Soft Comput (2007) 11:923–941
DOI 10.1007/s00500-006-0143-x

ORIGINAL PAPER

Improved genetic algorithm inspired by biological evolution

P. Kumar · D. Gospodaric · P. Bauer

Published online: 8 December 2006

Abstract The process of mutation has been studied extensively in the field of biology and it has been shown that it is one of the major factors that aid the process of evolution. Inspired by this a novel genetic algorithm (GA) is presented here. Various mutation operators such as *small mutation, gene mutation* and *chromosome mutation* have been applied in this genetic algorithm. In order to facilitate the implementation of the above-mentioned mutation operators a modified way of representing the variables has been presented. It resembles the way genetic information is coded in living beings. Different mutation operators pose a challenge as regards the determination of the optimal rate of mutation. This problem is overcome by using adaptive mutation operators. The main purpose behind this approach was to improve the efficiency of GAs and to find widely distributed Pareto-optimal solutions. This algorithm was tested on some benchmark test functions and compared with other GAs. It was observed that the introduction of these mutations do improve the genetic algorithms in terms of convergence and the quality of the solutions.

Keywords Genetic algorithms · Multi-objective optimization · Mutations

P. Kumar
Trimerics, 70794 Filderstadt, Germany
e-mail: p.kumar@trimerics.com

D. Gospodaric
Trimerics GmbH, 70794 Filderstadt, Germany
e-mail: dgo@trimerics.com

P. Bauer (✉)
Delft University of Technology, Delft, The Netherlands
e-mail: P.Bauer@tudelft.nl

1 Introduction

A large number of search and optimization techniques exist for optimization problems. Researchers and scientists in diverse fields, such as economics, political science, psychology, linguistics, immunology, biology and computer science, need an efficient tool to tackle the optimization problems. The complexity of the problem increases as the number of objectives increase because the objectives considered are often contradictory to one another. Such complex optimization problems have a large number of feasible solutions. However, only a few solutions among them are desirable. Genetic algorithms (GAs) go a long way in solving such problems. Genetic algorithms originated from the studies of cellular automata conducted by Holland and his colleagues. Since the idea of genetic algorithms was introduced by Holland in the early 1970s (Holland 1975), GAs have been applied to several optimization problems. GAs are utilised by many researchers who deal with optimization problems because they are not limited by restrictive assumptions about search space which concern continuity, existence of derivatives or uni-modality. Recently, Coello et al. (2002) gave a comprehensive account of GAs and their application to various multi-objective optimisation problems.

Genetic algorithms (GAs) are randomised, parallel-search algorithms that model the principles of natural selection that leads to evolution. Over time, natural selection in nature has produced a wide range of robust structures (life forms) that perform a broad range of functions efficiently. The success of natural selection provides proof of the viability to use an evolutionary process as a model for design. Like natural selection, GAs is a robust search method requiring little

information to search effectively in large and poorly understood search spaces. Bagley's (1967) work introduced the words genetic algorithms to the scientific literature and Holland's (1975) work: adaptation in natural and artificial systems, laid the first theoretical foundation. Since then there has been many attempts to understand their nature and characteristic to design a more improved GA and to use it effectively in many applications. A large number of researchers have provided contributions towards the development of genetic algorithms and as a result of this a variety of genetic algorithms have come into existence. The main focus of the research work here was to develop GAs that are robust and are able to find well spread solutions for multi-objective optimization problems. One of the pioneering works in this area was that of Goldberg (1989). Goldberg suggested the use of non-dominated sorting together with a niching mechanism. This resulted in an overwhelming enthusiasm on multi-objective evolutionary algorithms (MOEAs). Initial MOEAs, such as the multi-objective genetic algorithm (MOGA) (Fonseca and Fleming 1993), the non-dominated sorting genetic algorithm (NSGA) (Srinivas and Deb 1994) and the niched pareto genetic algorithm (NPGA) (Horn et al. 1994), were directly based on the suggestions of Goldberg (1989) and consisted of two primary steps: (1) the fitness of a solution was determined using its dominance within the population and (2) the diversity among solutions was preserved using a niching strategy. The above mentioned three genetic algorithms show that these steps can be implemented in different ways resulting in a variety of MOEAs that can then be conceived from the suggestions of Goldberg. The elitism operator was absent in these MOEAs which resulted in their poor performance. Hence the focus of later work was mostly concentrated on how elitism could be introduced in a MOEA. As a result of this a number of advanced algorithms emerged such as the strength pareto evolutionary algorithm (SPEA) (Zitzler and Thiele 1999), the pareto archived genetic algorithm (PAES) (Knowles and Corne 2000) and the non-dominated sorting genetic algorithm II (NSGA-II) (Deb et al. 2000), among others. In further attempts to improve the quality of the solutions and to obtain well spread solutions of the Pareto Front, algorithms with dynamic population size were developed by Tan et al. (2001). Adaptive mutation rates were implemented to further accelerate the search for optima and to enhance the ability to locate optima accurately. A detailed explanation and a review of various state-of-the-art evolutionary algorithms for solving multi-objective optimization problems are given by Coello et al. (2002).

Besides these excellent approaches to improve the performance of genetic algorithms, the impact of the mutation operator has been investigated by Aguirre and Tanaka (2005). In this work Aguirre et al. investigated the impact of selection, drift and mutation operators on the performance of evolutionary algorithms. The certain mutation operators: viz. duplication, segregation and transposition, were studied by Goldberg (1989). Similarly, Brizuela and Aceves (2003) performed experimental analysis of the genetic operators for a multi-objective GA applied to the Flowshop problem. Furthermore, Chan et al. (2005) used the concept of jumping genes and applied it to multi-objective resource management in wideband CDMA systems. In the jumping gene GA the transposition mutation operator was used to improve the performance of the GA.

Most genetic algorithms still use an elementary form of point mutation. Research in evolutionary biology has shown that mutation is one of the primary sources of diversity in nature. In our work different types of mutations are applied to the genetic algorithms to increase diversity in the solutions and to improve the convergence. In Sect. 2 a brief discussion is given on the evolution from a biological perspective. Section 3 considers the definition of a multi-objective optimization problem. The general structure of the GAs is discussed in Sect. 4. The different types of mutation that are implemented in our algorithms are discussed in Sect. 5. The structure of the non-dominated sorting biologically motivated genetic algorithm (NBGA), proposed by the authors, is given in Sect. 6. Section 7 gives an overview of the performance parameters that are used to evaluate the NBGA. The test functions on which the NBGA is evaluated and the results of the performance parameters for these test functions, along with the comparison with the other Gas, are given in Sect. 8. The importance of the proposed mutation types is stressed in Sect. 9. Finally, the conclusions are drawn in Sect. 10.

2 Biological perspective of evolution

In general evolution is defined as any process of change occurring with time. In terms of life sciences evolution is a change in gene frequency in a population. The genes are the fundamental physical and functional units of heredity. Genes are made up of DNA, many genes constitute a chromosome and each organism in turn has many chromosomes. For example, humans have 46 chromosomes.

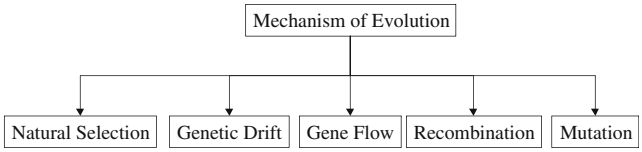

Fig. 1 Mechanisms of evolution

The mechanisms of evolution are: natural selection, mutation, recombination, genetic drift and gene flow, as depicted in Fig. 1.

Natural selection is the principle mechanism that causes evolution. Natural selection was expressed as a general law by Darwin and Huxely (1859), as quoted below:

1. "IF there are organisms that reproduce, and,
2. IF offspring inherit traits from their progenitor, and,
3. IF there is variability of traits, and,
4. IF the environment cannot support all members of a growing population,
5. THEN those members of the population with inferior traits will die out, and,
6. THEN those members with better traits will thrive."

Natural selection can be subdivided into two types:

1. Ecological selection
2. Sexual selection.

Ecological selection takes place in situations where inheritance of specific traits is solely determined by ecology. Sexual selection is the theory that states that competition for mates, between individuals of the same sex, drives the evolution of certain traits.

Natural selection occurs only when the individuals of a population have diverse characteristics. Natural selection ceases to operate when the population does no longer have any genetic variation. For evolution to continue, mechanisms that increase the genetic diversity are necessary. Mutation, recombination and gene flow are the mechanisms that increase the diversity in the population so that evolution can proceed onwards.

Genetic drift is the mechanism that acts in conjunction with natural selection and changes the characteristics of the species over a period of time. This is a stochastic process and is caused by random sampling in the reproduction of offspring. Like natural selection, genetic drift changes the frequencies of alleles but decreases the genetic variations.

Gene flow is the transfer of genes from one population to another. Migration into or out-of a population may be responsible for a significant change in the gene pool frequency. Addition of new genetic material is facilitated by immigration whereas emigration results in the removal of genetic material.

Recombination is the process by which the combination of genes in an organism's offspring differs from that of its parents. Recombination results in a shuffling of the genes. Recombination is a mechanism of evolution because it adds new alleles to the gene pool.

Mutations are permanent changes to the genetic material of a cell. The process of mutation introduces new genetic variations and this facilitates the process of evolution. Most biologists believe that adaptation occurs through the accumulation of many mutations that in themselves only have small effects. Neutral mutations do not have an impact on the organism's chance of survival but they accumulate over time and might result in, what is known as, punctuated equilibrium.

In essence, Genetic algorithms have all the features of these evolution mechanisms. If the greater details of all the mechanisms of biological evolution are understood and implemented in genetic algorithms then the efficiency of GAs will increase many fold. In the genetic algorithm presented here the concept of mutation is extended, based on the mutation idea as perceived in the biological field (explained elaborately in Sect. 5). Besides that, the expression of variables as binary strings is also modified to resemble the genetic makeup of a living being. The implementation of some of these evolutionary biology and genetic concepts in the algorithm developed here, has shown improvement in terms of convergence and quality of the solutions.

3 Multi-objective optimization problem

The multi-objective optimization problem is defined as

$$\begin{aligned} &\text{optimize } \mathbf{f} = \{f_1(\vec{x}), \ldots, f_m(\vec{x})\} \\ &\text{subject to } \mathbf{g}(\vec{x}) \leq 0 \\ &h(\vec{x}) = 0 \\ &\text{where } \vec{x} = (x_1, x_2, \ldots, x_n) \in X \\ &\mathbf{g}(\vec{x}) = \big(g_1(\vec{x}), g_2(\vec{x}), \ldots, g_p(\vec{x})\big) \\ &\mathbf{f}(\vec{x}) = \big(f_1(\vec{x}), f_2(\vec{x}), \ldots, f_q(\vec{x})\big) \end{aligned} \tag{1}$$

where $\boldsymbol{x}$ is the n-component decision vector, $\boldsymbol{X}$ is the decision space and $\boldsymbol{f}$ is the objective space which is a set of m objective vectors. The inequality constraint space is represented by $\mathbf{g}(\vec{x})$, which is a set of p functions. $\mathbf{h}(\vec{x})$ is the equality constraint space, which is a set of q

functions. In multi-objective optimization problems the situation may arise where it is required to minimize all the functions in the objective space, maximize all the functions or minimize some and maximize the others. In order to maintain the uniformity, all the functions in the objective space are converted to either their maximized or minimized form using the following identity:

$$\max f(\vec{x}) = -\min(-f(\vec{x})) \tag{2}$$

Hence, without loss of generality, it can be stated that all functions in the objective space in the above definition are to be minimised. In practical engineering problems the objective functions have different physical dimensions and are non commensurable. Therefore, multi-objective optimization has to look for the best compromise among these objectives in cases such as these. For the problem to be truly of the multi-objective type the pair of objective functions (f_i, f_j) should be conflicting in nature. In this situation the notion of Pareto dominance and Pareto optimality can be used to find the solution set.

Pareto Dominance For any two decision vectors

$$\vec{x} = (x_1, \ldots, x_n) \in \boldsymbol{X}$$

and

$$\vec{y} = (y_1, \ldots, y_n) \in \boldsymbol{X}$$

$\vec{x}$ is said to dominate $\vec{y}$ iff $\vec{x}$ is partially less than $\vec{y}$, i.e.

$$\forall i \in \{1, \ldots, n\}, x_i \leq u_i \wedge \exists i \in \{1, \ldots, n\} : x_i < y_i \tag{3}$$

Pareto optimality A solution $x_u \in X$ is said to be Pareto optimal iff there is no $x_v \in X$ for which $\vec{v} = \mathbf{f}(x_v) = (v_1, \ldots, v_n)$ dominates $\vec{u} = \mathbf{f}(x_u) = (u_1, \ldots, u_n)$.

Non-dominated sets and fronts Let $A \subseteq X_f$. The function $p(A)$ gives the set of non-dominated decision vectors in A:

$$p(A) = \{\mathbf{a} \in A | a \text{ is nondominated regarding } A\}$$

The set $p(A)$ is the non-dominated set regarding A, the corresponding set of objective vectors $f(p(A))$ is the non-dominated front regarding A. The set $X_p = p(X_f)$ is called the Pareto-optimal set and the set $Y_p = (X_p)$ is denoted as the Pareto-optimal front.

4 Genetic algorithms

A genetic algorithm is a heuristic technique to find solutions for difficult optimization problems. GAs use the principles of evolutionary biology. The GAs begin with random initialisation of the population. The transition from one generation of population to the next takes place by application of the genetic operators: *selection, crossover and mutation*. The *selection* operator selects chromosomes in the population for reproduction. The fitter the chromosomes, the more number of times it is likely to be selected for reproduction. The *crossover* operator randomly chooses a locus and exchanges the sub sequences before, and after, that locus between two chromosomes to create two offspring. For example, the strings 10001000 and 11111111 could be crossed over at the fourth locus to yield the two offspring: 100001111 and 11111000. The crossover operator roughly mimics biological recombination between two single chromosomes. The *mutation* operator randomly flips some of the bits in a chromosome. For example the string 11110011 may be mutated in its fifth position to yield 11100011. Mutation can occur at each bit position in a string with some probability. The pseudo code and flowchart of a simple genetic algorithm are shown in Figs. 2 and 3, respectively.

The genetic algorithms (GAs) have the following features:

- GAs operate with a population of possible solutions instead of single individuals. Thus the search is carried out in a parallel form.
- GAs are able to find optimal or sub-optimal solutions in complex and large search spaces. The GAs can be modified to solve multi-objective optimisation problems.
- GAs examine many possible solutions simultaneously, hence they have a high probability to converge to a global optimum.

In order to apply the genetic algorithms for the solution of multi-objective optimization problems the simple genetic algorithms need to be modified. To achieve this, the fitness of the solutions is determined using Pareto optimality, described in the previous section. Figure 4, taken from Srinivas and Deb (1994), shows the

```
Population Initialisation
Evaluate the objective function
Do (while termination criterion satisfied)
        Selection (based on the objective function value)
        Crossover
        Mutation
        Evaluate the objective functions
Loop
```

Fig. 2 Pseudo code of simple genetic algorithm

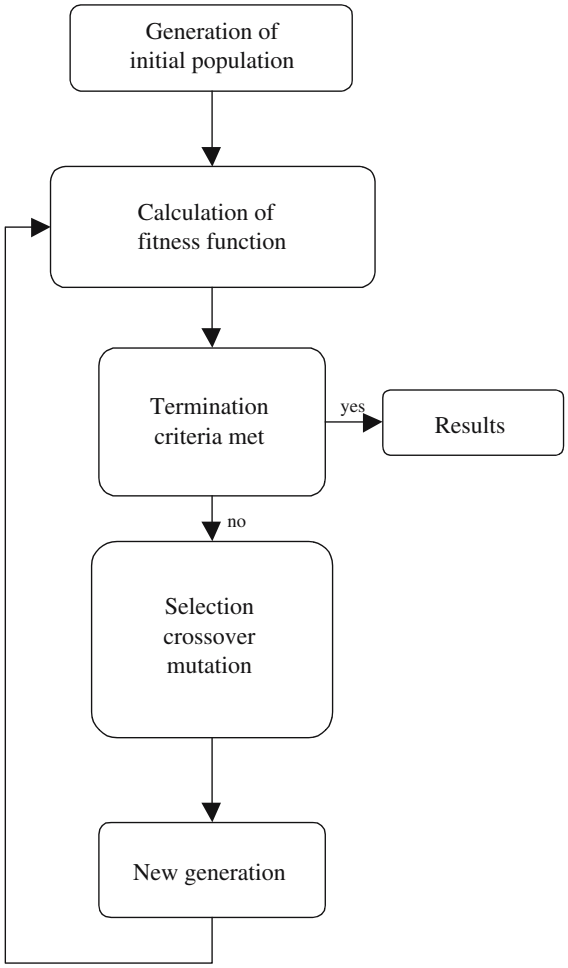

Fig. 3 Flow chart of simple genetic algorithm

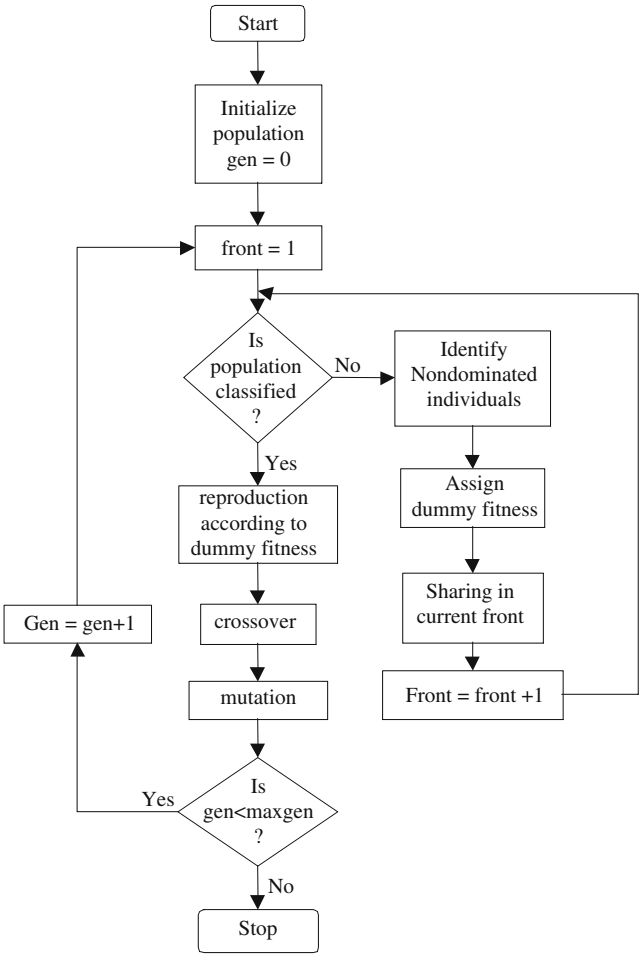

Fig. 4 Flow chart of NSGA

general flow chart of this approach. The detailed working principle of the algorithm is given in Deb (2001). Elitism is used to improve the quality of the solutions. Elitism provides a means to reduce genetic drift by ensuring that the best chromosome is allowed to pass on, or copy, their traits to the next generation. Genetic drift is a mechanism of evolution that acts in concert with natural selection to change the characteristics of species over a long period of time. It acts on the population: altering the frequency of alleles as well as the predominance of traits amongst members of a population and thereby changes the diversity of the population. Genetic drift is used to explain/measure stochastic changes in gene frequency through random sampling of the finite population. Some genes of chromosomes may turn out to be more important to the final solution than others. When the chromosomes, which represent decision variables that have a reduced "salience" to the final solution, do not experience sufficient selection pressure then the genetic drift may be stalled. In order to avoid this it is important to maintain adequate selection pressure, as demanded by the application. In other words, the arrest of genetic drift reflects the failure to exert adequate selection pressure, applied by increasing the tournament size or by some form of elitism. Since elitism can increase the selection pressure, by preventing the loss of low "salience" genes of chromosomes due to deficient selection pressure, it improves the performance with regard to the optimality and convergence of GAs in many cases. However, the degree of elitism should be adjusted properly and carefully because high selection pressure may lead to premature convergence.

Elitism was an important parameter that resulted in improvements in the solutions but an important aspect of evolution, namely mutation, was neither investigated in detail nor was different types of mutation, existing in nature, implemented in genetic algorithms. In the next section a discussion of different types of mutation existing in nature and a way to implement them in binary coded genetic algorithms are discussed.

5 Types of mutation

Mutations are permanent changes to the genetic material of an organism, which are transferred from one generation to the next. The importance of mutations in the evolution process was investigated by Nei (1986) and Li (1993). Molecular studies have shown that mutations include not only nucleotide substitutions but also important processes as gene duplication and recombination. Mutations are considered the driving force of evolution, where less favourable ones are removed by the process of selection and the favourable ones tend to propagate from generation to generation, thereby improving the fitness of individuals in the population. The various types of mutation can be broadly put into three categories namely:

- Point mutations
- Large mutations
- Chromosomal mutations

Each of the above mutations can be further subdivided into various classes. Figure 5 gives an over view of the possible mutations. These mutations have been implemented in the NBGA and a brief description of each of these mutations is given below. The way in which the variables are encoded in NBGA is investigated before discussing the implementation of the mutations in detail. For the sake of simplicity a function with only two variables, $f(x, y)$ is considered. The variables x and y are real valued and are bounded between upper and lower limits. For ease of implementation these variables are represented as binary strings. The binary string for each variable is called a *chromosome* and each chromosome in turn consists of subsequent strings known as *genes*. The chromosomes of both the variables x and y are known as the *chromosomal genome* (Fig. 6).

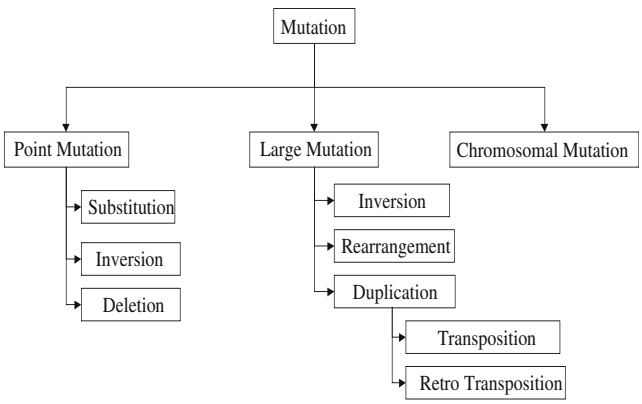

Fig. 5 Classification of types of mutation

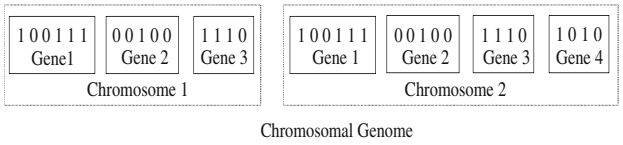

Fig. 6 Representation of variables in NBGA

As can be seen from Fig. 6 the variable x consists of three genes: gene 1 is a binary string of length 6, gene 2 is a binary string of length 5 and gene 3 is a binary string of length 4. Together these three genes constitute chromosome 1. Similarly, the variable y consists of four genes: gene 1 is of length 6, gene 2 of length 5, gene 3 of length 4 and gene 4 of length 4, and together they constitute chromosome 2 for variable y. The combination of these chromosomes constitutes the chromosomal genome. Having outlined the structure of how the variables are encoded in the NBGA, the explanation of the various mutations and their equivalent in the NBGA are discussed below. The chromosomal genome in Fig. 6 is taken as the reference for the discussion of mutations below.

1 *Point mutation* These are changes in the single DNA nucleotides. A point mutation may consist of the *deletion* of a nucleotide, the *insertion* of additional nucleotide or the *substitution* of one nucleotide for another. The deletion type point mutation is shown in Fig. 7a. In this case a bit from gene1 of chromosome 1 has been deleted. This type of mutation is very common. Figure 7b shows the case for insertion mutation. In this case a bit has been added to the binary string (marked in bold) of gene 2 from chromosome 2. In Fig. 7c the substitution

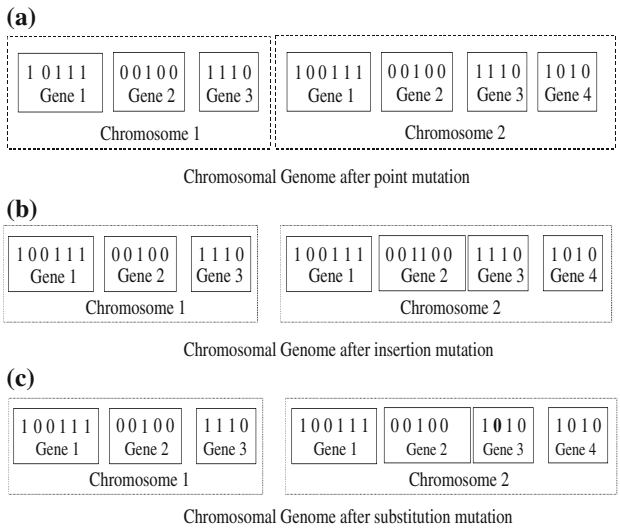

Fig. 7 a: Deletion type point mutation. b: Insertion type point mutation. c: Substitution type point mutation

mutation is shown. In substitution mutation a bit in gene3 from chromosome 2 is flipped (marked bold). Traditionally this type of mutation has been implemented in most of the genetic algorithms.

2 *Large mutation* These mutations involve a whole gene at a time. Various types of large mutation that are implemented in the NBGA are: *deletion*, *inversion, insertion* and *gene duplication*. Gene duplication can be categorised into *transposition* and *retro transposition*. Figure 8a shows the deletion type of large mutation. In this case gene 3 from chromosome 1 is deleted (chromosomal genome in Fig. 6 is used as reference). The inversion mutation is illustrated in Fig. 8b. The string in gene 1 of chromosome 2 is inverted backwards. In Fig. 8c a gene is inserted into chromosome 1, indicating that insertion mutation has occurred. Sometimes a whole gene is duplicated and then inserted at random in the chromosomal genome, such a mutation is known as transposition duplication. Figure 8d shows the transposition type of mutation. Gene 3 from chromosome 1 is copied and positions itself next to gene 3. The transposition mutation has been implemented in the jumping gene GA (Chan et al. 2005). However the

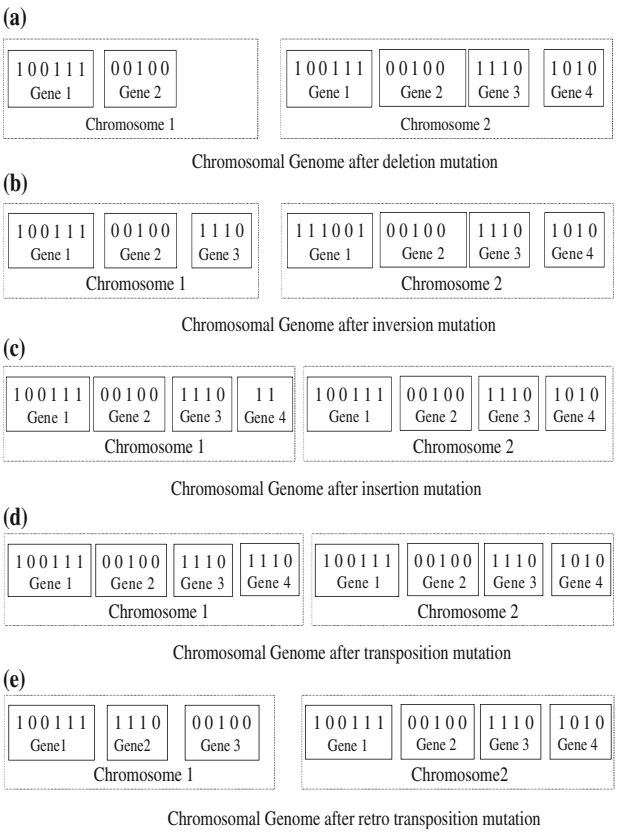

Fig. 8 a: Deletion type of mutation. b: Inversion type of mutation. c: Insertion type of mutation. d: Transposition type of mutation. e: Retro transposition type of mutation

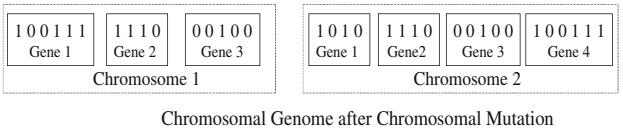

Fig. 9 Chromosomal mutation

difference between the two approaches is that in the NBGA an attempt is made to mimic the biological process of transposition mutation by representing the members of the population as a set of two chromosomes with different number of genes rather than a continuous binary string. Retro transposition is similar to transposition except that a gene is copied and repositioned in a new position and deleted from its original location. This situation is shown in Fig. 8e. In this case gene 2 from chromosome 1 is removed from its original position and replaced at the end.

3 *Chromosomal mutation* These are very large scale mutations and involve whole chromosomes or a piece of them and can alter many genes at a time in that chromosome. They are an important source of new genetic material. Figure 9 shows a chromosomal mutation where the gene sequence in chromosome 2 has been inverted.

These are the mutation operators that are used in our genetic algorithm: NBGA. The next section discusses the structure of the NBGA in detail.

6 Non-dominated sorting biologically motivated genetic algorithm (NBGA)

The NBGA designed by the authors implements Pareto ranking. For diversity preservation the crowding distance, as proposed by Deb et al. (2000), is used besides the mutation operators described in the previous section. The algorithm of the NBGA is as follows:

1. Randomly initialise the population
   (a) For $i = 1$ to Members in population
   (b) Initialise the population (binary string)
   (c) Initialise the rate of point mutation (binary string)
   (d) Initialise the rate of large mutation (binary string)
   (e) Initialise the rate of chromosome mutation (binary string)
2. Decode the population
3. Evaluate the objective functions
4. Classify the population into Pareto Fronts (Deb et al. 2000)
5. Assign the dummy fitness values (Deb et al. 2000)

6. Select the parents using tournament selection
7. Perform multipoint crossover
   (a) Perform the crossover of the chromosomes pertaining to the variables
   (b) Cross of the binary string representing rate of mutations
8. Perform point mutation
9. Perform large mutation
10. Perform Chromosomal Mutation
11. Combine the offspring and parent population.
12. If termination criteria satisfied then stop else go to step 3.

The NBGA differs from NSAG-II on the following accounts:

1. The method in which the variables are represented as binary strings (Sect. 4, Fig. 6). This method of representation of variables helps in the implementation of the proposed mutation operators.
2. The mutation operators implemented in the NBGA are different from the point mutation used in NSGA-II.

Based on the above mentioned features this algorithm has been named: NBGA. This algorithm uses the principle of non-dominated sorting as used by NSGA-II and the representation of the variables is similar to that of biological systems.

In the next section the performance parameters for the NBGA are discussed.

7 Performance parameters for NBGA

An important issue in multi-objective optimization is the quantitative performance comparison of the different algorithms. The most popular comparison methods are based on unary quality measures, i.e., the measure assigns a number to each approximation set that reflects a certain quality aspect. Usually a combination of them is used, e.g. Van Veldhuizen and Lamont (2000). Other methods are based on binary quality measures, which assign numbers to pairs of approximation sets, e.g. Zitzler and Thiele (1998). A comprehensive study on different performance measurement indices is given in Zitzler et al. (2003), Knowles et al. (2006), and Knowles and Corne (2002).

Four performance parameters are considered in order to analyse the performance of the NBGA for the test functions. These performance parameters are:

1 *Generational distance* The concept of generational distance was proposed by Van Veldhuizen and Lamont (1998). The purpose of this parameter is to estimate how far the Pareto front, obtained by a genetic algorithm, is from the actual Pareto front. Mathematically this parameter is defined as

$$\mathrm{GD} = \frac{\sqrt{\sum_{i=1}^{p} d_i^2}}{p} \tag{4}$$

where p is the number of points in the Pareto front obtained by the algorithm and d_i is the Euclidean distance between each solution point in the obtained Pareto front and the actual Pareto front for the problem under consideration. If the value of GD is zero then this indicates that all the points obtained by the algorithm lie on the true Pareto front. Any other value of GD will indicate how far the obtained Pareto front is from the actual one.

2 *Spacing* This factor indicates the spread of the solutions obtained. This metric was proposed by Schott (1995) and is defined as

$$S \triangleq \sqrt{\frac{1}{p-1}\sum_{i=1}^{p} (\bar{d} - d_i)^2} \tag{5}$$

where $d_i = \min_j(|f_1^i - f_1^j| + |f_2^i - f_2^j|)$ and $i, j = 1, 2, \ldots, p$, $\bar{d}$ is the mean of all d_i, and p is the number of non-dominated vectors found so far. If this value is zero then all the points in the obtained Pareto front are equidistantly placed.

3 *Error ratio* This factor was proposed by Van Veldhuizen and Lamont (1998) to determine (in percentage) the number of solutions in the obtained Pareto front that are not members of the true Pareto front. The mathematical definition of this factor is

$$e = \frac{\sum_{i=1}^{p} e_i}{p} \tag{6}$$

where p is the number of points in the obtained Pareto front and $e_i = 0$ if the point I is a member of the true Pareto front, else $e_i = 1$. If $e = 0$ then all the points of the obtained Pareto front lie on the true Pareto front.

4 *The two set coverage (SC)* This measurement index was proposed by Zitzler and Thiele (1998) to determine the relative coverage comparison of two sets. For two sets X' and X'', SC is defined as the mapping of the order pair (X', X'') to the interval [0, 1] as

$$\mathrm{SC}(X', X'') \triangleq \left|\left\{a'' \in X''; \exists a' \in X' : a' \preceq a''\right\}\right| / \left|X''\right| \tag{7}$$

Two set coverage is a binary performance measure index.

Using the above performance parameters the NBGA proposed by the authors in this paper was evaluated on some of the standard test functions for the sake of comparison. The NBGA was run 30 times on each of the test functions and the average values of the three parameters are reported here. The parameters of the NBGA used for the analysis in this paper are as follows:
Number of generations = 50
Number of individuals = 100
Crossover probability = 80%

Single point crossover was used.

The mutation rate was fixed between 0 and 10%. Since each type of mutation was represented as a binary string and subjected to cross-over the mutation rate for each individual changed during each generation.

These mutation rates were chosen after experimentation with different mutation rates.

8 Comparison of NBGA with other genetic algorithms

In this section the NBGA proposed by the authors is compared with other well known genetic algorithms, namely: non-dominated sorting genetic algorithm II (Deb et al. 2000), Micro-genetic algorithm for multi-objective optimization (Coello 1993) and Pareto archived evolution strategy (Darwin and Huxely 1859) which is an algorithm based on evolutionary strategy. The performance parameters considered for comparison are generational distance (GD), spacing (SP) and error ratio (ER). These parameters are discussed in Sect. 7. The setting of NBGA is the same as discussed in Sect. 7. The NBGA was run 30 times for each test function and the average value of the performance parameters is reported.

Test function 1

This test function was proposed by Kursawe (1991). Mathematically this function is defined as

$$\begin{aligned} \min f_1(\vec{x}) &= \sum_{i=1}^{n-1}\left(-10\exp\left(-0.2\sqrt{x_i^2+x_{i+1}^2}\right)\right) \\ \min f_2(\vec{x}) &= \sum_{i=1}^{n}\left(|x_i|^{0.8}+5\sin(x_i)^3\right) \end{aligned} \tag{8}$$

where $-5 \le x_1, x_2, x_3 \le 5$

The result of this test function is shown in Fig. 10 and the values of the performance parameters are shown in Table 1. The values of the performance parameters for non-dominated sorting genetic algorithm II, Pareto archived evolution strategy and micro-genetic algorithm for multi-objective optimization are taken from Coello et al. (1993). As in Coello et al. the fitness function was evaluated 12,000 times for the sake of comparison.

The NBGA is able to cover the entire Pareto Front. From Table 1 it can be seen that the NBGA performs better than the other three algorithms in terms of error ratio. For the generational distance performance

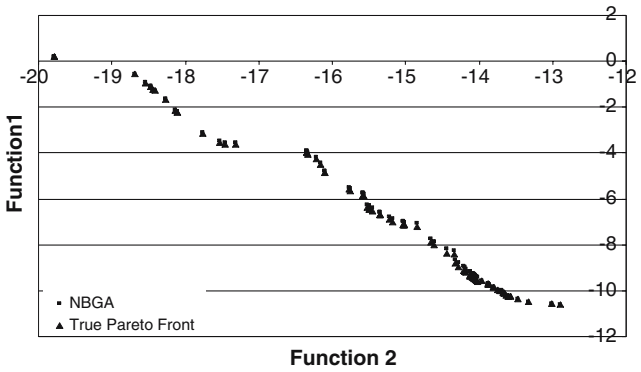

Fig. 10 Pareto Front produced by the NBGA and actual front for test function 1

Table 1 Result of error ratio, generational distance, spacing and two set coverage measure for test function 1

Error ratio	NBGA	NSGA II	Micro GA	PAES
Best	0.10	0.06	0.18	0.1
Worst	0.33	1.01	0.36	0.68
Average	0.17	0.56	0.27	0.27
Median	0.163	0.495	0.245	0.245
Standard deviation	0.05721	0.38452	0.05395	0.10489
Generational distance	**NBGA**	**NSGA II**	**Micro GA**	**PAES**
Best	0.00694	0.006905	0.006803	0.0147
Worst	0.10448	0.103095	0.010344	0.1572
Average	0.05	0.029255	0.008456	0.5491
Median	0.048	0.017357	0.008489	0.0494
Standard deviation	0.02916	0.02717	0.00099	0.03074
Spacing	**NBGA**	**NSGA II**	**Micro GA**	**PAES**
Best	0.03676	0.01842	0.07169	0.06411
Worst	0.10974	0.06571	0.20313	0.34096
Average	0.08622	0.03614	0.12890	0.19753
Median	0.08950	0.03609	0.12666	0.18663
Standard deviation	0.02173	0.01098	0.02993	0.06411
SC	**NBGA**	**Micro GA**	**NSGA-II**	**PAES**
NBGA	0.00	0.98	0.42	0.75
Micro GA	0.02	0.00	0.42	0.72
NSGA-II	0.04	0.04	0.00	0.68
PAES	0.03	0.03	0.17	0.00

parameter only micro GA performs better than the NBGA. The NSGA II performs better than the NBGA for spacing performance metric. The better performance of the NSGA-II compared to the NBGA for spacing performance metric and generational distance does not necessarily indicate that the NBGA performs worse compared to the NSGA-II. The spacing metric gauges how evenly the points in the Pareto Set, obtained by a GA, are distributed in the objective space (Knowles and Corne 2002) and it is quite possible that the True Pareto front has non-uniform distribution of points, i.e., the True Pareto Front might have higher concentration of solutions at some sections and lower concentration in other. Similarly, the generational distance metric has certain disadvantages. According to this metric it is better to find one solution close to the Pareto front than to find a set of solutions in which many solutions are on the true Pareto front and one solution is a small distance away from the true Pareto Front. Thus evaluation of the performance of any GA, based on spacing metric and generational distance parameter, may lead to erroneous conclusions. Hence, to compare two GAs the two set convergence metric is more suited. From Table 1 (two set coverage measure) it can be seen that SC(NBGA, MicroGA) = 0.98 and SC(MicroGA, NBGA) = 0.02, since SC(NBGA, MicroGA) > SC (MicroGA, NBGA), hence, the NBGA is relatively better than the MicroGA. Similarly, SC(NBGA, NSGA-II) = 0.98 and SC(NSGA - II, NBGA) = 0.04, i.e., SC(NBGA, NSGA-II) > SC(NSGA-II, NBGA) hence, it can be concluded that the NBGA again performs relatively better than the NSGHA-II. The analysis of the two set coverage measurement between the NBGA and the PAES gives SC(NBGA, PAES) = 0.75 and SC(PAES, NBGA) = 0.03, since SC(NBGA, PAES) > SC(PAES, NBGA), so the NBGA is *relatively* better than the PAES. From this analysis of the results of the two set coverage measurement it can be concluded that the NBGA performs better for this test function, as compared to the other comparison algorithms.

Test function 2

This test function was proposed by Kita et al. (1996). Mathematical definition of this problem is as follows

$$\max\ F = \left(f_1(x,y), f_2(x,y)\right) \tag{9}$$

where

$$f_1(x,y) = -x^2 + y, f_2(x,y) = \frac{1}{2}x + y + 1$$

subject to

$$0 \geq \frac{1}{6}x + y - \frac{13}{2}, \quad 0 \geq \frac{1}{2}x + y - \frac{15}{2}, \quad 0 \geq 5x + y - 30$$

and $x, y \geq 0$. The range used in this case is $0 \leq x, y \leq 7$ Kita et al. (1996). The results are shown in Fig. 11 and Table 2. The values of the performance parameters for the NBGA II, the PAES and the MicroGA are taken from Coello et al. (2004). As in Coello et al. the fitness function was evaluated 5,000 times for the sake of comparison.

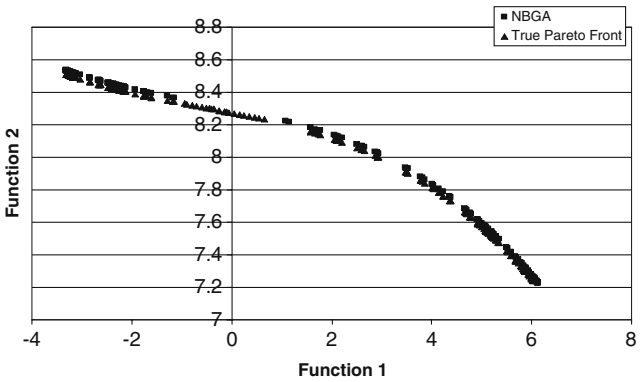

Fig. 11 Pareto Front produced by the NBGA and actual front for test function 2

Table 2 Results of error ratio, generational distance, spacing and two set coverage measure for test function 2

Error ratio	NBGA	NSGA II	Micro GA	PAES
Best	0.05405	0.75	0.734694	0.93
Worst	0.18	0.99	1.01639	1.01
Average	0.05	0.8965	0.927706	0.993
Median	0.042	0.92	0.936365	1.01
Standard deviation	0.04282	0.06714	0.06874	0.02536
Generational distance	NBGA	NSGA II	Micro GA	PAES
Best	0.04808	0.003885	0.00513	0.0113
Worst	0.11628	0.678449	0.912065	0.9192
Average	0.07	0.084239	0.150763	0.1932
Median	0.066	0.011187	0.089753	0.0333
Standard deviation	0.02125	0.16524	0.21656	0.24965
Spacing	NBGA	NSGA II	Micro GA	PAES
Best	0.00319	0.00103	0.06561	0.00667
Worst	0.01714	1.48868	1.64386	0.43287
Average	0.00992	0.09849	0.31502	0.11010
Median	0.01045	0.02717	0.12977	0.08200
Standard deviation	0.00416	0.32738	0.42174	0.09960
SC	NBGA	Micro GA	NSGA-II	PAES
NBGA	0.00	0.73	0.63	0.64
Micro GA	0.12	0.00	0.26	0.37
NSGA-II	0.21	0.08	0.00	1.00
PAES	0.21	0.21	0.00	0.00

From Table 2 it can be seen that SC(NBGA, MicroGA) = 0.73 and SC(MicroGA,NBGA) = 0.12, since SC(NBGA,MicroGA) > SC(MicroGA,NBGA), hence, the NBGA is relatively better than the MicroGA. Similarly, SC(NBGA,NSGA-II) = 0.63 and SC(NSGA-II,NBGA) = 0.21, i.e., SC(NBGA,NSGA-II) > SC(NSGA-II,NBGA) hence, it can be concluded that the NBGA again performs relatively better than the NSGA-II. The analysis of the two set coverage measurement between the NBGA and the PAES gives SC(NBGA,PAES) = 0.64 and SC(PAES,NBGA) = 0.21, since SC(NBGA,PAES) > SC(PAES,NBGA), so the NBGA is *relatively* better than the PAES. From this analysis of the results of the two set coverage measurement it can be concluded that the NBGA performs better for this test function as compared to the other comparison algorithms. The performance of the NBGA, based on error ratio, generational distance and spacing metric, is better than the MicroGA, the NSGA-II and the PAES (Table 2).

Test function 3

This test function was proposed by Deb (1999). Mathematically this function is represented as

$$\begin{aligned} &\min f_1(x_1, x_2) = x_1 \\ &\min f_2(x_1, x_2) = g(x_1, x_2) \cdot h(x_1, x_2) \end{aligned} \tag{10}$$

where

$$g(x_1, x_2) = 11 + x_2^2 - 10\cos(2\pi x_2)$$

$$h(x_1, x_2) = \begin{cases} 1 - \sqrt{\dfrac{f_1(x_1, x_2)}{g(x_1, x_2)}}, & \text{if } f_1(x_1, x_2) \le g(x_1, x_2) \\ 0, & \text{otherwise} \end{cases}$$

and $0 \le x_1 \le 1, -30 \le x_2 \le 30$.

The result of this test function is shown in Fig. 12. The values of the performance parameters are given in Table 3. The values of the performance parameters for the NBGA II, the PAES and the MicroGA are taken from Coello et al. (2004). As in Coello et al. the fitness function was evaluated 5,000 times for the sake of comparison.

From the above figure it can be seen that the NBGA was able to cover the entire Pareto front. From Table 3 it can be seen that in terms of error ratio the NBGA performs better than other comparison algorithms. In terms of generational distance the NSGA-II performs better than the other algorithms (Table 3). Based on spacing performance index the NBGA performs better than other algorithms (Table 3). Hence based on the unary performance metric, the NBGA performs better than other algorithms in terms of error ratio and spacing. The NSGA-II has better performance in terms of generational distance. From Table 1 (two set coverage measure) it can be seen that SC(NBGA,MicroGA) = 1 and SC(MicroGA,NBGA) = 0, since SC(NBGA, MicroGA) > SC(MicroGA,NBGA), hence, the NBGA is relatively better than the MicroGA. Similarly, SC(NBGA,NSGA-II) = 0.49 and SC(NSGA-II, NBGA) = 0.18, i.e., SC(NBGA,NSGA-II) > SC(NSGA-II,NBGA) hence it can be concluded that

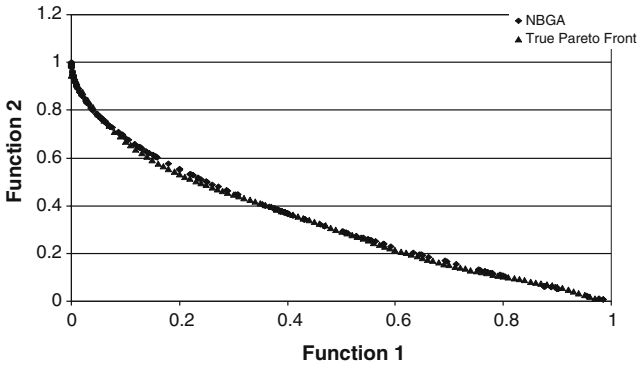

Fig. 12 Pareto Front produced by the NBGA and actual front for test function 3

Table 3 Results of error ratio, generational distance, spacing and two set coverage measure for test function 3

Error ratio	NBGA	NSGA II	Micro GA	PAES
Best	0.0070067	0	0.02	0.06
Worst	0.0186447	1.01	1.04545	1.01
Average	0.0113304	0.35	0.2568	0.4485
Median	0.0112476	0.2	0.19	0.24
Standard deviation	0.0035859	0.39615	0.25646	0.38199
Generational distance	**NBGA**	**NSGA II**	**Micro GA**	**PAES**
Best	0.008621	0.000133	8.74×10^{-5}	0.000114
Worst	0.073171	0.163146	0.811403	1.99851
Average	0.03	0.023046	0.047049	0.163484
Median	0.028	0.000418	0.000236	0.058896
Standard deviation	0.02072	0.04543	0.18116	0.44130
Spacing	**NBGA**	**NSGA II**	**Micro GA**	**PAES**
Best	0.00035	0.00021	0.00760	0.00916
Worst	0.00170	0.01023	5.56270	19.88640
Average	0.00087	0.00369	0.34166	1.11462
Median	0.00075	0.00209	0.29950	0.01876
Standard deviation	0.00037	0.00337	1.24756	4.43459
SC	**NBGA**	**Micro GA**	**NSGA-II**	**PAES**
NBGA	0.00	1.00	0.49	1
Micro GA	0.00	0.00	0.00	0.36
NSGA-II	0.50	1.00	0.00	0.29
PAES	0	0.00	0.00	0.00

the NBGA performs relatively better than the NSGA-II. The analysis of the two set coverage measurement between the NBGA and the PAES gives SC(NBGA, PAES) = 1 and SC(PAES,NBGA) = 0, since SC(NBGA, PAES) > SC(PAES,NBGA), so the NBGA is *relatively* better than the PAES. From this analysis of the results of two set coverage measure it can be concluded that the NBGA performs better for this test function as compared to the other comparison algorithms.

Test function 4

This test function was proposed by Deb (1999). The mathematical form of this test function is

$$\min f_1(x_1, x_2) = x_1$$
$$\min f_2(x_1, x_2) = \frac{g(x_2)}{x_1} \tag{11}$$

where

$$g(x_2) = 2 - \exp\left\{-\left(\frac{x_2 - 0.2}{0.004}\right)^2\right\} - 0.8\exp\left\{-\left(\frac{x_2 - 0.6}{0.4}\right)^2\right\}$$

and $0.1 \le x_1 \le 1.0, 0.1 \le x_2 \le 0.1$.

The result of this test function is shown in Fig. 13 and the values of the performance metric are given in Table 4. The values of the performance parameters for the NBGA II, the PAES and the MicroGA are taken from Coello et al. (2004). As in Coello et al. the fitness function was evaluated 10,000 times for the sake of comparison.

Figure 13 and Table 4 (error ratio, generational distance and spacing) show that the NBGA was able to cover the entire Pareto front and its performance was better than the other three algorithms in terms of all the performance parameters. From Table 4 (two set coverage measure) it can be seen that SC(NBGA, MicroGA) = 1 and SC(MicroGA,NBGA) = 0, since SC(NBGA,MicroGA) > SC(MicroGA,NBGA), hence, the NBGA is relatively better than the MicroGA. Similarly, SC(NBGA,NSGA-II) = 0.75 and SC(NSGA-II, NBGA) = 0.2, i.e., SC(NBGA,NSGA-II) > SC(NSGA-II,NBGA) hence, it can be concluded that the NBGA again performs relatively better than the NBGA. The analysis of two set coverage measure between the NBGA and the PAES gives SC(NBGA,PAES) = 0.62 and SC(PAES,NBGA) = 0.15, since SC(NBGA, PAES) > SC(PAES,NBGA), so the NBGA is *relatively* better than the PAES. From this analysis of the results of the two set coverage measurement it can be concluded that the NBGA performs better for this test function as compared to the other comparison algorithms.

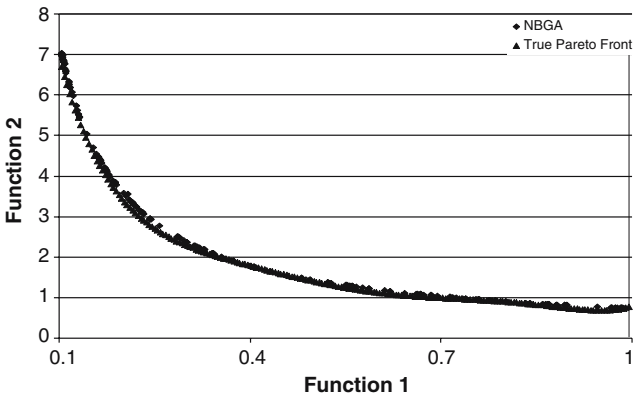

Fig. 13 Pareto Front produced by the NBGA and actual front for test function 4

Table 4 Results of error ratio, generational distance, spacing and two set coverage measure for test function 4

Error ratio	NBGA	NSGA II	Micro GA	PAES
Best	0.036251	0.02	0.08	0.02
Worst	0.14	1.01	1.01	1.01
Average	0.06	0.4145	0.252	0.489
Median	0.060	0.115	0.16	0.28
Standard deviation	0.02584	0.45939	0.23158	0.43812
Generational distance	**NBGA**	**NSGA II**	**Micro GA**	**PAES**
Best	0.00806	0.0007	0.00047	0.00045
Worst	0.02718	0.20847	0.1835	0.22167
Average	0.01274	0.04424	0.04347	0.19476
Median	0.016	0.00086	0.05004	0.07036
Standard deviation	0.00768	0.07368	0.04821	0.20469
Spacing	**NBGA**	**NSGA II**	**Micro GA**	**PAES**
Best	0.00031	0.02609	0.03027	0.04784
Worst	0.08113	0.06142	0.81764	0.66468
Average	0.08185	0.03745	0.21358	0.19477
Median	0.05368	0.03553	0.06301	0.07037
Standard deviation	0.08426	0.00924	0.25059	0.20469
SC	**NBGA**	**Micro GA**	**NSGA-II**	**PAES**
NBGA	0.00	1.00	0.75	0.62
Micro GA	0.00	0.00	0.04	0.46
NSGA-II	0.20	0.06	0.00	0.43
PAES	0.15	0.00	0.01	0.00

9 Performance of NBGA on multivariable test functions

Zitzler and Deb (2000a) have identified several features that may cause difficulties for multi-objective GAs in:

(1) converging to the Pareto optimal front and (2) maintaining diversity within the population. Concerning the first issue, multimodality, deception, and isolated optima are well known problem areas in single-objective evolutionary optimization. The second issue is important in order to achieve a well distributed non-dominated front. However, certain characteristics of the Pareto optimal front may prevent a GA from finding diverse Pareto optimal solutions: convexity or non-convexity, discreteness, and non-uniformity. For each of the six problem features mentioned, a corresponding test function is constructed, following the guidelines in Zitzler and Deb (2000a).

In the following analysis the population size was 200 and 100 iterations were performed; other parameters remain the same as discussed in Sect. 8.

Test Function 5

The test function is described mathematically as Zitzler et al. (2000b):

$$\begin{aligned} &\min f_1(x_1, x_2) = x_1 \\ &\min f_2(x_1, x_2) = g(\vec{x})h(f_1, g) \end{aligned} \tag{12}$$

where

$$g(x) = 1 + 9\sum_{i=2}^{m} x_i/(m-1)$$

$$h(f_1, g) = 1 - \sqrt{f_1/g}$$

and

$$m = 30 \quad \text{and} \quad x_i \in [0, 1]$$

The result of this test function is shown in Fig. 14 and the values of the performance metric are given in Table 5.

In terms of error ratio (Table 5), Generation distance (Table 5) and spacing (Table 5), the NBGA performs better than the other algorithms. From Table 5 (two set coverage measure) it can be seen that SC(NBGA, MicroGA) = 0.44 and SC(MicroGA,NBGA) = 0.06,

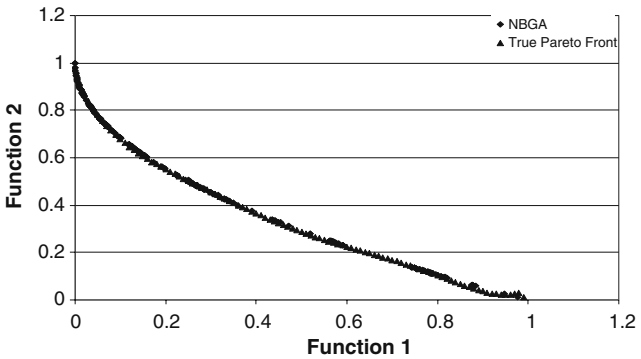

Fig. 14 Pareto Front produced by the NBGA and actual front for multivariable test function 5

Table 5 Results of error ratio, generational distance, spacing and two set coverage measure for test function 5

Error ratio	NBGA	NSGA II	Micro GA	PAES
Best	0.0047	0.0093	0.0102	0.0111
Worst	0.0367	0.0732	0.0804	0.0877
Average	0.0147	0.0293	0.0322	0.0351
Median	0.0121	0.0241	0.0265	0.0289
Standard deviation	0.0103	0.0206	0.0226	0.0247
Generation dist	**NBGA**	**NSGA II**	**Micro GA**	**PAES**
Best	0.0051	0.0101	0.0101	0.0102
Worst	0.0758	0.1513	0.1506	0.1521
Average	0.0367	0.0734	0.0730	0.0738
Median	0.0334	0.0667	0.0664	0.0670
Standard deviation	0.0200	0.0400	0.0398	0.0402
Spacing	**NBGA**	**NSGA II**	**Micro GA**	**PAES**
Best	0.00064	0.00128	0.00141	0.00154
Worst	0.00126	0.00251	0.00276	0.00301
Average	0.00083	0.00165	0.00182	0.00198
Median	0.00079	0.00157	0.00173	0.00188
Standard deviation	0.00017	0.00034	0.00038	0.00041
SC	**NBGA**	**Micro GA**	**NSGA-II**	**PAES**
NBGA	0.00	0.44	0.99	1.00
Micro GA	0.06	0.00	0.98	1.00
NSGA-II	0.02	0.02	0.00	0.39
PAES	0.00	0.05	0.63	0.00

since SC(NBGA,MicroGA) > SC(MicroGA,NBGA), hence the NBGA is relatively better than the MicroGA. Similarly, SC(NBGA,NSGA-II) = 0.99 and SC(NSGA-II,NBGA) = 0.02, i.e., SC(NBGA,NSGA-II) > SC(NSGA-II,NBGA) hence, it can be concluded that the NSGII again performs relatively better than the NBGA. The analysis of the two set coverage measurement between the NBGA and the PAES gives SC(NBGA,PAES) = 1 and SC(PAES,NBGA) = 0, since SC(NBGA,PAES) > SC(PAES,NBGA), so the NBGA is *relatively* better than the PAES. From this analysis of the results of the two set coverage measurement it can be concluded that the NBGA performs better for this test function as compared to the other comparison algorithms.

Test function 6

The test function is described mathematically as Zitzler et al. (2000b):

$$\begin{aligned} &\min f_1(x_1, x_2) = x_1 \\ &\min f_2(x_1, x_2) = g(\vec{x})h(f_1, g) \end{aligned} \tag{13}$$

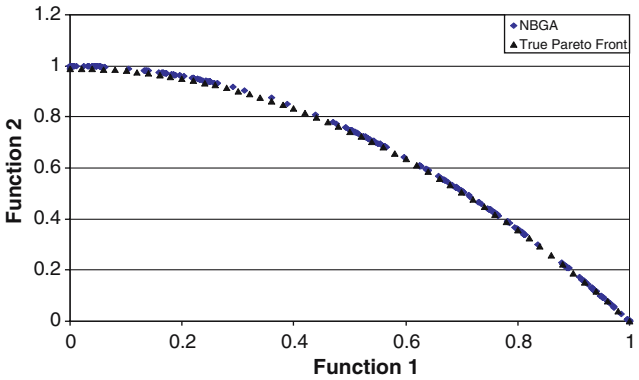

Fig. 15 Pareto Front produced by the NBGA and actual front for multivariable test function 6

where

$$g(x) = 1 + 9\sum_{i=2}^{m} x_i/(m-1)$$

$$h(f_1, g) = 1 - (f_1/g)^2$$

and

$$m = 30 \quad \text{and} \quad x_i \in [0, 1]$$

The result of this test function is shown in Fig. 15 and the values of the performance metric are given in Table 6.

In terms of error ratio (Table 6) the NBGA performs better than the other algorithms. The MicroGA performs best in terms of generation distance (Table 6) and in terms of spacing (Table 6), the NBGA performs better than the other algorithms. From Table 6 (two set coverage measure) it can be seen that SC(NBGA, MicroGA) = 0.68 and SC(MicroGA,NBGA) = 0.5, since SC(NBGA,MicroGA) > SC(MicroGA,NBGA), hence the NBGA is relatively better than the MicroGA. Similarly, SC(NBGA,NSGA-II) = 0.73 and SC(NSGA-II,NBGA) = 0.42, i.e., SC(NBGA,NSGA-II) > SC(NSGA-II,NBGA) hence, it can be concluded that the NSGII again performs relatively better than the NBGA. The analysis of the two set coverage measurement between the NBGA and the PAES gives SC(NBGA,PAES) = 0.91 and SC(PAES,NBGA) = 0.37, since SC(NBGA,PAES) > SC(PAES,NBGA), so the NBGA is *relatively* better than the PAES. From this analysis of the results of the two set coverage measurement it can be concluded that the NBGA performs better for this test function as compared to the other comparison algorithms.

Test function 7

The test function is described as Zitzler et al. (2000b):

$$\begin{aligned} &\min f_1(x_1, x_2) = x_1 \\ &\min f_2(x_1, x_2) = g(\vec{x})h(f_1, g) \end{aligned} \tag{14}$$

Table 6 Results of error ratio, generational distance, spacing and two set coverage measure for test function 6

Error ratio	NBGA	NSGA II	Micro GA	PAES
Best	0.00178	0.00196	0.00196	0.00194
Worst	0.02913	0.03204	0.03201	0.03178
Average	0.01182	0.01300	0.01298	0.01289
Median	0.00943	0.01037	0.01036	0.01029
Standard deviation	0.00807	0.00887	0.00886	0.00880
Generational distance	NBGA	NSGA II	Micro GA	PAES
Best	0.00503	0.30130	0.30063	0.29722
Worst	0.30435	0.30130	0.30063	0.29722
Average	0.03003	0.02973	0.02966	0.02933
Median	0.00559	0.00553	0.00552	0.00546
Standard deviation	0.07599	0.07523	0.07507	0.07422
Spacing	NBGA	NSGA II	Micro GA	PAES
Best	0.00011	0.00012	0.00014	0.00026
Worst	0.00533	0.00586	0.00644	0.01228
Average	0.00100	0.00110	0.00121	0.00230
Median	0.00056	0.00062	0.00068	0.00130
Standard deviation	0.00141	0.00155	0.00170	0.00324
SC	NBGA	Micro GA	NSGA-II	PAES
NBGA	0.0	0.68	0.73	0.91
Micro GA	0.5	0.0	0.5	0.5
NSGA-II	0.42	0.41	0.0	0.5
PAES	0.037	0.32	0.42	0.0

where

$$g(x) = 1 + 9\sum_{i=2}^{m} x_i/(m-1)$$

$$h(f_1, g) = 1 - \sqrt{f_1/g} - (f_1/g)\sin(10\pi f_1)$$

and

$$m = 30 \quad \text{and} \quad x_i \in [0, 1]$$

The result of this test function is shown in Fig. 16 and the values of the performance metric are given in Table 7.

In terms of error ratio (Table 7), generation distance (Table 7) and spacing (Table 7), the NBGA performs better than the other algorithms. From Table 7 (two set coverage measure) it can be seen that SC(NBGA, MicroGA) = 0.68 and SC(MicroGA,NBGA) = 0.28, since SC(NBGA,MicroGA) > SC(MicroGA,NBGA), hence the NBGA is relatively better than the Micro-GA. Similarly, SC(NBGA,NSGA-II) = 0.66 and SC(NSGA-II,NBGA) = 0.29, i.e., SC(NBGA,NSGA-II) > SC(NSGA-II,NBGA) hence, it can be concluded that the NSGII again performs relatively better than the

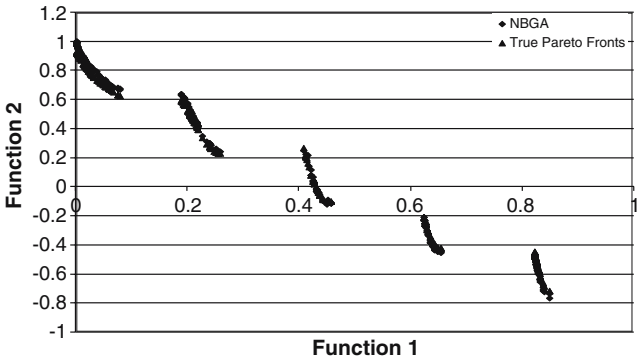

Fig. 16 Pareto Front produced by the NBGA and actual front for multivariable test function 7

Table 7 Results of error ratio, generational distance of error ratio, spacing of error ratio and two set coverage measure for test function 7

Error ratio	NBGA	NSGA II	Micro GA	PAES
Best	0.00178	0.00196	0.00195	0.00196
Worst	0.02777	0.03052	0.03039	0.03048
Average	0.01263	0.01388	0.01382	0.01386
Median	0.00994	0.01092	0.01088	0.01091
Standard deviation	0.00660	0.00726	0.00723	0.00725
Generational distance	**NBGA**	**NSGA II**	**Micro GA**	**PAES**
Best	0.00503	0.00552	0.00603	0.00660
Worst	0.30435	0.33403	0.36498	0.40003
Average	0.05602	0.06149	0.06718	0.07363
Median	0.02764	0.03034	0.03315	0.03633
Standard deviation	0.08309	0.09120	0.09964	0.10921
Spacing	**NBGA**	**NSGA II**	**Micro GA**	**PAES**
Best	0.00012	0.00014	0.00015	0.00016
Worst	0.05134	0.05630	0.06128	0.06695
Average	0.01521	0.01668	0.01816	0.01984
Median	0.01524	0.01672	0.01820	0.01988
Standard deviation	0.01478	0.01620	0.01764	0.01927
SC	**NBGA**	**Micro GA**	**NSGA-II**	**PAES**
NBGA	0	0.68	0.66	0.68
Micro GA	0.28	0	0.65	0.66
NSGA-II	0.29	0.28	0	0.65
PAES	0.26	0.289	0.28	0

NBGA. The analysis of the two set coverage measurement between the NBGA and the PAES gives SC(NBGA,PAES) = 0.68 and SC(PAES,NBGA) = 0.26, since SC(NBGA,PAES) > SC(PAES,NBGA), so the NBGA is *relatively* better than the PAES. From this analysis of the results of two set coverage measure it can be concluded that the NBGA performs better for this test function as compared to the other comparison algorithms.

10 Experimental evidence of importance of mutations in performance of NBGA

These experiments were designed to compare the performance of the NBGA, proposed in this paper, with and without different types of mutations (point mutation, Large mutation and chromosome mutation). In this work the four test functions, discussed in the previous section, have been used together with the three performance parameters: error ratio, generational distance and spacing. The following eight experiments were performed:

(a) *With mutation* In this case the NBGA was run with all the types of mutations and the performance parameters for the test functions, defined in Sect. 7, were considered.
(b) *Without mutation* The NBGA was run without point mutation, large mutation and chromosome mutation. The performance parameters for the test functions defined in Sect. 7 were considered.
(c) *Without point mutation* The NBGA was run without Point mutation and other mutations (large and chromosome mutation) were engaged. The performance parameters for the test functions defined in Sect. 7 were considered.
(d) *Without large mutation* The NBGA was run without Large mutation and other mutations (point and chromosome mutation) were engaged. The performance parameters for the test functions defined in Sect. 7 were considered.
(e) *Without chromosome mutation* The NBGA was run without Chromosome mutation and other mutations (point and large mutation) were engaged. The performance parameters for the test functions defined in Sect. 7 were considered.
(f) *Without point and large mutation* The NBGA was run without point and large mutation and chromosome mutation was engaged. The performance parameters for the test functions defined in Sect. 7 were considered.
(g) *Without point and chromosome mutation* The NBGA was run without point and chromosome mutation and large mutation was engaged. The performance parameters for the test functions defined in Sect. 7 were considered.
(h) *Without chromosome and large mutation* The NBGA was run without Chromosome and Large

mutation and point mutation was engaged. The performance parameters for the test functions defined in Sect. 7 were considered.

Table 8 Experimental values of performance parameters for test function 1 with all mutation, without chromosome mutation, without large mutation, without small mutation, without large and chromosome mutation, without small and chromosome mutation, without small and large mutation and without any mutation

	Error ratio	Generational distance	Spacing
With all mutation			
Best	0.0061284	0.0110497	0.01786
Worst	0.0250286	0.1348315	0.04249
Average	0.0152484	0.06	0.03495
Median	0.0143543	0.0569337	0.03637
Standard deviation	0.0061901	0.0338507	0.00707
Without chromosome mutation			
Best	0.076973	0.018541	0.086829
Worst	0.314360	0.226247	0.206617
Average	0.191520	0.098847	0.169921
Median	0.180290	0.095535	0.176864
Standard deviation	0.077748	0.056801	0.034391
Without large mutation			
Best	0.081447	0.009259	0.060763
Worst	0.388474	0.088710	0.157557
Average	0.197284	0.044053	0.113201
Median	0.178941	0.040239	0.113094
Standard deviation	0.100128	0.025787	0.032612
Without small mutation			
Best	0.123981	0.021429	0.123981
Worst	0.238703	0.147059	0.238703
Average	0.172170	0.059345	0.172170
Median	0.163728	0.044263	0.163728
Standard deviation	0.040283	0.044126	0.040283
Without large and chromosome mutation			
Best	0.107453	0.024000	0.087042
Worst	0.505248	0.131148	0.409273
Average	0.223468	0.085236	0.181019
Median	0.188319	0.093639	0.152547
Standard deviation	0.115602	0.038825	0.093643
Without small and chromosome mutation			
Best	0.105169	0.014358	0.10517
Worst	0.310676	0.093023	0.31068
Average	0.196150	0.038681	0.19615
Median	0.190054	0.033333	0.19005
Standard deviation	0.062143	0.031107	0.06214
Without small and large mutation			
Best	0.056609	0.018634	0.056609
Worst	0.467372	0.142857	0.467372
Average	0.176223	0.066027	0.176223
Median	0.148774	0.056141	0.148774
Standard deviation	0.125256	0.041151	0.125256
Without any mutation			
Best	0.162712	0.000000	0.100152
Worst	0.384013	0.107692	0.236367
Average	0.220684	0.058129	0.135835
Median	0.197426	0.046750	0.121519
Standard deviation	0.065886	0.039763	0.040554

Experiment result

These experiments were designed to establish whether the mutation operators implemented in the

Table 9 Experimental values of performance parameters for test function 2 with all mutation, without chromosome mutation, without large mutation, without small mutation, without large and chromosome mutation, without small and chromosome mutation, without small and large mutation and without any mutation

	Error ratio	Generational distance	Spacing
With all mutation			
Best	0.001042	0.020348	0.000963
Worst	0.058807	0.163800	0.054376
Average	0.012415	0.101232	0.011479
Median	0.006169	0.110466	0.005704
Standard deviation	0.017342	0.040208	0.016036
Without chromosome mutation			
Best	0.087042	0.043478	0.007137
Worst	0.402788	0.350000	0.402788
Average	0.181019	0.216307	0.085032
Median	0.152547	0.236037	0.042253
Standard deviation	0.118782	0.085914	0.118782
Without large mutation			
Best	0.064145	0.073171	0.011158
Worst	0.554539	0.354430	0.128159
Average	0.315458	0.275865	0.048853
Median	0.359667	0.318223	0.041975
Standard deviation	0.177693	0.089896	0.032078
Without small mutation			
Best	0.087042	0.225806	0.021045
Worst	1.069168	0.636364	1.069168
Average	0.181019	0.454252	0.237581
Median	0.152547	0.386364	0.061069
Standard deviation	0.377243	0.168480	0.377243
Without large and chromosome mutation			
Best	0.114225	0.017271	0.012622
Worst	1.297382	0.771350	0.335019
Average	0.360635	0.446554	0.104474
Median	0.221646	0.509777	0.084343
Standard deviation	0.356276	0.219659	0.094380
Without small and chromosome mutation			
Best	0.012430	0.066667	0.012430
Worst	1.335841	0.400000	1.335841
Average	0.192754	0.213556	0.192754
Median	0.029238	0.213821	0.029238
Standard deviation	0.412924	0.103887	0.412924
Without small and large mutation			
Best	0.010803	0.093220	0.008672
Worst	0.229761	0.325581	0.184448
Average	0.059341	0.202101	0.047638
Median	0.038890	0.226061	0.031220
Standard deviation	0.065954	0.097568	0.052947
Without any mutation			
Best	0.056318	0.057471	0.005632
Worst	0.338023	0.277228	0.338023
Average	0.129815	0.123162	0.069713
Median	0.088712	0.122200	0.041063
Standard deviation	0.092002	0.062070	0.099446

NBGA played a significant role, or not. The results of the experiments are summarised here.

1. *Test Function 1* The results for the first test function defined in Sect. 8 are summarised in Table 8.

Table 10 Experimental values of performance parameters for test function 3 with all mutation, without chromosome mutation, without large mutation, without small mutation, without large and chromosome mutation, without small and chromosome mutation, small and large mutation and without any mutation

	Error ratio	Generational distance	Spacing
With all mutation			
Best	0.005542	0.010155	0.011519
Worst	0.028164	0.046329	0.018156
Average	0.010171	0.028615	0.013965
Median	0.006356	0.030161	0.013573
Standard deviation	0.007376	0.011354	0.002165
Without chromosome mutation			
Best	0.044169	0.034014	0.027184
Worst	0.224457	0.155172	0.042846
Average	0.081062	0.095841	0.032954
Median	0.050652	0.101019	0.032030
Standard deviation	0.058788	0.038029	0.005109
Without large mutation			
Best	0.044691	0.015504	0.035335
Worst	0.093148	0.258065	0.241645
Average	0.061268	0.135715	0.061325
Median	0.062464	0.142857	0.039453
Standard deviation	0.014012	0.073680	0.063617
Without small mutation			
Best	0.011229	0.015038	0.000828
Worst	0.299241	0.080000	0.715964
Average	0.067761	0.040471	0.151429
Median	0.017965	0.037282	0.006117
Standard deviation	0.096177	0.019762	0.298000
Without large and chromosome mutation			
Best	0.008474	0.008000	0.000730
Worst	0.145713	0.148515	0.001378
Average	0.041155	0.045651	0.001106
Median	0.018487	0.035344	0.001107
Standard deviation	0.055232	0.042118	0.000246
Without small and chromosome mutation			
Best	0.007849	0.028986	0.000711
Worst	0.187073	0.241379	0.075598
Average	0.030406	0.076117	0.012564
Median	0.011690	0.067886	0.001123
Standard deviation	0.055333	0.061065	0.023720
Without small and large mutation			
Best	0.009351	0.018018	0.000943
Worst	0.273262	0.225806	0.153115
Average	0.066809	0.051788	0.023786
Median	0.030422	0.035057	0.001238
Standard deviation	0.08847	0.061907	0.050960
Without any mutation			
Best	0.008934	0.010101	0.001147
Worst	0.811530	0.368421	0.725834
Average	0.120845	0.098136	0.085323
Median	0.048608	0.057144	0.002334
Standard deviation	0.245662	0.115493	0.227382

Table 11 Experimental values of performance parameters for test function 4 with all mutation, without chromosome mutation, without large mutation, without small mutation, without large and chromosome mutation, without small and chromosome mutation, without small and large mutation and without any mutation

	Error ratio	Generational distance	Spacing
With all mutation			
Best	0.002816	0.011472	0.002360
Worst	0.224425	0.231899	0.198270
Average	0.055564	0.058757	0.044591
Median	0.018163	0.032950	0.007018
Standard deviation	0.089515	0.068470	0.069140
Without chromosome mutation			
Best	0.010183	0.026549	0.001259
Worst	0.811530	0.536679	0.105783
Average	0.200921	0.135981	0.020499
Median	0.065679	0.076255	0.003657
Standard deviation	0.323691	0.158458	0.037089
Without large mutation			
Best	0.048314	0.0265487	0.00126
Worst	0.3171449	0.5366789	0.10578
Average	0.1324197	0.1359808	0.02050
Median	0.0728534	0.0762553	0.00366
Standard deviation	0.10806	0.15846	0.03709
Without small mutation			
Best	0.048028	0.075188	0.001871
Worst	0.350790	0.183007	0.203832
Average	0.101910	0.141991	0.101278
Median	0.070038	0.150971	0.120175
Standard deviation	0.091231	0.036893	0.076806
Without large and chromosome mutation			
Best	0.038025	0.025641	0.002566
Worst	0.80086	0.165680	0.206318
Average	0.134931	0.124859	0.081911
Median	0.053114	0.152178	0.068889
Standard deviation	0.23569	0.053692	0.084912
Without small and chromosome mutation			
Best	0.045403	0.027972	0.003666
Worst	0.421778	0.164706	0.158907
Average	0.122842	0.131472	0.101494
Median	0.088802	0.155154	0.128731
Standard deviation	0.112174	0.054880	0.059716
Without small and large mutation			
Best	0.038670	0.034884	0.001479
Worst	0.287102	0.192308	0.177334
Average	0.143837	0.123554	0.050430
Median	0.076175	0.154696	0.003440
Standard deviation	0.108723	0.061616	0.077186
Without any mutation			
Best	0.041263	0.036364	0.004254
Worst	0.297187	0.194444	0.194298
Average	0.156210	0.157713	0.073749
Median	0.144896	0.164394	0.028736
Standard deviation	0.099514	0.047936	0.079097

From the above table it is clear that the best performance of the NBGA is obtained when all the mutation types are activated.

2. *Test Function 2* The results for the second test function are summarised in Table 9. From the table below it is evident that the performance of the NBGA is best when all mutation operators are engaged.

3. *Test Function 3* The results for the third test function are summarised in Table 10.

From the above table it can be seen that the NBGA performs best with all the types of mutation operators.

4. *Test Function 4*

The results for his test function are summarised in Table 11.

From Table 11 it is evident that when the NBGA is run with all the mutation operators best results are obtained.

From the results of this section it is evident that the performance of the NBGA is best, for all the test functions, when all the mutation operators are used.

11 Conclusions and future work

In this work the concept of mutation was introduced in genetic algorithms. These mutations are well studied in the field of evolutionary biology. Also in evolutionary biology, it is gradually being established that mutations are one of the prime sources of diversity in nature. A simplified implementation of these mutations is done in the NBGA proposed by here. The performance of the NBGA on various test functions was better than the other genetic algorithms. Furthermore, the influence of these mutation operators was validated by a series of experiments. These experiments prove that mutations improve the performance of the NBGA. The future direction of work will be to investigate the impact of the rate of mutation and the rate of reproduction on the performance of the NBGA. Furthermore, the study regarding impact of mass extinction on the performance of the NBGA will also be of interest.

Acknowledgment This work is supported by Delft University of Technology, The Netherlands, and Trimerics GmbH, Germany.

References

Aguirre H, Tanaka K (2005) Selection, drift, recombination, and mutation in multiobjective evolutionary algorithms on scalable MNK-Landscapes. Evolutionary multi-criterion optimization. 3rd international conference, EMO 2005, Lecture notes in computer science, vol 3410, pp 355–369

Bagley JD (1967) The behavior of adaptive systems which employ genetic and correlation algorithms. PhD Thesis, University of Michigan

Brizuela CA, Aceves R (2003) Experimental genetic operators analysis for the multi-objective permutation flowshop. volutionary multi-criterion optimization. In: 2nd international conference, EMO 2003, vol 2632, pp 578–592

Chan TM, Man KF, Tang KS, Kwong S (2005) A jumping gene algorithm for multiobjective resource management in wideband CDMA systems. Comput J 48:749–768

Coello CAC (1993) A micro-genetic algorithm for multiobjective optimization. Lecture Notes in Computer Science

Coello CAC, Veldhuizen V et al. (2002) Evolutionary algorithms for solving multi-objective problems. Kluwer, New York

Coello CAC, PulidoD GT, Lechuga MS (2004) Handling multiple objectives with particle swarm optimization. IEEE Trans Evol Comput 8:256–279

Darwin C, Huxely J (1859) Origin of Species. Signet Classics

Deb K (1999) Multiobjective genetic algorithms: problem difficulties and construction of test problems. Evol Comput 7:205–230

Deb K (2001) Multi-objective optimization using evolutionary algorithms. Wiley, Chichester

Deb K, Agrawal S, Pratap A, Meyarivan T (2000) A fast elitist nondominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: proceedings of the parallel problem solving from nature VI(PPSN-VI), pp 849–858

Fonseca CM, Fleming PJ (1993) Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In: Proceedings of the 5th international conference on genetic algorithms

Goldberg DE (1989) Genetic algorithms in search, optimization and machine learning. Addison-Wesley Publishing Company, Reading

Holland JH (1975) Adaptation in natural and artificial systems. MIT Press, Cambridge

Horn J, Nafploitis N, Goldberg DE (1994) A niched Paretogenetic algorithm for multi-objective optimization. In: proceedings of the 1st IEEE conference on evolutionary computation

Kita H, Yabumoto Y, Mori N, Nishikawa Y (1996) Multi-objective optimization by means of the thermodynamical genetic algorithm. In: Parallel problem solving from nature — PPSN IV. Voigt H-M, Ebeling W, Rechenberg I, Schwefel H-P, (eds) Springer Berlin Heidelberg New York, pp 504–512

Knowles J, Corne D (2002) On metrics for comparing nondominated sets. Congress on Evolutionary Computation CEC, 2002 Piscataway

Knowles JD, Corne DW (2000) Approximating the nondominated front using the Pareto archived evolution strategy. Evol Comput 8:149–172

Knowles J, Thiele L, Zitzler E (2006) A tutorial on the performance assessment of stochastic multiobjective optimizers no. 214. Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich

Kursawe F (1991) A variant of evolution strategies for vector optimization. Lecture Notes in Computer Science. In: Schwefel HP, Männer R, (eds) Springer Berlin Heidelberg New York, vol 496. pp 193–197

Li WH (1993) Molecular evolution. Sinauer, Sunderland, MA

Nei M (1986) Molecular evolutionary genetics. Columbia University Press, New York

Schott JR (1995) Fault tolerant design using single and multicriteria genetic algorithm optimization. M.S. Thesis, Dept. Aeronautics and Astronautics, MIT, Cambridge

Srinivas N, Deb K (1994) Multi-objective function optimization using non-dominated sorting genetic algorithms. Evol Comput J 3:221–248

Tan KC, Lee TH, Khor EF (2001) Evolutionary algorithm with dynamic population size and local exploration for multiobjective optimization. IEEE Trans Evol Comput 5:565–586

Van Veldhuizen DA, Lamont GB (1998) Multiobjective evolutionary algorithm research: A history and analysis. Dept. Elec. Comput. Eng., Graduate School of Eng., Air Force Inst. Technol., Tech. Rep-98–03

Van Veldhuizen DA, Lamont GB (2000) On measuring multiobjective evolutionary algorithm performance. In: Congress on evolutionary computation (CEC 2000), vol 1, pp 204–211

Zitzler E, Thiele L (1998) Multiobjective optimization using evolutionary algorithms-A comparitive study. Parallel Probl Solv Nat 5:292–301

Zitzler E, Thiele L (1999) Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE Trans Evol Comput 3:257–271

Zitzler E, Deb K, Thiele L (2000a) Multiobjective genetic algorithms: problem difficulties and construction of test problems. Evol Comput 7:205–230

Zitzler E, Deb K, Thiele L (2000b) Comparison of multiobjective evolutionary algorithms: empirical results. Evol Comput 8:173–195

Zitzler E, Thiele L, Laumanns M, Fonseca CM, Fonseca VG (2003) Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans Evol Comput 7:117–132