Prediction-based Population Re-initialization for Evolutionary Dynamic Multi-objective Optimization

Aimin Zhou, Yaochu Jin, Qingfu Zhang, Bernhard Sendhoff, Edward Tsang

2007

Preprint:

This is an accepted article published in The Fourth International Conference on Evolutionary Multi-Criterion Optimization. The final authenticated version is available online at: https://doi.org/[DOI not available]

To appear: The 4th Int. Conference on Evolutionary Multi-Criterion Optimization.
LNCS, Springer, 2007

Prediction-based Population Re-initialization for Evolutionary Dynamic Multi-objective Optimization

Aimin Zhou[1], Yaochu Jin[2], Qingfu Zhang[1], Bernhard Sendhoff[2], and Edward Tsang[1]

[1] Department of Computer Science, University of Essex, Colchester, CO4 3SQ, U. K.
[2] Honda Research Institute Europe, Carl-Legien-Str. 30, 63073, Offenbach, Germany

Abstract. Optimization in changing environment is a challenging task, especially when multiple objectives are to be optimized simultaneously. The basic idea to address dynamic optimization problems is to utilize history information to guide future search. In this paper, two strategies for population re-initialization are introduced when a change in the environment is detected. The first strategy is to predict the new location of individuals from the location changes that have occurred in the history. The current population is then partially or completely replaced by the new individuals generated based on prediction. The second strategy is to perturb the current population with a Gaussian noise whose variance is estimated according to previous changes. The prediction based population re-initialization strategies, together with the random re-initialization method, are then compared on two bi-objective test problems. Conclusions on the different re-initialization strategies are drawn based on the preliminary empirical results.

1 Introduction

In this paper, we consider the following continuous dynamic multi-objective optimization problems (DMOP):

$$\begin{aligned} &\text{minimize } \boldsymbol{F}(x,t) = (f_1(x,t), f_2(x,t), \ldots, f_m(x,t))^T, \\ &\text{subject to } \qquad\qquad x \in X, \end{aligned} \tag{1}$$

where $t = 0, 1, 2, \cdots$ represents time, $x = (x_1, \ldots, x_n)^T \in R^n$ is the decision variable vector and $X \subset R^n$ is the decision space. R^m is the objective space. $\boldsymbol{F} : (X, t) \rightarrow R^m$ consists of m real-valued objective functions $f_i(x,t)$ $(i = 1, 2, \ldots, m)$, each of which is continuous with respect to x over X. The Pareto front (PF) in the objective space and the Pareto set (PS) in the decision space change over time. The task of a dynamic multi-objective optimization algorithm is to trace the movement of the PF and PS with reasonable computational costs.

Inspired by the success of evolutionary algorithms on dynamic scalar optimization problems [1–3], research work on evolutionary dynamic multi-objective optimization (EDMO) has very recently been conducted by several researchers. In the following, we briefly review the current work on EDMO:

i **Test Problems:** Benchmarks are important for developing and testing algorithms for solving DMOPs. In [4], Jin and Sendhoff proposed a method for constructing

dynamic multi-objective test problems by aggregating different objectives of existing stationary multi-objective problems and changing the weights dynamically. Test problems in [5] and [6] are created by adding time-varying terms to the objectives in stationary MOP test problems.

ii **Algorithms:** Several attempts for solving DMOPs by evolutionary algorithms have been reported recently. Stationary multi-objective evolutionary algorithms such as NSGA-II [7], SPEA2 [8], MSOPS [9] and OMOEA-II [10] have been directly applied to DMOPs [6, 11]. A few evolutionary algorithms for solving dynamic single objective optimization problems have also been extended to the case of multi-objective problems [12]. Several strategies have been proposed to add to stationary multi-objective evolutionary algorithms problems for tracking the movement of the PS [13–15].

iii **Performance Indicators:** It is hard to measure the performance of algorithms for DMOPs for the following reasons. Firstly, the measure must be able to evaluate the quality of approximation of a solution set, which itself is not trivial. Secondly, the PS is changing over time. It is natural to draw the PF for stationary multi-objective optimization, but it is no longer practical to plot the changing PFs in dynamic environment. In [12], two convergence performance measures have been suggested. In [6], the generational distance with time was plotted to show the convergence. In addition, a distribution indicator, known as the PL-metric, has also been introduced [6].

Arguably, diversity maintenance is essential in dynamic scalar objective evolutionary optimization algorithms. It is however interesting to note that in multi-objective evolutionary algorithms, the diversity of population is inherently maintained due to the multi-objective nature. Thus, it is probably of greater importance to ensure that the population is able to follow the moving PF more quickly. To this end, a correct guess of the new location of the changed PS is of great interest.

In this paper, we study how to generate an initial population close to a changed PF when a change is detected in a dynamic environment. Inspired by [14, 15], we build prediction models to predict the location of the new PS based on the information collected from the previous search. Different to [14, 15] where only the new locations of two anchor points and the Closest-To-Ideal point are predicted, we predict the new locations of a number of Pareto solutions in the decision space once a change is detected. Individuals in the initial population for the changed problem are generated around these predicted points. In such a way, the changed PS and PF can be found more effectively by the algorithm.

Four methods for re-initialization have been studied and compared in this paper. They are 1) Random re-initialization method in which the initial populations are randomly generated in the search space; 2) Variation method in which the individuals in the current population are perturbed using a Gaussian noise whose variance is determined by changes in the history; 3) Prediction method in which the new trial solutions are generated around predicted locations; and 4) A naive hybrid method, in which half of population is generated by strategy 2 and half is created by strategy 3.

The remainder of the paper is organized as follows. In Section 2, the four re-initialization methods are described in detail. Section 3 presents the two test functions

and performance indicators used in this paper. The empirical results are shown in Section 4. The paper is concluded with Section 5.

2 Re-Initialization Strategies for Dynamic Multi-objective Optimization

2.1 The Algorithm Framework

In this paper, we concentrate on population re-initialization when a change in the environment is detected. Other genetic operators, such as offspring generation and selection, are based on a model-based approach for stationary multi-objective problems that we proposed recently [16–19]. The framework of the dynamic multi-objective evolutionary algorithm with predicted re-initialization (DMEA/PRI) suggested in this paper is described as follows.

DMEA/PRI

Step 0 Set generation index $\tau := 1$, time window $t := 1$, initialize population P_τ.
Step 1 If a change is detected,
  1.1 Store P_τ in memory: Set $Q_t = P_\tau$.
  1.2 Re-initialization: generate an initial population $P_{\tau+1}$ based on information from Q_k, $k = t, t-1, \cdots$.
  1.3 $t := t + 1$.
Step 2 If no change is detected, create the offspring population and do selection:
  2.1 Create an offspring population P^* from P_τ using an offspring generator for stationary optimization.
  2.2 Select $P_{\tau+1}$ from $P^* \cup P_\tau$.
Step 3 If the stop criterion is met, stop; else set $\tau := \tau + 1$ and go to **Step 1**.

We assume that inside a time window, there is no change in the environment and thus the dynamic optimization problem can be considered as a stationary problem. In the above framework, the algorithm works as an algorithm for solving stationary multi-objective optimization without **Step 1**. Details on **Step 1** will be discussed in the following subsection.

2.2 Prediction-based Population Re-initialization

As discussed, diversity is maintained inherently in multi-objective optimization. Thus, we concentrate ourselves on faster convergence to the new PF when a change is detected in the environment by predicting the new locations of the Pareto optimal solutions using historical information. We assume that the recorded solutions in the previous time windows when a change is detected, i.e., $Q_t, \cdots, Q_1$, can provide information for predicting the new location of the PS PS_{t+1} at time window $t+1$. We further assume that the location of PS_{t+1} is a function of the locations $Q_t, \cdots, Q_1$:

$$Q_{t+1} = F(Q_1, \cdots, Q_t, t),$$

where Q_{t+1} denotes the new location of the PS for time window $t+1$.

The problem now becomes how to use the historical information ($Q_1, \cdots, Q_t$) to generate new individuals as the initial population for time window $t+1$. In practice, function $F(\cdot)$ is not known and must be estimated using a certain technique. In the following, we discuss how to generate initial solutions for time window $t+1$.

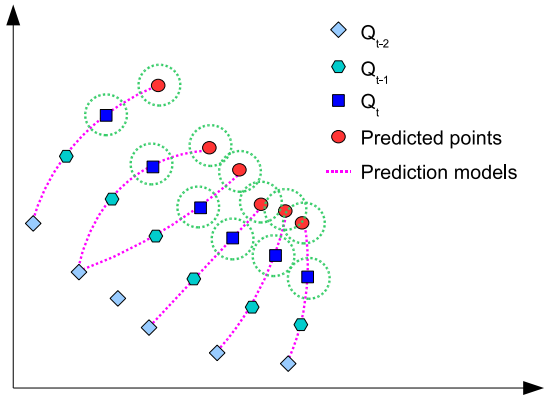

Fig. 1. Illustration of creating an initial population at the beginning of a time window (in decision space).

Prediction Model Suppose that $x_1, x_2, \ldots, x_t$, $x_i \in Q_i, i = 1, \cdots, t$ are a series of points (n-dimensional vectors) in the decision space that describes the movements of the PS, a generic model to predict the location of the initial individuals for the $(t+1)$-th time window can be formulated as follows:

$$x_{t+1} = F(x_t, x_{t-1}, \cdots, x_{t-K+1}, t), \tag{2}$$

where K represents the number of the previous time windows that x_{t+1} is dependent on in the prediction model. An example with $K = 3$ is illustrated in Fig. 1.

Any time series models [20] can be used for modeling F in (2). The major problem in making a prediction is that it is very difficult to identify the relationship between the stored solutions in $Q_1, \cdots, Q_t$ to build a time series. In this paper, we adopted a heuristic approach to identifying such time series. For a point $x_t \in Q_t$, its parent location in the previous time window can be defined as the nearest point in Q_{t-1}, i.e.,

$$x_{t-1} = \arg \min_{y \in Q_{t-1}} ||y - x_t||_2.$$

Once a time series is identified for each individual in the population, any linear or nonlinear prediction model can be used to predict the location of the individual for the next time window. In this paper, the following simple linear model is adopted:

$$x_{t+1} = F(x_t, x_{t-1}) = x_t + (x_t - x_{t-1}). \tag{3}$$

Variation with a "Predicted" Noise The assumption that the movement of the PS can be described by a time series might be too strict. To improve the chance of the initial population to cover the PS in the new time window, a "predicted" Gaussian noise can be added to the current population and/or predicted locations. The standard deviation of the noise is estimated by looking at the changes occurred before:

$$\varepsilon \sim N(0, I\delta), \tag{4}$$

where I is an identity matrix and δ is the standard deviation, which is defined by

$$\delta^2 = \frac{1}{4n}||x_t - x_{t-1}||_2^2,$$

where n is the number of decision vector. See the example in Fig. 1.

Re-initialization Methods In the following, we describe the four methods for re-initializing population that we will empirically study in this paper.

i **Random (RND) Method** All new solutions are randomly initialized in the search space:

$$x = \text{rand}(x^l, x^u),$$

where rand(x^l, x^u) returns a random vector within the lower boundary x^l and upper boundary x^u of the search space.
In this restart method, no historical information is used.

ii **Variation (VAR) Method** All new solutions are created by varying the solution in the last time window with a "predicted" Gaussian noise:

$$x = x_t + \varepsilon,$$

where ε is defined in (4).
In this method, only the information in the last time window is used and no models are built. We believe the new solutions should be close to the solutions in the last time window. Hopefully, the new trial solutions can cover the PS of the new time window.

iii **Prediction (PRE) Method** All new solutions are sampled around the predicted locations:

$$x = F(x_t, x_{t-1}) + \varepsilon,$$

where F is defined in (3) and ε is defined in (4).
In this method, the last two time windows are used to predict new trial locations. By considering historical information, we hope the prediction model can capture the moving trend.

iv **Variation and Prediction (V&P) Method** In this strategy, half of initial population for the $(t+1)$-th time window is sampled around the predicted locations and half is created by varying the points in the last time window (the current population when a change is detected). This method can be formulated as:

$$x = \begin{cases} F(x_t, x_{t-1}) + \varepsilon, & \text{if rand}() < 0.5; \\ x_t + \varepsilon & \text{otherwise.} \end{cases}$$

In the above equation, rand() returns a uniformly random number in $[0, 1]$, F is defined in (3) and ε is defined in (4).

By hybridizing VAR method and PRE method, both historical location information and prediction models are used in re-initializing population.

3 Experimental Setup

3.1 Benchmark Problems

Two test problems are used in our simulation studies. The first one is the FDA1 [12], which is defined as follows:

$$\begin{cases} f_1(x,t) = x_1 \\ f_2(x,t) = 1 - \sqrt{f_1/g} \\ g(x,t) = 1 + \sum_{i=2}^{n} (x_i - G(t))^2 \\ G(t) = \sin(0.5\pi t) \\ x \in [0,1] \times [-1,1]^{n-1}, t = \frac{1}{n_T} \lfloor \frac{\tau}{\tau_T} \rfloor \end{cases}, \tag{5}$$

where τ is the generation counter, τ_T is the number of generations in time window t, and n_T controls the distance between two consecutive PSs. In fact, τ_T and n_T represent the *frequency of change* and *severity of change* respectively.

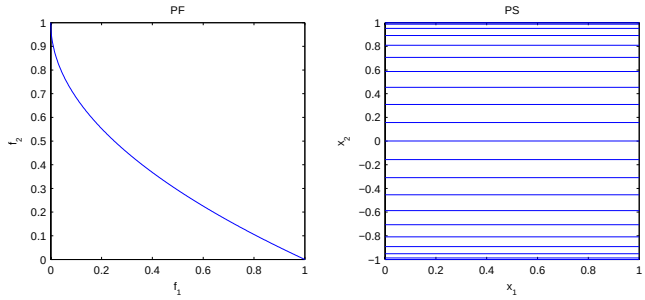

Fig. 2. Illustration of Pareto front and Pareto sets of FDA1 with $n_T = 10$.

As shown in Fig. 2, the PSs of FDA1 are line segments parallel to coordinates, and the PFs are convex and remains unchanged. As suggested in [21], a linear linkage between decision variables may mislead multi-objective evolutionary algorithms. To study the performance of our algorithms on problems with variable linkages, we modify the above FDA1 by using the method proposed in [21]. The modified test problem,

which is named ZJZ, is as follows:

$$\begin{cases} f_1(x,t) = x_1 \\ f_2(x,t) = 1 - (f_1/g)^{H(t)} \\ g(x,t) = 1 + \sum_{i=2}^{n}(x_i + G(t) - x_1^{H(t)})^2 \\ H(t) = 1.5 + G(t) \\ G(t) = \sin(0.5\pi t) \\ x \in [0,1] \times [-1,2]^{n-1}, t = \frac{1}{n_T}\lfloor \frac{\tau}{\tau_T} \rfloor \end{cases} \tag{6}$$

In ZJZ, both the PF and the PS are changing and there are nonlinear linkages between the decision variables. The PFs and PSs are illustrated in Fig. 3.

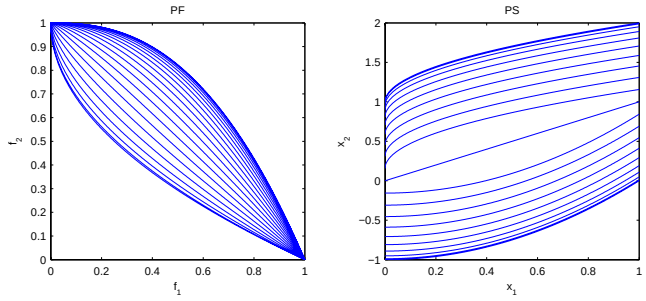

Fig. 3. Illustration of Pareto fronts and Pareto sets of ZJZ with $n_T = 10$.

3.2 Performance Indicators

It is not trivial to assess the performance of evolutionary algorithms for solving dynamic multi-objective optimization problems. Let $M(P_t)$ measure the performance (the smaller, the better) of population P_t at generation t, it is natural to plot performance indicator M against time, as shown in Fig. 4(a). From the plot, we can observe the performance at any given time or the trend within a longer time period. When comparing two algorithms, we can draw a conclusion that a method outperforms another one with respect to this performance indicator if there is no intersections between two M curves as shown in Fig. 4(b). But if there exist intersections, as shown in Fig. 4(c), it is hard to say which one is better. In this case, we may need a scalar value to indicate the quality of an algorithm on a given problem. Inspired by the idea of the offline error metric [1], we can calculate the integral of $M(P_t)$:

$$int(M) = \int_{t=0}^{T} M(P_t)dt,$$

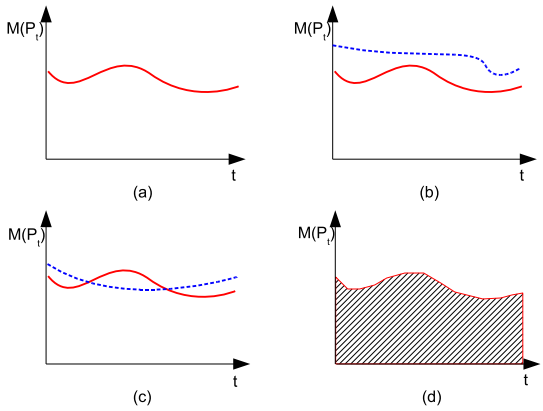

Fig. 4. Illustration of performance indicator against time.

Suppose an algorithm is run N times on a given problem, and P_τ^i is the population at generation τ in the i-th run. Then we use

$$Ave(M(P_\tau)) = \frac{1}{N}\sum_{i=1}^{N} M(P_\tau^i),$$

and

$$Std(M(P_\tau)) = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}[M(P_\tau^i) - Ave(M(P_\tau))]^2}$$

to denote the mean and standard deviation of the performance indicator M at generation τ.

To access the performance of an algorithm fairly, we record the following averages of the means and the deviations over t in our experiments:

$$Ave(M) = \frac{1}{T}\sum_{\tau=1}^{T} Ave(M(P_\tau)), \tag{7}$$

$$Std(M) = \frac{1}{T}\sum_{\tau=1}^{T} Std(M(P_\tau)). \tag{8}$$

In this paper, a distance-based performance indicator $D(P)$ suggested in [18] and the hypervolume difference ($I_H^-(P)$) proposed in [22] are used,

$$D(P) = \frac{1}{|P^*|}\sum_{x\in P^*} ||x - y(x)||_2,$$

where P is an obtained nondominated set, P^* is a reference PF, and $y(x) = \arg\min_{y\in P} ||x - y||_2$.

$$I_H^-(P) = I_H(P^*) - I_H(P),$$

where $I_H(P)$ is the hypervolume [23] of set P.

Both $D(P)$ and $I_H^-(P)$ can measure the approximation quality in convergence and diversity.

4 Experimental Results

4.1 Parameter Settings

In [16–19], we have proposed model-based algorithms to tackle stationary multi-objective optimization problems. By taking into account the regularity property of MOPs, these methods can approximate the PF efficiently. In this paper, the pure model-based method proposed in [18] is used as **Step 2** in the DMEA/PRI framework. The parameter setup of this step is as follows: In local PCA algorithm, the number of clusters is set to 5. The quasi-Newton method for minimizing the error function in the local PCA is run for 300 iterations.

In our experiments, the number of decision variables is 10 for both FDA1 and ZJZ. In all experiments, 60 time changes are performed. The severity of change, n_T is set to be 5 and 10. The frequency of change, τ_T is set to be 5, 10, 15, 20, 30 or 40 generations, thus the time window size will be 500, 1000, 1500, 2000, 3000 and 4000 in terms of fitness evaluations. The results are based on 20 independent runs.

4.2 Results and Discussions

The experimental results on FDA1 and ZJZ are shown in Table 1. Due to space limit, only $D(P)$ over time on FDA1 and I_H^- over time curves are drawn in Figs. 5 and 6. The results on both $D(P)$ and $I_H^-(P)$ are very consistent with each other.

For the FDA1 problem, the PRE method works better than RND and VAR strategies, no matter how wide the time window is. The results of V&P method are very similar to those of PRE method. The VAR method performs poorly, and the RND method does not work at all. These results indicate that for FDA1, the model is able to predict the new locations very well.

For the ZJZ problem, the results are somehow divergent. When the size of time window is 500, the VAR method and V&P method are comparable. When the time window increases, the V&P and PRE strategies outperform others. This can be attributed to the fact that when the width of the time window increases, the approximated sets are closer to the PSs and thus the prediction model can work more effectively. For the same reason, the RND method does not work at all on this FDA1.

Furthermore, we consider the influences of severity of change, i.e., n_T and frequency of change, i.e., the width of time windows. From Fig. 2 and Fig. 3, it is evident that the distances between the consecutive PSs vary over time. This has influences on the performance of the algorithm. The performance is better when the two consecutive PSs are closer to each other. The influence of the distance can be observed in Figs. 5, and 6. The influence will be even more significant when the width of the time window becomes smaller. However, it does not influence the RND method for the following reasons. First, the RND method does not consider the previous results in generating the

Table 1. Statistical Results on FDA1 and ZJZ

Strategies	Size of Time Window (Fitness Evaluations)					
	500	1000	1500	2000	3000	4000
FDA1,$Ave(D) \pm Std(D)$,$n_T = 5$						
RND	1.291 ± 0.345	0.996 ± 0.277	0.809 ± 0.220	0.658 ± 0.183	0.481 ± 0.131	0.364 ± 0.101
VAR	1.920 ± 0.528	0.694 ± 0.290	0.254 ± 0.091	0.149 ± 0.041	0.082 ± 0.015	0.057 ± 0.009
REF	0.280 ± 0.071	0.121 ± 0.028	0.073 ± 0.017	0.050 ± 0.011	0.032 ± 0.007	0.025 ± 0.005
V&P	0.416 ± 0.133	0.141 ± 0.033	0.075 ± 0.016	0.050 ± 0.012	0.032 ± 0.006	0.026 ± 0.005
FDA1,$Ave(D) \pm Std(D)$,$n_T = 10$						
RND	1.290 ± 0.353	1.009 ± 0.281	0.790 ± 0.227	0.657 ± 0.181	0.473 ± 0.128	0.368 ± 0.101
VAR	0.832 ± 0.398	0.173 ± 0.062	0.086 ± 0.024	0.059 ± 0.013	0.037 ± 0.008	0.030 ± 0.005
PRE	0.180 ± 0.041	0.072 ± 0.014	0.049 ± 0.009	0.035 ± 0.007	0.025 ± 0.006	0.020 ± 0.005
V&P	0.196 ± 0.046	0.075 ± 0.018	0.047 ± 0.008	0.037 ± 0.008	0.027 ± 0.005	0.020 ± 0.004
FDA1,$Ave(I_H^-) \pm Std(I_H^-)$,$n_T = 5$						
RND	0.950 ± 0.063	0.842 ± 0.070	0.742 ± 0.073	0.646 ± 0.073	0.494 ± 0.062	0.387 ± 0.052
VAR	0.899 ± 0.054	0.669 ± 0.140	0.372 ± 0.090	0.239 ± 0.050	0.130 ± 0.018	0.092 ± 0.011
PRE	0.399 ± 0.071	0.175 ± 0.028	0.095 ± 0.014	0.064 ± 0.008	0.041 ± 0.004	0.031 ± 0.003
V&P	0.560 ± 0.113	0.205 ± 0.035	0.106 ± 0.015	0.068 ± 0.010	0.043 ± 0.005	0.033 ± 0.004
FDA1,$Ave(I_H^-) \pm Std(I_H^-)$,$n_T = 10$						
RND	0.948 ± 0.064	0.840 ± 0.070	0.739 ± 0.077	0.645 ± 0.072	0.489 ± 0.063	0.387 ± 0.052
VAR	0.724 ± 0.153	0.256 ± 0.074	0.126 ± 0.025	0.083 ± 0.012	0.051 ± 0.006	0.040 ± 0.004
PRE	0.237 ± 0.030	0.075 ± 0.006	0.050 ± 0.004	0.039 ± 0.003	0.028 ± 0.003	0.022 ± 0.003
V&P	0.258 ± 0.040	0.086 ± 0.009	0.054 ± 0.004	0.041 ± 0.004	0.030 ± 0.003	0.024 ± 0.002
ZJZ,$Ave(D) \pm Std(D)$,$n_T = 5$						
RND	2.122 ± 0.583	1.552 ± 0.428	1.201 ± 0.332	0.978 ± 0.273	0.694 ± 0.198	0.524 ± 0.154
VAR	0.580 ± 0.528	0.141 ± 0.032	0.096 ± 0.021	0.075 ± 0.015	0.052 ± 0.010	0.041 ± 0.008
PRE	0.573 ± 0.198	0.124 ± 0.034	0.068 ± 0.014	0.050 ± 0.009	0.033 ± 0.006	0.026 ± 0.004
V&P	0.362 ± 0.110	0.116 ± 0.022	0.070 ± 0.013	0.049 ± 0.010	0.034 ± 0.006	0.027 ± 0.005
ZJZ,$Ave(D) \pm Std(D)$,$n_T = 10$						
RND	2.137 ± 0.588	1.555 ± 0.418	1.197 ± 0.333	0.974 ± 0.269	0.696 ± 0.196	0.528 ± 0.153
VAR	0.176 ± 0.043	0.089 ± 0.019	0.062 ± 0.011	0.045 ± 0.009	0.033 ± 0.007	0.026 ± 0.005
PRE	0.412 ± 0.102	0.090 ± 0.020	0.050 ± 0.010	0.038 ± 0.008	0.025 ± 0.005	0.019 ± 0.003
V&P	0.256 ± 0.083	0.084 ± 0.017	0.052 ± 0.010	0.039 ± 0.008	0.027 ± 0.006	0.020 ± 0.004
ZJZ,$Ave(I_H^-) \pm Std(I_H^-)$,$n_T = 5$						
RND	0.874 ± 0.008	0.848 ± 0.040	0.778 ± 0.060	0.689 ± 0.068	0.523 ± 0.064	0.410 ± 0.056
VAR	0.474 ± 0.118	0.197 ± 0.026	0.137 ± 0.018	0.108 ± 0.015	0.076 ± 0.011	0.059 ± 0.008
PRE	0.606 ± 0.094	0.178 ± 0.028	0.091 ± 0.010	0.064 ± 0.007	0.043 ± 0.004	0.033 ± 0.003
V&P	0.450 ± 0.069	0.166 ± 0.020	0.094 ± 0.011	0.067 ± 0.008	0.045 ± 0.005	0.035 ± 0.004
ZJZ,$Ave(I_H^-) \pm Std(I_H^-)$,$n_T = 10$						
RND	0.870 ± 0.009	0.847 ± 0.034	0.776 ± 0.058	0.684 ± 0.068	0.525 ± 0.066	0.412 ± 0.055
VAR	0.209 ± 0.026	0.104 ± 0.011	0.074 ± 0.008	0.058 ± 0.006	0.041 ± 0.005	0.033 ± 0.003
PRE	0.499 ± 0.059	0.113 ± 0.014	0.056 ± 0.006	0.040 ± 0.004	0.027 ± 0.002	0.021 ± 0.002
V&P	0.295 ± 0.056	0.096 ± 0.009	0.056 ± 0.006	0.041 ± 0.004	0.028 ± 0.003	0.022 ± 0.002

initial population for the next time window. Second, it is easier to tackle a stationary problem whose PS locates at the 'center' of search space than a problem whose PS lies near the boundary. This might also be the reason why the results of the RND method are inconsistent with those of the other strategies.

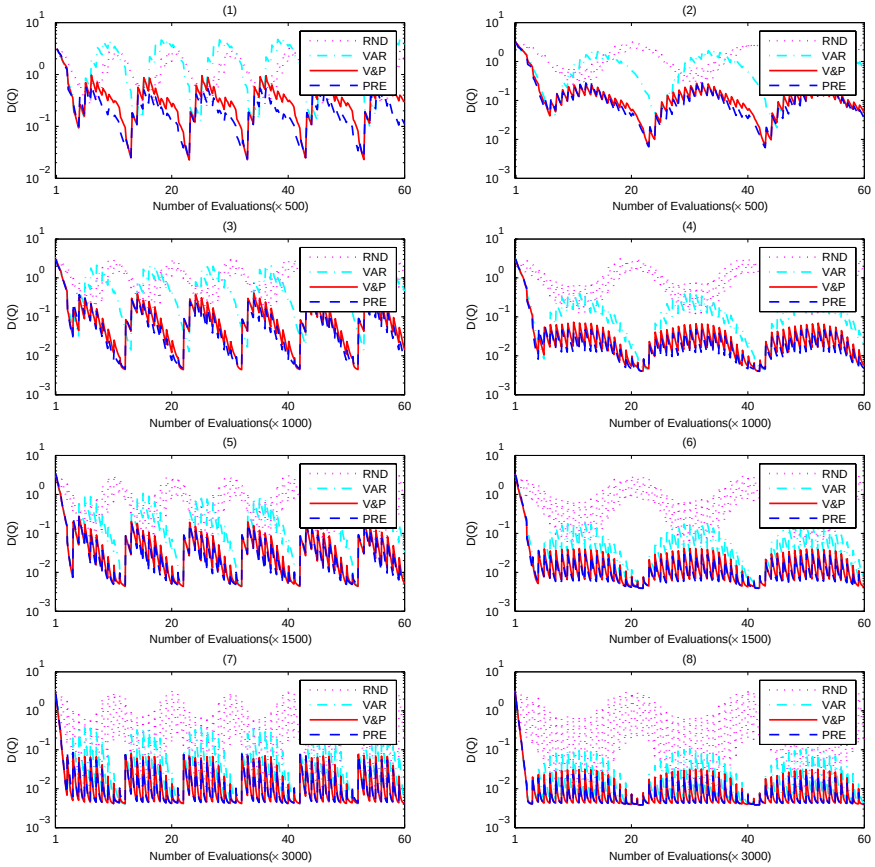

Fig. 5. The evolution of the average $D(Q_t)$ over time among 20 independent runs with 60 time changes for four re-initialization strategies on FDA1, the left column is with $n_T = 5$ and right column with $n_T = 10$.

We can also see that when the width of the time window increases, the performance improves as well. This is quite straightforward because with a wider time window, more function evaluations can be performed and thus more candidate locations can be checked. When the width of the time window is wide enough, the difference between different strategies becomes small, especially for PRE and V&P strategies. The reason is that the difference on the initial populations is covered by the long run in the time window.

Overall, among the four strategies, the RND method is not of practical interest. The PRE method and the VAR method show some advantages over the other two strategies, depending on the characteristics of test problems and the width of the time windows. The V&P method is more likely to perform better in practice.

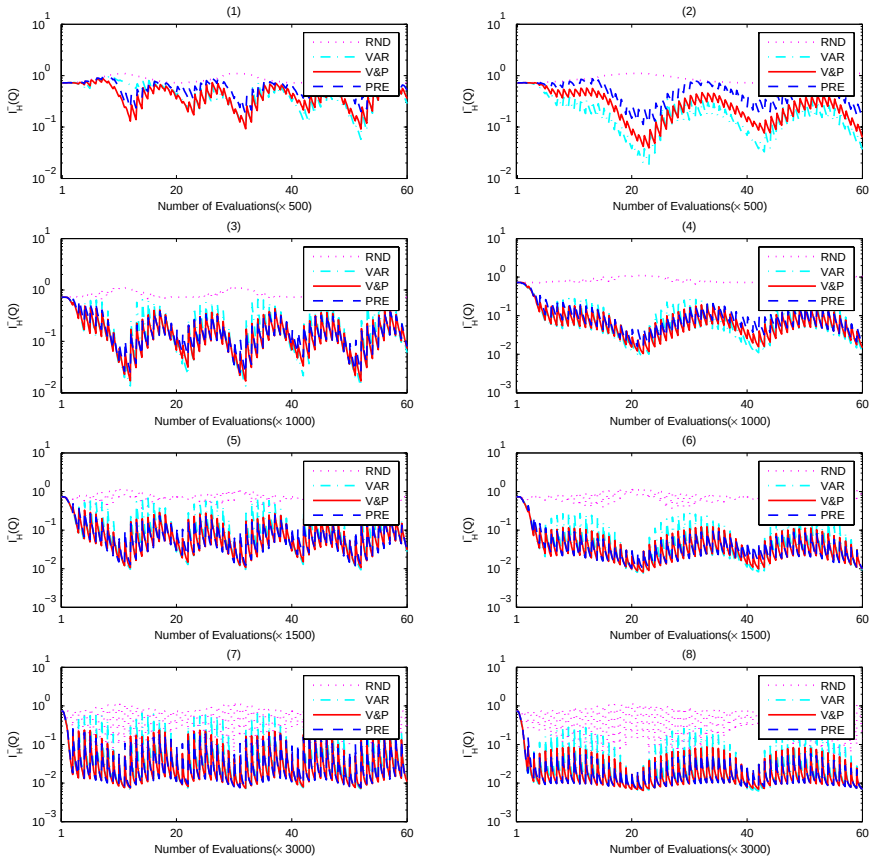

Fig. 6. The evolution of the average $I_H^-(Q_t)$ over time among 20 independent runs with 60 time changes for four re-initialization strategies on ZJZ, the left column is with $n_T = 5$ and right column with $n_T = 10$.

5 Conclusions

In this paper, four strategies for re-initializing populations when a change in environment is detected are empirically studied on two test problems. The experimental results indicate that:

- When a change occurs, strategies which utilize historical information can accelerate the search process. The method that re-initializes a population purely randomly does not work.
- The width of the time window has significant influence on the performance. The wider the time window is, the better the performance. Other characteristics of problems, such as the distance between the neighboring Pareto sets, will also affect the results.
- Different strategies should be applied in different situations. In general, a hybrid method, i.e., the V&P method, might be more recommendable when little information about the problems is known.

To verify the proposed algorithms, a test problem, called ZJZ, has also been introduced. The location of both the Pareto front and the Pareto set of ZJZ changes over time. In addition, there are nonlinear linkages between decision variables. To assess the performance of the proposed algorithms, a performance indicator is suggested as well by combining the offline error measure in dynamic single objective optimization and the performance indicators in multi-objective optimization.

The research on dynamic multi-objective optimization is still in its very infancy and our work presented in this paper is also rather preliminary. Much work remains to be done in the future, for example, analyzing the problem structure of DMOPs, designing dedicated offspring generators and selection strategies by taking into account the problem structure, testing the suggested methods on more benchmarks, and comparing them with other methods.

References

1. Jürgen Branke. *Evolutionary Optimization in Dynamic Environments*, volume 3 of *Genetic Algorithms and Evolutionary Computation*. Kluwer Academic Publishers, 2002.
2. Yaochu Jin and Jürgen Branke. Evolutionary optimization in uncertain environments-a survey. *IEEE Transactions on Evolutionary Computation*, 9(3):303–317, 2005.
3. Shengxiang Yang and Xin Yao. Experimental study on population-based incremental learning algorithms for dynamic optimization problems. *Soft Computing*, 9(11):815–834, 2005.
4. Yaochu Jin and Bernhard Sendhoff. Constructing dynamic test problems using the multi-objective optimization concept. In *Applications of Evolutionary Computing*, volume 3005 of *Lecture Notes in Computer Science*, pages 525–536, Coimbra, Portugal, April 2004. Springer.
5. Marco Farina, Kalyanmoy Deb, and Paolo Amato. Dynamic multiobjective optimization problems: test cases, approximations, and applications. *IEEE Transactions on Evolutionary Computation*, 8(5):425–442, Oct. 2004.
6. Jörn Mehnen, Tobias Wagner, and Günter Rudolph. Evolutionary optimization of dynamic multiobjective functions. Technical Report CI-204/06, University of Dortmund, 2006.
7. Kalyanmoy Deb. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.
8. Eckart Zitzler, Marco Laumanns, and Lothar Thiele. SPEA2: improving the strength Pareto evolutionary algorithm for multiobjective optimization. In *Evolutionary Methods for Design, Optimisation and Control*, pages 95–100, Barcelona, Spain, 2002. CIMNE.

9. Evan J. Hughes. Multiple single objective Pareto sampling. In *Proceedings of the Congress on Evolutionary Computation (CEC 2003)*, pages 2678–2684, Canberra, 8-12 December 2003. IEEE Press.
10. Sanyou Zeng, Shuzhen Yao, Lishan Kang, and Yong Liu. An efficient multi-objective evolutionary algorithm: OMOEA-II. In Carlos A. Coello Coello, Arturo Hernández Aguirre, and Eckart Zitzler, editors, *Third International Conference on Evolutionary Multi-Criterion Optimization (EMO 2005)*, volume 3410 of *Lecture Notes in Computer Science*, pages 108–119, Guanajuato, Mexico, March 2005. Springer.
11. Sanyou Zeng, Guang Chen, Liangfeng Zhang, Hui Shi, Hugo de Garis, Lixin Ding, and Lishan Kang. A dynamic multi-objective evolutionary algorithm based on an orthogonal design. In *Proceedings of the Congress on Evolutionary Computation (CEC 2006)*, pages 2588–2595, Vancouver, BC, Canada, July 2006. IEEE Press.
12. M Farina and P Amato. Linked interpolation-optimization strategies for multicriteria optimization problems. *Soft Computing*, 9(1):54–65, 2005.
13. P. Amato and M. Farina. An ALife-inspired evolutionary algorithm for dynamic multiobjective optimization problems. In *World Conference on Soft Computing*, 2003.
14. Iason Hatzakis and David Wallace. Dynamic multi-objective optimization with evolutionary algorithms: A forward-looking approach. In *Proceedings of Genetic and Evolutionary Computation Conference (GECCO 2006)*, pages 1201–1208, Seattle, Washington, USA, July 2006.
15. Iason Hatzakis and David Wallace. Topology of anticipatory populations for evolutionary dynamic multi-objective optimization. In *11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Portsmouth, Virginia, USA, September 2006.
16. Aimin Zhou, Qingfu Zhang, Yaochu Jin, Edward Tsang, and Tatsuya Okabe. A model-based evolutionary algorithm for bi-objective optimization. In *Proceedings of the Congress on Evolutionary Computation (CEC 2005)*, pages 2568–2575, Edinburgh, U.K, September 2005. IEEE Press.
17. Aimin Zhou, Yaochu Jin, Qingfu Zhang, Bernhard Sendhoff, and Edward Tsang. Combining model-based and genetics-based offspring generation for multi-objective optimization using a convergence criterion. In *Proceedings of the Congress on Evolutionary Computation (CEC 2006)*, pages 3234–3241, Vancouver, BC, Canada, July 2006. IEEE Press.
18. Qingfu Zhang, Aimin Zhou, and Yaochu Jin. Modelling the regularity in estimation of distribution algorithm for continuous multi-objective evolutionary optimization with variable linkages. *IEEE Transactions on Evolutionary Computation*, 2006. Submitted.
19. Aimin Zhou, Qingfu Zhang, Yaochu Jin, Bernhard Sendhoff, and Edward Tsang. Modelling the population distribution in multi-objective optimization by generative topographic mapping. In *Parallel Problem Solving From Nature - PPSN IX*, pages 443–452, 2006.
20. Christopher Chatfield. *The Analysis of Time Series: An Introduction*. CRC Press, 2004.
21. Hui Li and Qingfu Zhang. A multiobjective differential evolution based on decomposition for multiobjective optimization with variable linkages. In *International Conference on Parallel Problem Solving from Nature (PPSN IX)*, pages 583–592, 2006.
22. Joshua D. Knowles, Lothar Thiele, and Eckart Zitzler. A tutorial on the performance assessment of stochastic multiobjective optimizers. Technical Report 214, Computer Engineering and Networks Laboratory, ETH Zurich, Gloriastrasse 35, 8092 Zurich, Switzerland, 2006.
23. Eckart Zitzler and Lothar Thiele. Multiobjective optimization using evolutionary algorithms - a comparative case study. In *Parallel Problem Solving from Nature - PPSN V*, volume 1498 of *Lecture Notes in Computer Science*, pages 292–304, Amsterdam, The Netherlands, September 1998. Springer.