the same, we try to create version-II of the SymGA-II, with setting population sizes to 100 for diploid population and 20 for haploid populationwhich is called as SymGA-IIv2.

The main genetic algorithm steps are used in SymGA-Iv1,SymGA-Iv2,SymGA-IIv1, andSymGA-IIv2. First criteria is the initialization of the individual parameters with random values. The second criteria is the selection operation, which is performed with using the tournament selection for the mating pool. The third criteria is reproduction step, which consists of three phases: Crossover, Mutation and Formation of offspring.

In the crossover step, two parents (each parent give one chromosome from their's pair of chromosomes) exchange their chromosomes based on uniform crossover method. Then, their chromosomes randomly create one temporary offspring. That offspring has one chromosome each parent and hence the genotype materials from the parents are mixed and propagated to the offspring. Then that produced offspring goes to mutation phase, with using same mutation probablity pm, for each locus, mutation independently applied to each of the two genotype chromosomes. The main steps of the genetic algorithm is completed at the end of the formation of offsprings. A generational replacement strategy with simple elitism is used. The best individual from the previous generation replaces the current worst individual only if the fitness of the previous best individual is better than the fitness of the current worst individual.

Diploidy and dominance mechanisms has an important effect on the performance of improving the robustness and the adaptability of genetic algorithms in the area of real world applications. In such representation it may be expressed that the characteristic of diploidy acts as a source of diversity in the gene pool during the domination mechanism guides the phenotype towards an optimum. It means that we try to create adaptive domination change mechanism as a a balanced form between exploration and exploitation which is obtained from the combination of diploidy and

the adaptive domination. That is we try to adapt the best dominance characteristics for each individual through each subsequent populations. This model suggests an effective way to implement dominance mechanism as a separate population. At each iteration, one of the populations is fixed and the other is allowed to produce a new generation.

In suggested framework, diploid genotypic structure and haploid dominance structure are used for determining the phenotype of each individuals or potential solutions. Each determined phenotype is used based on the problem's objective function in order to determine the fitness of each individual.

Symbiotic genetic algorithm helps us for analysing the contribution of the major mechanisms which are used together. During the enhancement of this dynamic genetic algorithm we try to develop that algorithm with diploidy and dominance structure to create dynamically stable systems. We work in dynamic environments in sense of the predefined generation numbers in which the environment changed only for consecutive generations. For controlling the changing process, we used dynamic bit matching benchmark based on different levels of change severities and frequencies.

3.3 Research on Performance Improvements

3.3.1 Research in more efficient way to modelize GA dominance mechanism

Some attempts have been made to create dominance mechanism in order to apply diploid GA in changing-fitness problems.

The first attempt's structure is constructed by diploidy and dominance which are inspired from nature. In this structure, an important factor is the useof dominance chromosome as a part of the individual information like (seperate dominance chromosome). The diploid structure and dominance chromosome used in order to find the best final phenotype for each individuals or potential solutions with appliying the main steps of genetic algorithm. Then,each determined phenotype is used based on the problem's objective function in order to adapt the dominance characteristics. In that approach, we don't achive enough success percentage.

In the second attempt's structure, an important factor is the use of the dominance population (a seperate dominance population). Like the first attempt, we try to build

dominance mechanism for diploid genetic algorithm. When there is a heterozygosity (two different alleles) in a locus, the original diploid population and the population representing the dominance information (haploid) are co-evolved similar to an expectation-maximization approach. It means that the dominance mechanism is used as a seperate dominance population in order to determine the final phenotype of the individual. For each successive generation, we match each individual in the diploid population with each individual in seperate dominance population in order to find best final phenotype for each individuals or potential solutions with appliying the main steps of genetic algorithm. Then we set each individual's chromosome based on the each dominance chromosome which produce the best final phenotype. At each iteration,one of the populations is fixed and the other is allowed to produce a new generation. In that attempt, the population loses its genetic diversity and that causes premature convergencing to solutions. The dominance maps converged too early.Furthermore, the amount of information stored in the dominance maps was on the same order as all of the information stored in the genotypes of the individuals.

The third attempt's structure which is similar to second attempt. It means that the dominance mechanism is again used as a seperate dominance population in order to determine the final phenotype of the individual. For each successive generation we match each individual chromosome in the diploid population with eachseperate dominance chromosome in order to find the best final phenotype with appliying the main steps of genetic algorithm. Then, we set for each individual's dominance chromosome just the best final phenotype dominance chromosome.Again, at each iteration, one of the populations is fixed and the other is allowed to produce a new generation. In that attempt, beause search space consists from $\boldsymbol{n^2}$, we find the best fitness quickly and that causes premature convergencing to solutions.

At this point we introduce SymGA-I and SymGA-II,which are proposed in this thesiswhich are briefly described in the section 3.2.3.

3.3.2 Performance enhancement using different GA techniques

In this subsection, a dominance mechanism in nature into GAs to improve their performance for DOPs is emphasized. Cruz, Gonzolez and Pelta (2010) the challenge is to develop algorithms, methods, tools, and theoretical foundations that enable

effective design of adaptive systems which harness and control properties that emerge through the interaction of systems and their environment.

Modelling appropriate methods, tools or mechanisms forgenetic algorithm in dynamic environments are emphasized. It is aimed to construct improved mechanisms for genetic algorithms in dynamic environment.

Modelling appropriate methods, tools or mechanisms for genetic algorithm in dynamic environments emphasized. It is aimed to construct improved mechanisms for genetic algorithms in dynamic environment.

Researchers have the relevant works are briefly reviewed. Goldberg and Smith (1987) first proposed a diploidy-based GA with a tri-allelic dominance scheme for the time-varying knapsack problem. Thereafter, Ng and Wong (1995) investigated a dominance scheme with four possible alleles for a diploid GA and reported a better performance than the triallelic scheme. Hadad and Eick (1997) used multiploidy and a dominance vector as an additional part of an individual that breaks the ties whenever there are an equal amount of 0's and 1's at a specific gene location. Ryan (1997) used an additive multiploidy, where the genes determining one trait are added to determine the phenotypic trait. The phenotypic trait becomes 1 if a certain threshold is exceeded, or 0, otherwise. Lewis *et al.* (1997) compared several multiploid approaches and observed some interesting results. For example, a simple dominance scheme is not sufficient to track the changing optimum well, but much better results can be obtained if the method is extended with a dominance change mechanism. Uyar and Harmanci (2005) proposed an adaptive dominance change mechanism for diploid GAs, where the dominance characteristics for each locus are dynamically adjusted via the feedback from the current population.

4. RESULTS AND DISCUSSIONS

4.1 Discussion of Problem and Tests Results

We follow the test strategy given in (Uyar and Harmanci, 2005). The experiments are conducted in four stages. For all stages of the tests, a template of length 200 bits is used. The fitness of an individual is equal to the number of bits the phenotype matches in the given template. The highest fitness an individual can have is 200. For the experiments, 10 different test sets are used to explore the performance of each approach under the different levels of change severity and change frequency. These test sets are given and labeled in Table 4.1. In the table, LS denotes low severity changes where 5% of the bits are changed randomly, MS denotes medium severity changes where 20% of the bits are changed randomly, HS denotes high severity changes where 70% of the bits are changed randomly, LF denotes low frequency where changes occur every 100 generations, MF denotes medium frequency where changes occur every 50 generations, HF denotes high frequency where changes occur every 20 generations and RND denotes that change severity is determined randomly between 5 and 70% and the change frequency between 20 and 100 generations at each change instance.

To avoid unfair test instances, test set scenarios for each entry in the table is created in advance and the same test set scenario is applied to all approaches.

Table 4.1 : Test sets used in experiments (Uyar and Harmanci, 2005).

Test set no	Severity	Frequency	Type
Test-1	5%	100	LS, LF
Test-2	5%	50	LS, MF
Test-3	5%	20	LS, HF
Test-4	20%	100	MS, LF
Test-5	20%	50	MS, MF
Test-6	20%	20	MS, HF
Test-7	70%	100	HS, LF
Test-8	70%	50	HS, MF
Test-9	70%	20	HS, HF
Test-10	Rnd	Rnd	–

In all tested approaches the same set of parameters are used. There are 120 individuals in the population for SymGA-Iv1, SymGA-Iv2, SymGA-IIv1, SymGA-IIv2, SGA and domGA. In adaptive PDGA there are 100 individuals in the population.There are a chromosome consists of 200 genes for all approaches. SGA

and adaPDGA is used in the haploid representation, while domGA used diploid representation and SymGA-Iv1, SymGA-Iv2, SymGA-IIv2, SymGA-IIv1 co-evolved haploid and diploid representation with each other. For the selection step, tournament selection with tournament sizes of 2 is chosen for SymGA-Iv1, SymGA-Iv2, SymGA-IIv1, SymGA-IIv2, SGAand domGA. In adaptive PDGA fitness-proportional selection strategy is used for selection step. Used to selected individuals are placed in a mating pool. The individuals in the mating pool are paired off randomly. Uniform crossover with a probability of 1 and a crossover rate of 0.5 for each locus is used for SymGA-Iv1, SymGA-Iv2, SymGA-IIv1, SymGA-IIv2, SGA and domGA. In adaptive PDGA, one-point crossover is used. A mutation rate of 0.005 is used for the first three stages of the experiments. For the final stage, different mutation rates are applied to each approach to see the effect of the mutation rate selection on performance. A generational replacement strategy with simple elitism is used. The best individual from the previous generation replaces the current worst individual only if the fitness of the previous best individual is better than the fitness of the current worst individual. Since the environment is not static, each best individual from the previous generation needs to be re-evaluated in the current generation. A total of 10 changes occur during each run. A random initial population is used for all tests and a new random seed is used in the random number generator for each run. The tested approaches are evaluated based on the offline error.Mean and standart deviation of the offline error values are calculated at the end of the runs. The main objective of this section is to define and show the effect of the proposed SymGA-Iv1, SymGA-Iv2, SymGA-IIv1, SymGA-IIv2 on improving the performance of diploid GAs in dynamic environments. All test sets (test-1 through test-10) are applied to all chosen approaches. We try to measure offline error for the proposed SymGA-Iv1, SymGA-Iv2, SymGA-IIv1, SymGA-IIv2 algorithms based on the bitmatching benchmark problem.The environment is changed every τ generations and Parameter τ controls the speed of changes, whereas $\rho \in (0.0, 1.0)$ controls the severity of changes.Additionally, we try to test on bitmatching benchmark problem.

A bigger ρ means more severe changes, whereas a smaller τ means more frequent changes. The dynamics parameter ρ is set to 0.05, 0.2 and 0.7, respectively, to examine the performance of algorithms in dynamic environments with different severities of changes. For each experiment of an algorithm on a test problem 20

independent runs were executed. For each run of an algorithm on a DOP, ten environmental changes were allowed, and the best-of-generation fitness was recorded per generation.Since the environment is not static, each best individual from the previous generation needs to be re-evaluated in the current generation.

4.1.1 Dynamic Performance of SymGA-Iv1, SymGA-Iv2, SymGA-IIv1, SymGA-IIv2, adaPDGA, DOMGA and SGA on Test Environments

In this section, we try to make a convenient description on the experiments which applied on the SymGA-Iv1, SymGA-Iv2, SymGA-IIv1, SymGA-IIv2 , DomGA and adaptive PDGA. Which are modeled based on the dynamic environments by using different levels of change severity and change frequency in order to compare effect of dominance mechanisms in SymGA-Iv1, SymGA-Iv2, SymGA-IIv1, SymGA-IIv2 with DomGA and adaptive PDGA. For all test environments,a template of length 200 bits is used and fitness is calculated for SymGA-Iv1, SymGA-Iv2, SymGA-IIv1, SymGA-IIv2 and DomGA is number of bits in the phenotype matches in the given template,and for adaptive PDGA is number of bits in the indvidual matches in the given template.The fitness of an indivdual maximum can be equal to 200. For each dynamic environment, the landscape is periodically changed for predefined number of generations during the run of a GA.

In each periodically changed environment fitness function f(x) is computed for each of the individual in the population .

The best of generation fitness of algorithms ,aganist generation in dynamic environment is plotted in each figures. The experimental results presented for the ten environmental changes,where the data averaged over 20 runs. In the below figures, SymGA-Iv1, SymGA-IIv1, SymGA-Iv2, SymGA-IIv2, DomGA and adaptive PDGA respectively plotted in order to compare effect of the used different dominance mechanisms.

The figures below show the Best- of-Generational fitness on the test sets (test-1 through test-10) are applied to all chosen approaches. The extra parameter settings specific to each test environments are given below.

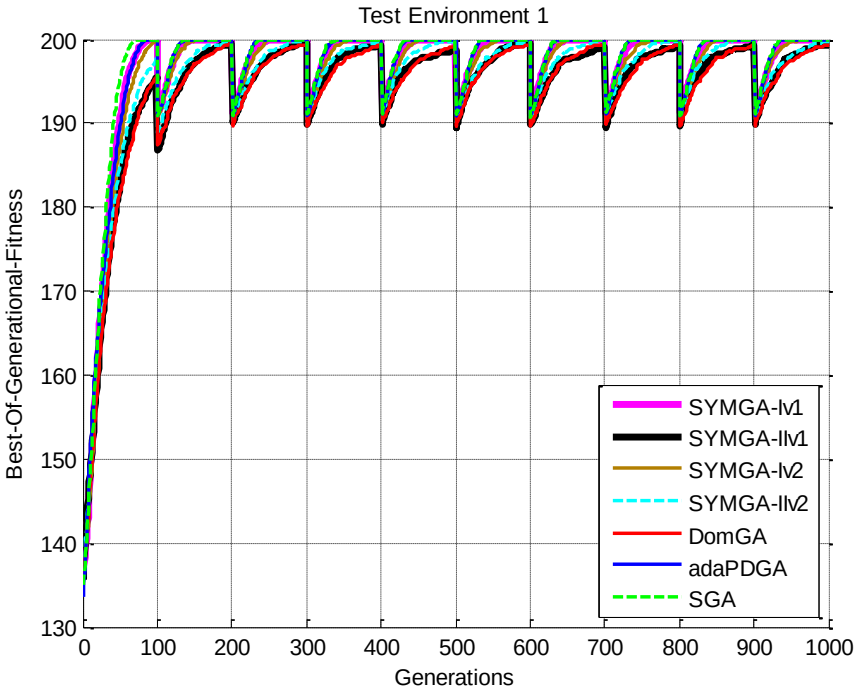


Figure 4.1 : Test Environment 1 for SymGA-Iv1, SymGA-IIv1, SymGA-Iv2, SymGA-IIv2,DomGA, ada PDGA and SGA.

In Figure 4.1: A bigger ρ means more severe changes, whereas a smaller τ means more frequent changes. The dynamics parameters ρ is set to 0.05 and τ is set to 50 to examine the performance of algorithms in dynamic environments with different severities of changes. To see the effects of the all chosen approaches for ten environmental changes, where the data averaged over 20 runs. Fitness value levels gives a better performance for SGA, adaPDGA, SymGA-Iv1, SymGA-Iv2, SymGA-IIv2, SymGA-IIv1,domGA respectively.

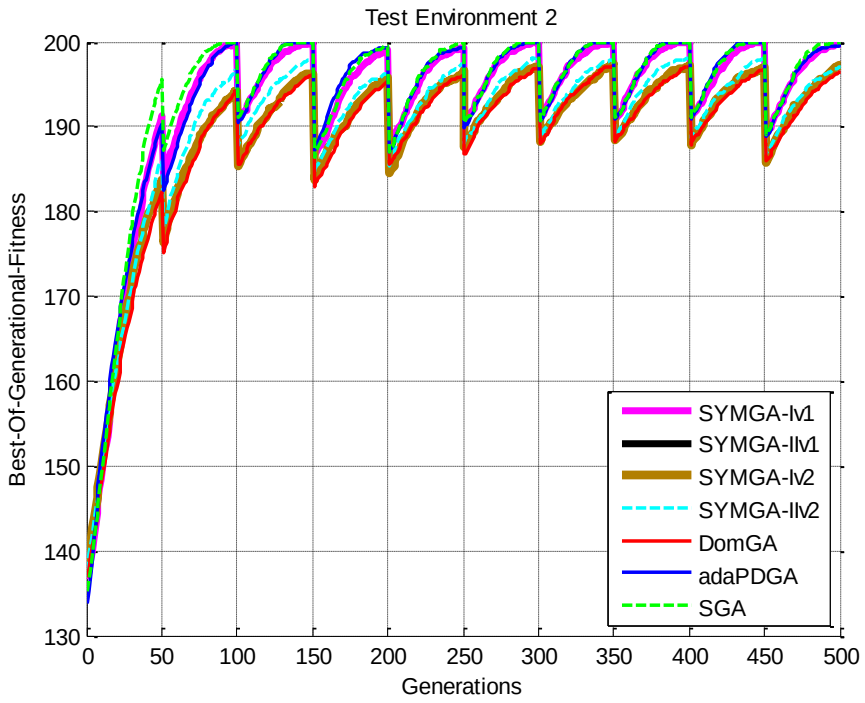


Figure 4.2 : Test Environment 2 for SymGA-Iv1, SymGA-IIv1, SymGA-Iv2, SymGA-IIv2, DomGA, ada PDGA and SGA.

In Figure 4.2: A bigger ρ means more severe changes, whereas a smaller τ means more frequent changes. The dynamics parameters ρ is set to 0.05 and τ is set to 50 to examine the performance of algorithms in dynamic environments with different severities of changes. To see the effects of the all chosen approaches for ten environmental changes, where the data averaged over 20 runs. Fitness value levels gives a better performance for SGA , adaPDGA, SymGA-Iv1, SymGA-Iv2, SymGA-IIv2, SymGA-IIv1, domGA respectively.

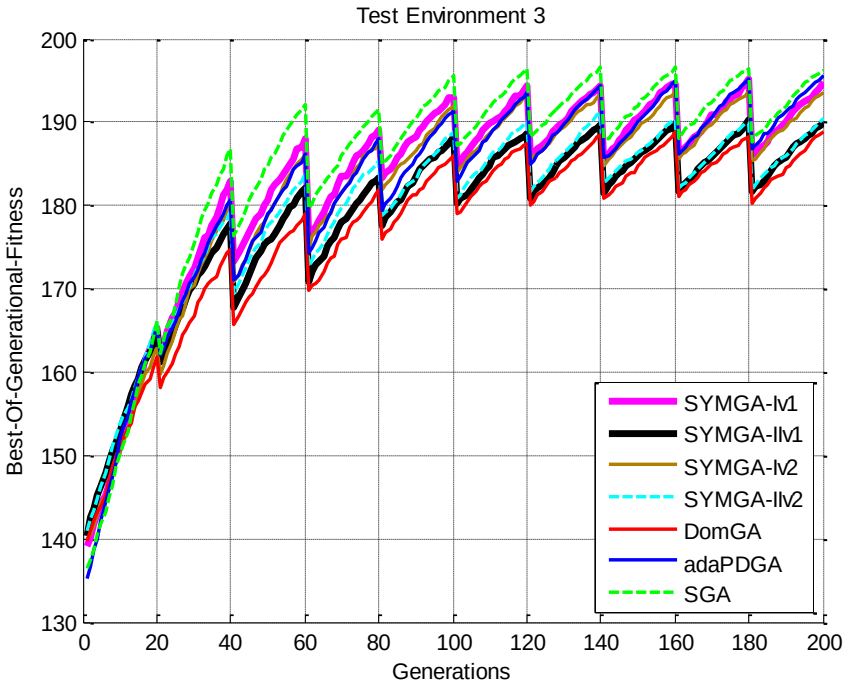


Figure 4.3: Test Environment 3 for SymGA-Iv1, SymGA-IIv1, SymGA-Iv2, SymGA-IIv2, DomGA, ada PDGA and SGA.

In Figure 4.3: A bigger ρ means more severe changes, whereas a smaller τ means more frequent changes. The dynamics parameters ρ is set to 0.05 and τ is set to 20 to examine the performance of algorithms in dynamic environments with different severities of changes. To see the effects of the all chosen approaches for ten environmental changes, where the data averaged over 20 runs. Fitness value levels gives a better performance for SGA , adaPDGA, SymGA-Iv1, SymGA-Iv2, SymGA-IIv2, SymGA-IIv1, domGA respectively.

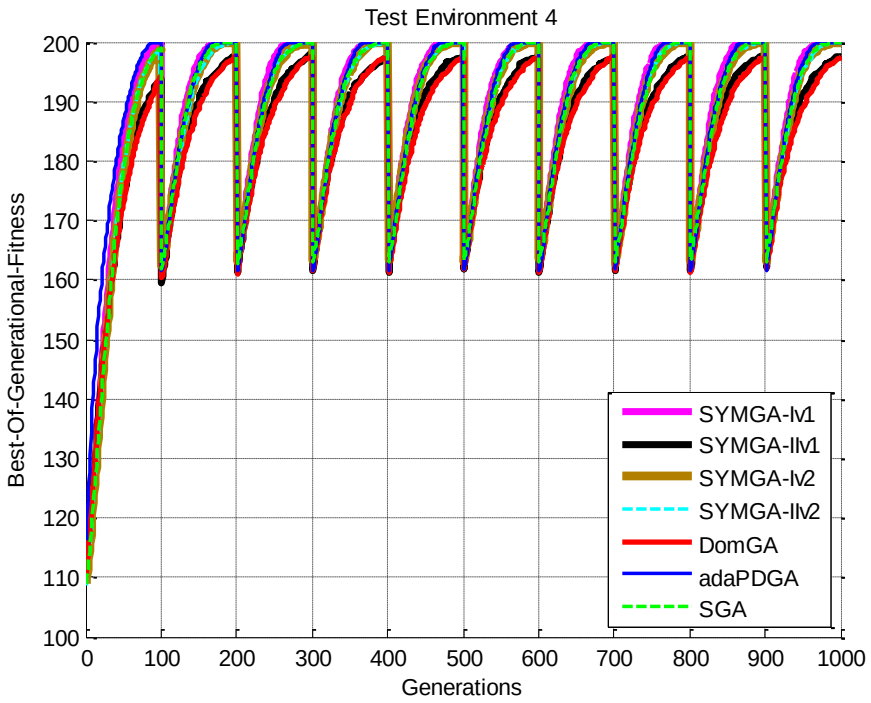


Figure 4.4 : Test Environment 4 for SymGA-Iv1, SymGA-IIv1, SymGA-Iv2, SymGA-IIv2, DomGA, ada PDGA and SGA.

In Figure 4.4: A bigger ρ means more severe changes, whereas a smaller τ means more frequent changes. The dynamics parameters ρ is set to 0.2 and τ is set to 100 to examine the performance of algorithms in dynamic environments with different severities of changes. To see the effects of the all chosen approaches for ten environmental changes, where the data averaged over 20 runs. Fitness value levels gives a better performance for adaPDGA , SymGA-Iv1 , SGA, SymGA-IIv2, SymGA-IIv1, domGA respectively.

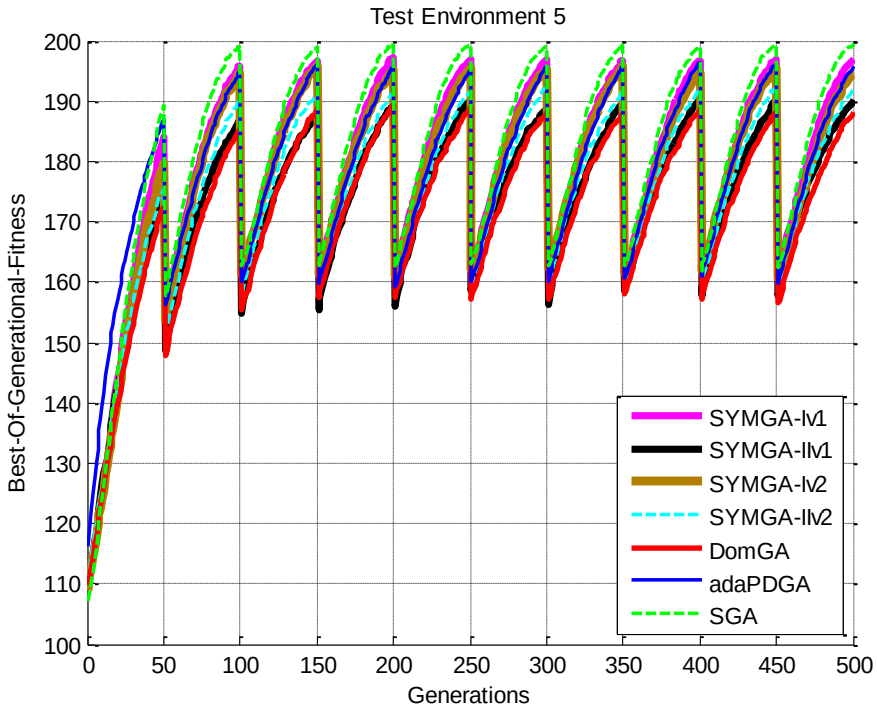


Figure 4.5 : Test Environment 5 for SymGA-Iv1, SymGA-IIv1, SymGA-Iv2, SymGA-IIv2, DomGA, ada PDGA and SGA.

In Figure 4.5: A bigger ρ means more severe changes, whereas a smaller τ means more frequent changes. The dynamics parameters ρ is set to 0.2 and τ is set to 50 to examine the performance of algorithms in dynamic environments with different severities of changes. To see the effects of the all chosen approaches for ten environmental changes, where the data averaged over 20 runs. Fitness value levels gives a better performance for SGA, SymGA-Iv1, adaPDGA, SymGA-Iv2, SymGA-IIv2, SymGA-IIv1, domGA respectively.

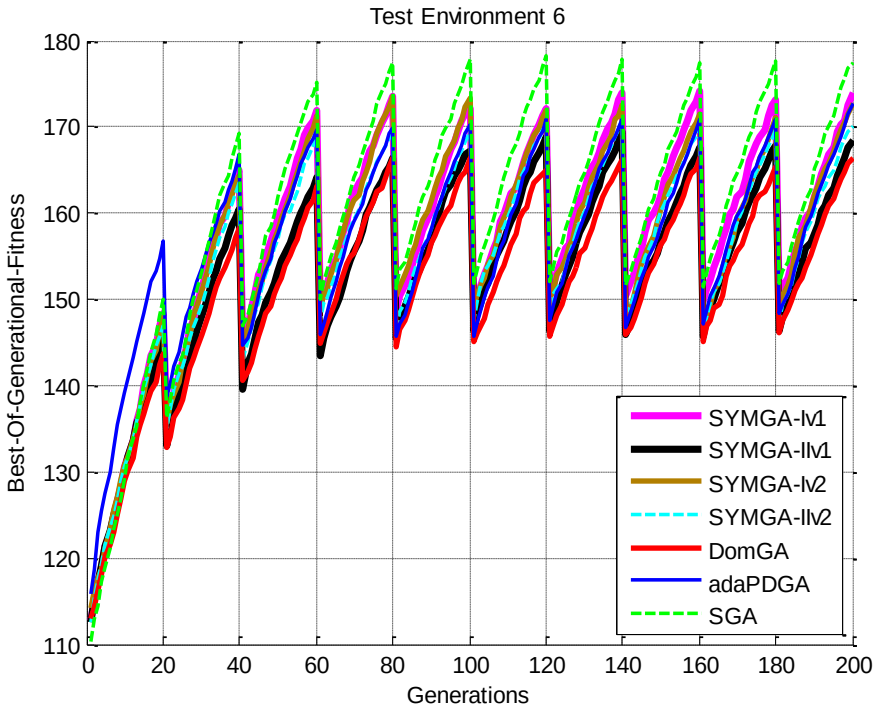


Figure 4.6 : Test Environment 6 for SymGA-Iv1, SymGA-IIv1, SymGA-Iv2, SymGA-IIv2, DomGA, ada PDGA and SGA.

In Figure 4.6: A bigger ρ means more severe changes, whereas a smaller τ means more frequent changes. The dynamics parameters ρ is set to 0.2 and τ is set to 20 to examine the performance of algorithms in dynamic environments with different severities of changes. To see the effects of the all chosen approaches for ten environmental changes, where the data averaged over 20 runs. Fitness value levels gives a better performance for SGA, SymGA-Iv1 , SymGA-Iv2, adaPDGA, SymGA-IIv2, SymGA-IIv1, domGA respectively.

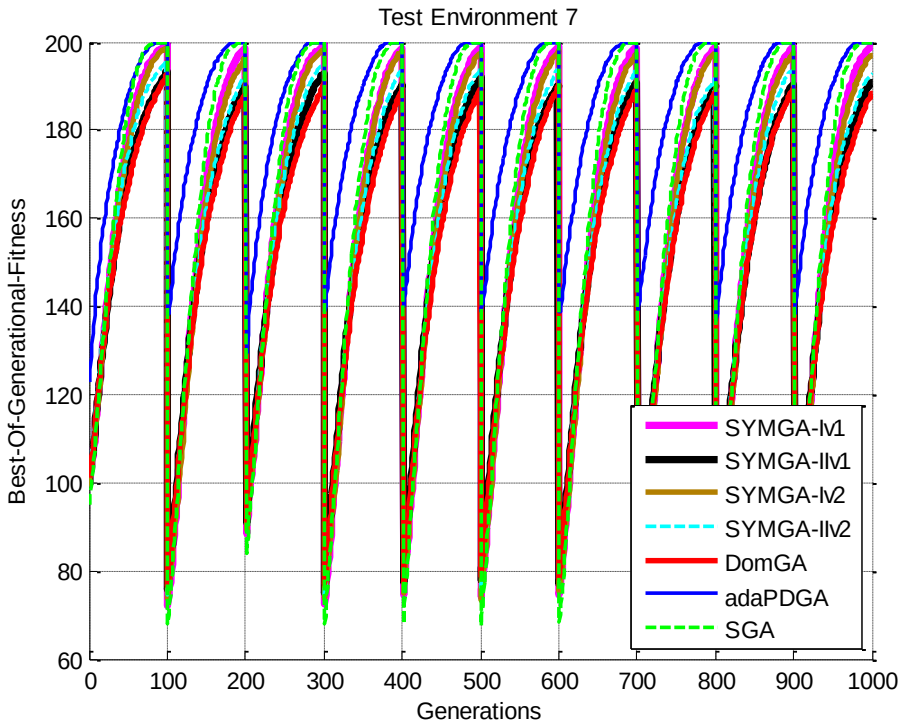


Figure 4.7 : Test Environment 7 for SymGA-Iv1, SymGA-IIv1, SymGA-Iv2, SymGA-IIv2, DomGA, ada PDGA and SGA.

In Figure 4.7: A bigger ρ means more severe changes, whereas a smaller τ means more frequent changes. The dynamics parameters ρ is set to 0.7 and τ is set to 100 to examine the performance of algorithms in dynamic environments with different severities of changes. To see the effects of the all chosen approaches for ten environmental changes, where the data averaged over 20 runs. Fitness value levels gives a better performance for adaPDAGA, SGA, SymGA-Iv1, SymGA-Iv2, ,SymGA-IIv2, SymGA-IIv1,domGA respectively.

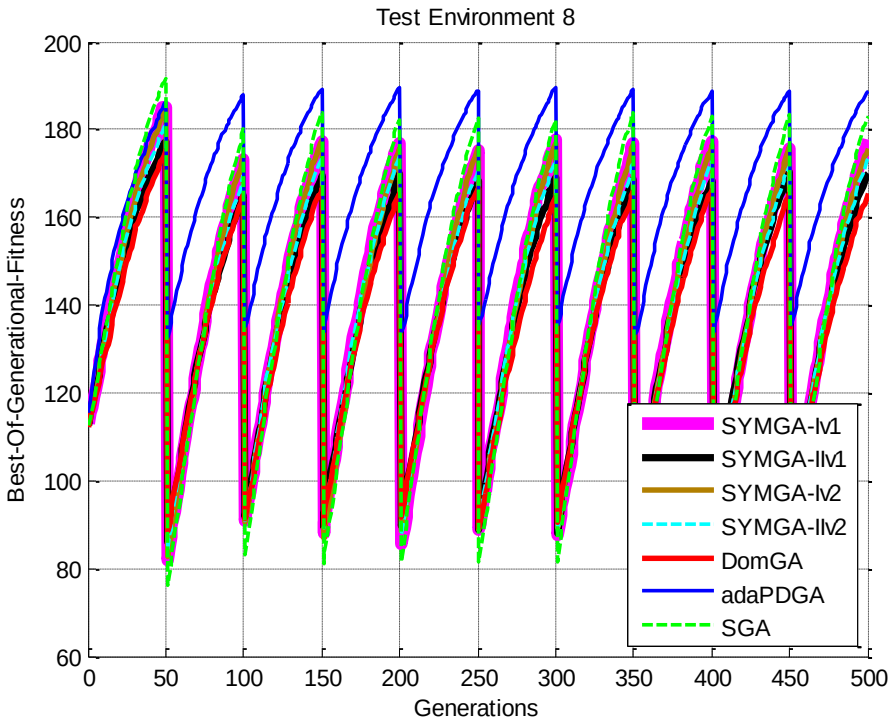


Figure 4.8 : Test Environment 8 for SymGA-Iv1, SymGA-IIv1, SymGA-Iv2, SymGA-IIv2, DomGA, ada PDGA and SGA.

In Figure 4.8: A bigger ρ means more severe changes, whereas a smaller τ means more frequent changes. The dynamics parameters ρ is set to 0.7 and τ is set to 50 to examine the performance of algorithms in dynamic environments with different severities of changes. To see the effects of the all chosen approaches for ten environmental changes, where the data averaged over 20 runs. Fitness value levels gives a better performance for adaPDAGA, SGA, SymGA-Iv1, SymGA-Iv2, SymGA-IIv2, SymGA-IIv1, domGA respectively.

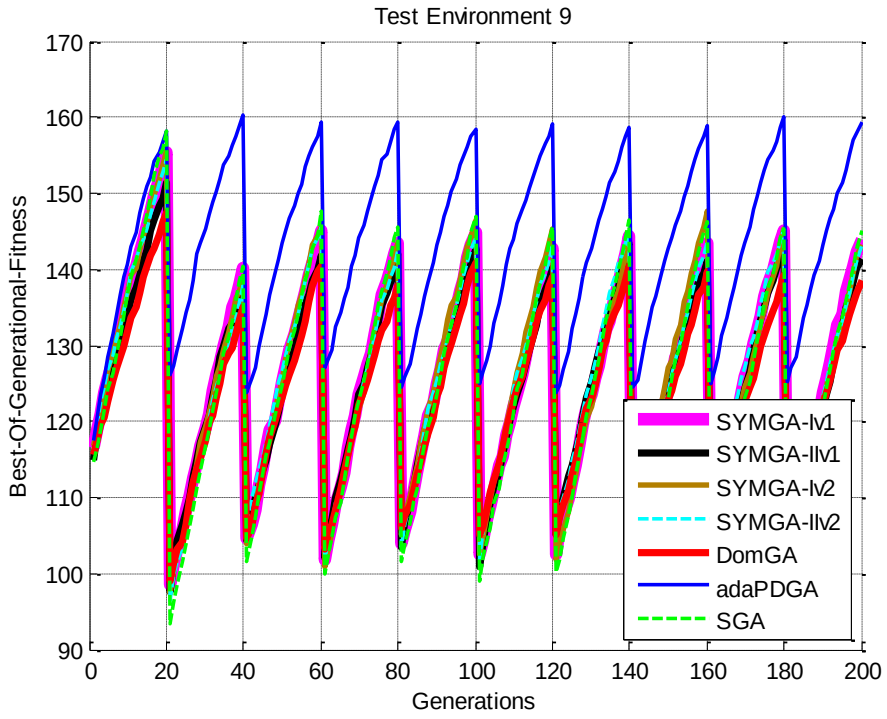


Figure 4.9 : Test Environment 9 for SymGA-Iv1, SymGA-IIv1, SymGA-Iv2, SymGA-IIv2, DomGA, ada PDGA and SGA.

In Figure 4.9: A bigger ρ means more severe changes, whereas a smaller τ means more frequent changes. The dynamics parameters ρ is set to 0.7 and τ is set to 20 to examine the performance of algorithms in dynamic environments with different severities of changes. To see the effects of the all chosen approaches for ten environmental changes, where the data averaged over 20 runs. Fitness value levels gives a better performance for adaPDAGA ,SymGA-IIv2, SymGA-Iv2, SymGA-Iv1, SymGA-IIv1,SGA, domGA respectively.

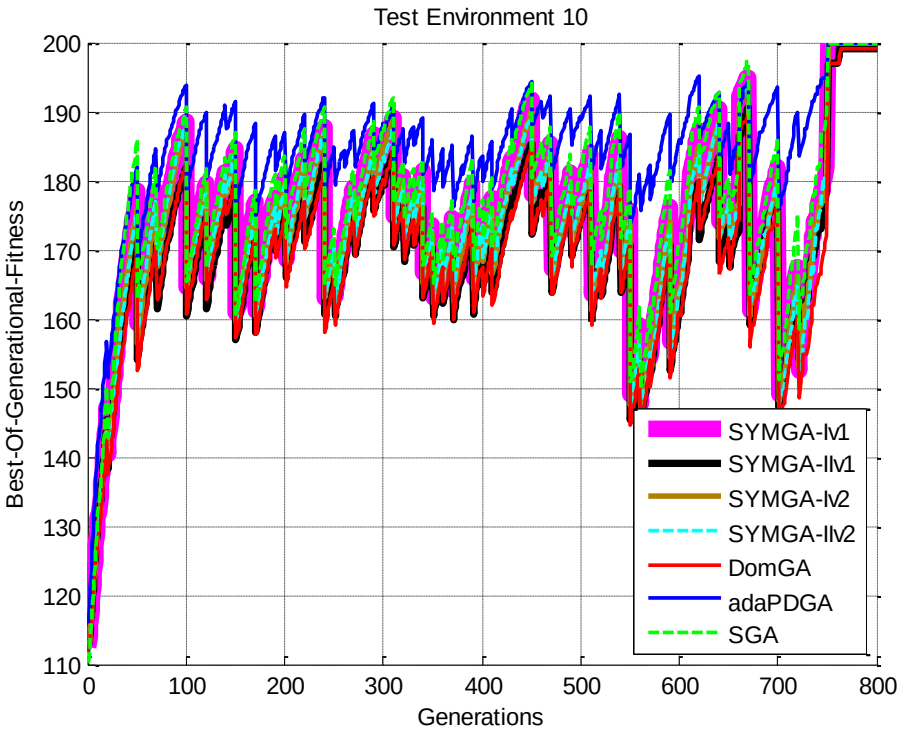


Figure 4.10 : Test Environment 10 for SymGA-Iv1, SymGA-IIv1, SymGA-Iv2, SymGA-IIv2, DomGA, ada PDGA and SGA.

In Figure 4.10: A bigger ρ means more severe changes, whereas a smaller τ means more frequent changes. The dynamics parameters ρ is set to randomly chosen from values of the 0.05, 0.2, 0.7 and τ is randomly chosen from values of the 100,50 and 20. To see the effects of the all chosen approaches for ten environmental changes, where the data averaged over 20 runs. Fitness value levels gives a better performance for adaPDAGA, SGA, SymGA-Iv1, SymGA-IIv2, SymGA-Iv2, SymGA-IIv1, domGA respectively.

In figure 4.7, 4.8 and 4.9 because there is high severe ,adaPDGA has advantage.That means, when population converges to one solution, applying high severe causes the finding best solution after the taking duality.

Table 4.2 shows the offline error levels for all approaches under the chosen test conditions and parameter settings. The tested approaches are evaluated based on the offline error. Mean and standart deviation of the offline error values are calculated at the end of the runs.

Table 4.2 : Mean and standard deviation values for offline errors with using test-10 (SymGA-Iv1, SymGA-Iv2, SymGA-IIv1, SymGA-IIv2,DomGA, adaPDGA).

	μSymGAI1 ±σSymGAI1	μSymGAI2 ±σSymGA2	μSymGAII1 ±σSymGAII1	μSymGAII2 ±σSymGAII2	μdomGA ±σdomGA	μPDGA ±σPDGA	μSGA ±σSGA
Test-1	3.111±0.30	3.58±0.353	5.292±0.42	4.50±0.459	5.62±0.45	2.9±0.19	2.7±0.2
Test-2	6.828±2.18	7.63±2.00	10.48±2.42	9.417±2.21	11.1±2.13	6.4±1.92	6.0±1.9
Test-3	17.08±3.01	18.29±2.98	21.09±2.94	20.26±2.64	22.9±3.22	17.9±4.2	15.0±2.9
Test-4	10.73±0.58	12.2±0.624	15.00±0.48	12.19±0.59	15.98±0.6	10.7±0.1	12.2±0.5
Test-5	21.1±1.0	22.5±1.01	26.8±1.24	25.20±0.97	28.2±1.13	21.6±0.6	18.6±0.7
Test-6	42.34±2.1	43.1±2.02	46.42±2.41	44.4±2.16	47.9±1.84	43.3±1.4	39.8±1.8
Test-7	41.7±0.64	43.4±0.57	46.37±0.92	44.51±0.68	48.4±0.91	17.4±0.3	37.8±0.5
Test-8	60.5±0.70	61.05±0.71	63.06±0.68	61.7±0.806	65.3±0.77	34.1±0.8	59.7±0.6
Test-9	74.6±0.92	74.39±1.06	75.42±1.47	74.22±1.12	76.7±0.89	56.9±0.8	75.4±1.0
Test-10	27.5±7.24	29.07±7.39	32.06±7.29	29.00±7.30	32.9±7.30	18.7±6.7	25.0±7.0

5. CONCLUSION

5.1 Important Contributions

In this section we try to investigate improvement of thesymbiotic population as an important contribution, according to main characteristics of Genetic Algorithms. Symbiotic population is created for learning dominance mechanism in dynamic environment.

In this thesis, a new population based on the dominance mechanism scheme is constructed based on the diploidy and dominance for genetic algorithms in changing environment. Constructing of the dominance mechanism is formed by co-evalution of the two populations. It is useful to emphasize the note that populations are called Diploid population which include individual's diploidy genotype structure and Haploid population, which consist of haploid dominance chromosomes. These populations are used to map genotype to phenotype in terms of expressing alleles in the phenotype. According to Symbiotic Genetic Algorithms model, our experiments are performed using different test environments. We set experiments in order to show performance of the Symbiotic populations for learning the dominance mechanism in dynamic environments.

5.2 General Conclusion

We propose the symbiotic dominance scheme in order to improve the performance of dynamic genetic algorithm. During the enhancement of this Symbiotic genetic algorithm we try to developed that algorithm with diploidy and dominance structure to create dynamically stable systems. This thesisproposes a way to improve a standart Genetic algorithm which is more applicable in changing environments. We work in dynamic environments in sense of the predefined generation numbers in which the environment changed after each of the predefined generation numbers.

Predefined generation numbers were denoted as change frequency. Diploidy genotype structured populations and haploid structured dominance populations implemented in our study with the genetic operators are based on the Genetic algorithm's main steps. Also, that work helps us for analysing the contribution of the major mechanisms which were used together.

The aim of this work is to illustrate applications of one of the evolutionary computation type, which is Genetic Algorithm (GA), to problems in the biological sciences, with particular emphasis on problems in optimisation. The offline errors of domGA, SGA, adaptive PDGA, Symbiotic Genetic Algorithm-I version1(SymGA-Iv1),Symbiotic Genetic Algorithm-I version2 (SymGA-Iv2), Symbiotic Genetic Algorithm-II version1 (SymGA-IIv1)and Symbiotic Genetic Algorithm-II version2 (SymGA-IIv2) calculated with evolution as a diploidy and dominance theme tested on bit match benchmark problem. All of those models are tried for the optimization bit match benchmark problem in order to improve robustness of Genetic Algorithm in dynamic environments. Parameter settings for dynamic bit match problem. According to domGA, adaptive PDGA, SymGA-Iv1, SymGA-Iv2, SymGA-IIv1 and SymGA-Iv2 the best-of-generation fitness was recorded per generation. The results of our experiments show the fitness value of the adaptive PDGA is near the fitness value of the SymGA-I, and the SymGA-II maintain a higher fitness level than DomGA.

5.3 Future Work

The research presented in this thesis seems to have raised more questions than it has answered. There are several lines of research arising from this work which should be pursued.

First, parameter of population size, different type of crossover and mutation should be studied in more detail. Second, the tests may be performed on two oscillating target instead of random targets. Finally, the proposed method should be compared with the existing methods on a real world problem.

REFERENCES

Cruz C., Gonzolez J. R., Pelta D.A. (2010). Optimization in dynamic environments: a survey on problems, methods and measures.

Dan S. (2009). An Analysis of Diploidy and Dominance in genetic Algorithms, Cleveland University, Cleveland, Ohio.

Deviller J. edited by (1996).Genetic Algorithms in Molecular Modeling, Academic Press, Lyon, France.

Eiben, G. and Smith, J. (2003).Introduction to Evolutionary Computing, Amsterdam, Bristol, Budapest.

Grefenstette J. J. (1992). Genetic Algorithms for changing environments, Navy Research Laboratory, Washington, USA.

Grefenstette J. J. (1999). Evolvability in Dynamic Fitness Landscapes: A Genetic Algorithm Approach, Institute for Biosciences, Bioinformatics and Biotechnology, Manassas.

Johnson N. (2008) Sewall Wright and the development of shifting balance theory.

Knysh D. S. and Kureichik V. M. (2010).Parallel Genetic Algorithms: a Survey and problem State Of the Art , No. 4, pp. 72–82.

Leardi R. (2003) Nature-inspired methods in chemometrics: genetic algorithms and artificial neural networks.

Liu L., Wang D. , and Ip W. H. (2008). A permutation -based dual genetic Algorithm for dynamic optimization problems.

L. E. A. Santana, L. Silva, A. M. P. Canuto, F. Pintro, and K. O. Vale. (2010) A comparative analysis of genetic algorithm and ant colony optimization to select attributes for a heterogeneous ensemble of classifiers. IEEE Congress on Evolutionary Computation, 2010, pp. 1-8.

Menon A. edited by. (2004).Frontiers of Evolutionary Computation, Kluwer academic Publishers, Pittsburgh, USA.

Na L., Qing-dao-er-ji L. (2010). The Application of Dominant-Recessive Diploid Codes in MOGA, China.

Oh S., Lee J.S., Moon B.R. (2004).Hybrid Genetic Algorithms for Feature Selection .Vol. 26, No.11, November.

Reeves C.R. and Rowe J.E. (2003).Genetic Algorithms –Principles and Perspectives A Guide to GA Theory, Kluwer academic Publishers, London, Moscow.

Rothlauf, F.Spahr, T. (2006).Representations for Genetic and Evolutionary Algorithms, Springer, Mannheim, Germany.

Saito T., Hamagami T. (2010). Adaptive Adjustment of weight parameters for

Diploid genetic Algorithm with a Network structure. Vol. 1,

pp. 249-253.

Schaefer (2006) R. (2006). Using a Diploid Genetic Algorithm to create and maintain a complex system in dynamic equilibrium, Stanford University, Stanford.

Shakya S. K. (2006). DEUM: A framework for an Estimation of Distribution Algorithm based on Markov Random Fields, Aberdeen, UK.

Tuv E. , Borisov A., Runger G., Torkkola K., Guyon I., and Saffari A. R. (2009) Feature selection with ensembles, artificial variables, and redundancy elimination. JMLR.

Uludag, G.; Kiraz, B.; Etaner Uyar, A.S.; Ozcan, E. (2012) Heuristic selection in a multi-phase hybrid approach for dynamic environments. Computational Intelligence (UKCI), 2012 12th UK Workshop on, On page(s): 1 – 8.

Uyar Ş. , Harmancı E. (2005) A new population based adaptive domination change mechanism for diploid genetic Algorithms in dynamic environments.

Wang H., Yang S. and Wang D. (2009). Adaptive Primal-Dual Genetic Algorithms in Dynamic Environments, Vol : 39 ,No:6.

Woldesenbet, Y.G.; Yen, G.G. (2009). Dynamic Evolutionary Algorithm With Variable Relocation. Evolutionary Computation, IEEE Transactions on, On page(s): 500 - 513 Volume: 13, Issue: 3.

Zhu. Tao, Luo. W, Li. Z. (2011) An adaptive strategy for updating the memory in Evolutionary Algorithms for dynamic optimization. Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), 2011 IEEE Symposium on, On page(s): 8 – 15.

CURRICULUM VITAE

Name Surname: Canan BATUR

Place and Date of Birth: Siirt /1984

Address:

E-Mail: cbatur@itu.edu.tr

B.Sc.: CIU, Computer Engineering 2009.

Adress : İzzet Paşa Mah. Susam Sok. 1/3 Şişli/Istanbul

PUBLICATIONS/PRESENTATIONS ON THE THESIS

- **Batur. C,** Saraç Ö.S. (2013).Model of population-based dominance mechanism for diploid genetic algorithms in dynamic environments. International Science and Technology Conference (ISTEC 2013), Rome, Italy.