2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

An Adaptive Diversity Introduction Method for Dynamic Evolutionary Multiobjective Optimization

Min Liu [1,2,3]
1. School of Computer Science and Engineering
Hunan University of Science and Technology
Xiangtan, China
2. Key Laboratory of Knowledge Processing and Networked Manufacturing
College of Hunan Province
Xiangtan, China
3. School of Computer
Minnan Normal University
Zhangzhou, China
Email: liumin1974@gmail.com

Jinhua Zheng [4]
4. Institute of Information Engineering
Xiangtan University
Xiangtan, China
Email: jhzheng@xtu.edu.cn

Junnian Wang[2], Yuzhen Liu[2], Lei Jiang[2]
2. Key Laboratory of Knowledge Processing and Networked Manufacturing
College of Hunan Province
Xiangtan, China

***Abstract*—This paper investigates how to use diversity introduction methods to enhance the dynamic evolutionary multiobjective optimization algorithms in dealing with dynamic multiobjective optimization problems (DMOPs). Although diversity introduction method is easy used to response to the dynamic change, current diversity introduction methods still have a difficulty in identifying the correct proportion of diversity introduction. To overcome this difficulty, this paper proposes an adaptive diversity introduction (ADI) method. Specifically, the proportion of diversity introduction can be dynamically adjusted rather than being hand designed and fixed in advance. In addition, an adaptive relocation operator is designed to adapt the evolving individuals to the new environmental condition. The effectiveness of the ADI method is validated against various diversity introduction methods upon five DMOPs test problems. The simulation results show that the proposed ADI has better robustness and total performance than other diversity introduction methods.**

Keywords—dynamic multi-objective optimization; evolutionary algorithm; diversity introduction; adaptive

I. INTRODUCTION

Optimization problems occur in many situations and aspects of modern life. Optimization problems include single-objective optimization problems (SOPs) and multiobjective optimization problems (MOPs). MOPs can be classified as stationary multiobjective optimization problems (SMOPs) and dynamic multiobjective optimization problems (DMOPs). Evolutionary multiobjective optimization (EMO) algorithms are a class of stochastic optimization techniques that simulate biological evolution to solve MOPs [1]. Most of EMO works are confined to SMOPs currently [2]. However, many real-world problems are DMOPs, their objective vector-valued function, constraints and problem parameters may change with time [3]. Accordingly, the Pareto optimal front is unlikely to remain invariant. Hence, the optimization goal is not only to evolve a near-optimal and diverse Pareto optimal front, but also to track the front as it changes with time [4]. Therefore, the research topic, dynamic EMO (DEMO), has obtained growing attention among researchers recently [2].

Based on Evolutionary algorithm (EA), some DEMO algorithms have been proposed to solve DMOPs. Deb *et al*. [5] extended the well-known NSGA-II algorithm [6] to solve DMOPs, and they denoted the dynamic version of NSGA-II as DNSGA-II. Goh and Tan [4] proposed a new coevolutionary paradigm that hybridizes competitive and cooperative mechanisms to solve DMOPs. Liu and Zeng [7] presented a memory enhanced dynamic multi-objective evolutionary algorithm based on decomposition. Recently, Zhou *et al*. [8] suggested a population prediction strategy for dynamic evolutionary multiobjective optimization.

Besides EAs, other nature-inspired optimization methods, such as particle swarm optimization (PSO) and artificial immune systems (AIS), *etc*, have also been introduced to deal with DMOPs [9-13]. In [9], a dynamic multiobjective PSO, maximinPSOD, was proposed. Greff and Engelbrecht [10] presented a vector evaluated particle swarm optimiser (VEPSO) to solve DMOPs. In the area of AIS, Zhang [11] investigated immune-based optimization techniques for a class of DMOPs. Shang *et al*. [12] designed a clonal selection algorithm for dynamic multiobjective optimization.

Generally, DEMO algorithm must make a balance between convergence and diversity for dealing with DMOPs [14],[15]. This is because if the dynamic landscape changes in one area and there is no member of the algorithm in this area. Once the algorithm is converged, it is hard to escape from an old optimum and hence might fail to track the

This work was supported by the Scientific and Technology Project of Fujian Province Education Department (No. JA12221), Scientific Research Fund of Hunan Science and Technology Department (No.2013FJ3002), Fund of Hunan University of Science and Technology (No. E51368).

moving global optimum. In particular, when a new change happens, the DEMO algorithm must be capable of solving diversity problem within the evolving population. The different techniques proposed to handle population diversity are based on the following three approaches, i.e., diversity introduction after a change, diversity maintaining and multiple populations [4].

Diversity introduction (DI) method is a strategy which introduces a certain degree of diversity to the evolutionary population during the transition phase. The DEMO algorithm is run in standard fashion, but as soon as a change in the environment is detected, explicit actions are taken to increase diversity and, thus, to facilitate the shift to the new optimum [4]. If the change is radical, and the new problem bears little resemblance to the previous problem, random restart (RR) or reinitialization may be the only viable option. If the optimal solutions of the new environment are similar to those of the old environment, some percentage of new individuals had better be introduced either through random initialization or mutation [5].

The problem of DI method is that increasing diversity is basically equivalent to replacing information about previously successful individuals by random information. It is difficult to determine a useful amount of diversity: Too much will resemble restart, while too little does not solve the problem of convergence [15]. Therefore, an ideal approach may be the adaptive diversity introduction method, in which the proportion of diversity introduction is dynamically determined according to the actual environmental change rather than being hand designed and fixed in advance. However, to the best of our knowledge, most of current methods use a fixed proportion of diversity introduction. Motivated by this observation, we propose an adaptive diversity introduction (ADI) method for DEMO algorithms. The major contributions of this paper include the following.

- The appropriate proportion of diversity introduction is determined by estimating the actual extent of environmental change.
- An adaptive relocation operator is designed to adapt already converged or currently evolving individuals to the new environmental condition.

The paper is organized as follows. Section II introduces the DMOPs and discusses related diversity introduction methods. Our adaptive diversity introduction method is presented in Section III. Then, sections IV reports and analyzes experimental results of our method with a comparison to other DI methods. The paper concludes with a summary and some ideas for future work in Section V.

II. BACKGROUND

A. Problem Statement

Definition 1: The DMOPs [3] can be formally defined as

$$\begin{cases} \min\limits_{\boldsymbol{x}\in\boldsymbol{\Omega}} \boldsymbol{y} = \boldsymbol{F}(\boldsymbol{x},t) = \left(f_1(\boldsymbol{x},t), f_2(\boldsymbol{x},t), \ldots, f_m(\boldsymbol{x},t)\right)^T \\ s.t.\ \ g_i(\boldsymbol{x},t) \le 0 \quad i = 1,2,\ldots,p \\ \qquad h_j(\boldsymbol{x},t) = 0 \quad j = 1,2,\ldots,q \end{cases} \tag{1}$$

where $\boldsymbol{x}$ is the vector of decision variables bounded by the decision space, $\boldsymbol{\Omega}$; The evaluate function, $\boldsymbol{F}(\boldsymbol{x},t)$ is a mapping from $\boldsymbol{\Omega} \times t$ to objective space, $\boldsymbol{\Lambda}$. $\boldsymbol{y}$ is the set of objectives to be minimized with respect to time, t. m is the number of objective function. The functions of $\boldsymbol{g}$ and $\boldsymbol{h}$ represent the set of inequality and equality constraints that changes with t, respectively.

Definition 2: Pareto Dominance: A vector $\boldsymbol{u} = (u_1, u_2, ..., u_m)^T$ is said to dominate another vector $\boldsymbol{v} = (v_1, v_2, ..., v_m)^T$, denoted by $\boldsymbol{u} \prec \boldsymbol{v}$ if and only if $\boldsymbol{u}$ is partially less than $\boldsymbol{v}$, ie., $\forall k \in \{1,2,...,m\}$, $u_k \leqslant v_k$, and, $\exists l \in \{1,2,...,m\}$, $u_l < v_l$.

Definition 3: Dynamic Pareto Optimal Set: The dynamic Pareto optimal set, denoted as POS(t), is the set of solutions that are non-dominated in the decision space such that $POS(t) := \{\boldsymbol{x} \in \boldsymbol{\Omega} \mid \neg\exists \boldsymbol{x}' \in \boldsymbol{\Omega}, \boldsymbol{F}(\boldsymbol{x}',t) \prec \boldsymbol{F}(\boldsymbol{x},t)\}$.

Definition 4: Dynamic Pareto Optimal Front: The dynamic Pareto optimal front, denoted as POF(t), is the set of solutions that are non-dominated in the objective space such that $POF(t) := \{y = \boldsymbol{F}(\boldsymbol{x},t) \mid \boldsymbol{x} \in POS(t)\}$.

B. Related Work

This section summarizes some related work on DI methods and further presents a framework of DEMO algorithm with DI method.

In a certain sense, the DMOP can be considered as the consecutive optimization of different time-constrained MOPs with varying complexities [4]. It is imperative that DEMO algorithm must be capable of attaining a fast convergence in order to find the optimal solution set before it changes and becomes obsolete. However, a fast convergence also implies a rapid loss of diversity during the optimization process, which inevitably leads to the difficulty of tracking the dynamic Pareto optimal front. It is, thus, necessary to introduce sufficient diversity in order to explore the search space when the DMOPs changes in a dynamic environment.

A typical representative of DI method is reinitialization. Other common techniques include hypermutation [16] where the mutation rate is increased drastically, and variable local search [17] where mutation rate is increased gradually. According to the amount of diversity introduction, the existing diversity introduction methods can be classified as whole DI and partial DI.

1) Whole diversity introduction: Random restart (RR) or reinitialization of the whole population is one of the simplest whole DI for generating diversity [4]. The main drawback of this approach is that information gained is lost after the introduction of diversity [18]. Recently, a memory like reinitialization (MLR) strategy is adopted for the new environment [19]. In MLR, the individuals of new generated population are randomly generated within the bounds of the search space or generated by the Gaussian local search operator.

2) Partial diversity introduction: Deb *et al*. [5] proposed dynamic NSGA-II (DNSGA-II) for DMOPs. In order to detect problem changes, 10% of individuals in the population are selected randomly and reevaluated in every generation. When a change is detected, all outdated solutions are

reevaluated, and diversity introduction is executed either through random initialization or mutation. Specifically, a ζ% of the new population is replaced with randomly created solutions or with mutated solutions of existing solutions. Accordingly, in this paper we denote these two diversity introduction methods as random diversity introduction (RDI) and mutation diversity introduction (MDI) respectively. Greff and Engelbrecht [10] presented vector evaluated particle swarm optimizer (VEPSO) to solve DMOPs. When a change is detected, several responses with different amount diversity introduction are used. Either 10%, 20% or 30% of the swarm's population is reinitialized. Reinitialization of particles is either done for all swarms, or only for the swarm that is solving the objective function that has changed. The results showed that different percent adapts to different dynamic multiobjective benchmark problems, and none of these percentages can achieve optimal on all benchmark problems. Later on, Helbig and Engelbrecht [20] further adopted a partial DI method with fixed proportion of diversity in their dynamic VEPSO algorithm. If a change has been detected, 30% of the particles of the swarm(s) whose objective function changed are reinitialized.

Instead of reinitialization or subjecting the entire subpopulation to hypermutation, Goh and Tan [4] adopted a competitive process to regulate diversity introduction. The idea of competitive process is to compare the potential of new regions in the search space and the past information to decide whether the subpopulation should be initialized. However, it is not clear how this diversity introduction is implemented.

Since DI method is common used to enhance the DEMO for solving DMOPs, we summarize a general DEMO algorithm framework with DI, which pseudo-code is described in algorithm 1.

Algorithm 1: A DEMO Framework with DI Method

```
1  Set time step t = 0;
2  Initialize population P_t
3  while not terminate do
4     if change() then
5        Execute DI on P_t;
6        t = t+1;
7     else
8        Evolve P_t to optimize the t-th MOP by using
         an EMO algorithm;
9     end if
10 end while
```

In algorithm 1, more diversity are introduced into the evolving population to response the new change, the more useful information gained will be lost. On the other hand, too little diversity introduction does not solve the problem of convergence. An extreme example is none diversity introduction (NDI) method, in which the last population of the previous environment is just set as the starting population for the new environment [12]. This NDI method may be feasible only if there is a large degree of similarity between the old environment and the new environment. Therefore, it is difficult to determine a useful amount of diversity for an uncertain new change [15]. Bearing these observations in mind, an adaptive diversity introduction method for DEMO is suggested, investigated, and discussed in the following sections.

III. Adaptive Diversity Introduction Method

Firstly, a measure method is given to estimate the extent of environmental change. Secondly, the appropriate proportion of diversity introduction is calculated based on the extent of change. Later on, an adaptive relocated operator is designed. Finally, the algorithm of ADI is given.

A. Estimating the Extent of Environmental Change

Firstly, the extent of environmental change is defined as the degree of deviation between the old POF and the new POF. Since POF usually can not be known in advance, the difference of evolutionary populations before and after change is used to estimate the extent of change. Specifically, we simply use the following formula for estimating the extent of environmental change in objective space:

$$\delta(t) = \frac{\sum_{i=1}^{K} \left(\left\| \boldsymbol{F}(\boldsymbol{x}^i, t) - \boldsymbol{F}(\boldsymbol{x}^i, t-1) \right\| \right)}{K} \tag{2}$$

where the operator $\|\bullet\|$ is the Euclidean distance. After K different individuals are randomly sampled from the population, the average displacement of their objective function vectors is calculated to approximate the extent of environmental change.

B. Adaptive Proportion of the Diversity Introduction

Here, the proportion of diversity introduction is adaptive and can be dynamically adjusted rather than being hand designed and fixed. Formally, the proportion of diversity introduction can be calculated as follows:

$$\zeta(t) = \eta(\delta(t)) = Min(\lambda \times \frac{\delta(t) - \delta_{\min}}{\delta_{\max} - \delta_{\min}}, 1.0) \tag{3}$$

where $\zeta(t)$ is the adaptive proportion of the diversity introduction. $\delta(t)$ is the extent of environmental change. $\delta_{\min}$ and $\delta_{\max}$ are the minimum and maximum change recorded in the history, respectively. With the increasement of the objective space in size, the extent of environmental change is gradually enlarged. Therefore, we introduce a scale factor λ, and set λ=m-1, where m is the number of objective function. Finally, the adaptive proportion of diversity introduction is determined by normalization and multiplication. If the result is greater than 1, it will be revised as 1 by Min function.

Based on the $\zeta(t)$, $\zeta(t) \times N$ individuals are sampled from the population randomly, where N is the size of the population. Then, these individuals are updated by using an adaptive relocation operator described in the next section to complete diversity introduction.

C. Adaptive Relocation Operator

For every sampled individual $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]$, either Gaussian local search or random initialization is chosen to relocate the individual. Specifically, the probability of random initialization is set to be $\zeta(t)$, while the probability of

Gaussian local search is 1-$\zeta(t)$. Here, the Gaussian local search is defined as follows:

$$x_i = x_i + N(0, \zeta(t)), \ i = 1,2, \ldots, n. \quad (4)$$

where n is the number of decision variable, $N(0,\zeta(t))$ is a Gaussian distribution, its mean value and standard deviation are 0 and ζ(t), respectively. The adaptive relocation operator is also shown in Fig 1.

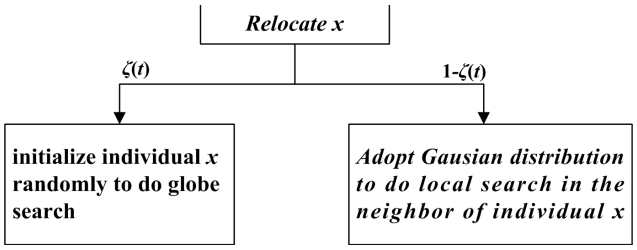

Fig. 1. Adaptive relocation operator.

From this figure, the relocation operator owns adaptive ability in the following two areas:

- Both the probability of random initialization and that of Gaussian local search are dynamically adjusted according to the estimation of extent of environmental change.
- If the environmental change is significant, the individual is more likely to be randomly reinitialized. On the other hand, if the environmental change is small, the individual tends to do local search in its neighborhood. Furthermore, the neighborhood radius (i.e., $\zeta(t)$) of the Gaussian local search is also adaptively determined according to the estimated extent of environmental change.

D. Algorithm Description of the Adaptive Diversity Introduction Method

The pseudo-code description of the ADI method is given in Algorithm 2.

Algorithm 2: Adaptive diversity introduction method

```
1  Estimate the extent of environmental change δ(t) by
   using formula 2;
2  Calculate the adaptive proportion of the diversity
   introduction ζ(t) by using formula 3;
3  for (i = 0; i < ζ(t)×N; i++) {
4     Choose an individual x from Population P_t
      randomly;
      /*Perform adaptive relocation operator as follows.
      Pr is a random decimal between 0 and 1.0 */
5     Pr = randomperc();
6     if (Pr < ζ(t)) then
7        initialize individual x randomly;
8     else
9        Gaussian_local_search(x,0,ζ(t)).
10    end if
11 }  // end for
```

In the above algorithm, the major computational costs are in line 1 and line 3. Since the line 1 samples K individuals and the line 3 needs do N loop at most, the computational complexity of Algorithm 2 is $O(N)$. Therefore, the ADI can response to the change rapidly.

It should be noted that although both ADI and MLR methods own Gaussian local search and random initialization, ADI is still quite different from MLR. It is because that some related parameters, such as the proportion of diversity introduction, the ratio of global search and local search, the radius of neighborhood search, used in MLR are fixed in advance, whereas those used in ADI can all be adaptively adjusted according to the real environmental change.

IV. Experimental Results and Discussion

This section is devoted to the experimental comparison for investigating the performance of different DI methods. First, we will explain how to generate several DNSGA-II variants. Second, we will use these DNSGA-II variants to solve five dynamic multiobjective test problems. Finally, the results of the experiments are tabulated and analyzed.

A. DNSGA-II Variants

We briefly choose five common used DI methods: NDI [12], MDI [5], RDI [5], MLR [19], and RR [15], which are used to validate the proposed ADI method. For a fair comparison of various DI methods, we choose DNSGA-II as a basic DEMO algorithm, and integrate them into DNSGA-II to generate different DNSGA-II variants. For example, after integrated the ADI method into DNSGA-II by replacing the original RDI method with the ADI method, we can get a new DNSGA-II variant, which is denoted as DNSGA-II-ADI.

B. Dynamic Multiobjective Test Problems

Five different dynamic multiobjective test problems are applied here to examine the performance of ADI in enhancing the ability of DEMO algorithm. These test problems come from FDA suites [3], [21].

C. Performance Metric

The reversed generational distance ($rGD(t)$) [9] is chosen as performance metric. The metric indicator, $rGD(t)$, can be used to evaluate both the convergence and spread of solution set obtained by some DEMO algorithm. The lower value of $rGD(t)$, the better performance of the algorithm.

$$rGD(t) = \frac{\sum_{i=1}^{|POF^*(t)|} d_i}{|POF^*(t)|}, d_i = \min_{k=1}^{|Q(t)|} \sqrt{\sum_{j=1}^{m} (f_j^{*(i)} - f_j^{(k)})^2} \quad (5)$$

D. DNSGA-II Variants

The experiments are conducted at different severity levels (n_T) and different frequencies (τ_T) so as to study the impact of dynamics in uncertain environments. In particular, a low value of n_T implies that the number of different change is small. Likewise, a larger value of τ_T will result in an increasingly condition of static environments.

The simulations are implemented in C++ on an Intel Core i3 2.93GHz personal computer. Thirty independent runs are performed for each of the test functions to obtain the statistical information. All the algorithms here are implemented using the same real coding scheme. The

simulated binary crossover (SBX) and polynomial mutation are used in DNSGA-II. The experimental parameter settings are listed in Table I. In this table, there are four versions of RDI method. Furthermore, this table includes six different settings for the proportion of DI.

TABLE I. PARAMETER SETTING FOR EXPERIMENT

Parameter	Value
Population size (N)	Two objectives DMOPs: 100; Three objectives DMOPs: 300
Crossover probability	0.9
Distribution index for crossover	10
Mutation probability	$1/n$
Distribution index for mutation	20
proportion of diversity introduction	NDI:0; MDI and RDI1: 0.2; RDI2:0.4; RDI3:0.6; RDI4:0.8; MLR and RR:1.0
Frequency of change (τ_T)	5, 10, 15, 20
Severity of change (n_T)	5, 10, 20

E. Discussion of the Results

All the DNSGA-II variants are ranked based on their $rGD(t)$ results under every dynamic change combination (τ_T, n_T). The sorting rule is as follows: between two variants with different mean value of $rGD(t)$, we prefer the variant with the lower (better) mean. Otherwise, if both variants have the same mean value, then we prefer the one with the smaller variance. Furthermore, we calculated the scores of all the DNSGA-II variants based on their rank values. Given an algorithm A, its scores can be calculated as follows:

$$\text{score(A)} = \sum_{i=1}^{num} (10 - rank_i(\text{A})) \quad (6)$$

where num is the number of different dynamic change combinations, and $rank_i$(A) is the rank value of the algorithm A under i-th change combination. The higher score is the algorithm, and its performance is more excellent.

Table II is experiment results of nine different DNSGA-II variants on solving FDA1 problem. For the sake of convenience, DNSGA-II is also denoted as DNSGA2 in this paper. The last variant adopts adaptive DI method, while the others use fixed proportion of DI. Clearly, this table shows that DNSGA-II-ADI has the best rank value in the most of dynamic change combinations, which indicates that DNSGA-II-ADI is robust to different dynamic changes. In addition, DNSGA-II-ADI has the best performance since it has the highest score. It is interest to note that four different DNSGA-II-RDI variations have close scores, no matter they use different proportion of diversity introduction. Finally, two variants using reinitialization method come in the eighth and ninth positions. In particular, DNSGA-II-RR has the lowest score, since the old useful information is lost after the random restart.

Tables III - VI are the results of different variants on solving the other FDA test problems. DNSGA-II-ADI still has the best score on these FDA problems, except FDA3mod. Although the score of DNSGA-II-ADI is 43 in FDA3mod problem, the gap between it and the best score (46) is not significant.

Finally, for every DNSGA-II variant, we sum up its scores in the above five test problems. The total scores of different DNSGA-II variants are compared in the Fig. 2. This bar chart clearly shows that DNSGA-II-ADI has the highest total scores, which significantly surpasses the results of other variants. Therefore, we can draw a conclusion that the proposed ADI method can better enhance the performance of DNSGA-II, compared with the methods using fixed proportion of DI.

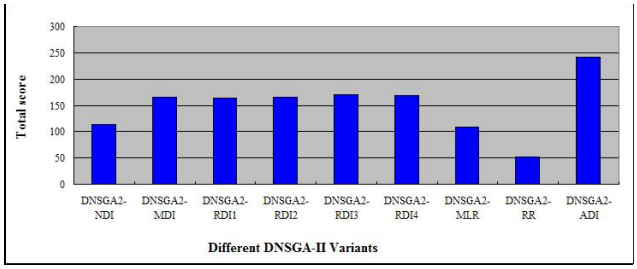

Fig. 2. Total scores of different DNSGA-II variants.

F. Analysis of Adaptive Proportion

In this section we will analyze whether the proportion of diversity introduced in the DNSGA-II-ADI is adaptive to the real dynamic change. Here we take FDA1 as an example, and the dynamic change combination is (20,5). The time-varying POS(t) of FDA1 is shown in Fig. 3. For convenience, variations on only the first two decision variables are shown for 20 time steps.

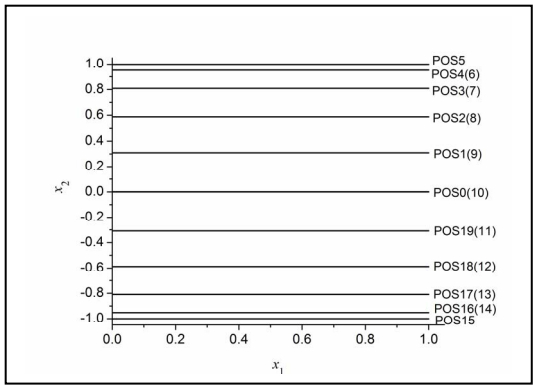

Fig. 3. Time-varying POS(t) of FDA1

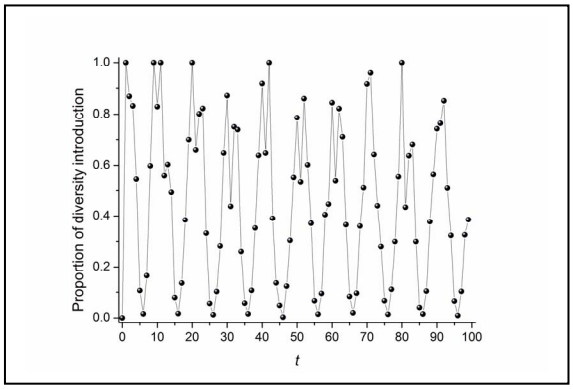

Fig. 4. Adaptive proportion of diversity introduction in the DNSGA-II-ADI algorithm on solving FDA1.

Accordingly, Fig. 4 illustrates the proportion of diversity introduction in the DNSGA-II-ADI algorithm on solving FDA1 in 100 time steps. In the beginning, the population is randomly generated. Consequently, no diversity introduction is needed at the time step 0 in the Fig. 4. When the first dynamic change took place, the new Pareto optimal set jumped to POS1 from POS0 as shown in Fig. 3. It can easily be seen that the gap between POS0 and POS1 is the biggest in 20 time steps. Correspondingly, the proportion of diversity introduction at time step 1 reached the maximum in the Fig. 4. In the next four environmental changes, the gaps between the old and the new POS are decreased gradually. As a result, a continual decline in the proportion of diversity introduction at the related time steps is shown in Fig. 4. By further comparison of these two Figures, it can be seen that the proportion of diversity introduction in the DNSGA-II-ADI is basically consistent with the extent of environmental change. Therefore, we can draw a conclusion that the proposed ADI method can introduce appropriate diversity to the evolving population according to the actual needs of dynamic changes.

V. Conclusions and Future Work

In this paper, we have proposed an adaptive diversity introduction (ADI) method to enhance the performance of dynamic evolutionary multiobjective optimization algorithms in dealing with DMOPs. When an environmental change is detected, the appropriate proportion of diversity introduction is determined by estimating the extent of the environment change. In addition, an adaptive relocation operator is designed to quickly response to the environment change. The main advantages of the proposed ADI method over other diversity introduction methods are as follows.

- Rather than using fixed proportion of diversity introduction, ADI studies the gap between two consecutive POFs by estimating the extent of the new change, and uses this information to obtain an appropriate proportion of diversity introduction.
- Based on the appropriate proportion, an adaptive relocation operator is designed to adapt the evolving population to the needs of the new environment.

The work presented in this paper is preliminary, and there are some possible directions for future work. For example, this paper only investigates ADI for boundary-constrained DMOPs. It is worth to test ADI on more problems with different types of changes, with constraints. If there are some complex constrains to be handled with, the estimation method of environmental change in the ADI maybe need to be modified. In addition, we are planning to combine ADI method with other strategies, such as memory [7], [22] or prediction strategy [8], and investigate hybrid methods for DMOPs in the future.

References

[1] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-objective Problems*. 2nd ed., New York: Springer-Verlag, 2007.

[2] M. Helbig, "Solving dynamic multi-objective optimisation problems using vector evaluated particle swarm optimization", Ph.D. dissertation, Dept. of Computer Science, University of Pretoria, Pretoria, South Africa, 2012.

[3] M. Farina, K. Deb, and P. Amato, "Dynamic multiobjective optimization problems: Test cases, approximations, and applications," *IEEE Trans. on Evolutionary Computation*, vol. 8, no. 5, pp. 425-442, 2004.

[4] C. K. Goh and K. C. Tan, "A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization," *IEEE Trans. on Evolutionary Computation*, vol. 13, no. 1, pp. 103-127, 2009.

[5] K. Deb, U. V. Rao, and S. Karthik, "Dynamic multi-objective optimization and decision-making using modified NSGA-II: A case study on hydro-thermal power scheduling", in *Evolutionary Multi-Criterion Optimization (EMO 2007)*, 2007, vol. LNCS 4403, pp. 803–817.

[6] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II", *IEEE Trans. on Evolutionary Computation*, vol. 6, no.2, pp. 182-197, 2002.

[7] M. Liu and W. H. Zeng, "Memory enhanced dynamic multi-objective evolutionary algorithm based on decomposition", *Ruan Jian Xue Bao/Journal of Software*, vol. 24, no. 7, pp. 1571-1588, 2013.

[8] A. Zhou, Y. Jin, and Q. F. Zhang, "A population prediction strategy for evolutionary dynamic multiobjective optimization", *IEEE Transactions on Cybernetics*, vol. 44, no. 1, pp. 40-53, 2013.

[9] X. D. Li, J. Branke, and M. Kirley, "On performance metrics and particle swarm methods for dynamic multiobjective optimization problems", In *Conf. of the 2007 Congress on Evolutionary Computation (CEC 2007)*, 2007, pp. 576-583.

[10] M. Greeff, and A. Engelbrecht, "Dynamic multi-objective optimisation using PSO", in *Multi-Objective Swarm Intelligent Systems*. N. Nedjah Ed. Berlin Heidelberg: Springer-Verlag, 2010, pp. 105-123.

[11] Z. H. Zhang, "Multiobjective optimization immune algorithm in dynamic environments and its application to greenhouse control," *Applied Soft Computing*, vol. 8, no. 2, pp. 959–971, 2008.

[12] R. H. Shang, L. C. Jiao, M. G. Gong, and W. P. Ma, "An immune algorithm for dynamic multi-objective optimization", *Ruan Jian Xue Bao/Journal of Software*, vol. 18, no. 11, pp. 2700-2711, 2007.

[13] L. Huang, I. H. Suh, and A. Abraham, "Dynamic multi-objective optimization based on membrane computing for control of time-varying unstable plants", *Information Sciences*, vol. 181, no. 11, pp. 2370-2391, 2011.

[14] I. Hatzakis and D. Wallace, "Dynamic multi-objective optimization with evolutionary algorithms: A forward-looking approach", in *Genetic and Evolutionary Computation* Conference (*GECCO'06*), 2006, pp. 1201-1208.

[15] Y. C. Jin and J. Branke, "Evolutionary optimization in uncertain environments - A survey", *IEEE Trans. on Evolutionary Computation*, vol. 9, no. 3, pp. 303-317, 2005.

[16] H. G. Cobb, "An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environments," Naval Res. Lab., Washington, DC, Tech. Rep. AIC-90-001, 1990.

[17] F. Vavak, K. Jukes, and T. C. Fogarty, "Adaptive combustion balancing in multiple burner boiler using a genetic algorithm with variable range of local search," in *Proc. Int. Conf. Genetic Algorithms*, T. Back, Ed., 1997, pp. 719–726.

[18] Y. G. Woldesenbet and G. G. Yen, "Dynamic evolutionary algorithm with variable relocation", *IEEE Trans. on Evolutionary Computation*, vol. 13, no. 3, pp. 500-513, 2009.

[19] Y. Wang and B. Li, "Multi-strategy ensemble evolutionary algorithm for dynamic multi-objective optimization," *Memetic Computing*, vol. 2, no. 1, pp. 3–24, 2009.

[20] M. Helbig and A. P. Engelbrecht, "Dynamic multi-objective optimization using PSO", in *Metaheuristics for Dynamic Optimization,* E. Alba *et al*. Eds. Berlin Heidelberg: Springer, 2013, pp. 147-188.

[21] M. Cámara Sola, "Parallel processing for dynamic multi-objective optimization", Ph.D. dissertation, Dept. of Computer Architecture and Computer Technology, University of Granada, Granada, Spain, 2010.

[22] S. X. Yang and X. Yao, "Population-Based incremental learning with associative memory for dynamic environments", *IEEE Trans. on Evolutionary Computation*, vol. 12, no. 5, pp. 542-561, 2008.

TABLE II. EXPERIMENT RESULTS OF DIFFERENT DNSGA-II VARIANTS ON FDA1 PROBLEM

τ_T, n_T	Statistic	DNSGA2 -NDI	DNSGA2 -MDI	DNSGA2 -RDI1	DNSGA2 -RDI2	DNSGA2 -RDI3	DNSGA2 -RDI4	DNSGA2 -MLR	DNSGA2 -RR	DNSGA2 -ADI
5,20	mean	8.23E-01†	4.92E-01†	5.43E-01†	5.63E-01†	4.97E-01†	5.22E-01†	2.24E+00†	2.75E+00†	2.99E-01
	variance	5.24E-01	1.85E-01	1.75E-01	1.53E-01	1.07E-01	9.35E-02	2.81E+00	3.15E+00	6.66E-02
	rank	7	2	5	6	3	4	8	9	1
10,20	mean	8.32E-02	7.98E-02‡	9.02E-02†	9.33E-02†	1.04E-01†	1.10E-01†	5.00E-01†	1.19+00†	8.56E-02
	variance	6.07E-03	5.89E-03	6.93E-03	7.19E-03	8.11E-03	8.46E-03	1.69E-01	5.75E-01	5.39E-03
	rank	2	1	4	5	6	7	8	9	3
10,10	mean	5.68E-01†	4.41E-01†	3.86E-01†	4.22E-01†	3.99E-01†	3.61E-01†	7.88E-01†	1.20E+00†	2.28-01
	variance	2.66E-01	1.53E-01	9.60E-02	1.14E-01	8.16E-02	6.31E-02	3.61E-01	5.87E-01	3.90E-02
	rank	7	6	3	5	4	2	8	9	1
15,10	mean	1.25E-01	1.30E-01	1.33E-01	1.44E-01†	1.63E-01†	1.52E-01†	2.26E-01†	6.37E-01†	1.14E-01
	variance	9.62E-03	1.05E-02	1.12E-02	1.40E-02	1.76E-02	1.55E-02	2.59E-02	1.56E-01	8.60E-03
	rank	2	3	4	5	7	6	8	9	1
15,5	mean	1.60E+00†	1.13E+00†	4.94E-01†	4.08E-01†	4.19E-01†	3.77E-01†	4.23E-01†	6.21E-01†	2.64E-01
	variance	2.08E+00	1.03E+00	1.18E-01	7.21E-02	7.94E-02	5.92E-02	8.84E-02	1.56E-01	3.58E-02
	rank	9	8	6	3	4	2	5	7	1
20,5	mean	4.70E-01†	3.72E-01†	2.82E-01†	2.50E-01†	2.33E-01†	2.40E-01†	4.70E-01†	3.57E-01†	1.45E-01
	variance	1.65E-01	1.04E-01	4.76E-02	3.46E-02	2.42E-02	2.77E-02	1.65E-01	5.04E-02	1.08E-02
	rank	8	7	5	4	2	3	8	6	1
	score	25	33	33	32	34	36	15	11	52

† indicates DNSGA2-ADI performs better than the other DNSGA2 Variant with 95% certainty by t-test. ‡ means that corresponding DNSGA2 variant is better than DNSGA2-ADI.

TABLE III. EXPERIMENT RESULTS OF DIFFERENT DNSGA-II VARIANTS ON FDA2 PROBLEM

τ_T , n_T	Statistic	DNSGA2 -NDI	DNSGA2 -MDI	DNSGA2 -RDI1	DNSGA2 -RDI2	DNSGA2 -RDI3	DNSGA2 -RDI4	DNSGA2 -MLR	DNSGA2 -RR	DNSGA2 -ADI
5,20	mean	4.83E-01	4.45E-01	3.89E-01‡	5.10E-01†	4.30E-01‡	5.12E-01†	1.12E+00†	1.25E+00†	4.71E-01
	variance	1.10E-01	1.08E-01	1.06E-01	1.07E-01	9.58E-02	1.04E-01	2.72E-01	3.48E-01	1.28E-01
	rank	5	3	1	6	2	7	8	9	4
10,20	mean	3.81E-01	3.72E-01	3.75E-01	3.26E-01‡	3.66E-01	4.05E-01†	8.86E-01†	9.66E-01†	3.62E-01
	variance	1.08E-01	1.06E-01	1.04E-01	9.78E-02	1.01E-01	1.01E-01	7.11E-02	1.31E-01	1.06E-01
	rank	6	4	5	1	3	7	8	9	2
10,10	mean	2.92E-01	2.81E-01	3.15E-01	2.51E-01‡	3.60E-01†	3.20E-01†	9.04E-01†	9.49E-01†	2.87E-01
	variance	8.08E-02	7.87E-02	8.58E-02	5.90E-02	9.46E-02	7.51E-02	8.38E-02	1.16E-01	7.96E-02
	rank	4	2	5	1	7	6	8	9	3
15,10	mean	2.51E-01†	2.55E-01†	2.56E-01†	2.80E-01†	2.44E-01†	2.49E-01†	8.21E-01†	8.48E-01†	1.96E-01
	variance	7.86E-02	8.61E-02	8.21E-02	9.36E-02	7.36E-02	7.48E-02	4.67E-02	4.96E-02	5.60E-02
	rank	4	5	6	7	2	3	8	9	1
15,5	mean	2.35E-01	2.31E-01	2.42E-01	2.54E-01†	2.50E-01	2.90E-01†	7.94E-01†	8.71E-01†	2.24E-01
	variance	5.15E-02	5.21E-02	4.46E-02	4.73E-02	4.93E-02	6.20E-02	5.18E-02	4.85E-02	4.13E-02
	rank	3	2	4	6	5	7	8	9	1
20,5	mean	1.80E-01	1.73E-01	1.84E-01	1.94E-01	2.00E-01	2.06E-01	7.92E-01†	8.24E-01†	1.84E-01
	variance	4.13E-02	4.12E-02	3.75E-02	4.00E-02	4.19E-02	4.24E-02	3.24E-02	3.09E-02	4.01E-02
	rank	2	1	3	5	6	7	8	9	4
	score	36	43	36	34	35	23	12	6	45

TABLE IV. EXPERIMENT RESULTS OF DIFFERENT DNSGA-II VARIANTS ON FDA3MOD PROBLEM

τ_T , n_T	Statistic	DNSGA2 -NDI	DNSGA2 -MDI	DNSGA2 -RDI1	DNSGA2 -RDI2	DNSGA2 -RDI3	DNSGA2 -RDI4	DNSGA2 -MLR	DNSGA2 -RR	DNSGA2 -ADI
5,20	mean	9.45E-01‡	8.41E-01‡	9.47E-01‡	9.59E-01‡	1.03E+00	1.11E+00	6.46E+00†	7.03E+00†	1.05E+00
	variance	1.27E+00	8.73E-01	6.75E-01	5.68E-01	5.49E-01	6.00E-01	1.69E+01	1.76E+01	1.87E+00
	rank	2	1	3	4	5	7	8	9	6
10,20	mean	2.63E-01	2.46E-01‡	2.65E-01	2.96E-01	3.15E-01†	3.21E-01†	2.93E+00†	4.02E+00†	2.69E-01
	variance	9.60E-02	9.65E-02	9.76E-02	1.08E-01	1.12E-01	1.07E-01	5.25E+00	7.51E+00	7.66E-02
	rank	2	1	3	5	6	7	8	9	4
10,10	mean	8.26E-01	7.44E-01	7.21E-01‡	8.69E-01†	1.00E+00†	9.16E-01†	3.46E+00†	4.08E+00†	8.26E-01
	variance	8.91E-01	7.20E-01	4.50E-01	5.20E-01	6.64E-01	4.86E-01	6.32E+00	7.09E+00	1.03E+00
	rank	3	2	1	5	7	6	8	9	4
15,10	mean	3.64E-01†	3.41E-01	3.46E-01	4.05E-01†	4.79E-01†	4.13E-01†	1.46E+00†	2.17E+00†	3.32E-01
	variance	1.45E-01	1.26E-01	1.28E-01	1.70E-01	2.08E-01	1.47E-01	1.73E+00	2.47E+00	1.45E-01
	rank	4	2	3	5	7	6	8	9	1
15,5	mean	1.18E+00†	1.05E+00†	1.04E+00†	1.15E+00†	1.22E+00†	1.10E+00†	1.78E+00†	2.28E+00†	7.83E-01
	variance	2.54E+00	1.88E+00	6.23E-01	8.72E-01	9.45E-01	6.98E-01	2.16E+00	2.74E+00	4.14E-01
	rank	6	3	2	5	7	4	8	9	1
20,5	mean	9.65E-01†	7.67E-01†	6.68E-01†	7.74E-01†	7.17E-01†	6.73E-01†	1.08E+00†	1.33E+00†	4.98E-01
	variance	1.49E+00	8.13E-01	2.78E-01	3.82E-01	3.27E-01	2.89E-01	8.33E-01	1.11E+00	1.93E-01
	rank	7	5	2	6	4	3	8	9	1
	score	36	46	46	30	24	27	12	6	43

TABLE V. EXPERIMENT RESULTS OF DIFFERENT DNSGA-II VARIANTS ON FDA4 PROBLEM

τ_T , n_T	Statistic	DNSGA2 -NDI	DNSGA2 -MDI	DNSGA2 -RDI1	DNSGA2 -RDI2	DNSGA2 -RDI3	DNSGA2 -RDI4	DNSGA2 -MLR	DNSGA2 -RR	DNSGA2 -ADI
5,20	mean	3.72E-01†	2.69E-01†	1.89E-01	1.69E-01	1.60E-01	1.65E-01	3.05E-01†	3.75E-01†	1.72E-01
	variance	8.84E-02	3.90E-02	1.21E-02	7.98E-03	7.03E-03	7.45E-03	1.43E-02	2.52E-02	1.03E-02
	rank	8	6	5	3	1	2	7	9	4
10,20	mean	9.70E-02†	8.73E-02†	9.22E-02†	9.01E-02†	9.01E-02†	8.86E-02†	1.08E-01†	1.99E-01†	7.84E-02
	variance	2.87E-03	2.03E-03	2.49E-03	2.06E-03	2.06E-03	1.81E-03	1.86E-03	5.96E-03	1.44E-03
	rank	7	2	6	3	3	5	8	9	1
10,10	mean	3.29E-01†	2.46E-01†	1.46E-01†	1.36E-01	1.32E-01	1.31E-01	1.45E-01†	2.04E-01†	1.24E-01
	variance	1.02E-01	4.15E-02	5.91E-03	6.18E-03	4.13E-03	3.77E-03	2.44E-03	6.49E-03	3.82E-03
	rank	9	8	6	4	3	2	5	7	1
15,10	mean	1.48E-01†	1.21E-01†	9.99E-02†	9.06E-02†	8.88E-02†	8.75E-02	7.83E-02‡	1.24E-01†	8.51E-02
	variance	1.35E-02	5.79E-03	2.32E-03	1.71E-03	1.53E-03	1.47E-03	5.50E-04	2.49E-03	1.49E-03
	rank	9	7	6	5	4	3	1	8	2
15,5	mean	4.49E-01†	3.34E-01†	1.38E-01†	1.26E-01	1.21E-01	1.17E-01	1.04E-01	1.27E-01	1.09E-01
	variance	3.23E-01	1.02E-01	7.63E-03	6.02E-03	4.82E-03	4.10E-03	2.33E-03	2.18E-03	3.44E-03
	rank	9	8	7	5	4	3	1	6	2
20,5	mean	2.99E-01†	2.11E-01†	1.02E-01†	8.94E-02†	8.63E-02†	8.62E-02†	7.03E-02‡	8.22E-02†	7.59E-02
	variance	9.84E-02	3.95E-02	4.20E-03	2.81E-03	2.20E-03	2.08E-03	7.32E-04	8.97E-04	1.28E-03
	rank	9	8	7	6	5	4	1	3	2
	score	6	21	23	34	40	41	37	18	48

TABLE VI. EXPERIMENT RESULTS OF DIFFERENT DNSGA-II VARIANTS ON FDA5 PROBLEM

τ_T , n_T	Statistic	DNSGA2 -NDI	DNSGA2 -MDI	DNSGA2 -RDI1	DNSGA2 -RDI2	DNSGA2 -RDI3	DNSGA2 -RDI4	DNSGA2 -MLR	DNSGA2 -RR	DNSGA2 -ADI
5,20	mean	4.24E-01†	2.96E-01†	2.30E-01†	2.06E-01	2.12E-01	2.24E-01†	6.23E-01†	6.71E-01†	1.93E-01
	variance	1.16E-01	3.94E-02	1.14E-02	8.92E-03	7.88E-03	7.62E-03	1.65E-01	1.66E-01	8.51E-03
	rank	7	6	5	2	3	4	8	9	1
10,20	mean	1.28E-01	1.18E-01	1.23E-01	1.19E-01	1.19E-01	1.19E-01	2.11E-01†	3.49E-01†	1.10E-01
	variance	2.20E-03	1.59E-03	1.75E-03	1.41-03	1.45E-03	1.34E-03	1.93E-02	5.37E-02	1.24E-03
	rank	7	2	6	4	5	3	8	9	1
10,10	mean	3.61E-01†	2.85E-01†	1.88E-01†	1.73E-01†	1.66E-01	1.62E-01	2.32E-01†	3.47E-01†	1.45E-01
	variance	9.79E-02	3.99E-02	6.49E-03	4.71E-03	3.88E-03	3.09E-03	1.37E-02	4.48E-02	2.83E-03
	rank	9	7	5	4	3	2	6	8	1
15,10	mean	1.71E-01†	1.51E-01†	1.26E-01	1.20E-01	1.17E-01	1.16E-01	1.08E-01	1.89E-01†	1.06E-01
	variance	9.15E-03	4.70E-03	1.44E-03	1.37E-03	9.34E-04	9.50E-04	5.72E-04	9.04E-03	6.36E-04
	rank	8	7	6	5	4	3	2	9	1
15,5	mean	4.89E-01†	3.84E-01†	1.70E-01†	1.57E-01†	1.53E-01	1.48E-01	1.44E-01	2.01E-01†	1.30E-01
	variance	3.31E-01	1.34E-01	4.24E-03	3.84E-03	3.57E-03	3.32E-03	2.14E-03	1.85E-02	1.91E-03
	rank	9	8	6	5	4	3	2	7	1
20,5	mean	3.23E-01†	2.58E-01†	1.25E-01†	1.18E-01†	1.16E-01†	1.13E-01†	1.02-01	1.35E-01†	9.84E-02
	variance	1.02E-01	4.48E-02	1.67E-03	1.25E-03	1.49E-03	1.24E-03	6.04E-04	1.18E-02	5.10E-04
	rank	9	8	6	5	4	3	2	7	1
	score	11	22	26	35	37	42	32	11	54