# Parameters control in GAs for dynamic optimization

**Khalid Jebari [1] *, Abdelaziz Bouroumi [1,2], Aziz Ettouhami [1]**

[1] *Laboratoire Conception et Systèmes Microélectronique et Informatique,*
*Faculty of Sciences, Mohammed V-Agdal*
*University UM5A, Rabat Morocco.*
*E-mail: khalid.jebari@gmail.com; touhami@fsr.ac.ma*

[2] *Modeling and Instrumentation Laboratory, Ben Msik Faculty of Sciences*
*Hassan II Mohammedia-Casablanca University, UH2MC*
*Casablanca, Morocco.*
*E-mail: a.bouroumi@gmail.com*

Received 1 February 2012
Accepted 20 August 2012

## Abstract

The Control of Genetic Algorithms parameters allows to optimize the search process and improves the performance of the algorithm. Moreover it releases the user to dive into a game process of trial and failure to find the optimal parameters.

Yet the control of parameters has received much attention in the case of static optimization problems, its investigation in the case of dynamic optimization problems (DOPs) is certainly a promising area of search. Indeed, in the case of DOPs the problem is not just to find the optima but to track the moving optima over time, so the parameters must be adapted to this dynamic environment.

The proposed algorithm Parameters Control for Dynamic Optimization (PCDO) is based on Genetic Algorithm with Fitness Sharing (GAFS). To solve DOPs by controlling GAs parameters, PCDO uses several strategies. First, an unsupervised fuzzy clustering method is used to track multiple optimums and to perform GAFS. Second, a modified enthusiasm selection is used to adjust the selection pressure. Third, a clustering multi non uniform Mutation is utilized to locate an unexplored search space. Fourth, a novel technique with multiple crossover is applied to guide the algorithm in promising regions of the search space. Fifth, a self adaptive mutation rate is evolved through generation with a learned parameter, in order to control the diversity of the population.

In the concern of maintaining the diversity of the population, a new genetic operator called Fertilization is proposed. PCDO is tested on six problems generated from Generalized Dynamic Benchmark Generator (GDBG). Experimental results demonstrate that PCDO outperforms other GAs on DOPs. Moreover, the ability of PCDO to maintain diversity is demonstrated by a new diversity measure.

*Keywords:* Genetic Algorithms, Unsupervised Learning, Fuzzy Clustering, Dynamic Optimization, Evolutionary Algorithms, Fitness Sharing, Multi-modal function optimization, Generalized dynamic benchmark generator (GDBG).

*Corresponding Author: khalid.jebari@gmail.com

Published by Atlantis Press

## 1. Introduction

In the real world, many optimization problems are time-varying [1,2]. In fact, the challenge of these optimization problems consists in locating and tracking the moving optima over time.

Several researchers have focused on solving DOPs. GAs are the most used in DOPs because GAs are inspired from the Darwinian evolution, which is confronted with a dynamic environment in nature. However, when solving DOPs, Standard GAs suffer from a major problem which is the convergence. Consequently, they can not be adapted to the new environment when change occurs and they progressively loose diversity through generations.

To address this problem of diversity [4], several approaches based on GAs have been developed such as increasing mutation rate [10], using the immigrants approaches [17,9,28,22], using explicit or implicit memory [29,5,24], spreading out the population by multiple sub-populations approaches [6,27], promoting diversity by an artificial objective function [31], and improving GAs by hybridization strategies [8,32].

To maintain diversity in population, GAs parameters must be taken into consideration and adapted dynamically. This issue of adapting parameters is an important area of search in GAs, especially, static problems that use adaptive tuning parameters have received considerable attention [33]. Indeed, mutation and crossover rates have been investigated [34,35,36] and fuzzy logic controller that sets the parameters has been tested [37]. Nevertheless, adjusting parameters would probably turn out to be most beneficial in connection with DOPs. But, it is not sufficient to adapt one parameter like the hyper-mutation factor, or to increase an exploration parameter for solving DOPs. Rather, we should maintain a balance between the exploration and exploitation parameters.

In PCDO, the mechanism of diversity is based on the concept of fitness sharing and this mechanism is enhanced by adapting dynamically GAs parameters. In order to maintain a balance between exploration and exploitation parameters, the system architecture of PCDO is built on two components. The first component is a GA with Fitness Sharing (GAFS) enhanced by four remedies, the second one is an unsupervised fuzzy clustering algorithm.

For the purpose of enhancing the mechanism of diversity, some parameters of the first component are dynamically adapted. First, a new genetic operator called Fertilization is presented. The principle of this operator is to add new individuals in the current population. These new individuals are the prototypes ($V_i$) given by the second component. Then, we apply mutation and restricted crossover for only those prototypes. This new sub-population is added to the current population with elitism replacement. Second, a Modified Enthusiasm Selection [16], [18] (MES), is used to dynamically adjust the selection pressure. Third, a new mutation operator named Clustering Multi Non Uniform Mutation (CMNUM) with a self adaptive mutation rate is used to find a new peak in new search space. Fourth, a new crossover based on multiple-crossover (CMC) between parents, prototypes ($V_i$), and best individuals is used to find dynamically the regions that have a high probability of containing good solutions. This crossover method is able to explore other areas of the search space.

To form the niches, the second component is used in order to identify clusters that correspond to niches. The number of clusters (C), as well as the characteristics of each detected cluster ($C_i$), are automatically provided by this unsupervised fuzzy clustering without requiring any prior knowledge on the distribution of population. The characteristics (C, the prototypes $V_i$, the cluster radius $r_i$) are used to overcome the weakness of fitness sharing.

This paper is presented in the following manner. First, we review the Clustering Algorithms used in our approach. Second, we provide a description for Fitness Sharing. Third, we present our proposed method, followed by experimental results and an analysis of these results. Finally, we conclude with a discussion and directions for future research in PCDO.

## 2. Clustering Algorithm

In GAs, individuals are usually created at random manner. Accordingly, information on the distribution of individuals may, therefore, be missing and clusters may overlap. The benefit of using fuzzy clustering consists in the fact that it efficiently deals with overlapping clusters, and does not require prior knowledge on individuals distribution. Drawing on an inter-points similarity measure, this clustering algorithm is founded in two major steps. The first step is an Unsupervised Fuzzy Learning (UFL) [7] procedure that explores the individuals of population P(t) for discovering the clusters. Besides, the number of clusters (C), UFL provides the initial prototypes ($V_i$). The second step is an optimization procedure that applies Fuzzy C-Means (FCM)[3] algorithm with the aim to optimize the C learned prototypes.

UFL starts with generating the first cluster around the first encountered individual. The other individuals are then sequentially explored. A new cluster is automatically created whenever the current individual presents a similarity less than a prefixed threshold $S_{min}$, to the already existing prototypes.

The similarity measure controls how the clusters are formed. In this study, the similarity measure is based on Euclidean distance. To measure the similarity between any pair of p-dimensional vectors (individuals) $I_i$ and $I_j$ ($I_i, I_j \in \Re^p$), the following expression is used:

$$S(i,j) = 1 - \frac{d(I_i, I_j)}{\sqrt{p}} \qquad (1)$$

Where d($I_i$ , $I_j$) is the Euclidean distance measure, calculated on the basis of the normalized values of $I_i$ and $I_j$.

In equation(1), if $I_i$ is replaced by a cluster prototype $V_j$, S(i,j) can be interpreted as the membership degree $\mu_{i,j}$ of $I_j$ to the $j^{th}$ cluster.

$$\mu_{i,j} = 1 - \frac{d(I_i, V_j)}{\sqrt{p}} \qquad (2)$$

The equation(2) is used in each iteration of the learning process for assessing the membership degree of the current individual to all the C existing clusters (C being variable). Hence, the condition to create a new cluster is formulated as follows:

$$\max_{1 \leqslant j \leqslant C} (\mu_{i,j}) < S_{min}$$

If this condition is not verified, the prototype of each detected cluster should be updated according to the following equation:

$$V_j = \frac{\sum_{k=1}^{i} \mu_{k,j} I_j}{\sum_{k=1}^{i} \mu_{k,j}} ; 1 \leqslant j \leqslant C \qquad (3)$$

Where i denotes the current individual index.

Generally, in the absence of any prior information to the selection of good value for $S_{min}$, the algorithm is automatically iterated for different values of $S_{min}$, belonging to an interval [$\xi_{min}, \xi_{max}$], where:

$$\xi_{min} = \min_{i \neq k} (S(I_i, I_k)) \qquad (4)$$

$$\xi_{max} = \max_{i \neq k} (S(I_i, I_k)) \qquad (5)$$

However, this clustering algorithm is sensitive to the choice of the similarity threshold value $S_{min}$, in our clustering algorithm $S_{min}$ is varied within the range $\xi_{min}, \xi_{max}$ with a step equal to 0.01 (step=0.01). Consequently, the different values of $S_{min}$ may lead to different results. So, a criterion validity is used in the third phase. This validation (VAL) has used the normalized partition entropy [3] which is defined as follows :

$$h(U) = -\frac{1}{\log(C)} \frac{1}{N} \sum_{i=0}^{N} \sum_{j=0}^{C} [u_{ij} \log(u_{ij})] \qquad (6)$$

Where U=[$\mu_{i,j}$] matrix of membership degrees; C:Number of clusters; N:Population Size.

The best solution $U^*$ is the one that minimizes h(), in this case the number of clusters $C^*$ given by UFL() is the optimum found and also the prototype $V^*$ given by FCM(). The proposed clustering algorithm UFL-FCM-VAL is described as follows:

```
Data: Individuals: $(I_1, I_2, .....I_N)$
Result: Prototypes $(V_1^*, V_2^*, ..., V_C^*)$; $C^*$:
        Number of Clusters
Initialization;
$S_{min} \leftarrow \xi_{min}$ ; $h_{min} \leftarrow 1$ ; $C^* \leftarrow 2$;
while $S_{min} < \xi_{max}$ do
    Apply UFL();
    Apply FCM();
    if $(h_{min} < h())$ then
        $h_{min} \leftarrow h()$;
        $C^* \leftarrow C$;
        $V^* \leftarrow V$;
        $U^* \leftarrow U$;
    end
    $S_{min} \leftarrow S_{min} + step$;
end
```

**Algorithm 1:** Clustering Algorithm: UFL-FCM-VAL()

The solution $U^*$ is then defuzzified in order to assign each individual to its natural cluster. The decision rule used for this operation consists in assigning each individual to the cluster for which it presents the maximum membership degree. The result is a final hard C-partition with C prototypes $V_1, V_2, ..., V_C$. These prototypes represent the tracking optima and also the individuals for fertilization operator.

## 3. Fitness Sharing

The mechanism of niching methods are based on the principle of promoting the formation of artificial niches during the evolutionary process. Niching methods try to preserve a diversity within the population to avoid premature convergence of the algorithm. This paper is focused more specifically to fitness sharing [30] which combined with the method (UFL-FCM-VAL), presented previously, to obtain a more efficient hybrid model.

Fitness sharing is used for preserving and enhancing variation within population. The fitness sharing maintains genetic diversity in a population by encouraging the niche concept and permit the investigation of many optima in the area search space. This concept is built by penalizing the individuals in densely populated regions. In this fact, the individual fitness is divided by its niche count. The shared fitness value f$_{sh}$ for an individual is as follows:

$$f_{sh,i} = \frac{f_i}{\sum_{j=1}^{N} sh(d(i,j))} \tag{7}$$

where f$_i$ is the Fitness value, N is the population size and d(i, j) is a distance measure between the individuals i and j. The sharing function (sh) measures the similarity between two individuals:

$$sh(d(i,j)) = \begin{cases} 1 - \left(\frac{d(i,j)}{\sigma_s}\right)^{\alpha} & \text{if d(i,j)} < \sigma_s \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

Where $\sigma_s$ denotes a threshold of dissimilarity and $\alpha$ is a constant which regulates the shape of the sharing function.

The primary problems of GAFS are the number q and the niche radii are often estimated. Furthermore, making $\sigma_s$ the same for all individuals means that the peaks are considered as nearly equidistant in the domain. [26]

To overcome these limitations, our approach uses a fuzzy clustering technique in order to determine automatically the number of niches q (q=number of clusters given by UFL-FCM-VAL) . Moreover, the radius of each niche is continuously updated.

## 4. PCDO algorithm

The aim of PCDO is to determine and to track different optima of dynamic functions. Moreover, the number of peaks q and the characteristics corresponding to niches (center, radius, cardinal) are provided by the first component of PCDO. The idea is to apply UFL-FCM-VAL to population of solutions produced by GAFS, in order to detect the presence of homogeneous and well separated clusters. If the entropy of the fuzzy c-partition obtained [7] is higher than ($10^{-2}$), these solutions will again be evolved by GAFS and classified by UFC-FCM-VAL.

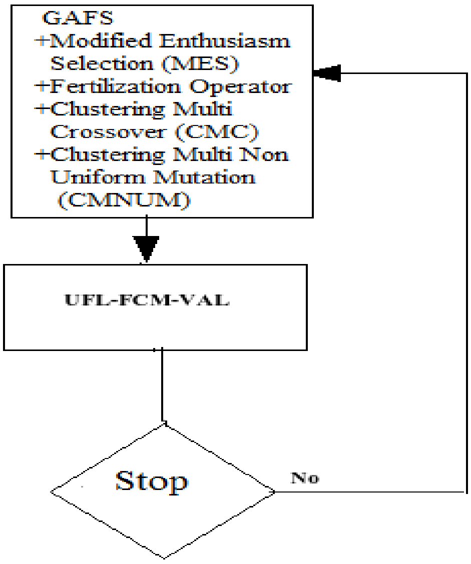
GAFS
+Modified Enthusiasm Selection (MES)
+Fertilization Operator
+Clustering Multi Crossover (CMC)
+Clustering Multi Non Uniform Mutation (CMNUM)
UFL-FCM-VAL
Stop
No

Fig. 1. PCDO Structure.

PCDO is an iterative technique (Figure 1) based on two components. The first component (GAFS) is a GA which combines fitness sharing, in order to maintain diversity and to encourage speciation. The second component (UFL-FCM-VAL) is based on an unsupervised fuzzy clustering algorithm. UFL-FCM-VAL performs the partition of the individuals, which are provided by GAFS, into a set of C clusters. Each of the C clusters corresponds to a niche. Hence, individuals once again undergo a cyclic process throughout the two components. PCDO system is based on the following:

1. the number of clusters (C) is computed by UFL -FCM-VAL;
2. prototypes, provided by UFL-FCM-VAL, represent the individuals used in fertilization operator;
3. each cluster represents a niche;
4. The subpopulations and their appropriated subspaces are generated using the characteristics (center and radius) of each identified cluster;
5. In order to identify a non-detected niche in the previous cycle, GAFS is used again.

The different components are described in the next sections.

### 4.1. GAFS component

To solve the problem, a set of candidate solutions is randomly created. This set of solutions is called population. Then, the quality of each individual is measured by the fitness sharing and the best ones are selected as parents. The chosen individuals reproduce and undergo a variation process by means of genetic operators, such as crossover and mutation, creating a new offspring population. Finally, the next generation of the population is formed by combining parent and offspring populations. This process is repeated until a certain stop condition, e.g., number of generations is reached. The details are as follows:

The initialization of the population of individuals is generated by a random process, the population size is $N = 100$. The real encoding scheme is used in the PCDO. For the crossover operator, we have used a Clustering Multiple Crossover , with $crossover\_rate = (1 - mutation\_rate)$. For the mutation, we have considered Clustering Multi Non Uniform Mutation. For the mutation rate, we have chosen an adaptive mutation rate starting at 0.4. For selection operator, We have used a Modified Enthusiasm Selection.

GAFS component makes the computation of the following for each cluster or niche i:

- Cardinal of the niche: card (i) = total number of individuals assigned to the niche i
- Prototypes (Vi ): the mean vector of the cluster elements;

$$V_i = \frac{1}{card(i)} \sum_{j=1}^{card(i)} I_j \tag{9}$$

- radius of the niche

$$\sigma_i = max_{I \in niche}(d(I, V_i)) \tag{10}$$

$$\sigma_s = max_{1 \leqslant i \leqslant C}(\sigma i) \tag{11}$$

### 4.1.1. Fertilization Operator

After calculating and updating the prototypes by UFL-FCM-VAL, the fertilization operator allows to insert new individuals in the population. The first individuals in this sub population are the prototypes F(t), we evolved the individuals by respecting the following steps:

- Apply crossover, Let $I_f$, $I_m$ are two parents, the children are:
  $C_1=\alpha*I_f+(1-\alpha)*I_m$
  $C_2=\alpha*I_m+(1-\alpha)*I_f$; Where $\alpha \in[0,1]$
  the result is the sub population F́(t);
- apply Gaussian mutation (the result is the sub population F̈(t));
- each individual in {F(t)U F́(t)U F̈(t)} replaces the nearest individual in the population P(t) if it has a higher fitness.

### 4.1.2. Modified Enthusiasm Selection

Modified Enthusiasm Selection (MES) [16] , is based on standard tournament selection. But MES provides a new opportunity for individuals who are excluded by the tournament selection. These individual are ranked according to their fitness and a percentage of enthusiasm $\lambda$ is given to these individuals in order to increase their fitness. After the enthusiasm individual has been selected, MES put it to its raw fitness. MES guards also in each iteration the best individual. The tournament size (k) is the number of clusters (C) given by UFL-FCM-VAL [7]. This tournament size controls better the selection pressure [18]. In the pseudo code of MES, TabR represents an array of integer containing, with random manner, the indices of individuals in the population; TabW represents an array of individuals indices's who will be selected by the selection method; TabS represents an array of individuals indices's, these individuals are sorted in a decreasing order of Shared Fitness ($f()$). TabT represent an array of $k-1$ indidual fitness.

**MES**() summarizes the pseudo code of the selection method.

```
Data: Array: (TabR(), TabS() )
Result: Array: (TabW())
Initialization ;
$k \leftarrow C$;
$l \leftarrow 0$;
for $i \leftarrow 0$ to $k$ do
    Shuffle TabR() ;
    $j \leftarrow 0$;
    while $j < N$ do
        $C1 \leftarrow TabR(j)$ ;
        for $m \leftarrow 1$ to $k$ do
            $C2 \leftarrow TabR(j+m)$;
            if f(C1)< f(C2) then
                $C1 \leftarrow C2$
            end
            if m> 1 then
                $aux \leftarrow m$;
                for $m \leftarrow 1$ to $k$ do
                    $TabT(m) \leftarrow f(I_{j+m})$;
                    f($I_{j+m}$)=$\lambda$ f(($I_{j+m}$);
                end
                $m \leftarrow aux$;
            end
        end
        for $m \leftarrow 1$ to $k$ do
            $f(I_{j+m}) \leftarrow TabT(m)$;
        end
        $j \leftarrow j+k+1$;
        $TabW(l) \leftarrow C1$;
        $TabW(l+1) \leftarrow TabS(l)$ ;
        $l \leftarrow l+2$ ;
    end
end
```

**Algorithm 2:** Modified Enthusiasm Selection MES()

### 4.1.3. Clustering Multi Non Uniform Mutation

The aim of the mutation operator is to find an unexplored search space. However, the classical mutation operator cannot satisfy this requirement in an effective manner. If the offspring fitness is less than the parent fitness, it is difficult to be retained. Even if this offspring is located in an unexplored search space. So, this unexplored search space cannot be considered by this mutation operation.

To ensure the population diversity, the Clustering Multi Non Uniform Mutation (CMNUM) [14] is constructed in the purpose of keeping the offspring which is located in the unexplored search space. For that, CMNUM has inserted the offspring whose fitness function is less than those of the parents but only if the offspring do not belong to already existing clusters.

The pseudo code CMNUM() summarizes the mutation operator:

```
Data: (Individual: Parent)
Result: (New Individual: Child)
// Let Class1={C1,C2,...,C_c};
// Class1 set of clusters given by
UFL-FCM-VAL ;
Child is created conforming to equation 12;
if (Child ∈ Class1) then
    if (f(child)<f(Parent)) then
        Child ← Parent
    end
else
    P(t)=P(t)∪{Child}
end
```

**Algorithm 3:** Clustering Multi Non Uniform Mutation, CMNUM()

$$Child = \begin{cases} Parent + (X^{UB} - Pareent).A & \text{if r} < 0.5 \\ Parent + (Parent - X^{LB}).A & \text{if r} \geqslant 0.5 \end{cases} \tag{12}$$

Where

$$A = \langle r_1 (1 - \frac{c}{c_{max}}) \rangle^b \tag{13}$$

where $r$ and $r_1$ are uniform random numbers in $(0,1)$, and $b$ is the shape parameter [14]. $X^{UB}$ is the upper bound and $X^{LB}$ is the lower bound. $C$ is the number of clusters given by UFLFCM-VAL and $C_{max}$ is equal to Maximum number of clusters.

### 4.1.4. Self Adaptive Mutation rate

The most technique used to adapt the mutation rate is the self adaptive method where the value of $p_m$ is modified by evolving it. The idea is that good parameter value will provide an evolutionary advantage to individual it belongs to and therefore it will itself breed in the population. During the dynamic optimization process, clustering algorithms can extract information about the diversity of the population. If the number of clusters is high, we can conclude that the population is diversified. Otherwise, we must diversify the population. That is why, the mutation rate formula used depends on the number of clusters.

$$p_m(t+1) = (1 + \frac{1 - p_m(t)}{p_m(t)} exp(-\gamma N(0,1)))^{-1} \tag{14}$$

Such the mutation rate is distributed according to a logistic normal distribution;

The learning rate $\gamma$ controls the size of adaptation steps.

$$\gamma = (\lambda \frac{C}{C_{max}} + (1-\lambda) |\frac{(O(t) - B(t))}{max(O(t), B(t)}|) \tag{15}$$

$\lambda$ depends the nature of the problem, C is the number of clusters given by UFL-FCM-VAL, B$_t$ is the best solution obtained by an algorithm just before the environmental change, O$_t$ is the optimum value of the environment at time t.

If C $<$ 5, the cluster number is low then the population should be diversified, in this case the mutation rate increases. But if the number of cluster is hight and $|\frac{(O(t)-B(t))}{max(O(t),B(t)}|$ is low, i.e., the performance begins to decline in this case the mutation rate increases.

### 4.1.5. Clustering Multiple Crossover

Crossover plays an important role in the implementation of robust adaptive GAs. Clustering Multiple Crossover (CMC) operates in three ways. First, a random pair of parents (x,y) produces two new offsprings (K11,K12):

$$K11 = 0.5[(1+\beta_q)x + (1-\beta_q)y] \tag{16}$$

$$K12 = 0.5[(1-\beta_q)x + (1+\beta_q)y] \tag{17}$$

if u< 0.5 then

$$\beta_q = exp(-\alpha u) \tag{18}$$

else

$$\beta_q = exp(\alpha(u-1)) \tag{19}$$

Where u a random number in (0,1)

$$\alpha = \frac{C}{C_{max}} \tag{20}$$

Where C the number of clusters given by UFL-FCM-VAL and C$_{max}$ =maximum number of clusters, in this study we choose C$_{max}$=70.
Second, a set S(t)= {T% of the best individuals in the population} gives the first parent P1. The second parent P2 is chosen randomly in the population, but P1 and P2 belong to different clusters. This pair of parents (P1,P2) produces (K21,K22):

$$K21 = \alpha * P1 + (1-\alpha) * P2 \tag{21}$$

$$K22 = \alpha * P2 + (1-\alpha) * P1 \tag{22}$$

Third, a set Prot(t) of the prototypes given by UFL-FCM-VAL gives the first parent P1. The second parent P2 is chosen randomly in the population, but P1 is not the prototype of the cluster that belongs P2. This pair of parents (P1,P2) produces (K31,K32) according to equations 21 and 22. We choose the best pair of CHIL1={K11,K12,K21,K22,K31,K33}. We insert also in the population the elements of CHIL1 which present a small similarity, less than $S_{min}$, to the entire already existing prototypes. Our method, Clustering multiple crossover, allows to exploit the search space and also to explore other areas of the search space.

### *4.2. UFL-FCM-VAL component*

Once UFL-FCM-VAL() is applied, a defuzzification procedure is performed in order to affect definitely each individual to its natural cluster for which it presents the maximum membership degree. This results in a final hard C-partition with C cluster centers V$_1$; V$_2$; . . . ; V$_c$, which allow to track the expected optima. In addition to the cluster prototypes, the algorithm also provides some interesting characteristics:

- the number of individuals assigned to the cluster;
- the prototype V$_i$ and the radius of the cluster r$_i$;
- the maximum and minimum of inter points similarities within the cluster, $\xi_{max}$ and $\xi_{min}$.

## 5. Experimental results

In this section, four groups of experiments were carried out based on the GDBG [20]. The objective of the first group of experiments is to investigate the work mechanism of PCDO, analyze the advantage of using different remedies, and study the diversity of PCDO. In the second group of experiments, the performance of PCDO is compared with a number of well known genetic algorithms taken from the literature. The aim of the third group is to study the mechanism of diversity with a new diversity measure. To understand the sensitivity of the parameter specifications on the performance of PCDO, the last experiment examines PCDOs with some parameters turn off.
PCDO is tested on six problems generated by the benchmark proposed by Li et al [20]. There are seven change types of the system control parameters in the benchmark test. They are small step change, large step change, random change, chaotic change, recurrent change, recurrent change with noise and dimensional change. The seven change types of GDBG are described as follows:

- Small step

$$\Delta\phi = \alpha.\left\|\phi\right\|.r.\phi_{severity} \tag{23}$$

- Large step

$$\Delta\phi = \left\|\phi\right\|.\left(\alpha.sign\left(r\right) + r.\left(\alpha_{max} - \alpha\right)\right).\phi_{severity} \tag{24}$$

- Random

$$\Delta\phi = N\left(0,1\right).\phi_{severity} \tag{25}$$

- Chaotic

$$\phi\left(t+1\right) = A.\left(\phi\left(t\right) - \phi_{min}\right).\frac{\left(1 - \left(\phi\left(t\right) - \phi_{min}\right)\right)}{\left\|\phi\right\|} \tag{26}$$

- Recurrent

$$\phi\left(t+1\right) = \phi_{min} + \left\|\phi\right\|.\frac{\left(\sin\left(\frac{2\pi}{P}t + \varphi\right) + 1\right)}{2} \tag{27}$$

- Recurrent with noisy

$$\phi\left(t+1\right) = \phi_{min} + \left\|\phi\right\|.\frac{\left(\sin\left(\frac{2\pi}{P}t + \varphi\right) + 1\right)}{2} + \lambda \tag{28}$$

where $\lambda = N(0,1).noisy_{severity}$

- Dimensional

$$D(t+1) = D(t) + sign.\Delta D \qquad (29)$$

where $\|\phi\|$ is the change range of $\phi$, $\phi_{severity}$ is a constant number that indicates change severity of $\phi$, $\phi_{min}$ is the minimum value of $\phi$, $noisy_{severity} \in \lceil 0,1 \rceil$ is noisy severity. $\alpha \in \lceil 0,1 \rceil$ and $\alpha_{max} \in (0,1)$ are constant values, which are set to 0.04 and 0.1 in the GDBG system. $\varphi$ is the initial phase, r is a random number in (-1, 1), sign(x) returns 1 when x is greater than 0, returns -1 when x is less than 0, otherwise, returns 0. ΔD is equal to 1.

$$D(f) = Max\_D \Longrightarrow sign = -1 \qquad (30)$$

$$D(f) = Min\_D \Longrightarrow sign = 1 \qquad (31)$$

Where Max_D and Min_D are the maximum and minimum number of dimensions. The six test problems defined in [21] are: F1 Rotation peak function, F2 Composition of Spheres function, F3 Composition of Rastrigins function, F4 Composition of Griewanks function, F5 Composition of Ackleys function and F6 Hybrid Composition function. The parameters of the six problems are the same chosen in [21].

Our first experiment is to investigate the PCDOs mechanism of diversity and analyze the fertilization operator in association with the parameters control of GAs. The second experiment is devoted to the performance of PCDO compared with SGA and GAFS. In this experiments, Standard Genetic Algorithm (SGA) parameters are as follows: simulated binary crossover [12], with crossover rate=0.8. For the mutation, we considered the polynomial mutation [11], with mutation rate=0.2. We have used a Tournament Selection with size=4. The population size is N =100 For PCDO and SGA on a DOPs, 30 independent runs were executed with the same set of random seeds. For each run, 50 environmental changes were allowed and the best-of-generation fitness was recorded every generation. The best-of-generation fitness averaged over 30 runs and formulated below:

$$\overline{F_{BOG}} = \frac{1}{G}\sum_{i=1}^{G}\left(\frac{1}{R}\sum_{j=1}^{R} F_{BOG_{i,j}}\right) \qquad (32)$$

where G = 50, R = 30 is the total number of runs, and $F_{BOG_{i,j}}$ is the best-of-generation fitness of generation i of run j. The best-of-generation fitness against generation on the DOPs for PCDO is shown in Figure 2.

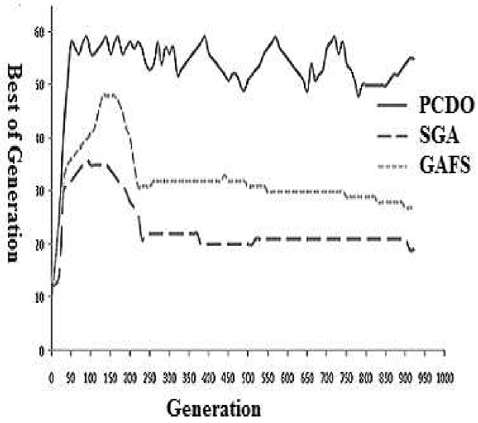

Fig. 2. Dynamic performance of PCDO compared to SGA and GAFS

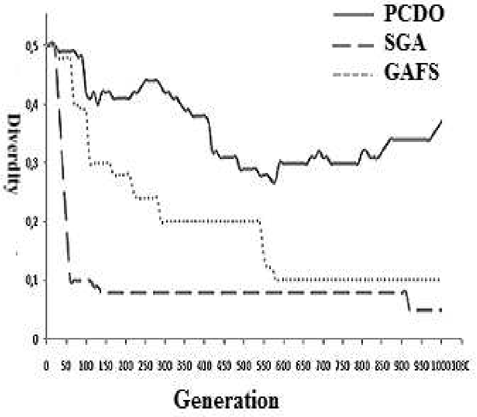

Fig. 3. Diversity Comparison

PCDO significantly outperforms SGA, PCDO give better results than GAFS. This result validates the benefit of introducing Fertilization into GAFS and the most significant role of adapting mutation and using new crossover, mutation operators for DOPs. In order to understand the effect of fertilization scheme and parameters control influence on the population diversity, we recorded the diversity of the population every generation for each run of a PCDO on a DOPs. The mean population diversity of a PCDO on a DOPs at generation over 20 runs is calculated according to the formula:

$$\overline{Div(t)} = \frac{1}{30}\sum_{k=1}^{30}\left(\frac{1}{ln(n-1)}\sum_{i=1}^{n}\sum_{j\neq i}^{n} d_{i,j}(k,t)\right) \quad (33)$$

$D_{i,j}(k,t)$ Euclidean distance between the i-th and j-th individuals at generation t of the k-th run. The diversity dynamics over generation for SGA and PCDO on DOPs is shown in Figure 3. It can be seen that PCDO does maintain the highest diversity level in the population while SGA maintains the lowest diversity level. This interesting result shows that the approach that aims at maintaining a high diversity level in the population in dynamic environments is the one with fertilization operator and parameters control.

In the second experiment we used GDBG system, the parameters of the test problems are set as the same as in [21]. PCDO is compared with SGA, Random Immigrants Genetic Algorithms (RIGA) [28] with the total number of immigrants N$_i$=30, and also with Hyper-mutation Genetic Algorithm (HMGA)[10]. For this second experiment, SGA, HMGA, RIGA parameters are as follows: simulated binary crossover [12] , with crossover rate=0.85. For the mutation, we considered the Gaussian mutation, with mutation rate=0.15. We have used a Tournament Selection with size=4. The population size is N =100.

All tests (table 1, table 2, table 3, table 4, table 5, table 6) present the results of the average best, average mean, average worst values and the standard error (STD). The results allow us to compare the quality of the solution founded by each technique. Table 1 shows that the performance of the four methods seems to be almost similar with a marked improvement in PCDO, because F1 is the simplest one to optimize. But when the number of peaks is equal to 50, PCDO outperforms the other techniques. Table 2, presents the results for a more complicated function, which has multiple optima and with a dimension =10. The SGA technique loses diversity due to the convergence of the algorithm. RIGA loses the quality of the solution due to a random individuals included in the population, and diversity is much lower compared to PCDO. Table 3 shows that for a function F3 which has multiple optima, the quality of the solution to PCDO is much better, but what is interesting in this table is the diversity of the population in the case of PCDO is much higher compared other techniques. Table 4 shows the performance of each method for a complicated multi-modal function with a huge number of local optima. PCDO uses the fitness sharing performed by UFL-FCM-VAL, then PCDO assigned each individual in the population to its niche by SS. So, PCDO does not fall into the trap of local optima, it is able to find the global optimum. Therefore, other techniques (SGA, HMGA, RIGA) have a problem of finding the global optimum and are diving into the local optima, so the error values are high. The results of function f5 are presented in table 5, the performance of PCDO is much better compared to SGA, HMGA and also RIGA. Table 6 presents the results of a composition of complicated functions, especially the Weierstrass's function, which is continuous at all points in research space, but is not differentiable at any point. In this complicated function, PCDO presents the best results compared to the other techniques By observing the results in table 7, it can be seen that the challenge of different change types is quite different. Looking through the difficulty of the problems starting from the simplest,the small step, and through the most complicated test to optimize, SGA, RIGA and RIGA have difficulties to optimize the problems, especially chaotic change and dimensional change. But, in all tests, PCDO performs much better than SGA, RIGA and HMGA.

Figure 4 provides sample results related to another aspect of this study. It is the aspect of diversity of SGA, RIGA, HMGA and PCDO. To measure the concept of diversity of the population, we adopt a criterion measure often used in genetic studies namely the Hill's diversity index. It is a measure of abundance proportional to associate the indices of Shannon-Weaver and Simpson. The Hill's diversity index provides a more precise view of the diversity observed. This index varies between 0 and 1. It is maximal when individuals belong to several clusters, i.e., the existence of several clusters.

It is minimal when one cluster dominates the population. Its formula is as follows:

$$H' = 1 - Hill = 1 - \frac{(\frac{1}{\lambda})}{exp(H)} \tag{34}$$

$$\lambda = \sum_{i=1}^{C} \frac{N_i(N_i - 1)}{N(N-1)} \tag{35}$$

$$H = -\sum_{i=1}^{C} ((\frac{N_i}{N}) * Log_2(\frac{N_i}{N})) \tag{36}$$

Where $\lambda$ is Simpson's index;
H is Shannon-Weaver's index;
C: number of clusters given by UFL-FCM-VAL;
N$_i$ : Cardinal of each cluster;
N: Population Size.

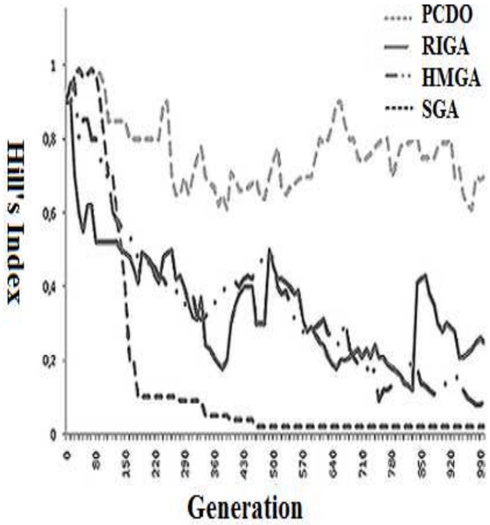

Fig. 4. Hill's Diversity index

The diversity curve of PCDO presents a high level of diversity and this diversity is continuous for several generations. This PCDO's curve shows no drop in values. The maintenance of diversity is due to the control by PCDO in the process of evolution. PCDO controls several parameters such as the selection pressure by Modified Tournament Selection. The mutation is controlled by Clustering Multi Non Uniform Mutation and an adaptive mutation rate. The crossover is improved by Clustering multiple crossover. Fitness sharing is performed by UFL-FCM-VAL. While SGA loses diversity of population easily. RIGA has a low diversity compared to PCDO. But what is striking in this study is the random variation of the diversity of RIGA in the process of evolution. HMGA lost the diversity, this is due to such problems as how to adapt the mutation rate over time and how to choose the hyper-mutation factor. PCDO adapt this factor by clustering, and this factor depends on the number of clusters.

The last experiment is devoted to understanding the sensitivity of the parameter specifications on the performance of PCDO and also the importance of using adaptive exploitation parameters and adaptive exploration parameters together, as presented in PCDO. To claim the effect of PCDO components, PCDO is examined with some functionality turned off. The first case is devoted to PCDO. In the second case, we used GAFS with only the new mutation operator CMNUM (GAFS+CMNUM). In the third case, GAFS is used with the selection operator MES (GAFS+MES). In the fourth case, the new crossover operator CMC represented the functionality turned on (GAFS+CMC). The fifth case, GAFS is used with all functionality turned off (GAFS).

Based on the Hills diversity index, we have measured the concept of diversity of the population for the five cases. Figure 5 shows clearly that PCDOs, in terms of diversity, outperforms all other cases. GAFS with four remedies, as proposed in PCDO, maintains efficiently a balance between exploitation and exploration.

Table 1. Error Values Achieved for Problems $F_1$.

| Technique used | Peaks(m) | Errors | T1 | T2 | T3 | T4 | T5 | T6 | T7 |
|---|---|---|---|---|---|---|---|---|---|
| SGA | 10 | Avg_best | $0.03e^{-7}$ | $1.34e^{-7}$ | $4.13e^{-5}$ | $8.7e^{-6}$ | $0.65e^{-5}$ | $6.5e^{-6}$ | $1.24e^{-5}$ |
| | | Avg_worst | 30.53 | 42.38 | 64.16 | 69.74 | 32.15 | 54.84 | 37.05 |
| | | Avg_mean | 10.45 | 14.19 | 12.23 | 23.39 | 7.69 | 30.81 | 8.05 |
| | | STD | 5.11 | 8.01 | 10.54 | 21.41 | 5.23 | 21.46 | 5.73 |
| | 50 | Avg_best | $8.74e^{-5}$ | $4.12e^{-4}$ | $6.33e^{-4}$ | $5.32e^{-4}$ | $4.11e^{-4}$ | $5.35e^{-5}$ | $2.21e^{-4}$ |
| | | Avg_worst | 41.35 | 58.53 | 61.21 | 56.84 | 44.62 | 54.61 | 23.62 |
| | | Avg_mean | 17.61 | 20.46 | 14.09 | 27.18 | 9.78 | 28.91 | 14.78 |
| | | STD | 9.28 | 10.38 | 13.41 | 21.08 | 4.38 | 18.38 | 4.46 |
| HMGA | 10 | Avg_best | 0 | 0 | $8.71e^{-7}$ | 0 | $5.66e^{-5}$ | 0 | $2.31e^{-6}$ |
| | | Avg_worst | 10.24 | 21.63 | 40.62 | 15.78 | 19.39 | 14.98 | 21.03 |
| | | Avg_mean | 5.04 | 9.89 | 26.35 | 8.16 | 7.79 | 8.76 | 9.18 |
| | | STD | 1.12 | 4.23 | 17.12 | 2.75 | 4.66 | 2.15 | 4.02 |
| | 50 | Avg_best | $1.34e^{-6}$ | $1.52e^{-5}$ | $3.41e^{-6}$ | $6.23e^{-5}$ | $1.57e^{-5}$ | $5.43e^{-6}$ | $2.45e^{-5}$ |
| | | Avg_worst | 13.12 | 15.13 | 19.31 | 19.17 | 20.24 | 15.76 | 14.83 |
| | | Avg_mean | 5.92 | 6.27 | 7.38 | 10.07 | 10.43 | 8.97 | 9.49 |
| | | STD | 3.46 | 5.42 | 4.57 | 3.28 | 6.84 | 3.37 | 4.03 |
| RIGA | 10 | Avg_best | 0 | $1.52e^{-7}$ | $6.25e^{-7}$ | $8.51e^{-7}$ | $4.33e^{-7}$ | $9.12e^{-7}$ | $4.11e^{-7}$ |
| | | Avg_worst | 1.02 | 2.75 | 3.51 | 5.16 | 5.91 | 0.18 | 17.38 |
| | | Avg_mean | 1.05 | 2.81 | 7.95 | 2.26 | 3.29 | 1.46 | 1.89 |
| | | STD | 1.42 | 6.28 | 12.02 | 1.75 | 4.66 | 2.73 | 4.03 |
| | 50 | Avg_best | 0 | $2.52e^{-6}$ | $1.03e^{-5}$ | $3.51e^{-6}$ | $4.71e^{-5}$ | $4.36e^{-6}$ | $0.52e^{-5}$ |
| | | Avg_worst | 1.32 | 4.03 | 7.21 | 3.27 | 2.13 | 2.15 | 0.98 |
| | | Avg_mean | 1.62 | 4.27 | 6.37 | 8.69 | 2.64 | 3.47 | 2.49 |
| | | STD | 1.98 | 4.07 | 10.41 | 2.84 | 2.84 | 3.94 | 2.71 |
| PCDO | 10 | Avg_best | 0 | 0 | 0 | 0 | 0 | $0.43e^{-12}$ | $0.52e^{-12}$ |
| | | Avg_worst | 0.01 | 0.41 | 1.65 | 1.28 | 3.77 | 0.21 | 0.43 |
| | | Avg_mean | 0.01 | 0.06 | 1.13 | 0.06 | 0.05 | 0.06 | 0.03 |
| | | STD | 0.02 | 1.43 | 2.04 | 0.05 | 1.16 | 0.07 | 0.14 |
| | 50 | Avg_best | 0 | 0 | $4.53e^{-10}$ | 0 | 0 | 0 | $1.04e^{-10}$ |
| | | Avg_worst | 0.04 | 3.03 | 3.21 | 0.29 | 0.14 | 0.28 | 0.32 |
| | | Avg_mean | 0.02 | 0.83 | 2.12 | 0.19 | 0.05 | 0.03 | 0.24 |
| | | STD | 0.04 | 2.13 | 6.12 | 0.23 | 0.05 | 0.34 | 0.29 |

Table 2. Error Values Achieved for Problems $F_2$.

| Technique used | Errors | T1 | T2 | T3 | T4 | T5 | T6 | T7 |
|---|---|---|---|---|---|---|---|---|
| SGA | Avg_best | $0.21e^{-5}$ | $3.14e^{-5}$ | $5.83e^{-5}$ | $9.58e^{-5}$ | $0.91e^{-5}$ | $6.5e^{-5}$ | $0.94e^{-5}$ |
| | Avg_worst | 98.65 | 458.12 | 489.45 | 123.15 | 398.54 | 123.15 | 178.54 |
| | Avg_mean | 34.17 | 83.91 | 128.52 | 32.54 | 94.28 | 123.15 | 278.54 |
| | STD | 0.48 | 4.12 | 7.12 | 1.02 | 3.56 | 2.78 | 5.33 |
| HMGA | Avg_best | $2.04e^{-7}$ | $7.02e^{-6}$ | $4.56e^{-7}$ | $1.21e^{-7}$ | $3.57e^{-7}$ | $1.21e^{-7}$ | $10.57e^{-6}$ |
| | Avg_worst | 7.43 | 11.05 | 21.72 | 7.86 | 9.29 | 8.29 | 12.52 |
| | Avg_mean | 1.76 | 8.31 | 9.62 | 2.27 | 12.76 | 7.29 | 25.83 |
| | STD | 3.16 | 8.09 | 11.25 | 2.63 | 8.61 | 7.37 | 12.83 |
| RIGA | Avg_best | $1.68e^{-7}$ | $6.07e^{-6}$ | $4.96e^{-7}$ | $2.31e^{-7}$ | $4.67e^{-7}$ | $2.11e^{-7}$ | $11.97e^{-6}$ |
| | Avg_worst | 6.13 | 9.15 | 18.37 | 6.81 | 7.27 | 9.13 | 10.43 |
| | Avg_mean | 1.36 | 7.31 | 8.92 | 2.37 | 10.21 | 5.08 | 11.13 |
| | STD | 2.62 | 7.07 | 10.35 | 2.23 | 6.51 | 6.45 | 10.53 |
| PCDO | Avg_best | 0 | $3.27e^{-9}$ | $1.62e^{-9}$ | $4.33e^{-9}$ | $2.16e^{-8}$ | $4.63e^{-8}$ | $7.81e^{-8}$ |
| | Avg_worst | 1.21 | 4.76 | 7.23 | 2.24 | 7.34 | 6.38 | 7.31 |
| | Avg_mean | 1.03 | 4.27 | 5.01 | 1.12 | 3.53 | 1.98 | 6.12 |
| | STD | 43.68 | 91.15 | 31.12 | 58.12 | 76.59 | 23.15 | 10.54 |

Table 3. Error Values Achieved for Problems $F_3$.

| Technique used | Errors | T1 | T2 | T3 | T4 | T5 | T6 | T7 |
|---|---|---|---|---|---|---|---|---|
| SGA | Avg_best | $4.31e^{-3}$ | $8.35e^{-3}$ | $4.23e^{-3}$ | $9.85e^{-3}$ | $1.47e^{-3}$ | $6.51e^{-3}$ | $2.81e^{-3}$ |
|  | Avg_worst | 792.45 | 958.12 | 925.45 | 1123.15 | 1043.54 | 523.85 | 878.94 |
|  | Avg_mean | 141.85 | 582.51 | 554.72 | 497.23 | 597.63 | 529.93 | 692.64 |
|  | STD | 243.14 | 400 | 431.12 | 518 | 246.59 | 323.15 | 271.42 |
| HMGA | Avg_best | $14.32e^{-4}$ | $18.25e^{-4}$ | $9.17e^{-}$ | $21.47e^{-4}$ | $8.43e^{-4}$ | $16.63e^{-4}$ | $9.87e^{-4}$ |
|  | Avg_worst | 91.84 | 102 | 97.14 | 123.45 | 98.15 | 83.80 | 178.24 |
|  | Avg_mean | 41.18 | 45.89 | 154.87 | 121 | 114.17 | 115.73 | 129.04 |
|  | STD | 81.47 | 101.12 | 76.15 | 119.85 | 94.57 | 23.79 | 79.42 |
| RIGA | Avg_best | $4.31e^{-4}$ | $7.74e^{-4}$ | $2.47e^{-4}$ | $15.47e^{-4}$ | $4.23e^{-4}$ | $8.95e^{-5}$ | $6.83e^{-4}$ |
|  | Avg_worst | 73.74 | 96 | 43.14 | 98.85 | 76.45 | 63.75 | 108.02 |
|  | Avg_mean | 23.28 | 26.91 | 134.77 | 98.09 | 98.74 | 101.13 | 97.08 |
|  | STD | 41.46 | 89.82 | 56.95 | 98.87 | 64.07 | 13.69 | 59.72 |
| PCDO | Avg_best | $6.41e^{-8}$ | $3.32e^{-8}$ | $2.47e^{-8}$ | $4.38e^{-7}$ | $6.47e^{-8}$ | $6.41e^{-8}$ | $3.73e^{-8}$ |
|  | Avg_worst | 16.05 | 25.68 | 16.06 | 12.67 | 14.83 | 19.76 | 20.87 |
|  | Avg_mean | 1.74 | 20.57 | 12.32 | 8.03 | 13.63 | 6.93 | 13.87 |
|  | STD | 343.13 | 410 | 431.22 | 510 | 346.51 | 313.15 | 371.42 |

Table 4. Error Values Achieved for Problems $F_4$.

| Technique used | Errors | T1 | T2 | T3 | T4 | T5 | T6 | T7 |
|---|---|---|---|---|---|---|---|---|
| SGA | Avg_best | $2.13e^{-3}$ | $6.34e^{-4}$ | $15.12e^{-3}$ | $9.97e^{-3}$ | $11.17e^{-3}$ | $21.54e^{-3}$ | $10.714e^{-3}$ |
|  | Avg_worst | 172.5 | 248 | 431.02 | 351.62 | 558.43 | 261.27 | 454.21 |
|  | Avg_mean | 41.41 | 125.15 | 214.54 | 97.23 | 84.79 | 63.64 | 102.34 |
|  | STD | 103.14 | 200.05 | 212.02 | 148 | 156.49 | 133.74 | 352.48 |
| HMGA | Avg_best | $4.51e^{-4}$ | $8.05e^{-4}$ | $4.03e^{-3}$ | $1.72e^{-3}$ | $2.47e^{-3}$ | $19.34e^{-3}$ | $7.164e^{-3}$ |
|  | Avg_worst | 31.84 | 110 | 187.13 | 121.2 | 298.15 | 61.29 | 64.34 |
|  | Avg_mean | 41.41 | 125.15 | 214.54 | 97.23 | 84.79 | 63.64 | 102.34 |
|  | STD | 14.18 | 35.85 | 25.89 | 14.88 | 56.48 | 121.74 | 152.56 |
| RIGA | Avg_best | $1.71e^{-5}$ | $2.45e^{-5}$ | $2.13e^{-5}$ | $2.28e^{-5}$ | $0.41e^{-5}$ | $9.38e^{-5}$ | $5.34e^{-5}$ |
|  | Avg_worst | 19.54 | 95.76 | 125.18 | 121.2 | 198.12 | 81.39 | 74.24 |
|  | Avg_mean | 21.67 | 108.65 | 119.81 | 97.23 | 84.29 | 57.44 | 107.04 |
|  | STD | 10.68 | 48.25 | 35.89 | 11.86 | 77.98 | 128.24 | 102.96 |
| PCDO | Avg_best | $1.36e^{-10}$ | $0.08e^{-10}$ | $2.83e^{-10}$ | $5.04e^{-9}$ | $1.14e^{-10}$ | $1.27e^{-10}$ | $7.34e^{-10}$ |
|  | Avg_worst | 7.35 | 26.53 | 15.24 | 20.34 | 28.21 | 19.59 | 62.58 |
|  | Avg_mean | 0.17 | 22.23 | 0.17 | 16.72 | 20.25 | 14.43 | 29.87 |
|  | STD | 2.38 | 16.61 | 1.23 | 8.37 | 25.24 | 13.12 | 10.72 |

Table 5. Error Values Achieved for Problems $F_5$.

| Technique used | Errors | T1 | T2 | T3 | T4 | T5 | T6 | T7 |
|---|---|---|---|---|---|---|---|---|
| SGA | Avg_best | $6.83e^{-3}$ | $7.61e^{-3}$ | $5.71e^{-3}$ | $3.87e^{-3}$ | $8.46e^{-3}$ | $5.12e^{-4}$ | $6.05e^{-3}$ |
|  | Avg_worst | 181.53 | 197.82 | 112.15 | 120.65 | 189.84 | 112.15 | 201.24 |
|  | Avg_mean | 28.35 | 31.57 | 28.62 | 32.84 | 14.29 | 41.23 | 21.34 |
|  | STD | 43.67 | 55.75 | 61.02 | 50.32 | 78.32 | 64.25 | 41.74 |
| HMGA | Avg_best | $1.58e^{-4}$ | $3.22e^{-4}$ | $3.33e^{-4}$ | $4.85e^{-5}$ | $1.37e^{-4}$ | $2.07e^{-4}$ | $2.05e^{-5}$ |
|  | Avg_worst | 81.64 | 97.89 | 101.05 | 138.75 | 169.84 | 102.75 | 173.14 |
|  | Avg_mean | 38.25 | 31.58 | 48.52 | 12.14 | 23.39 | 37.13 | 14.54 |
|  | STD | 33.62 | 45.15 | 21.72 | 37.42 | 38.39 | 39.85 | 14.74 |
| RIGA | Avg_best | $5.58e^{-5}$ | $2.28e^{-5}$ | $7.03e^{-5}$ | $2.55e^{-6}$ | $0.35e^{-5}$ | $1.07e^{-5}$ | $1.75e^{-5}$ |
|  | Avg_worst | 71.64 | 87.82 | 71.25 | 83.75 | 69.84 | 82.95 | 73.14 |
|  | Avg_mean | 18.25 | 21.58 | 68.52 | 72.14 | 24.59 | 47.53 | 34.54 |
|  | STD | 33.67 | 45.85 | 51.72 | 30.42 | 38.39 | 19.85 | 19.74 |
| PCDO | Avg_best | 0 | 0 | 0 | 0 | 0 | $7.27e^{-11}$ | $3.15e^{-10}$ |
|  | Avg_worst | 8.03 | 2.86 | 9.76 | 8.66 | 5.67 | 6.65 | 7.75 |
|  | Avg_mean | 1.02 | 1.37 | 2.24 | 0.12 | 3.31 | 2.12 | 4.12 |
|  | STD | 2.06 | 4.12 | 3.32 | 9.08 | 4.67 | 4.43 | 10.86 |

Table 6. Error Values Achieved for Problems $F_6$.

| Technique used | Errors | T1 | T2 | T3 | T4 | T5 | T6 | T7 |
|---|---|---|---|---|---|---|---|---|
| SGA | Avg_best | $1.31e^{-2}$ | $4.44e^{-3}$ | $9.61e^{-3}$ | $1.48e^{-2}$ | $0.93e^{-3}$ | $5.04e^{-3}$ | $0.724e^{-3}$ |
| | Avg_worst | 247.6 | 558.2 | 683.6 | 723 | 308.5 | 530.1 | 698.3 |
| | Avg_mean | 39.55 | 129.61 | 98.53 | 63.74 | 160.28 | 53.45 | 68.34 |
| | STD | 63.88 | 208.3 | 101.18 | 218.42 | 106.49 | 187.95 | 134.12 |
| HMGA | Avg_best | $5.34e^{-3}$ | $2.14e^{-3}$ | $6.31e^{-3}$ | $9.78e^{-3}$ | $0.23e^{-3}$ | $1.37e^{-3}$ | $0.22e^{-3}$ |
| | Avg_worst | 87.79 | 252.5 | 304.8 | 198.6 | 475.8 | 265.7 | 424.5 |
| | Avg_mean | 51.12 | 58.71 | 80.27 | 62.83 | 103.56 | 49.56 | 65.63 |
| | STD | 10.97 | 63.77 | 83.88 | 24.23 | 160.65 | 96.76 | 75.91 |
| RIGA | Avg_best | $4.67e^{-4}$ | $0.04e^{-3}$ | $7.81e^{-3}$ | $10.08e^{-4}$ | $7.93e^{-4}$ | $0.86e^{-3}$ | $1.32e^{-3}$ |
| | Avg_worst | 37.81 | 192.6 | 214.9 | 208.6 | 365.7 | 275.2 | 364.7 |
| | Avg_mean | 48.82 | 38.61 | 79.67 | 52.89 | 93.76 | 39.96 | 58.73 |
| | STD | 10.37 | 53.67 | 81.28 | 20.03 | 130.85 | 92.06 | 71.23 |
| PCDO | Avg_best | $2.75e^{-6}$ | $8.82e^{-7}$ | $9.16e^{-6}$ | $0.12e^{-7}$ | $3.12e^{-7}$ | $0.46e^{-7}$ | $4.36e^{-7}$ |
| | Avg_worst | 22.85 | 28.65 | 39.98 | 30.73 | 70.69 | 38.56 | 53.87 |
| | Avg_mean | 10.73 | 21.67 | 31.62 | 20.36 | 56.63 | 12.63 | 53.23 |
| | STD | 6.02 | 14.56 | 13.22 | 9.43 | 31.24 | 25.76 | 43.76 |

Table 7. Algorithm Overall Performance.

| | | $F_1(10)$ | $F_1(50)$ | $F_2$ | $F_3$ | $F_4$ | $F_5$ | $F_6$ |
|---|---|---|---|---|---|---|---|---|
| SGA | T1 | 0.851 | 0.716 | 0.698 | 0.125 | 0.1993 | 0.282 | 0.299 |
| | T2 | 0.767 | 0.729 | 0.567 | 0.0761 | 0.145 | 0.345 | 0.261 |
| | T3 | 0.845 | 0.674 | 0.578 | 0.0421 | 0.174 | 0.378 | 0.221 |
| | T4 | 0.695 | 0.612 | 0.445 | 0.0328 | 0.0812 | 0.345 | 0.328 |
| | T5 | 0.591 | 0.589 | 0.432 | 0.0485 | 0.108 | 0.353 | 0.385 |
| | T6 | 0.523 | 0.558 | 0.351 | 0.0789 | 0.158 | 0.351 | 0.289 |
| | T7 | 0.458 | 0.523 | 0.256 | 0.0241 | 0.123 | 0.256 | 0.241 |
| Mark | | 0,068660 | 0,063400 | 0,077800 | 0,010067 | 0,022740 | 0,053392 | 0,046648 |
| HMGA | T1 | 0.889 | 0.788 | 0.769 | 0.242 | 0.288 | 0.469 | 0.342 |
| | T2 | 0.848 | 0.678 | 0.658 | 0.145 | 0.278 | 0.558 | 0.345 |
| | T3 | 0.746 | 0.748 | 0.512 | 0.189 | 0.248 | 0.412 | 0.489 |
| | T4 | 0.723 | 0.858 | 0.351 | 0.0859 | 0.158 | 0.451 | 0.389 |
| | T5 | 0.758 | 0.523 | 0.256 | 0.0641 | 0.223 | 0.256 | 0.241 |
| | T6 | 0.423 | 0.458 | 0.131 | 0.0489 | 0.258 | 0.351 | 0.289 |
| | T7 | 0.658 | 0.523 | 0.456 | 0.0831 | 0.163 | 0.456 | 0.391 |
| Mark | | 0,072385 | 0,066025 | 0,071544 | 0,019927 | 0,037480 | 0,067224 | 0,056536 |
| RIGA | T1 | 0.958 | 0.978 | 0.858 | 0.295 | 0.478 | 0.658 | 0.645 |
| | T2 | 0.887 | 0.845 | 0.745 | 0.161 | 0.245 | 0.645 | 0.561 |
| | T3 | 0.845 | 0.874 | 0.781 | 0.221 | 0.374 | 0.678 | 0.521 |
| | T4 | 0.795 | 0.712 | 0.645 | 0.128 | 0.312 | 0.445 | 0.328 |
| | T5 | 0.791 | 0.689 | 0.432 | 0.185 | 0.389 | 0.532 | 0.485 |
| | T6 | 0.623 | 0.658 | 0.651 | 0.189 | 0.258 | 0.451 | 0.489 |
| | T7 | 0.458 | 0.523 | 0.256 | 0.141 | 0.123 | 0.456 | 0.441 |
| Mark | | ,078065 | 0,076570 | 0,102784 | 0,025752 | 0,051312 | 0,089112 | 0,071912 |
| PCDO | T1 | 0.998 | 0.988 | 0.894 | 0.798 | 0.978 | 0.878 | 0.725 |
| | T2 | 0.987 | 0.945 | 0.854 | 0.745 | 0.975 | 0.801 | 0.689 |
| | T3 | 0.935 | 0.902 | 0.785 | 0.658 | 0.897 | 0.798 | 0.614 |
| | T4 | 0.895 | 0.879 | 0.714 | 0.548 | 0.847 | 0.712 | 0.598 |
| | T5 | 0.891 | 0.789 | 0.658 | 0.548 | 0.801 | 0.651 | 0.524 |
| | T6 | 0.812 | 0.768 | 0.578 | 0.579 | 0.689 | 0.601 | 0.475 |
| | T7 | 0.759 | 0.754 | 0.548 | 0.651 | 0.589 | 0.512 | 0.456 |
| Mark | | 0.09036 | 0.086455 | 0.116432 | 0.10344 | 0.133912 | 0.114776 | 0.094216 |

Performance: SGA : 34.27 —— HMGA: 39.11——— RIGA :49.55 ——— PCDO: 73.96

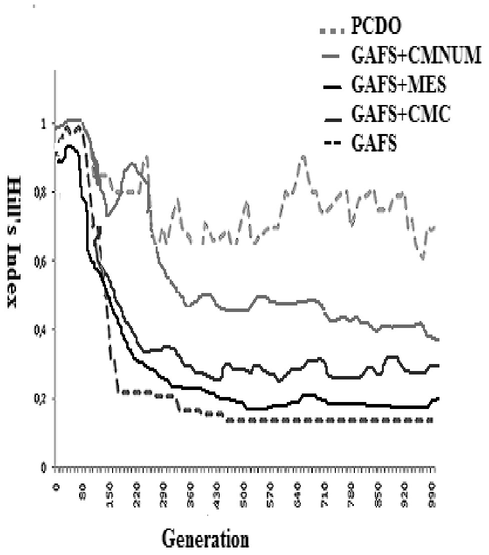

Fig. 5. PCDO examined with some functionality turned off

## 6. Conclusion

In this work, we have conceived a new approach for solving DOPs. PCDO method has taken into account several remedies for finding and tracking optima in dynamic environments. First, the fitness sharing performed by the unsupervised fuzzy clustering UFL-FCM-VAL is used to find the near optimal solutions in promising regions and to track optima. Second, the new genetic operator called Fertilization is introduced to maintain diversity. Third, the modified enthusiasm selection is used to dynamically adjust the selection pressure and to enthuse worst individuals . Fourth, the new mutation which is called Clustering Multi Non Uniform Mutation is used to find an unexplored search space by the insertion of new individuals which belong to no clusters already detected. Fifth, the new crossover named Clustering multiple crossover has exploited the best area search space and also has explored other promising regions in the search space.

The system architecture of PCDO was designed with the purpose to strike a balance between exploration parameters and exploitation parameters. Experimental results, by using a new diversity measure, have demonstrated that this aim has been achieved. In addition, PCDO has given a better result than the use of an adaptive exploration parameter or an adaptive exploitation parameters separately. Another advantage of PCDO, is that it has liberated the user from choosing himself the optimal GAs parameters. GDBG test problems have shown the efficiency of the PCDO. This performance is due to the remedies proposed in this study. The proposed clustering method is efficient to ameliorate fitness sharing; the fertilization operator maintains much better population diversity; and the adaptive parameters control used in this study maintains a dynamic balance between exploration and exploitation. Generally speaking, PCDO can effectively locate and track multiple optima in dynamic environments.

It would also be interesting to combine other techniques into PCDO to further improve its performance in dynamic environments. For example, Particle Swarm continually looks for new areas of the search space, and PCDO exploits the previously detected promising areas. Another idea is to use PCDO with Fuzzy rules to adapt crossover and mutation rates.

## References

1. Y. Jin, J. Branke, “Evolutionary optimization in uncertain environments: a survey”, *IEEE Trans Evol Comput*, **9(3)**, 303-317, (2005).
2. J. Branke, E. Salihoglu, and S. Uyar, “Towards an analysis of dynamic environments”, *Genetic and Evolutionary Computation Conference, GECCO 2005, ACM (2005)*, 1433-1440, (2005).
3. J. Bezdek, “Pattern Recognition with Fuzzy Objective Function Algorithms”, Plenum Press, New York (1981)
4. J. Branke, “Evolutionary Optimization in Dynamic Environments”, *Kluwer Academic Publishers*, Dordrecht (2002).
5. S. Yang, X. Yao, “Population-based incremental learning with associative memory for dynamic environments”, *IEEE Trans Evol Comput.* (2007).
6. J. Branke, T. Kau, L. Schmidt and H. Schmeck, “A multi-population approach to dynamic optimization problems”, *In: 4th International Conference on Adaptive Computing in Design and Manufacture (ACDM 2000)* (2000).
7. A. Bouroumi, and A. Essaïdi, “Unsupervised fuzzy learning and cluster seeking”,*Intelligent Data Analysis* **4(3)**, 241–253 (2000).

8. S. Yang, X. Yao, “Population-based incremental learning with associative memory for dynamic environments”, *IEEE Trans Evol Comput*, **12(5)**, 542-561, (2008).
9. W. Cedeno, V. R. Vemuri, “A self-organizing random immigrants genetic algorithm for dynamic optimization problems”. Genetic Programming and Evolvable Machines, 8(3):255-286 (2007).
10. H.G. Cobb, “An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environments”, *Technical Report 6760 (NLR Memorandum). Washington, D.C* (1990).
11. K. Deb , and A. Kumar, “Real-coded Genetic Algorhithms with Simulated Binary Crossover:Studies on Multimodel and Multiobjective Problems”, *Complex Systems*, **9(6):**, 431–454 (1995).
12. K. Deb, and R.B. Agrawal, “Simulated binary crossover for continuous search space”, *Complex systems*, **9(2)**, 115–148 (1995).
13. A.E. Eiben, and S. K. Smit, “Parameter tuning for configuring and analyzing evolutionary algorithms”, *Swarm and Evolutionary Computation* **1(1)**, 19–31 (2011).
14. P.H. Tanga, and M.H. Tsenga, “Adaptive directed mutation for real-coded genetic algorithms”, *Applied Soft Computing* **13(1)**, 600–614 (2013).
15. D.E. Goldberg, and Wang L, “Adaptative niching via coevolutionary sharing”, *Genetic algorithms in engineering and computer science, Wiley, Chichester*, pp 21–38 (1997).
16. A. Agrawal, and I. Mitchell, “Selection enthusiasm’, *Proceedings of the 6th international conference on Simulated Evolution And Learning, Springer-Verlag*, pp 449–456 (2006).
17. J.J. Grefenstette, “A hybrid immigrants scheme for genetic algorithms in dynamic environments”, *Int J Autom Comput*, **4(3)**, 243–254 (2007).
18. K. Jebari, A. Bouroumi, A. Ettouhami, et al, “Unsupervised fuzzy tournament selection”, *Applied Mathematical Sciences*, **5(58)**, 2863–2881 (2011).
19. L.I. Kuncheva, J.C. Bezdek, “Nearest prototype classification: clustering, genetic algorithms or random search”, *IEEE Trans. Systems Man Cybernet*, **C 28 (1)**, 160–164 (1998).
20. C. Li and S. Yang, “A generalized approach to construct benchmark problems for dynamic optimization”, *Proceedings of the 7th Int. Conf. on Simulated Evolution and Learning, Springer*,(2008).
21. C. Li, S. Yang , T.T. Nguyen, H. Jin, G. Beyer,and P.N. Suganthan, “Benchmark Generator for CEC’2009 Competition on Dynamic Optimization”, *Technical Report 2008, Department of Computer Science, University of Leicester, U.K*, (2008).
22. S.J. Louis, Z. Xu , “An immigrants scheme based on environmental information for genetic algorithms in changing environments”, *In: Proceedings of the 2008 IEEE congress on evolutionary computation*, 1141-1147, (2008).
23. Z. Michalewicz, “Genetic algorithms + data structures = evolution programs”, *Springer-Verlag, London, UK*,(1996).
24. Kondrashov A. S, and Crow J. F , “Haploidy or Diploidy: Which is Better?” Nature , **351**, 314-315, (1994).
25. M. Pelikan, D.E. Goldberg, “Escaping hierarchical traps with competent genetic algorithms”, in: Proceedings of the Genetic and Evolutionary Computation Conference, 511–518 (2001).
26. B. Saäreni, L. Krähenbühl, “Fitness sharing and niching methods revisited”, *IEEE Trans. Evol. Comput*,**2(3)**, 97–106 (1998).
27. R.K. Ursem, “Multinational GAs: Multimodal optimization techniques in dynamic environments”, *In: Proc. of (GECCO’2000), Morgan Kaufmann, San Francisco*,19–26 (2000).
28. S. Yang, “Genetic algorithms with elitism-based immigrants for changing optimization problems”, *Applications of Evolutionary Computing, LNCS* **4448**, 627–636 (2007).
29. S. Yang, “Memory-based immigrants for genetic algorithms in dynamic environment”, *: Proc. of (GECCO’05), ACM Press, New York*, 1115–1122 (2005).
30. K. deb, and D. Goldberg, “An investigation of niche and species formation in genetic function optimization”, *:In: Schaffer (Ed) Proceedings of the Third International Conference on Genetic Algorithms. Morgan Kaufmann*, 42-50 (1989).
31. L.T. Bui, H.A. Abbass, and J. Branke, “Multiobjective optimization for dynamic environments”, *in Proc. Congr. Evol. Comput.*, **3**, 2349-2356, (2005).
32. Trojanowski K, Wierzchon S “Immune-based algorithms for dynamic optimization” *Applied Soft Computing*, **(8):2**, 1495-1515, (2008).
33. A. Eiben, Z. Michalewicz, M. Schoenauer, and J. Smith, “Parameter control in evolutionary algorithms”, *In F. G. Lobo, C. F. Lima, and Z. Michalewicz (Eds.), Parameter Setting in Evolutionary Algorithms, Vol. 54 of Studies in Computational Intelligence, Berlin: Springer Verlag*, 19-46, (2007).
34. M. Serpell, and J.E. Smith, “Self-adaptation of mutation operator and probability for permutation representations in genetic algorithms”, *Evolutionary Computation, MIT Press*, **18(3)**, 491-514, (2010).
35. O. Boyabalti, I. Sabuncuoglu, “Parameter Selection in Genetic Algorithms”, *System, Cybernatics and Informatics*, **2(4)**, 78-83, (2007).
36. M. Smetek, and B. Trawiński, “Investigation of genetic algorithms with self-adaptive crossover, muta-

tion, and selection", *Hybrid Artificial Intelligent Systems*, 116-123, (2011).

37. J. Zhang, H.S.H. Chung , and W.L. Lo, "Clustering-based adaptive crossover and mutation probabilities for genetic algorithms", *Evolutionary Computation, IEEE Transactions*. **11(3)**, 326-335, (2007).