Proceedings of the 30th Chinese Control Conference
July 22-24, 2011, Yantai, China

On the Effect of Environment-Triggered Population Diversity Compensation Methods for Memory Enhanced UMDA

PENG Xingguang[1], XU Demin[1, 2], ZHANG Fubin[1, 2]

1. School of Marine Engineering, Northwestern Polytechnical University, Xi'an 710072, P. R. China
E-mail: pxg0510@gmail.com; xudm@nwpu.edu.cn; zhangfb@nwpu.edu.cn

2. National Key Laboratory for Underwater Information Processing and Control, Xi'an 710072, P. R. China
Email: xudm@nwpu.edu.cn; zhangfb@nwpu.edu.cn

Abstract: This paper focuses on the effect of population diversity to environment identification-based memory scheme (EI-MMS) which heuristically compensates population diversity through the storage and retrieving process of historic information. We introduced several diversity compensation measures and combined them with EI-MMS based univariate marginal distribution algorithm (UMDA) from two aspects. First, a basic diversity compensation measure was used to fight against the inherent diversity loss of UMDA. Second, two environment-triggered compensation measures were added in the sense of dynamic environment. Based on the experimental results on three dynamic test problems, the dynamics of population diversity of the corresponding EI-MMS based UMDAs were analyzed and several conclusions about how does the population diversity affect the performance of the algorithm in dynamic environments were drawn.

Key Words: Dynamic Optimization Problem, Estimation Of Distribution Algorithm, Memory Scheme, Population Diversity

1 Introduction

Evolutionary algorithms (EAs) are inspired by the evolutionary process in nature. From the biologic point of view, the nature process simulated by EAs is changing, random and uncertain in itself. Therefore, it is very reasonable to use EAs to solve dynamic optimization problems (DOPs). The most intuitive way to react to an environmental change is to regard each change as the arrival of a new optimization problem, and solve it from scratch. However, this method is lack of efficiency and can not satisfy most of real-world DOPs. Researchers have developed many methods to maintain a sufficient diversity level for EAs to continuously adapt to the changing landscape. They can be classified into four categories [1]: (1) generating diversity after a change, such as the hyper-mutation method [2]; (2) maintaining the diversity throughout the run, such as the random immigrants [3], sharing or crowding mechanisms [4], and the thermodynamical genetic algorithm (GA) [5]. (3) memory-based approaches[6, 7]; (4) multi-population approaches, such as the self-organizing scouts GA [8], the multi-national GA [9], and the shift balance GA [10]. Comprehensive surveys on EAs applied to dynamic environments can be found in [1, 11-13].

The essence of DOPs is to search the optimum in the solution space dynamically. For such a dynamic process, the historic information generated in the previous searching process is very useful. An intuitional method is to store the high-performance historic solutions and reuse them at some special moments so as to improve the searching process. Peng has proposed an environment identification-based memory scheme (EI-MMS) in his previous work [14] to enhance the estimation of distribution algorithms (EDAs). While the general property, parameter sensitivity and algorithms comparison were given in [14], in this work, we focus on the dynamic of population diversity to have a deeper insight about their effects to the EI-MMS. In fact, the essence of various methods for adapting EAs to changing environment is diversity maintenance. As for EDAs, there are two aspects of diversity maintenance should be considered simultaneously.

First, the diversity of conventional EDAs is likely to loss gradually while the learning and sampling processes of the probability models are executed alternately. Considering this, researchers have developed some methods to counteract the diversity loss dynamically. Shaprio has analyzed the dynamics of EDAs and proposed Detailed Balance and Bayesian Mutation methods to counteract the diversity loss [15]. These two methods aim at guaranteeing the detailed balance property of EDAs. Branker et al. [16] have proposed four diversity counteracting methods and performed some experiments on how to set the parameters. These methods are designed for conventional EDAs and work in every generation of a static evolutionary process and their effects will be tested and analyzed in Section 4.

Second, in order to persistently track the dynamic optimum in changing environment additional population diversity should be compensated to keep the search ability of EDAs. EI-MMS is indeed a heuristic diversity compensation method which saves historic information into memory and retrieves heuristic diversity according to the memory. However, as indicated by many researchers, memory based methods are especially suitable for cyclic environment. Can other diversity compensation methods be combined with EI-MMS to deal with week-cyclic environments? How do they affect the diversity of EI-MMS based EDAs? To answer these questions we introduce random immigrant method [3] by adding a parameter which control the ratio of the population generated according to the memory. In addition, we also introduce a bi-population

This work was supported by foundation of the National Key Laboratory for Underwater Information Processing and Control (9140C230503090C23).

method to the EI-MMS. Both of the above additional diversity compensation methods are tested on three dynamic test functions and their diversity dynamics is also analyzed.

The rest of this paper is organized as follows. Section 2 presents the description of EI-MMS. In section 3, the diversity loss reason is analyzed firstly and the basic diversity loss counteracting methods are introduced. Then, two additional diversity compensation methods are proposed. Section 4 presents the experimental results and analysis. Conclusions are drawn in Section 5.

2 The EI-MMS

2.1 Framework of EI-MMS

As mentioned above, in order to utilize the intervals between every two environmental changes to learn a high quantity probability model, EI-MEDA updates its memory just after the environmental change. As shown in Fig. 1, the whole dynamic optimal process is divided into many static optimal processes. In each static process, EDA searches the optimum in its conventional way. When the environment changes, EI-MMS manages the memory in three major steps. First, it stores the probability model obtained from the generation just before the environmental change into the memory. Then, it finds a memory element which best fits the new environment to retrieve using an environment identification method. Finally, the memory element is sampled to generate the first generation population in the new environment.

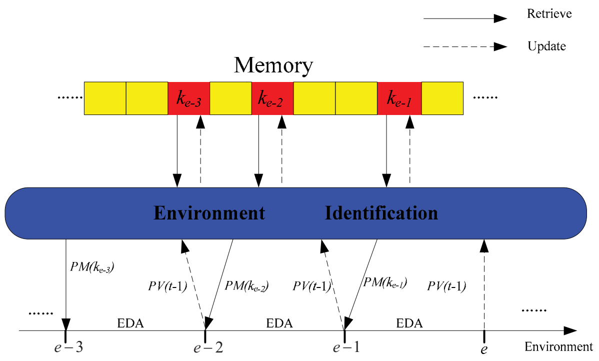

Fig. 1 Illustration of the EI-MMS

This memory management scheme should be designed specially because the elements stored in the memory are not individuals but the probability models. Hence, we explain the EI-MMS by describing its three main components: storing strategy, replacement strategy, and retrieve strategy. Also, they are relative to the main factors on how to design an explicit memory scheme summarized in [17]. Denote the size of the memory M by m (i.e. m elements can be stored in the memory M). For an l-dimensional DOP, the k-th element in M is denoted by $PM(k)$ $(k=1,2,...,m)$,which is initialized according to the uniform distribution. The probability model learned from the high-performance solutions at generation t is denoted by $PV(t)$. Denote the retrieved memory element in the *e*-th environment by $M(k_e)$. The environment changes every τ generations.

(a) Storing strategy: This strategy decides when and which probability model should be stored in the memory. In the EI-MMS, whenever the environment changes, EI-MEDA updates its memory M. The probability model $PV(t-1)$ should be stored after the procedure described in (b).

(b) Replacement strategy: This strategy decides which probability model should be replaced to make space for a new one. In the EI-MMS, the size of the memory is fixed (i.e. m) and the elements in the memory are randomly initialized. When the memory is updated, the incoming element $PV(t-1)$ will replace the memory element $PM(k_e-1)$ which was selected to retrieve in the previous environment. That is, once a memory element is selected to retrieve, it will be updated at the beginning of the next environment. This direct update method simplifies the replace scheme.

(c) Retrieve strategy: This strategy decides which probability model should be retrieved to generate a new population. In the EI-MMS, the memory information feeds back to the population via the environment identification technique. As shown in Fig. 1, the memory element $PM(k_e)$, which is most suitable for the new environment, is selected to generate the initial population in the new environment with the environment identification technique.

In summary, the EI-MMS can be described like follows: if an element in the finite-size memory is selected when the environment changes, it will go through the retrieving, evolution and update processes in turn. The retrieving and update processes start at the beginning of two consequent environments.

2.2 Environment Identification Method

Considering the computational complexity and the accuracy, we propose a Samples Averaging + Best Individual (SA + BI) method to evaluate the elements in the memory and select the suitable one. Suppose $BM(k)$ is the the best individual in $M(k)$. Fig. 2 shows the pseudo-code of SA+BI method.

```
Algorithm 1 The SA+BI Method
Input: The memory elements
$M(i)=<BM(i),PM(i)>,(i=1,2,\dots,m)$
Output: The suitable memory element $M(index)$
1: Set $maxfit:=0$ and $index:=1$;
2: for $i = 1$ to $m$ do
3:   $f_M(k)=f(BM(k))$
4:   if $f_M(i)>maxfit$ then $maxfit:=f_M(i)$ and $index:=i$;
5:   else if $f_M(i)=maxfit$ then $f_M(k)=\frac{1}{N_S}\sum_{i=1}^{N_S} f^k_{ind}(i)$
6:     if $f_M(i)>f_M(index)$ then $index:=i$;
7: end for
8: return $M(index)$;
```

Fig. 2 Pseudo-code of the SA+BI method

3 Diversity Compensation Methods

The conventional EDA is likely to search the space where it has visited, just like the genetic algorithm without a mutation operation. When the probability distribution of a decision variable is close to 1 or 0, it is difficult to change its value anymore. This is the so-called fixed-point problem and it may mislead the searching process to a local optimum.

Some researchers have contributed to address this problem [15, 16, 18, 19].

According to the experimental study in [16], the method that combines the Loss Correction and Boundary Correction methods, denoted LC+BC in this paper, is outstanding to counteract the diversity loss. Hence, we use it as the basic diversity compensation method.

The diversity compensation methods discussed above are designed for conventional EDAs. In fact, these methods work between every two environmental changes. That is, they compensate the population diversity in static environments. In order to track the changing environment, we propose two additional diversity compensation methods: Partial Random Retrieve (PRR) and Multiple Population with Restart (MPR). Both are designed to adapt to environmental changes. They are described as follows.

For PRR, it is developed from EI-MEDA by adding a parameter α to control the proportion of the retrieved population and the rest population is generated randomly. This is similar to apply a random immigration to the retrieved population in a 1-α proportion. Here we apply the PRR to UMDA and denote the corresponding algorithm by αEI-MUMDA. For MPR, the idea comes from [6], where the multiple population method is applied to the Population-Based Incremental Learning (PBIL) algorithm for DOPs. Yang has employed two size-adjustable populations to track the changing optimum in [6]. When the environment changes, one population searches with the associative memory scheme while the other searches from scratch. In this paper, this method is modified and integrated into our EI-MMS framework and applied to UMDA and the corresponding algorithm is denoted by EI-MUMDA2r. Fig.3 shows the pseudo-code of EI-MUMDA2r.

```
Algorithm 3 EI-MUMDA2r
1:  $t$:=0, $e$:=0, $\beta$:=0.5;
2:  $PV^1(0,i)$=0.5, $PV^2(0,i)$=$rand$(0,1), $PM(k,i)$=0.5, ($i=1,2,\ldots,l;\ k = 1, 2, \ldots, m$ )
3:  Sample $PV^1(0)$ and $PV^2(0)$ to generate $P^1_0$ of size $\beta N$ and $P^2_0$ of size $N$-$\beta N$
4:  repeat
5:    if the $e$-th environment change is detected in generation $t$ then
6:      $e$:=$e$+1
7:      if there are still some initial memory elements (i.e. $e \leqslant m$) then
8:        $PM(e)$:= $PV^B(t$-1$)$
9:      else $PM(k_e$-1$)$:= $PV^B(t$-1$)$
10:       Re-evaluate the best individual of each memory element
11:       Select $PM(k_e)$ using Algorithm 1
12:       $PV^1(t)$= $PM(k_e)$, $PV^2(t,i)$=0.5
13:       Sample $PV^1(t)$ and $PV^2(t)$ to generate $P^1_t$ and $P^2_t$ respectively
14:       Evaluate $P^1_t$ and $P^2_t$
15:     else
16:       Adjust the sizes of $P^1_t$ and $P^2_t$ to $\beta N$ and $N$-$\beta N$ respectively
17:       Compensate the diversity for $PV^1(t)$ and $PV^2(t)$ using the LC+BC method
18:       Sample $PV^1(t)$ and $PV^2(t)$ to generate $t$-th offspring $O^1_t$ and $O^2_t$
19:       Evaluate $O^1_t$ and $O^2_t$ and replace the worst individuals of $P^1_t$ and $P^2_t$
20:       Denote the best individuals of $P^1_t$ and $P^2_t$ by $B^1_t$ and $B^2_t$ respectively
21:       if $B^1_t$ is better than $B^2_t$ then $\beta$:=$\beta$+0.05, $\beta \in [\beta_l, \beta_h]$
22:       if $B^1_t$ is worse than $B^2_t$ then $\beta$:=$\beta$-0.05, $\beta \in [\beta_l, \beta_h]$
23:       Denote the probability vector of the better of $B^1_t$ and $B^2_t$ by $PV^B(t)$
24:       Select $fN$ best individuals from $P^1_t$ and $P^2_t$ to construct $PV^1(t+1)$ and $PV^2(t+1)$
25:     $t$:=$t$+1
26: until terminated = $true$
```

Fig. 3 Pseudo-code of the EI-MUMDA2r

4 Experimental Study

4.1 Methods of Study

Here we introduce a bitwise exclusive-or (XOR) DOP generator which was proposed in [6]. This DOP generator can construct dynamic environments for any binary-encoded function by an XOR operator.

Decomposable unitation-based functions (DUFs have been widely studied in the EA community in the attempt to understand what constructs difficult problems for EAs, especially for GAs [20]. In this paper, three DUFs (denoted DUF1, DUF2 and DUF3 respectively) are used as the stationary test functions. DUF1 is a simple OneMax problem, DUF2 is a deceptive function and DUF3 is a fully deceptive function. Generally speaking, these three DUFs form an increasing difficulty for EAs in the order from DUF1 to DUF2 to DUF3. Now, the dynamic test problems could be constructed by applying the XOR DOP generator to the DUFs and got three dynamic DUFs denoted DDUF1, DDUF2 and DDUF3 respectively.

In order to measure the performance of algorithms, the Collective Mean Fitness [21] is introduced. This measurement is calculated by averaging the best-of-generation fitness of every generation. Suppose each experiment is performed NE (NE=50) times independently with the same experimental settings, the Collective Mean Fitness (FCMF) is formulated as below:

$$F_{CMF} = \frac{1}{G}\sum_{i=1}^{G}\left(\frac{1}{N_E}\sum_{j=1}^{N_E}F_{BOG}\ (i,j)\right) \tag{4}$$

where G ($G = N_e \times \tau$, N_e denotes the quantity of the environment periods) is the total generations in each run and $F_{BOG}(i, j)$ denotes the best-of-generation fitness of i -th generation in j -th run.

In order to understand the effect of population diversity, we also recorded the diversity of the population every generation. The diversity of the population at time *t* in the *k*-th run of an algorithm on a DOP is defined as

$$Div(k,t) = \frac{1}{ln(n-1)} \sum_{i=1}^{n} \sum_{j \neq i}^{n} HD(i, j) \qquad (5)$$

where l is the encoding length, n is the population size, and $HD(i, j)$ is the Hamming distance between the i -th and *j*-th individual in the population. The mean population diversity of an algorithm on a DOP at time t over NE runs is calculated as follows:

$$\overline{Div}(t) = \frac{1}{N_E} \sum_{k=1}^{N_E} Div(k,t) \qquad (6)$$

4.2 Effect of Environment-Triggered Diversity Compensation Methods

In order to analyze the environment-triggered diversity compensation methods in different dynamic environments, our experiments are performed on 100-dimentions DDUFs. The experimental parameters are set as: α={0.2, 0.5, 0.8, 1.0}, β_l =0.3, β_h =0.6, p_n =0.2, N_{pop} =100, m=20. The αEI-MUMDA and the EI-MUMDA2r are compared in Table 1 and their dynamic population diversity is shown in Fig. 4. Several conclusions can be drawn and analyzed as follows.

Table 1: Performance of αEI-MUMDA and EI-MUMDA2r in different environments

τ	ρ	DDUF1					DDUF2					DDUF3				
		0.2EI	0.5EI	0.8EI	1.0EI	EI2r	0.2EI	0.5EI	0.8EI	1.0EI	EI2r	0.2EI	0.5EI	0.8EI	1.0EI	EI2r
		Cyclic														
5	0.1	95.28	**96.25**	95.92	95.93	95.72	90.01	**91.14**	90.28	90.04	90.43	50.32	80.32	84.66	**85.10**	84.44
	0.2	98.39	99.01	**99.04**	97.83	98.80	96.47	**97.82**	97.50	95.94	97.38	51.11	84.33	85.52	**85.98**	85.88
	0.5	99.20	99.54	**99.55**	99.44	99.52	98.28	**99.04**	**99.04**	98.63	98.90	69.79	85.97	86.56	**86.69**	**86.69**
	1.0	99.57	**99.78**	99.76	99.70	99.73	99.02	99.50	**99.51**	99.25	99.41	75.85	86.74	86.99	87.08	**87.13**
20	0.1	99.82	99.83	99.83	**99.84**	99.78	99.57	**99.66**	99.64	99.58	99.51	87.27	**87.49**	87.31	87.33	87.37
	0.2	**99.81**	99.79	99.78	99.78	99.78	**99.55**	99.40	99.34	99.18	99.43	87.25	87.27	**87.31**	**87.31**	87.28
	0.5	**99.90**	99.89	99.87	99.75	99.86	99.78	**99.77**	99.74	99.17	99.72	87.31	87.34	87.37	87.42	**87.50**
	1.0	**99.95**	99.94	99.94	99.80	99.92	**99.89**	99.88	99.87	99.17	99.81	87.45	87.43	87.50	87.44	**87.63**
Cyclic with noise																
5	0.1	**79.00**	76.13	74.59	74.81	77.75	**58.47**	54.96	53.56	53.48	56.59	49.83	55.32	59.06	**59.32**	58.95
	0.2	77.17	**79.04**	78.30	78.42	77.44	55.33	**58.97**	57.82	57.78	56.51	49.65	50.83	58.36	**58.69**	57.38
	0.5	85.70	87.45	87.38	**87.67**	86.25	70.45	74.10	73.48	**74.97**	71.43	49.89	70.03	72.21	**72.68**	70.28
	1.0	90.83	93.51	94.00	**94.35**	93.13	80.94	86.70	87.58	**88.24**	85.58	50.29	79.16	80.75	**81.43**	79.26
20	0.1	95.27	95.39	95.40	**95.53**	94.24	89.59	90.30	**90.48**	90.14	88.03	70.83	71.09	71.10	**71.33**	70.47
	0.2	**90.99**	89.49	89.07	89.42	90.01	**80.45**	76.98	76.46	74.57	78.34	**76.75**	76.61	75.62	75.81	74.66
	0.5	97.14	97.37	**97.43**	97.35	96.73	94.10	94.51	**94.66**	94.38	93.31	84.03	84.90	85.12	**85.22**	84.23
	1.0	98.68	98.89	**98.95**	98.88	98.59	97.33	97.70	**97.88**	97.40	97.10	86.00	86.46	86.59	**86.63**	86.19
Random																
5	0.1	83.16	84.16	83.59	**84.22**	82.47	65.96	**67.54**	66.51	67.51	64.50	49.90	59.74	60.77	**61.19**	59.50
	0.2	**76.99**	75.97	74.22	73.95	75.56	**54.85**	54.53	51.57	50.27	52.77	49.64	50.27	54.15	**55.27**	54.05
	0.5	72.23	**72.57**	71.03	68.66	71.11	47.17	**48.09**	47.01	43.93	46.22	49.63	49.91	52.18	**54.21**	53.00
	1.0	99.57	**99.76**	**99.76**	99.69	99.72	99.02	**99.50**	**99.50**	99.24	99.41	70.83	86.68	87.03	87.11	**87.18**
20	0.1	96.10	96.32	96.41	**96.51**	95.52	91.88	92.27	92.50	**92.66**	90.64	71.83	72.12	72.22	**72.38**	71.96
	0.2	**91.50**	90.71	90.37	90.55	90.23	**82.10**	80.02	79.28	79.58	78.73	**67.84**	67.75	67.47	67.70	66.18
	0.5	**87.23**	83.19	80.40	78.82	86.19	**72.63**	65.70	61.03	58.49	71.02	57.84	**64.96**	64.91	64.18	63.29
	1.0	**99.95**	**99.95**	99.94	99.80	99.92	**99.89**	99.88	99.87	99.20	99.82	87.41	87.48	87.46	87.43	**87.55**

First, the additional diversity compensation is positive for solving the DDUF1 and the DDUF2 problems while negative for solving the DDUF3 problems. This means a high diversity level may be harmful to highly deceptive DOPs. This is because, for static DUF1 and DUF2, the memory-based EDAs with basic diversity compensation are able to search the optimum effectively. And the pulse-like population diversity introduced by the additional diversity compensation makes the algorithms search more widely to adapt for the environment change. However the situation is different when the fully deceptive DUF3 is considered. Because, unlike DUF1 and DUF2, the DUF3 is very difficult for static EDAs and the search is likely to be trapped into some sub-optima. Therefore, the additional random individuals introduced by the additional diversity compensation may lead to more serious deception. This is demonstrated by the fact that 1.0EI-MUMDA performs the best in most of the DDUF3 conditions shown in Table 1.

Accordingly, when solving the DDUF3, one would rather make the algorithms go itself guided by its memory than introduce some additional diversity.

Second, for each DDUF the sensitivity of the αEI-MUMDA and the EI-MUMDA2r to ρ and τ is similar to the EI-MUMDA's analyzed above. Concretely, it can be summarized as follows. (1) The slower the environment changes, indexed by τ , the better can the algorithms track the dynamic optimum. (2) The environments changing severity, indexed by ρ , can offset the noise in some degree. If the noise can be effectively offset, a violently changing environment is good for the environment identification. Otherwise, large environment changing severity may make the situation worse.

Third, in most situations for DDUF3 and all situations for DDUF1 and DDUF2, the EI-MUMDA2r can not overcome the single-population αEI-MUMDA (if we regard the algorithms with a series α value as a whole). This is because although the EI-MUMDA2r can compensate some pulse-like diversity like the αEI-MUMDA by randomly generating its second population, the parallel and independent search processes of its two populations slow down the speed at which the whole population converges to the optimum. This is demonstrated in Fig. 4 the dynamic population diversity level of the EI-MUMDA2r is higher than most of the αEI-MUMDA's. This reveals that although the population diversity is necessary to make the algorithm adapt to the environment change it may hamper the population to converge to the optimum when it is on a improper level.

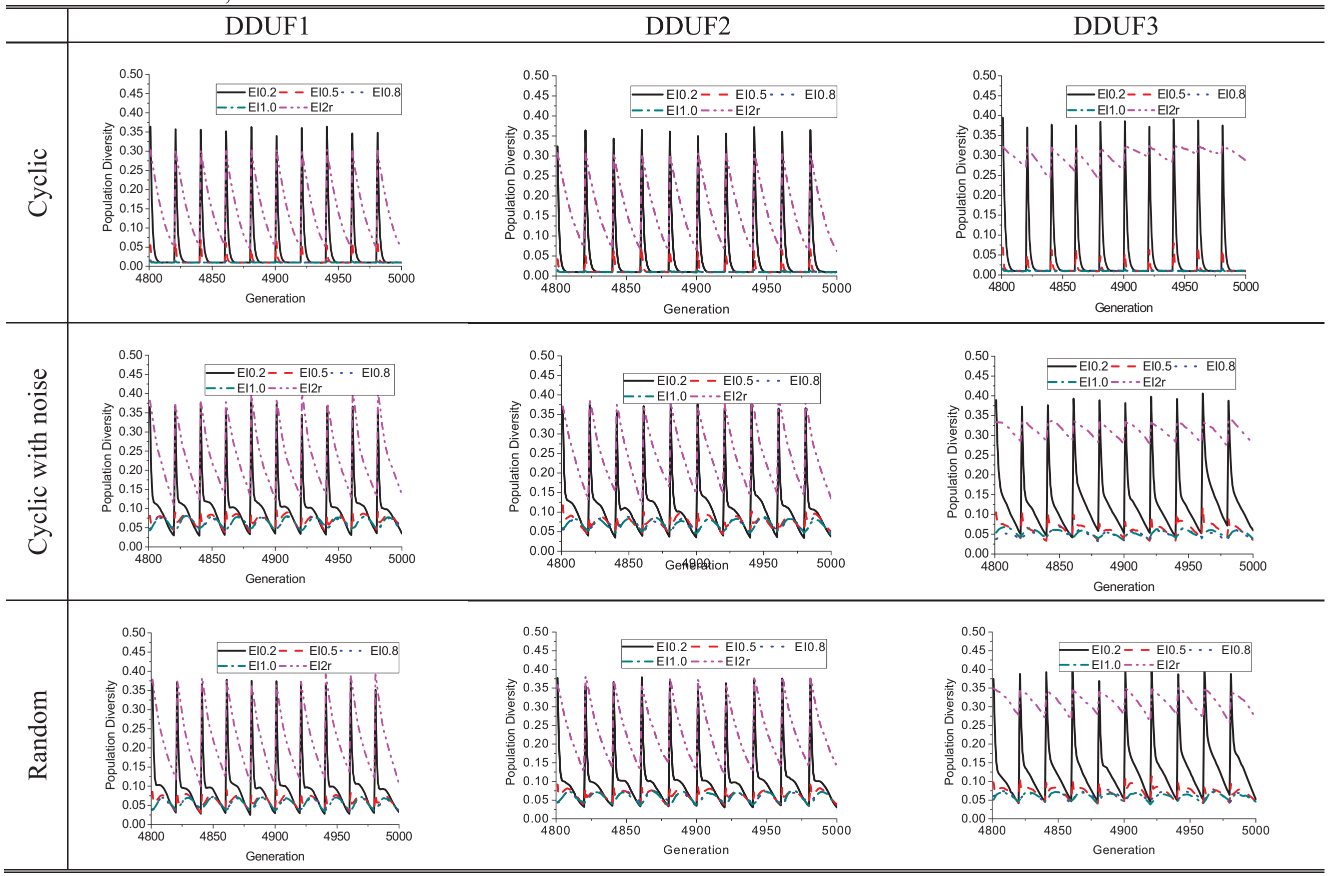

Fig. 4 Mean dynamic population diversity of algorithms for the last ten environmental changes on DDUFs with τ = 10 and ρ=0.2.

5 Conclusions

The essence of both conventional EDAs and their application in dynamic environments is maintenance of population diversity. Accordingly, this paper focuses on analyzing the effect of population diversity on EDAs in dynamic environments. We have introduced the details of EI-MMS including its storing, replacement and retrieve strategies and the environment identification method. The reasons to the inherent diversity loss of EDAs have been analyzed and several counteracting measurements have been introduced into EI-MMS based UMDA to compensate the diversity in both static and dynamic optimization aspects. Concluded from the experimental results, the following conclusions are drawn:

First, the additional diversity compensation methods are positive for solving the DDUF1 and the DDUF2 while negative for solving the fully deceptive DDUF3.

Second, the bi-population scheme of EI-MUMDA2r makes the dynamic population diversity maintain at a higher level and change smoother in comparison with αEI-MUMDA.

Third, although it is important to maintain the population diversity to adapt to the changing environment, an improper diversity level may be harmful for converging to the optimum effectively.

References

[1] Y. Jin, J. Branke. Evolutionary Optimization in Uncertain Environments -A Survey. *IEEE Transaction on Evolutionary Computation*, 9(3): 303-317, 2005

[2] H.G. Cobb, An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environments. 1990, Naval Res. Lab: Washington, DC.

[3] J.J. Grefenstette. Genetic algorithms for changing environments, in *proceedings of 2nd international conference on parallel problem solving from nature*, 1992: 137-144.

[4] W. Cedeno, V.R. Vemuri. On the use of niching for dynamic landscapes, in *proceedings of 1997 congress on evolutionary computation (CEC 97)*, 1997: 361-366.

[5] N. Mori, H. Kita, Y. Nishikawa. Adaptation to a changing environment by means of the thermodynamical genetic algorithm, in *proceedings of parallel problem solving from nature (PPSN IV)*, 1996: 513-522.

[6] S. Yang, X. Yao. Population-based incremental learning with associative memory for dynamic environments. *IEEE Transactions on Evolutionary Computation*, 2008, 12 (5): 542-561.

[7] S. Yang, H. Cheng, F. Wang. Genetic algorithms with immigrants and memory schemes for dynamic shortest path routing problems in mobile ad hoc networks. *IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews*, 2010, 40 (1): 52-63.

[8] J. Branke, P. Funes, C. Schmidt, et al. A multipopulation approach to dynamic optimization problems, in *proceedings of Adaptive Computing in Design and Manufacturing (ACDM 2000)*, 2000: 299--308.

[9] R.K. Ursem. Multinational GA optimization techniques in dynamic environments, in *proceedings of 2nd annual conference on genetic and evolutionary computation conference (GECCO 2000)*, 2000: 19-26.

[10] M. Wineberg, F. Oppacher. Enhancing the GA's ability to cope with dynamic environments, in *proceedings of 2nd annual conference on genetic and evolutionary computation conference (GECCO 2000)*, 2000: 3-10.

[11] H.G. Cobb, J.J. Grefenstette. Genetic algorithms for tracking changing environments, in *proceedings of 5th International Conference on Genetic Algorithms*, 1993: 523-530.

[12] J. Branke, *Evolutionary Optimization in Dynamic Environments*. 2001: Kluwer.

[13] R.W. Morrison, *Designing Evolutionary Algorithms for Dynamic Environments*. 2004: Springer.

[14] X. Peng, X. Gao, S. Yang. Environment identification based memory scheme for estimation of distribution algorithms in dynamic environments. *Soft Computing*, 2011, 15 (2): 311-326.

[15] J.L. Shapiro. Diversity Loss in General Estimation of Distribution Algorithms, in *proceedings of parallel problem solving from nature (PPSN IX)*, 2006: 92-101.

[16] J. Branke. Addressing Sampling Errors and Diversity Loss in UMDA, in *proceedings of 9th Annual Conference on Genetic and Evolutionary Computation Conference (GECCO 2007)*, 2007: 508-515.

[17] J. Branke. Memory enhanced evolutionary algorithms for changing optimization problems, in *proceedings of 1999 congress on evolutionary computation (CEC 99)*, 1999: 1875-1882.

[18] J.L. Shapiro. Scaling of Probability-Based Optimization Algorithms, in *proceedings of Advances in Neural Information Processing Systems*, 2003: 399-406.

[19] J.L. Shapiro. Drift and Scaling in Estimation of Distribution Algorithms. *Evolutionary Computation*, 13(1): 99-123, 2005

[20] D.E. Goldberg, *The Design of Innovation: Lessons from and for Competent Genetic Algorithms*. 2002, Norwell, MA, USA Kluwer. 272.

[21] R. Morrison, K.D. Jong. A test problem generator for non-stationary environments, in *proceedings of 1999 congress on evolutionary computation (CEC 99)*, 1999: 2047-2053.